Ektron

Redefining Web Content Management

Ektron CMS400.NET Developer Reference Manual

Version 5.5 for .NET 2.0, Rev 1

Ektron® Inc.
5 Northern Blvd., Suite 6
Amherst, NH 03031
Tel: +1 603-594-0249
Fax: +1 603-594-0258
Email: sales@ektron.com
http://www.ektron.com

Copyright 2006 Ektron®, Inc. All rights reserved.

EKTRON is a registered trademark of Ektron, Inc.

Version 5.5 for .NET 2.0, Rev 1, February 27, 2006

For updated versions of this manual, please visit:
http://www.ektron.com/cms400-web-cms.aspx?id=1701#documentation

EKTRON, INC. SOFTWARE LICENSE AGREEMENT

** READ THIS BEFORE LOADING SOFTWARE**

YOUR RIGHT TO USE THE PRODUCT DELIVERED IS SUBJECT TO THE TERMS AND CONDITIONS SET OUT IN THIS LICENSE AGREEMENT. USING THIS PRODUCT SIGNIFIES YOUR AGREEMENT TO THESE TERMS. IF YOU DO NOT AGREE TO THIS SOFTWARE LICENSE AGREEMENT, DO NOT DOWNLOAD.

CUSTOMER should carefully read the following terms and conditions before using the software program(s) contained herein (the “Software”). Downloading and/or using the Software or copying the Software onto CUSTOMER’S computer hard drive indicates CUSTOMER’S acceptance of these terms and conditions. If CUSTOMER does not agree with the terms of this agreement, CUSTOMER should not download.

Ektron, Inc. (“Ektron”) grants, and the CUSTOMER accepts, a nontransferable and nonexclusive License to use the Software on the following terms and conditions:

1. Right to use: The Software is licensed for use only in delivered code form. Each copy of the Software is licensed for use only on a single URL. Each license is valid for the number of seats listed below (the “Basic Package”). Any use of the Software beyond the number of authorized seats contained in the Basic Package without paying additional license fees as provided herein shall cause this license to terminate. Should CUSTOMER wish to add seats beyond the seats licensed in the Basic Package, the CUSTOMER may add seats on a block basis at the then current price for additional seats (see product pages for current price). The Basic Packages are as follows:

Ektron eMPower for ColdFusion -- Licensed for ten seats (10 named users) per URL.

Ektron CMS100 -- Licensed for five seats (5 named users) per URL.

Ektron CMS200 -- Licensed for ten seats (10 named users) per URL.

Ektron CMS300 -- Licensed for ten seats (10 named users) per URL.

Ektron CMS400 -- Licensed for ten seats (10 named users) per URL.

Ektron eWebEditPro -- Licensed for ten seats (10 named users) per URL.

Ektron eWebEditPro+XML — Licensed for ten seats (10 named users) per URL.

Ektron eWebWP — Licensed for unlimited seats per URL.

For purposes of this section, the term “seat” shall mean an individual user provided access to the capabilities of the Software.

The CUSTOMER may not modify, alter, reverse engineer, disassemble, or decompiler the Software. This software product is licensed, not sold.

2. Duration: This License shall continue so long as CUSTOMER uses the Software in compliance with this License. Should CUSTOMER breach any of its obligations hereunder, CUSTOMER agrees to return all copies of the Software and this License upon notification and demand by Ektron.

3. Copyright: The Software (including any images, “applets,” photographs, animations, video, audio, music and text incorporated into the Software) as well as any accompanying written materials (the “Documentation”) is owned by Ektron or its suppliers, is protected by United States copyright laws and international treaties, and contains confidential information and trade secrets. CUSTOMER agrees to protect the confidentiality of the Software and Documentation. CUSTOMER agrees that it will not provide a copy of this Software or Documentation nor divulge any proprietary information of Ektron to any person, other than its employees, without the prior consent of Ektron; CUSTOMER shall use its best efforts to see that any user of the Software licensed hereunder complies with this license.

4. Limited Warranty: Ektron warrants solely that the medium upon which the Software is delivered will be free from defects in material and workmanship under normal, proper and intended usage for a period of three (3) months from the date of receipt. Ektron does not warrant the use of the Software will be uninterrupted or error free, nor that program errors will be corrected. This limited warranty shall not apply to any error or failure resulting from (i) machine error, (ii) Customer's failure to follow operating instructions, (iii) negligence or accident, or (iv) modifications to the Software by any person or entity other than Company. In the event of a breach of warranty, Customer’s sole and exclusive remedy, is repair of all or any portion of the Software. If such remedy fails of its essential purpose, Customer’s sole remedy and Ektron’s maximum liability shall be a refund of the paid purchase price for the defective Products only. This limited warranty is only valid if Ektron receives written notice of breach of warranty within thirty days after the warranty period expires.

5. Limitation of Warranties and Liability: THE SOFTWARE AND DOCUMENTATION ARE SOLD “AS IS” AND WITHOUT ANY WARRANTIES AS TO THE PERFORMANCE, MERCHANTIBILITY, DESIGN, OR OPERATION OF THE SOFTWARE. NO WARRANTY OF FITNESS FOR A PARTICULAR PURPOSE IS OFFERED. EXCEPT AS DESCRIBED IN SECTION 4, ALL WARRANTIES EXPRESS AND IMPLIED ARE HEREBY DISCLAIMED.

NEITHER COMPANY NOR ITS SUPPLIERS SHALL BE LIABLE FOR ANY LOSS OF PROFITS, LOSS OF BUSINESS OR GOODWILL, LOSS OF DATA OR USE OF DATA, INTERRUPTION OF BUSINESS NOR FOR ANY OTHER INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY KIND UNDER OR ARISING OUT OF, OR IN ANY RELATED TO THIS AGREEMENT, HOWEVER,

CAUSED, WHETHER FOR BREACH OF WARRANTY, BREACH OR REPUDIATION OF CONTRACT, TORT, NEGLIGENCE, OR OTHERWISE, EVEN IF COMPANY OR ITS REPRESENTATIVES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH LOSS.

6. Miscellaneous: This License Agreement, the License granted hereunder, and the Software may not be assigned or in any way transferred without the prior written consent of Ektron. This Agreement and its performance and all claims arising from the relationship between the parties contemplated herein shall be governed by, construed and enforced in accordance with the laws of the State of New Hampshire without regard to conflict of laws principles thereof. The parties agree that any action brought in connection with this Agreement shall be maintained only in a court of competent subject matter jurisdiction located in the State of New Hampshire or in any court to which appeal therefrom may be taken. The parties hereby consent to the exclusive personal jurisdiction of such courts in the State of New Hampshire for all such purposes. The United Nations Convention on Contracts for the International Sale of Goods is specifically excluded from governing this License. If any provision of this License is to be held unenforceable, such holding will not affect the validity of the other provisions hereof. Failure of a party to enforce any provision of this Agreement shall not constitute or be construed as a waiver of such provision or of the right to enforce such provision. If you fail to comply with any term of this License, YOUR LICENSE IS AUTOMATICALLY TERMINATED. This License represents the entire understanding between the parties with respect to its subject matter.

YOU ACKNOWLEDGE THAT YOU HAVE READ THIS AGREEMENT, THAT YOU UNDERSTAND THIS AGREEMENT, AND UNDERSTAND THAT BY CONTINUING THE INSTALLATION OF THE SOFTWARE, BY LOADING OR RUNNING THE SOFTWARE, OR BY PLACING OR COPYING THE SOFTWARE ONTO YOUR COMPUTER HARD DRIVE, YOU AGREE TO BE BOUND BY THIS AGREEMENT'S TERMS AND CONDITIONS. YOU FURTHER AGREE THAT, EXCEPT FOR WRITTEN SEPARATE AGREEMENTS BETWEEN EKTRON AND YOU, THIS AGREEMENT IS A COMPLETE AND EXCLUSIVE STATEMENT OF THE RIGHTS AND LIABILITIES OF THE PARTIES.

Copyright 1999 - 2005 Ektron, Inc. All rights reserved.

LA10031, Revision 1.4

What's New in the 5.5 Release

- The Poll Server Control allows you to add a Poll or Survey to a Web form, scale it, and have it appear in an isolated display (an iFrame). *See Also:* "Poll Server Control" on page 191.
- Five Blog server controls have been added to give you flexibility when displaying a blog. The Blog Server Control displays all of the blog parts together in one server control. The other four: BlogEntries, BlogPost, BlogRoll and BlogCalendar display each blog part individually. *See Also:* "Blog Server Controls" on page 44.
- The AnalyticTracker Server Control tracks statistics about visits to your Web site. *See Also:* "Content Analytics Server Control" on page 111.
- To collect ratings and feedback about your Web site, use the ContentRating Server Control. It places a scale a Web page that allows visitors to rate and comment on content on your site. *See Also:* "ContentRating Server Control" on page 123.
- The PostResultsAt property in the FormBlock Server Control has been removed. This is now taken care of in the form on the Post Back Message tab. See the User Manual section "Working with HTML Forms" > "Creating a New Form" > "Redirect to an Action Page and Forwarding Form Data".

Summary Table of Contents

What's New in the 5.5 Release i

About This Manual 1

Introduction to Ektron CMS400.NET Server Controls 2

- Blog Server Controls 44
- BreadCrumb Server Control 57
- Calendar Server Control 68
- Collection Server Control 103
- Content Analytics Server Control 111
- ContentBlock Server Control 113
- ContentRating Server Control 123
- FormBlock Server Control 129
- LanguageAPI Server Control 134
- LanguageSelect Server Control 137
- ListSummary Server Control 139
- Membership Server Control 146
- Menu Server Control 153
- DhtmlMenu Server Control 170
- MetaData Server Control 178
- MetaDataList Server Control 183
- Login Server Control 188
- Poll Server Control 191
- RssAggregator Server Control 196
- Search Server Control 200
- IndexSearch Server Control 234

Dreamweaver Extension Support 252

Standard ASP.NET Methods 363

Custom ASP Functions 370

- Login Function 373
- Content Block Function 375
- Add Content 385

Search Functions 389
Search Display Function 400
List Summary Function 406
XML List Summary Function 414
Archive List Summary Function 419
Single Summary Function 420
RSS Summary Function 424
Collection Function 430
RSS Collection Function 436
XML Collection Function 438
Random Content 443
Random Summary 444
Event Calendar Function 445
Calendar Function 448
Form Function 452
MultiPurpose Function 455
Metadata Function 456
Show Site Language Function 464
DHTML Menus 465
XML Menu 470

Custom ColdFusion Functions 472

Custom PHP Functions 500
Custom API 528

Custom JSP Functions 529

Web Services 551

Syndicating Web Content as RSS 575
Syndicating via .NET 577

Learning About Visual Studio .NET 580

Customizing the Ektron CMS400.NET Workarea .. 582

Customizing Validation 583

Additional Information 602

Detailed Table of Contents

What's New in the 5.5 Release i

About This Manual 1

Introduction to Ektron CMS400.NET Server Controls 2

Opening the Sample Project 3
Setting Up a Template 4
Making the Server Controls Available 5
Removing the Server Controls 7
Updating the Server Controls 7
Inserting Server Controls Using Drag and Drop 7
Adding a Server Control to a Template 7
ServerControlWS.asmx 9
Using CMS Explorer to Browse Your Ektron CMS400.NET Site... 10
Build the Project 11
Inserting Server Controls Programmatically 11
Steps to Insert a Server Control Programmatically 12
Step 1: Declare the Namespace 13
Step 2: Create an Instance of the New Control 13
Step 3: Set the Properties 14
Step 4: Call the Fill Method 15
Step 5: Use .text Property to Display the Control 15
Server Control Properties in Code Behind 16
Displaying Properties for a Control 16
Mutual Server Control Properties in Code Behind 17
Accessing Additional Properties 19
Accessing Items in an Array 25
Referencing the Page Property 25
Customizing Server Controls 26
Recognizing the Server Control Within the HTML 27
Recognizing the Server Control Within the Code Behind 27
Customizing the Server Control in the Code Behind 28
Troubleshooting Error Creating Control Message 28
Using Drag & Drop and Programmatically Together 29
Data Binding with Server Controls 31
Steps to Data Binding Using Drag and Drop Server Controls 32
Customizing the DataGrid Table View 35
Caching with Server Controls 37
Caching Individual Content 39

Page Level Caching 40
List of Server Controls 41
Blog Server Controls 44
Using the Blog Server Control vs. Using the Individual Blog Server Controls 44
Blog Server Control 45
Blog Server Control Properties 46
BlogEntries Server Control 46
BlogEntries Server Control Properties 47
BlogPost Server Control 49
BlogPost Server Control Properties 49
BlogCalendar Server Control 51
BlogCalendar Server Control Properties 51
Using the BlogCalendar and BlogEntries Server Controls Together 52
Steps to Use BlogCalendar and BlogEntries Together. 52
BlogRoll Server Control 55
BlogRoll Server Control Properties 55
BreadCrumb Server Control 57
BreadCrumb Properties Apply to Web Forms, Not Web Pages 59
Types of BreadCrumb Server Control Properties 59
Properties that Determine a Breadcrumb Trail's Appearance 60
Properties that Determine How Pages Associated with the Form Appear on a Breadcrumb Trail 60
BreadCrumb Server Control Properties 61
BreadCrumb Metadata Type 64
Using the BreadCrumb Server Control 65
Making a Content Block's Title Appear in the Breadcrumb Trail 66
Steps to Use a Content Block's Title as a Breadcrumb 66
Calendar Server Control 68
Calendar Server Control Properties 70
Calendar Style Sheet 74
calendarStyles.css Class Description 75
Single Day View (Day) 75
Month View (month) 79
Inline View (inline) 91
Month-Up with Event (monthupwithevent) 93
Event Type Control (used in month display) 100
Collection Server Control 103
Collection Server Control Properties 104
Example of ecmNavigation Display 105
ecmNavigation XSL code 106
Example of ecmTeaser Display 106
ecmTeaser XSL code 107

Retrieving the XML Structure of a Collection.....107
Using the Collection Server Control Programmatically Example.....108
Content Analytics Server Control.....111
Content Analytics Server Control Properties..... 111
Ektron Properties.....111
ContentBlock Server Control..... 113
Static Content Block..... 113
ContentBlock Server Control Properties.....114
Dynamic Content Block 115
XML Content Block..... 118
The OverrideXslt and DisplayXslt Properties 119
Retrieving the XML Structure of an XML Content Block..... 121
Using the ContentBlock Server Control Programmatically Example 122
ContentRating Server Control 123
Content Rating Server Control Properties 123
Ektron Properties..... 124
Ektron Label Properties..... 125
Initial Graph Display..... 125
Initial Graph Properties 126
Complete Graph Display..... 127
Complete Graph Properties 127
FormBlock Server Control 129
FormBlock Server Control Properties 130
Automatic versus Manual Generation of Form Tags.....132
LanguageAPI Server Control 134
LanguageSelect Server Control vs. LanguageAPI Server Control 134
LanguageAPI Server Control Properties 134
Code Behind Only Properties.....135
Using the LanguageAPI Server Control Programmatically .136
LanguageSelect Server Control..... 137
LanguageSelect Server Control vs. LanguageAPI Server Control 137
LanguageSelect Server Control Properties.....138
ListSummary Server Control 139
ListSummary Server Control Properties 139
OrderKey Property..... 143
List Summary Example 144
Retrieving the XML Structure of a List Summary 145
Membership Server Control 146
Membership Server Control Properties.....147
DisplayMode..... 151

Menu Server Control **153**
Menu Server Control vs. DhtmlMenu Server Control **153**
Menu Server Control Properties **154**
Using DisplayXslt Samples **156**
SampleMenu 156
SampleMenu XSL code 157
TreeMenu 159
TreeMenu XSL code 161
Retrieving the XML Structure of a Menu **169**
DhtmlMenu Server Control **170**
Menu Server Control vs. DhtmlMenu Server Control **171**
DhtmlMenu Server Control Properties **171**
Using a Style Sheet 175
MetaData Server Control **178**
MetaData Server Control vs. MetaDataList Server Control **178**
MetaData Server Control Properties **178**
Using the Simple Dublin Core Metadata Standard 180
Using the MetaData Server Control **181**
MetaDataList Server Control **183**
MetaData Server Control vs. MetaDataList Server Control **184**
MetaDataList Server Control Properties **184**
Retrieving the XML Structure of a MetadataList **187**
Login Server Control **188**
Placing a Login Button **189**
Login Server Control Properties **189**
Poll Server Control **191**
Poll Server Control Properties **192**
Automatic versus Manual Generation of Form Tags **195**
RssAggregator Server Control **196**
RSSAggregator Server Control Properties **197**
Using the RssAggregator Server Control **198**
Search Server Control **200**
Introduction **200**
Search Server Control vs. IndexSearch Server Control **202**
Search Server Control Properties **202**
Property Usage Table 202
Property Descriptions 204
Text Box Examples 214
Button Image Source Examples 215
Search Button Text Examples 216
Display Examples 217
ShowDate Examples 217
ShowExtendedSearch 218
TargetFolderList 220
Using the Advanced Search Properties **221**

Basic and Standard Searches 223
Advanced Search 225
Using Standard and Advanced Searches Together 226
Fieldset Properties 229
Retrieving the XML Structure of a Search 230
Displaying Search Results on a Separate Page 231
IndexSearch Server Control 234
IndexSearch Server Control vs. Search Server Control 235
IndexSearch Server Control Properties 235
SearchParmXML Property 240
Using the IndexSearch Server Control Programmatically Example 243
Programmatically Predefined General Search Results Replaced By Specific Search Results 246
Turn Off Caching In Netscape Navigator and Mozilla Firefox 250

Dreamweaver Extension Support 252
Requirements 252
Preparing Dreamweaver for Ektron CMS400.NET Functions 253
Installing the Extensions 253
Setup 258
Using the Ektron CMS400.NET Extensions 262
Custom Function Symbols 264
Ektron CMS400.NET Command Menus 266
Inserting Custom Functions 271
CMS Function 271
Accessing CMS Content Functions 271
Login 272
Inserting a Content Block 275
Inserting a Dynamic Content Block 279
Inserting a Collection 281
Inserting a List Summary 285
Inserting a Metadata Function for ASP.NET pages 290
Inserting a Metadata Function for ASP, CF, JSP or PHP pages 293
Inserting a Search Function 296
Inserting a Search Display Function 305
Inserting a Single Summary Function 307
Inserting a Random Content Function 310
Inserting a Random Summary Function 312
Inserting an Event Calendar Function 314
Inserting a Form 317
Inserting a Multipurpose content block 321
Inserting an Advanced Calendar Tag 322

Inserting an XML Index Search 324
Inserting Display By Meta Value 330
Inserting a Standard Menu 335
Inserting a DHTML Menu 338
Inserting a Language Select Box 344
Inserting a Language API 346
Inserting an RSS Aggregator 349
Inserting a BreadCrumb Trail 351
Inserting a Membership Control 355
XML Functions... 360
Accessing XML Function Commands 360
Insert Content XSLT Tag... 360

Standard ASP.NET Methods 363
Replacement of Standard Methods 363
Referencing Methods in Visual Basic 365
Adding a New Web Form 365
Requirements for Standard Methods 366
Standard Methods 367
Addcontent 368
Add Content Example 369

Custom ASP Functions 370
Login Function 373
Login Tag Example 374
Placing Your Login Button 374
Content Block Function 375
Content Block Examples 378
Static Content Block 379
Dynamic Content Block 380
XML Content Block 381
The Function's Arguments 382
Dynamic and Static Combination 384
Add Content 385
Add Content Example 387
Search Functions 389
Search Tag Example 392
Search Tag Attributes 393
Starting Folder 393
Recursive 393
Target Page 394
Text Box 394
Maximum Characters 395
Button Image Source 395
Button Text 396

Font Face 397
Font Size 398
Horizontal 399
Spare 1 399
Search Display Function 400
Search Display Example 401
Search Display with Default Content 401
Search Display Attributes 402
Max Number of Content Blocks Returned 403
Style Info 404
Show Date 404
List Summary Function 406
List Summary Example 408
List Summary Attributes 409
Starting Folder 409
Recursive 409
Show Summary 410
Style Info 410
Order By 411
Ascending/Descending 411
Show Info 412
Max Number 413
XML List Summary Function 414
Archive List Summary Function 419
Single Summary Function 420
Single Summary Example 421
Single Summary Attributes 421
Content ID 421
Show Summary 422
Style Info 422
Show Info 423
Spare 1 423
RSS Summary Function 424
RSS Summary Example 425
RSS Summary Attributes 428
Starting Folder 428
Recursive 428
Order By 428
Ascending/Descending 429
Max Number 429
Collection Function 430
Collection Examples 430
The ecmNavigation Display Function 431
The ecmTeaser Display Function 432
Collection Attributes 435
ID 435

DisplayFunction 435
RSS Collection Function 436
XML Collection Function 438
Random Content 443
Random Summary 444
Event Calendar Function 445
Event Calendar Style Sheet 445
Event Calendar Sample 446
Calendar Function 448
Calendar Style Sheet 450
Event Calendar Sample 450
Form Function 452
MultiPurpose Function 455
Metadata Function 456
Creating an Item List 456
ecmMetadata Example 457
More ecmMetadata Examples 459
Using a Dynamic Metadata Tag 460
ecmMetadata Attributes 461
Item List 461
Spare 1 463
Show Site Language Function 464
DHTML Menus 465
Additional Documentation in Ektron Knowledge Base 465
Displaying a Menu on a Web Page 465
GenerateMenu and DropMenu 465
Displaying Menus in a Multi-language System 466
Customizing the Menu's Appearance 466
ek_Menu.js 467
FontFamily 468
menuLiteBgColor 468
XML Menu 470

Custom ColdFusion Functions 472
Login Function 475
Login Tag Example 475
Content Block Function 475
Static Content Block 476
Dynamic Content Block 476
XML Content Block 476
Add Content 476
Add Content Example 477
Random Content Function 478
Random Summary Function 478

Search Function **478**
Search Tag Example 479
Search Display Function **480**
Search Display Example 480
List Summary Function **480**
List Summary Example 481
XML List Summary Function **481**
XML List Summary Example 482
RSS List Summary Function **482**
List Summary Example 483
Archive List Summary Function **483**
Single Summary Function **483**
Single Summary Example 484
Collection Function **484**
Example Using the ecmNavigation Display Function 485
Example Using the ecmTeaser Display Function 487
Collection Variables 489
DisplayFunction 489
XML Collection Function **489**
RSS Collection Function **490**
Event Calendar Function **490**
Event Calendar Sample 493
Form Function **493**
MultiPurpose Function **494**
Metadata Function **494**
DHTML Menus **497**
Additional Documentation in Ektron Knowledge Base 497
Displaying a Menu on a Web Page 497
GenerateMenu and DropMenu 497
Customizing the Menu's Appearance 498
ek_Menu.js 498
XML Menu Function **498**
Language Select Function **499**

Custom PHP Functions **500**
Login Function **503**
Login Function Example 504
Content Block Function **504**
Static Content Block 504
Dynamic Content Block 505
XML Content Block 505
Add Content **505**
Add Content Example 506
Search Function **506**
Search Function Example 507

Search Display Function **507**
Search Display Function Example 508
List Summary Function **508**
List Summary Example 509
Archive List Summary Function **509**
Single Summary Function **509**
Single Summary Example 510
XML List Summary Function **510**
RSS List Summary Function **511**
Collection Function **511**
Collection Examples 512
Example Using the ecmNavigation Display Function 512
Example Using the ecmTeaser Display Function 513
Collection Variables 515
DisplayFunction 515
XML Collection Function **515**
RSS Collection Function **516**
Random Content Function **516**
Random Summary Function **517**
Event Calendar Function **517**
Event Calendar Sample 520
Form Function **520**
MultiPurpose Function **521**
Metadata Function **522**
DHTML Menus **524**
Additional Documentation in Ektron Knowledge Base 525
Displaying a Menu on a Web Page 525
GenerateMenu and DropMenu 525
Customizing the Menu's Appearance 526
ek_Menu.js 526
XML Menu Function **526**
Language Select Function **527**
Custom API **528**

Custom JSP Functions **529**
Calendar Function **532**
Calendar Sample 532
Event Calendar Function **533**
Event Calendar Sample 533
Collection Function **533**
Collection Sample 534
XML Collection Function **534**
Random Content Function **535**
Random Summary Function **535**
RSS Collection Function **536**

Add Content **536**
Add Content Example 537
Content Block Function **537**
Static Content Block 537
Dynamic Content Block 538
XML Content Block 538
Form Function **538**
Language Select Function **540**
Login Function **540**
Login Function Example 541
DHTML Menus **541**
Additional Documentation in Ektron Knowledge Base 542
Displaying a Menu on a Web Page 542
GenerateMenu and DropMenu 542
Customizing the Menu's Appearance 543
ek_Menu.js 543
XML Menu Function **543**
Metadata Function **544**
Search Function **546**
Search Function Example 547
Search Display Function **548**
Search Display Function Example 548
List Summary Function **548**
List Summary Example 549
XML List Summary Function **549**
RSS List Summary Function **550**
Single Summary Function **550**
Single Summary Example 550

Web Services **551**
Introduction to Web Services **551**
CMS400.NET's Implementation of Web Services 552
Benefits of Using Web Services 552
WS Components and CMS400.NET Architecture **552**
Software and Hardware Requirements **552**
Installation of Files **553**
Setup Instructions **553**
Providing Web Service Information in Your CMS **555**
CMS400.NET Methods 555
The CMS400.NET Content Sample Page 559
The Search Display Method Parameters 562
Sample Output from One Method 564
GetContentBlock 564
The News.aspx Sample (non-XML) 565
Identifying the Code Behind File 565

Setting Up an Instance of Web Services 566
Getting Content Blocks 566
The News.aspx Sample (XML) 568
Setting Up Instances of Web Services 568
Getting Content Blocks 568
Creating a Page that Provides a Web Service 573

Syndicating Web Content as RSS 575
What is RSS Syndication? 575
The RSS Specification 575
RSS Syndication in Ektron CMS400.NET 576
Syndicating via .NET 577
Architecture of RSS Syndication 577
aspx pages 577
Retrieving a Collection or List Summary as an RSS Feed 577
Creating Your Own aspx Collection Page 577
Example 578
Remarks 578
Example 578
Creating Your Own aspx List Summary Page 578
Example 579
Remarks 579
Example 579

Learning About Visual Studio .NET 580
Grid Layout vs. Flow Layout 580

Customizing the Ektron CMS400.NET Workarea .. 582

Customizing Validation 583
Customizing Validation Options 583
Validation Elements in the Configuration Data 583
Validation Attributes 584
Validation Sub-elements 584
Choice Sub-element 584
Choice Attributes 585
Choice Sub-Elements 586
Defining a Schema Fragment 586
Defining a Calculation 587
Defining Validation 589
Defining an Error Message 589
Defining Custom Validation 589
Saving Invalid Documents 593

Custom Script that Handles Saving Invalid Files 594
The onerror Event 594
Calculated Fields 595
Icons on the Select Field or Group Screen..... 595
Learn More about XPath 596
Xpath Operators..... 596
Xpath Functions 598
XPath References 600

Additional Information 602
Supporting Documents 602
Support 602
Sales 602
Information 602

About This Manual

This manual can be used as a reference for the developer, or the person is who is setting up your Ektron CMS400.NET Web site.

To utilize the full potential of an Ektron CMS400.NET driven Web site, learning and using the server controls supplied by Ektron is essential. These server controls help make your site more robust and easier to maintain. For more information, see ”Introduction to Ektron CMS400.NET Server Controls” on page 2.

This document explains how to manually use custom tags for each platform supported by Ektron CMS400.NET:

- ”Standard ASP.NET Methods” on page 363
- ”Custom ASP Functions” on page 370
- ”Custom ColdFusion Functions” on page 472
- ”Custom PHP Functions” on page 500

Besides explaining how to insert the functions to your Web templates manually, this manual also explains how to create Ektron CMS400.NET templates in a WYSIWYG environment by using the Ektron supplied server controls for Visual Studio .NET or by using the Dreamweaver extension supplied with the installation. See ”Dreamweaver Extension Support” on page 179 for additional information.

Ektron CMS400.NET has also implemented support for Web Services as well as syndication using RSS. Information about configuration and use of these features is also found in the following sections:

- ”Web Services” on page 551
- ”Syndicating Web Content as RSS” on page 575

Introduction to Ektron CMS400.NET Server Controls

Ektron CMS400.NET's server controls let you insert many standard methods and properties within the Visual Studio .NET environment. This means that you can see the effect of your changes in real time -- you don't have to modify a page then compile a sample project to see the results.

You can insert server controls using drag and drop or programmatically. You can also use databinding to retrieve and display data from **Ektron CMS400.NET**. This chapter explains the use of the Ektron CMS400 Server Controls through the following topics.

- "Opening the Sample Project" on page 3
- "Setting Up a Template" on page 4
- "Making the Server Controls Available" on page 5
- "Inserting Server Controls Using Drag and Drop" on page 7
- "Inserting Server Controls Programmatically" on page 11
- "Using Drag & Drop and Programmatically Together" on page 29
- "Data Binding with Server Controls" on page 31
- "Caching with Server Controls" on page 37
- "List of Server Controls" on page 41

Additional information and examples for Ektron Server Controls are available online by going to the Developers section of the sample site that was installed. Here is the path:

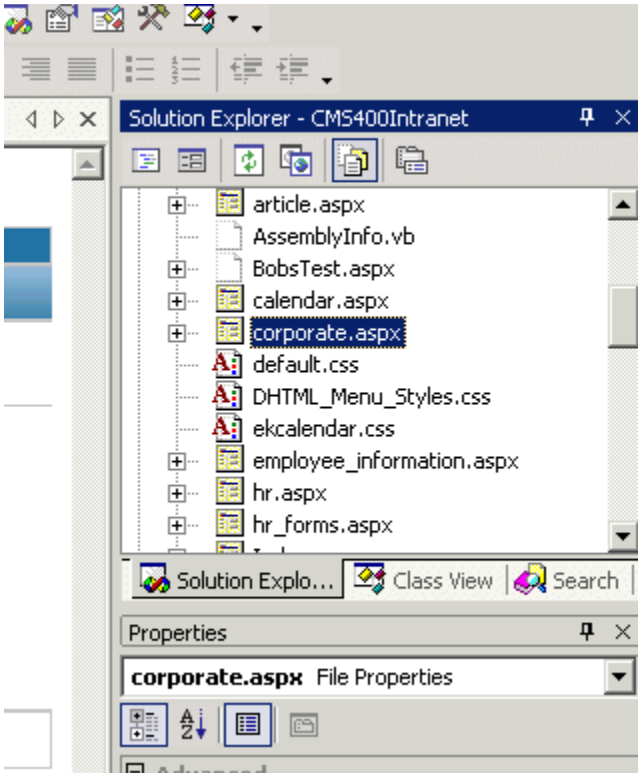
- `http://localhost/CMS400Example/Developer.aspx`

Replace `localhost` with the `webroot` where you installed the Example site.

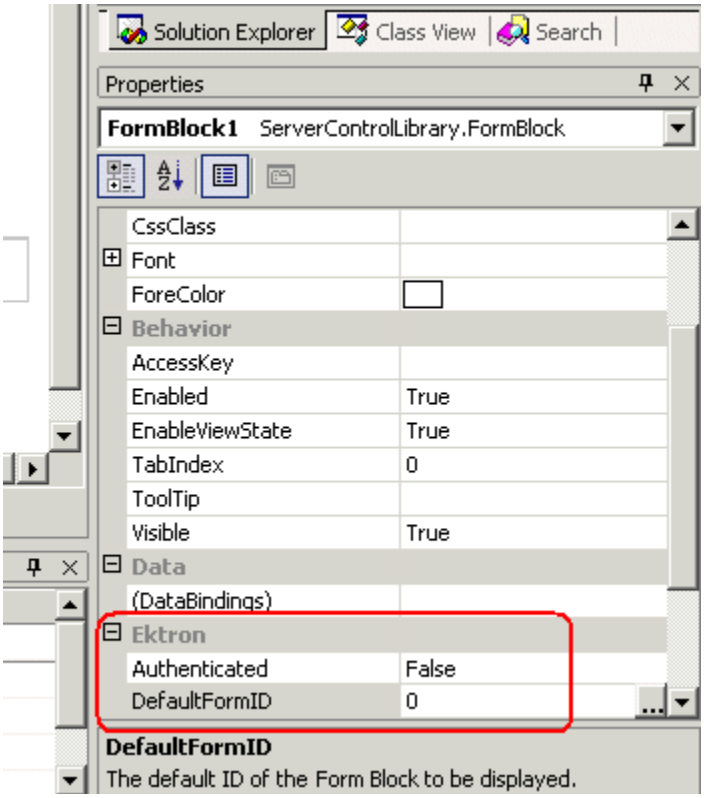
Opening the Sample Project

To help get you started, these directions explain how to open the sample site delivered with **Ektron CMS400.NET**.

1. Browse to and double click Ektron CMS400.NET's solution file, `localhost/CMS400Example/CMS400Example.sln`.
2. The sample site project opens.
3. To work on a template page, click it from the Solution Explorer.



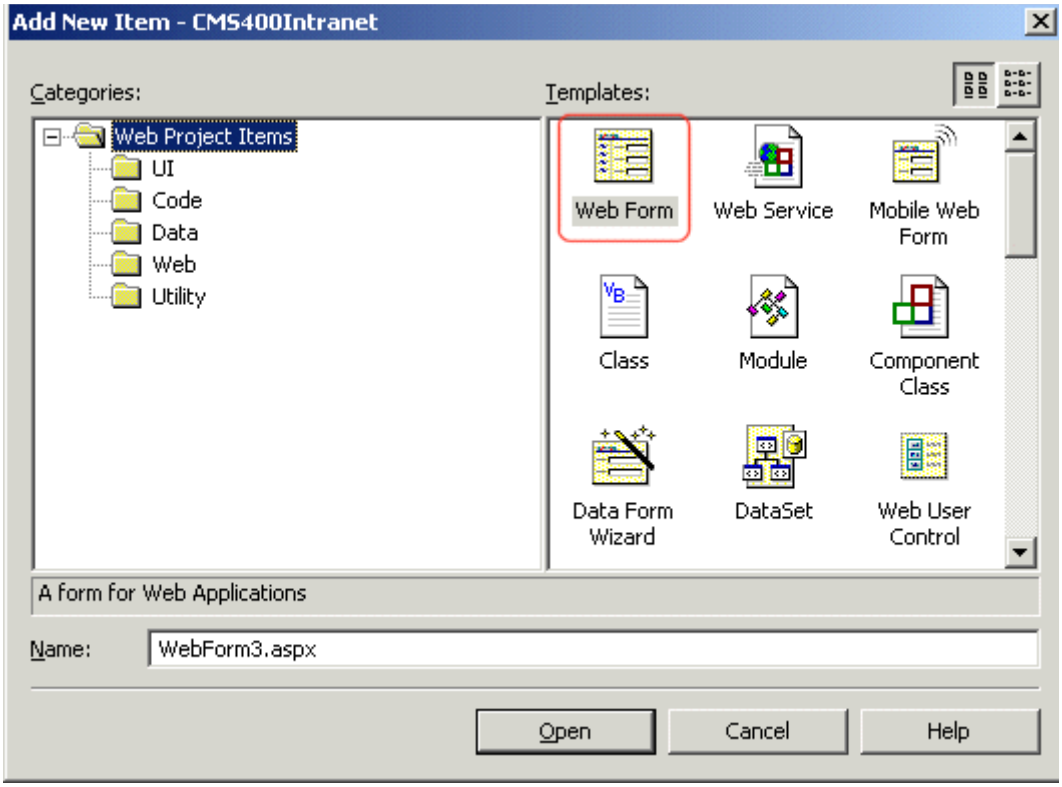
When you select a page, its properties appear in the Properties area (below Solution Explorer), and the page appears in the center of the screen. A control's properties include several standard .NET properties along with Ektron-specific ones. The Ektron properties are labeled as illustrated below.



Setting Up a Template

To create a new template (.aspx) page, follow these steps.

1. Click **Project > Add Web Form**.
2. On the Add New Item screen, click Web Form and assign a name.



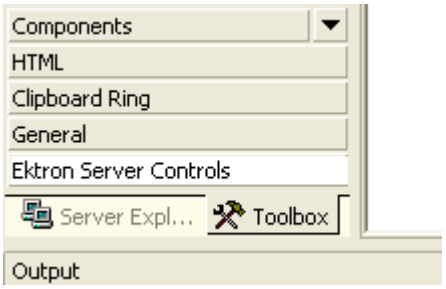
3. Add controls to determine the page content.

Making the Server Controls Available

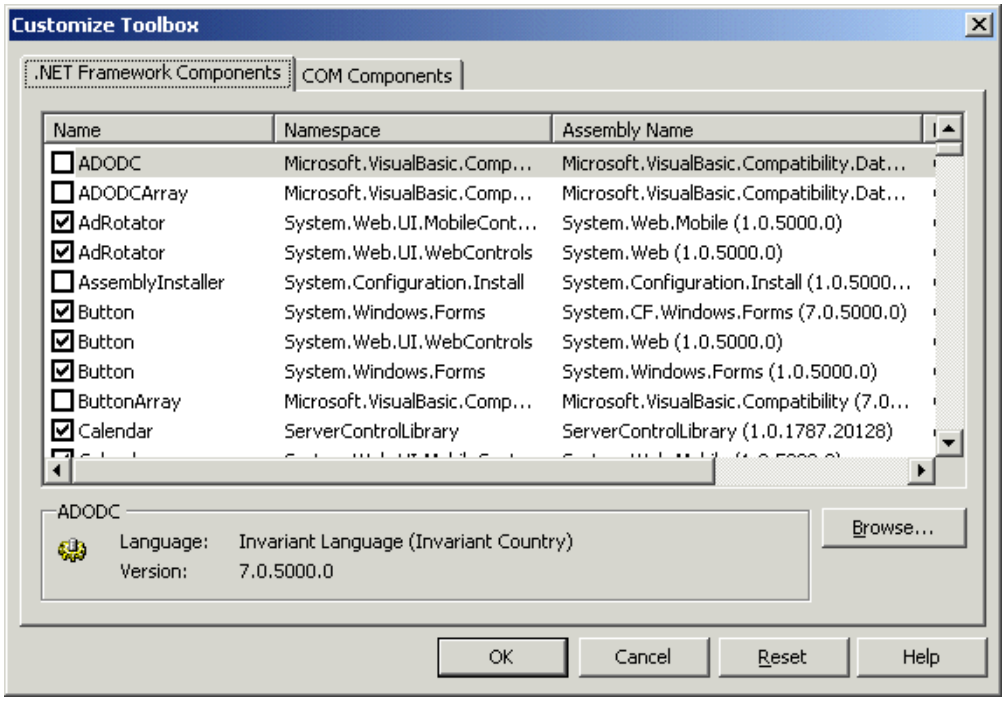
NOTE You must copy the dlls to a local drive before installing them. You cannot copy them from a network drive.

1. Display the Visual Studio .NET toolbox (**View > Toolbox**).
2. Right click the mouse within the Toolbox.
3. Click **Add Tab**.

4. Type **Ektron Server Controls** then press enter.



5. Click the Ektron Server Controls Tab.
6. Right click the mouse in the empty area.
7. Click **Add/Remove Items**. If using Visual Studio 2005, click **Choose Items**.
8. The Customize Toolbox dialog appears.



9. Select the **.NET Framework Components** tab.
10. Browse to the directory that stores Ektron CMS400.NET's dll files, `localhost/CMS400Example/bin`.

11. Add the Ektron.Cms.Controls.dll file and press **OK**. This file provides access to **Ektron CMS400.NET**'s server controls.

Removing the Server Controls

1. Display the Visual Studio .NET toolbox (**View > Toolbox**).
2. Right click the mouse within the Toolbox.
3. Click **Add/Remove Items**. If using Visual Studio 2005, click **Choose Items**.
4. Click **Namespace** or **Assembly Name** to sort the Server Controls by manufacturer.
5. Uncheck all boxes that are Ektron.Cms.Controls.
6. Click **OK**
7. Right click the Ektron Server Control Tab.
8. Click **Delete Tab** to remove it.

Updating the Server Controls

To update the server controls you must first remove the existing server controls in Visual Basic, then add the new server controls.

For information on removing server controls, see "Removing the Server Controls" on page 7

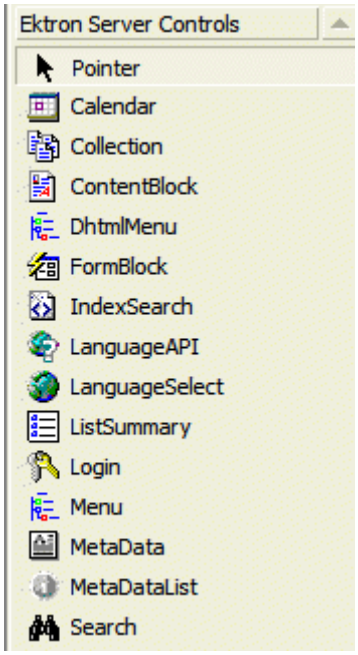
For information on adding server controls, see "Making the Server Controls Available" on page 5

Inserting Server Controls Using Drag and Drop

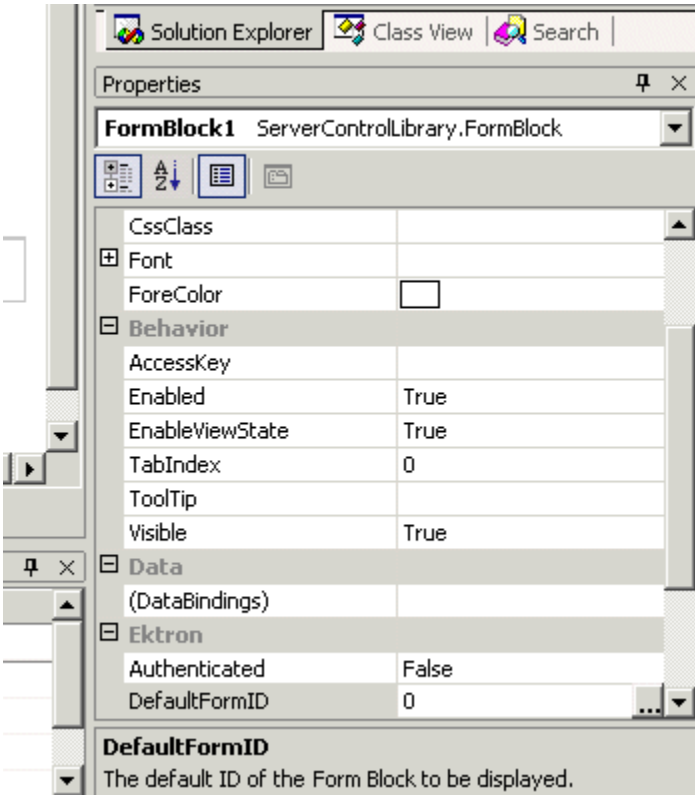
Because Visual Studio .NET is a visual environment, you can watch the page layout change as you add or move a control and adjust its properties. And, whenever you need to identify an **Ektron CMS400.NET** object (such as content block or collection), you can use the CMS Explorer to visually browse to and select it.

Adding a Server Control to a Template

1. Display the Visual Studio .NET toolbox (**View > Toolbox**).
2. Click the **Ektron Server Control** tab. **Ektron CMS400.NET**'s server controls appear.



3. Drag a server control and drop it into the desired location on the page.
4. As desired, modify the control's properties using the Properties area of Visual Studio .NET.



For a description of server control properties, see "List of Server Controls" on page 41.

ServerControlWS.asmx

> Best Practice
>
> For improved security, Ektron recommends renaming or removing the Web services file when you move it to your production server. After installation, this file is named ServerControlWS.asmx and resides in the /CMS400Example/Workarea/ folder in your Web root.

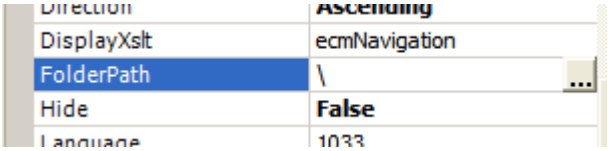
ServerControlWS.asmx is the Web service that lets the server controls talk to **Ektron CMS400.NET**. The path is coded in the web.config file. This is how it appears:

```
<!-- Web Service URL for Server Controls design time  --> <add key="WSPath"
value="http://localhost/cms400example/Workarea/ServerControlWS.asmx" />
```

You will need to edit this line if you change the location or name of the ServerControlWS.asmx file.

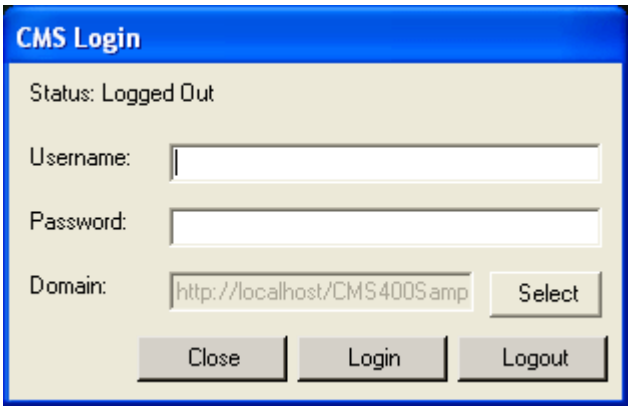
Using CMS Explorer to Browse Your Ektron CMS400.NET Site

Ektron CMS400.NET provides a window (called the CMS Explorer) that lets you browse through your Web site to identify CMS objects such as folders, calendars, blogs and content blocks. For example, if you insert a List Summary Server Control, its **FolderPath** property identifies the folder whose contents are displayed. If you are not sure of the folder path, click the gray button.

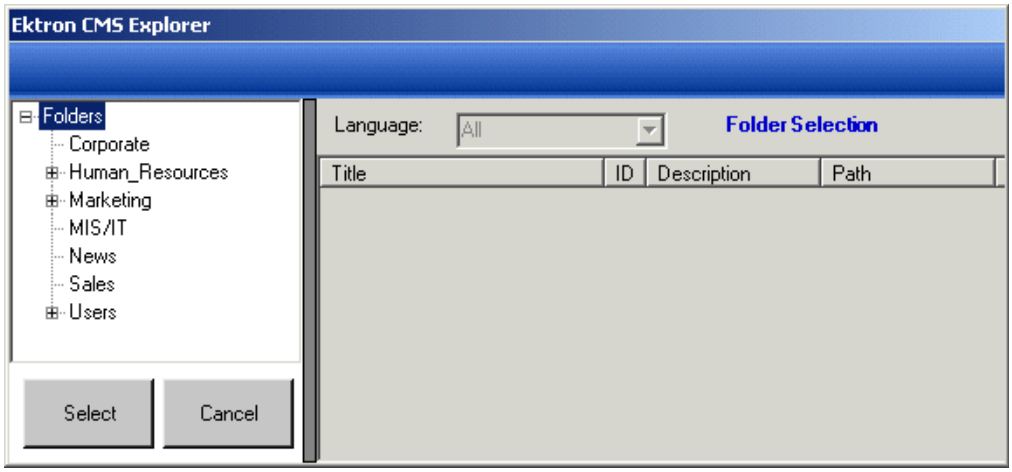


When you do, a login screen appears (illustrated below).

NOTE A user cannot be logged into the CMS Explorer and the CMS400.NET Workarea at the same time. If you log into the CMS Explorer while logged into the Workarea, you will receive the following error when you try to use the Workarea: `Error: The current user does not have permission to carry out this request [CBR R31].` If you log into the Workarea while logged into the CMS Explorer, you will need to re-login to the CMS Explorer when you return to using it.



Enter your **Ektron CMS400.NET** username and password. When you do, the CMS Explorer window appears, providing a visual way to browse your **Ektron CMS400.NET** Web site.



Navigate through the folders, select an object, then click the **Select** button. The selected object is pasted to the server control property.

NOTE Although you see the object in the selected language in Visual Studio .NET, the language is not stored. For example, if you select a German content block whose ID=2, **Ektron CMS400.NET** only stores `content block ID=2`. When a visitor to your site browses to that page, the content block does not appear in the selected language. Instead, it is determined by a cookie or the user's language selection.

Build the Project

After you open the solution file in Visual Studio .NET and add the required files, build the project. This may take a while the first time you do so. After the project is built, a browser opens and it is displayed as a Web page.

You can also view a Web page while working on it by right clicking on the Web form and clicking **View in Browser**.

Inserting Server Controls Programmatically

Besides using the drag-and-drop method to insert a server control (explained in "Inserting Server Controls Using Drag and Drop" on

page 7), you can insert one programmatically. You might prefer to insert server controls programmatically for several reasons, such as

- you want the control to be loaded into memory only under certain conditions. In this case, insert the logic that only displays the control if the condition exists.
- you want to display only certain properties of an object, such as the title of last edited date of a content block.

Note that you can drag and drop controls on your page then customize them using code behind. For more information, see:"Using Drag & Drop and Programmatically Together" on page 29.

The following sections explain how to use the server controls programmatically.

- "Steps to Insert a Server Control Programmatically" on page 12
- "Server Control Properties in Code Behind" on page 16
- "Customizing Server Controls" on page 26

Steps to Insert a Server Control Programmatically

To insert an Ektron server control programmatically, follow these steps.

"Step 1: Declare the Namespace" on page 13

"Step 2: Create an Instance of the New Control" on page 13

"Step 3: Set the Properties" on page 14

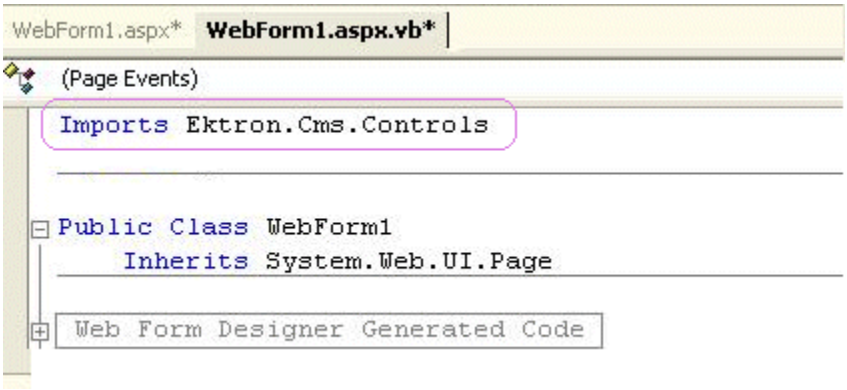
"Step 4: Call the Fill Method" on page 15

"Step 5: Use .text Property to Display the Control" on page 15

Each step is described below. These steps show an example of using the Collection Server Control.

Step 1: Declare the Namespace

Before inserting the server control, you can declare the namespace at the top of the code behind Visual Basic file, as shown below.



NOTE You do not need to declare a namespace. However if you do not, you must fully qualify objects that you create.

> Best Practice
>
> For any customization of the CMS, classes or controls that inherit from Ektron classes, it is recommend that you create your own namespace within 'Ektron.Cms.Custom'. For example, if your company is 'AcmeExampleTech, Inc.' you should create all of your custom classes within the namespace 'Ektron.Cms.Custom.AcmeExampleTech'.

Step 2: Create an Instance of the New Control

Next, declare a control as an object in the code behind. In this example, we create a collection named `MyColl`.

```
dim MyColl as New Collection
```

or

```
dim MyColl as new Ektron.Cms.Controls.Collection
```

With C#, use this syntax.

```
Ektron.Cms.Controls.Collection MyColl = new

Ektron.Cms.Controls.Collection();
```

You can declare any server control as an object by using the server control name in the "List of Server Controls" on page 41. Another example would be: `Dim MyMdl as New MetaDataList.`

Step 3: Set the Properties

NOTE For descriptions of the properties and how to use them, see "Server Control Properties in Code Behind" on page 16.

After inserting the control, you can set the properties that you want to display on the page. For example:

```
Dim MyColl as New Collection

MyColl.DefaultCollectionID = 1
```

or

```
dim MyColl as new Ektron.Cms.Controls.Collection

MyColl.DefaultCollectionID = 1
```

With C#, use this syntax.

```
Ektron.Cms.Controls.Collection MyColl = new

Ektron.Cms.Controls.Collection();

MyColl.DefaultCollectionID = 1;
```

This line tells the page to display CollectionID 1 unless otherwise specified.

NOTE If you do not know an object's ID number, you can switch to Design mode, drag and drop the object, then use the CMS Explorer to find the ID number. (See "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10.) If you do this, remember to delete the dropped object when you are done.

You can also obtain the ID number via the Workarea.

```
Dim MyColl as New Collection

MyColl.DefaultCollectionID = 1

  MyColl.Random = True
```

This line sets the Random property to **true**. *See Also:* "Random" on page 142.

Step 4: Call the Fill Method

Because there is no render event when using objects as components not as controls, use the Fill method to fill an object's properties on the page. An example is below.

```
Dim MyColl as New Collection

MyColl.DefaultCollectionID = 1

MyColl.Random = True

  MyColl.Fill()
```

Step 5: Use .text Property to Display the Control

Finally, use the `.text` property to determine what appears on the Web page. For example, to display the first item in a collection, use this syntax.

NOTE Before adding this line you need to drag and drop a label on your Web form.

```
Dim MyColl as New Collection

MyColl.DefaultCollectionID = 1

MyColl.Random = True

MyColl.Fill()

Label1.Text = myColl.EkItems(0).Title
```

To display *all* items in a collection, use this syntax.

```
  Dim myColl As New Collection
  Dim ekitem As New Ektron.Cms.Common.ContentBase
  myColl.DefaultCollectionID = 2
  myColl.Fill()
Label1.Text = "<ul>"

For Each ekitem In myColl.EkItems

    Label1.Text &= "<li><a href=""" & ekitem.QuickLink & """>" & ekitem.Title & "</a>"

Next

Label1.Text &= "</ul>"
```

This example displays the quick link for every content block in the collection, formatted as a bulleted list. You can use similar code to display a List Summary or search results. The following explains the new (red) code above.

- `label1.Text = "<ul>"` displays the opening tag for the bulleted list
- `For Each ekitem In myColl.EkItems` creates a loop for all content blocks in the collection
- `label1.Text &= "<li><a href=" & ekitem.QuickLink & ">" & ekitem.Title & "</a></li>"` for each content block in the collection, displays its quicklink and title
- `Next` loops through all content blocks in the collection
- `label1.Text &= "</ul>"` closes the bulleted list

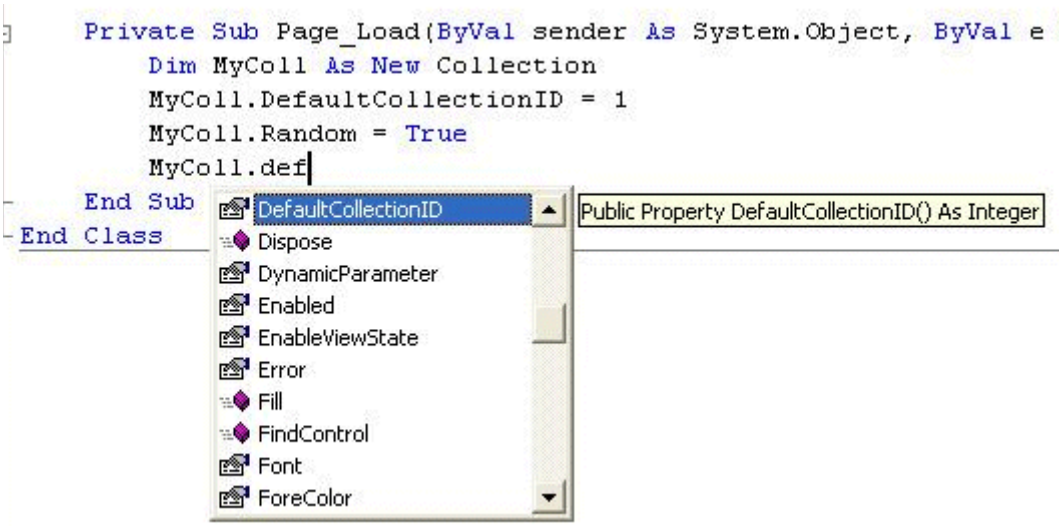
Server Control Properties in Code Behind

Every server control has properties associated with it that you can only access programmatically. This subsection explains, what they are, how to access them, and how to use them.

- ”Displaying Properties for a Control” on page 16
- ”Mutual Server Control Properties in Code Behind” on page 17
- ”Accessing Additional Properties” on page 19
- ”Accessing Items in an Array” on page 25
- ”Referencing the Page Property” on page 25

Displaying Properties for a Control

You can use Visual Studio .NET’s intellisense feature to display a control’s properties. The intellisense box appears as soon as you insert the period (.) after the object, as illustrated below.



The intellisense box displays *all* properties that can be applied. For a description of Ektron's standard server control properties, see "List of Server Controls" on page 41. To learn about native VS .NET properties, see its documentation.

The property's tooltip text indicates its type. In the above example, you can see that the DefaultCollectionID's type is integer.

For more information about accessing **Ektron CMS400.NET** object properties, see "Customizing the Server Control in the Code Behind" on page 28.

Mutual Server Control Properties in Code Behind

There are several read-only properties that can be called in the code behind and used by every server control. The list below explains these properties.

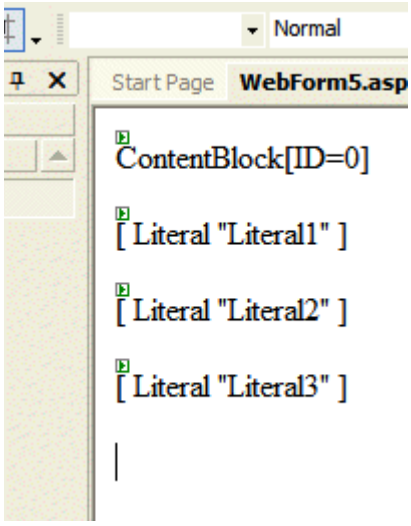
NOTE The properties do not display values within Visual Studio.NET during design time. Instead, they only display values at run time, which are dependent on the user's login status.

Property	Value	Data Type
IsLoggedIn	Tells if a user is logged in to **Ektron CMS400.NET**. **True** = User is logged in **False** = User is not logged in	Boolean
LoggedInUserName	Gets the **Ektron CMS400.NET** user name to display.	String
LoggedInUserID	Gets the **Ektron CMS400.NET** ID of the user to display.	Integer

These properties will allow you to personalize any page with your users names and IDs, and show if they are logged in. Here is an example of using these properties in code behind.

NOTE You must be logged in to **Ektron CMS400.NET** for this example to show your name and ID.

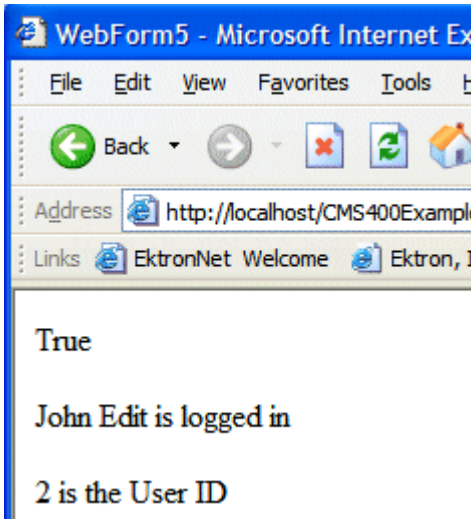
1. Drag an **Ektron CMS400.NET** server control onto a Web form.
2. Drag three Literals onto the Web form.



3. Open the code behind for the Web form.
4. Add the following code to the Page_Load event.

```
Literal1.Text = ContentBlock1.IsLoggedIn

  If ContentBlock1.IsLoggedIn Then

   Literal2.Text = ContentBlock1.loggedInUserName & " is logged in "

   Literal3.Text = ContentBlock1.loggedInUserID & " is the User ID "

  End If
```

5. Build and run the solution.
6. Browse to the login page and log in.
7. Browse to the new Web form you added.
8. The login information is displayed.



Accessing Additional Properties

In addition to the standard properties, **Ektron CMS400.NET** provides access to additional properties for the following objects.

- ListSummary
- Collection
- Search
- ContentBlock
- FormBlock

To access additional properties, use the same syntax you use for standard properties but add `.ekitem` or `.ekitems` after the object. Here is an example.

```
dim MyCB as New ContentBlock

label1.text = MyCB.EkItem.dateCreated
```

or

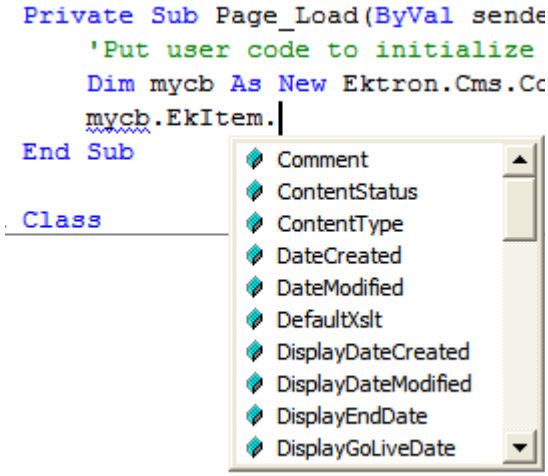
```
dim MyCB as new Ektron.Cms.Controls.ContentBlock

label1.text = MyCB.EkItem.dateCreated
```

IMPORTANT! To access additional properties for the Collection, ListSummary, and Search objects, use `ekitems`, not `ekitem`. For example: `MyColl.ekitems(0).dateCreated`. where (0) is the index of the array. For more Information on using ekitems, see "Accessing Items in an Array" on page 25.

With C#, use this syntax.

```
Ektron.Cms.Controls.ContentBlock MyCB = new

Ektron.Cms.Controls.ContentBlock();

label1.text = MyCB.EkItem.dateCreated
```

You can use intellisense to select from a list of additional object properties, as shown below.I



The additional properties are listed below.

NOTE The following properties are read-only. For example, you can get a content block's ID and pass it through to another part of the code, however you cannot set a content blocks ID to be shown. `mycb.Ekitem.id = 8` will not set a content block's ID. The correct way to set a content blocks ID is `DefaultContentID = 8`.

Property	Description	For more information, see
Comment	The content block's comment	
ContentStatus	The status of the content block. • approved • checked out • checked in • expired • pending deletion • pending expiration • pending start date • submitted	**Ektron CMS400.NET** User Manual, "Appendix A: Content Statuses."
ContentType	One of the following • all types • archived content • archived forms • content • forms	To learn about archived content, see the **Ektron CMS400.NET** User Manual, "Scheduling Content to Begin and End" > "Setting Archive Options."
DateCreated	The date when the content block was created, formatted as a .NET date type	
DateModified	The date when the content block was modified, formatted as a .NET date type	
DefaultXslt	The default Xslt	
DisplayDateCreated	The date when the content block was created. It is formatted as a string that represents Ektron CMS400's display of the date.	

Property	Description	For more information, see
DisplayDateModified	The date when the content block was edited. It is formatted as a string that represents Ektron CMS400's display of the date.	
DisplayEndDate	The content block's end date. It is formatted as a string that represents Ektron CMS400's display of the date.	**Ektron CMS400.NET** User Manual, "Scheduling Content to Begin and End" > "Setting an End Date on a Content Block"
DisplayGoLiveDate	The content block's start date. It is formatted as a string that represents Ektron CMS400's display of the date.	**Ektron CMS400.NET** User Manual, "Scheduling Content to Begin and End" > "Setting a Start Date"
DisplayStartDate	The content block's start date. It is formatted as a string that represents Ektron CMS400's display of the date.	**Ektron CMS400.NET** User Manual, "Scheduling Content to Begin and End" > "Setting a Start Date"
EndDate	The content block's end date, formatted as a .NET date type	**Ektron CMS400.NET** User Manual, "Scheduling Content to Begin and End" > "Setting an End Date on a Content Block"
EndDateAction	• archive display • archive expire • refresh report	**Ektron CMS400.NET** User Manual, "Scheduling Content to Begin and End" > "Setting Archive Options."
FolderID	The ID of the folder that contains each content block	
GoLiveDate	The content block's start date formatted as a .NET date type	**Ektron CMS400.NET** User Manual, "Scheduling Content to Begin and End" > "Setting a Start Date"
Html	The content that makes up the content block. If content block is in XML it will return it as raw XML content	
Hyperlink	Content block title wrapped by `<a href>` tags	

Property	Description	For more information, see
Id	The content block 's ID number	
InheritedFrom	If folder permissions are inherited, the folder from which they are inherited	**Ektron CMS400.NET** Administrator manual "Setting Permissions" > "Inheritance"
IsInherited	Whether a content block's permissions are inherited	**Ektron CMS400.NET** Administrator manual "Setting Permissions" > "Inheritance"
IsPrivate	Whether or not a content block is private	**Ektron CMS400.NET** Administrator manual "Managing Users & User Groups" > "Private Content & Memberships"
Language	The content block's language	**Ektron CMS400.NET** Administrator manual "Multi-Language Support" > "Working with Multi-Language Content"
LastEditorFname	The first name of the last person to edit the content block	
LastEditorLname	The last name of the last person to edit the content block	
PackageDisplayXSLT	If the content block is XML, the name of its XSLT	
QuickLink	The content block's quicklink	**Ektron CMS400.NET** User manual "Library Folder" > "Quicklinks"
StartDate	The content block's start date formatted as a .NET date type	**Ektron CMS400.NET** User Manual, "Scheduling Content to Begin and End" > "Setting a Start Date"

Property	Description	For more information, see
Status	The status of the content block • approved • checked out • checked in • expired • pending deletion • pending expiration • pending start date • submitted	**Ektron CMS400.NET** User manual "Appendix A: Content Statuses"
Teaser	The content block summary	"Adding a Content Block Summary" in the **Ektron CMS400.NET** User manual
TemplateLink	Currently empty and not being used with the ContentBlock server control	
Title	The content block title	
UserID	Last user who edited the content	
Xslt1	The content block's first Xslt, as defined in **Ektron CMS400.NET**	
Xslt2	The content block's second Xslt, as defined in **Ektron CMS400.NET**	
Xslt3	The content block's third Xslt, as defined in **Ektron CMS400.NET**	
Xslt4	The developer can use this property programmatically. **Ektron CMS400.NET** only uses Xslt1, 2 and 3 in the workarea.	
Xslt5	The developer can use this property programmatically. **Ektron CMS400.NET** only uses Xslt1, 2 and 3 in the workarea.	

Accessing Items in an Array

To access and manipulate content blocks returned by an object, use the common class `Ektron.Cms.Common.ContentBase`. *EkItems* is an array of Ektron.Cms.Common.ContentBase. *EkItem* is a single Ektron.Cms.Common.ContentBase.

Search, Collection and ListSummary have EkItems (an array of ContentBase), while and ContentBlock have a single EkItem. Here is an example of how to use this feature.

```
dim MyC as new Ektron.Cms.Controls.Collection

MyC.DefaultCollectionID = 1

MyC.Fill()

dim item as Ektron.Cms.Common.ContentBase

MyC.Text = "<ul>"

for each item in MyC.EkItems

    MyC.Text &= "<li>" & item.Title & "</lI>"

next

MyC.Text &= "</ul>"

Response.Write(MyC.Text())
```

This example formats every item in the collection in a bulleted list.

Referencing the Page Property

If a control requires a reference to its parent page (for example, utilizing the DynamicParameter property on a content block to check for a query string), you must provide access to the page object if you declared your control in the code behind. To do this, set the control's page property to the Web page you're working on. For an example, see the code in red and italics below.

```
dim search as new Ektron.Cms.Controls.Search()

MySearch.Page = me

MySearch.Fill()
```

With C#, use this syntax.

```
Ektron.Cms.Controls.Search MySearch = new
Ektron.Cms.Controls.Search();

MySearch.Page = this;

MySearch.Fill();
```

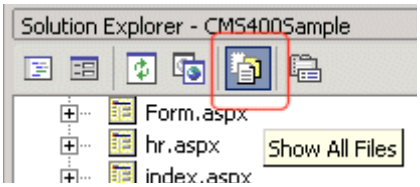
This relationship is only required when inserting a control in the code behind. When dragging and dropping, even if you make changes in code behind, the relationship is automatically generated.

> Best Practice
>
> Ektron recommends including the page property reference when using the server controls as components in the code behind.

Customizing Server Controls

One of Visual Studio .NET's strengths is its separation of coding and logic from presentation. Web page formatting is handled by a page's HTML, while the logic is handled by the code behind, which is stored in the corresponding .vb file. For example, if the ASP.NET page is mypage.aspx, the code behind file is mypage.aspx.vb.

NOTE If you do not see the code behind files, click the Show All Files button on the VS .NET Solution Explorer toolbar.



Within the vb file, you can use Visual Basic to insert code to manipulate the events that occur on the page.

This subsection contains the following topics:

- "Recognizing the Server Control Within the HTML" on page 27
- "Recognizing the Server Control Within the Code Behind" on page 27
- "Customizing the Server Control in the Code Behind" on page 28

- "Troubleshooting Error Creating Control Message" on page 28

Recognizing the Server Control Within the HTML

Within a Web page's HTML, a `<cms>` tag wraps the VS. NET object, as shown below.

```
<cms:Search id="Search1" runat="server" ButtonText="Search"
Display="Vertical"></cms:Search>
```

Above is an example of a Search Server Control. Below is an example of a ContentBlock Server Control.

```
<cms:ContentBlock id="ctrlMainContentBlock" runat="server"
DefaultContentID="1" DynamicParameter="id" OverrideXslt="Default"></
cms:ContentBlock>
```

Recognizing the Server Control Within the Code Behind

Within the VS .NET code behind file, the Ektron server controls appear (along with the VS .NET controls) in the Web Form Designer Generated Code section. When you click the plus sign (+) to display this section, you see something like the following. The content block listed in HTML above is circled below to help you see their relationship.

```
#Region " Web Form Designer Generated Code "

    'This call is required by the Web Form Designer.
    <System.Diagnostics.DebuggerStepThrough()> Private Sub InitializeComponent()...
    Protected WithEvents MetaDataArea As System.Web.UI.WebControls.Literal
    Protected WithEvents DhtmlJavaScript As System.Web.UI.WebControls.Literal
    Protected WithEvents DropDownMenu As System.Web.UI.WebControls.Literal
    Protected WithEvents ctrlMainContentBlock As Ektron.Cms.Controls.ContentBlock
    Protected WithEvents ctrlTopContentBlock As Ektron.Cms.Controls.ContentBlock
    Protected WithEvents ctrlBottomLeftContentBlock As Ektron.Cms.Controls.ContentBl
    Protected WithEvents ctrlBottomRightContentBlock As Ektron.Cms.Controls.ContentB
    Protected WithEvents ctrlLanguageSelect As Ektron.Cms.Controls.LanguageSelect
    'NOTE: The following placeholder declaration is required by the Web Form Designe
    Private designerPlaceholderDeclaration As System.Object
```

The next section of the code behind page loads the page into the browser.

```
Private Sub Page_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
```

We want our events to occur while the page is loading, so we'll add custom code following this line.

Customizing the Server Control in the Code Behind

To customize an Ektron server control in the code behind, insert code similar to the following *after* the Page_load command.

```
Dim MyObj As New Ektron.Cms.Controls.ContentBlock
```

This code declares a variable named MyObj and assigns to it the value of a content block. The content block is part of the Ektron.CMS.Controls content base, so it has access to the **Ektron CMS400.NET** database.

After defining MyObj as a content block, you can access its properties. For example, to assign a defaultID of 24, insert the following.

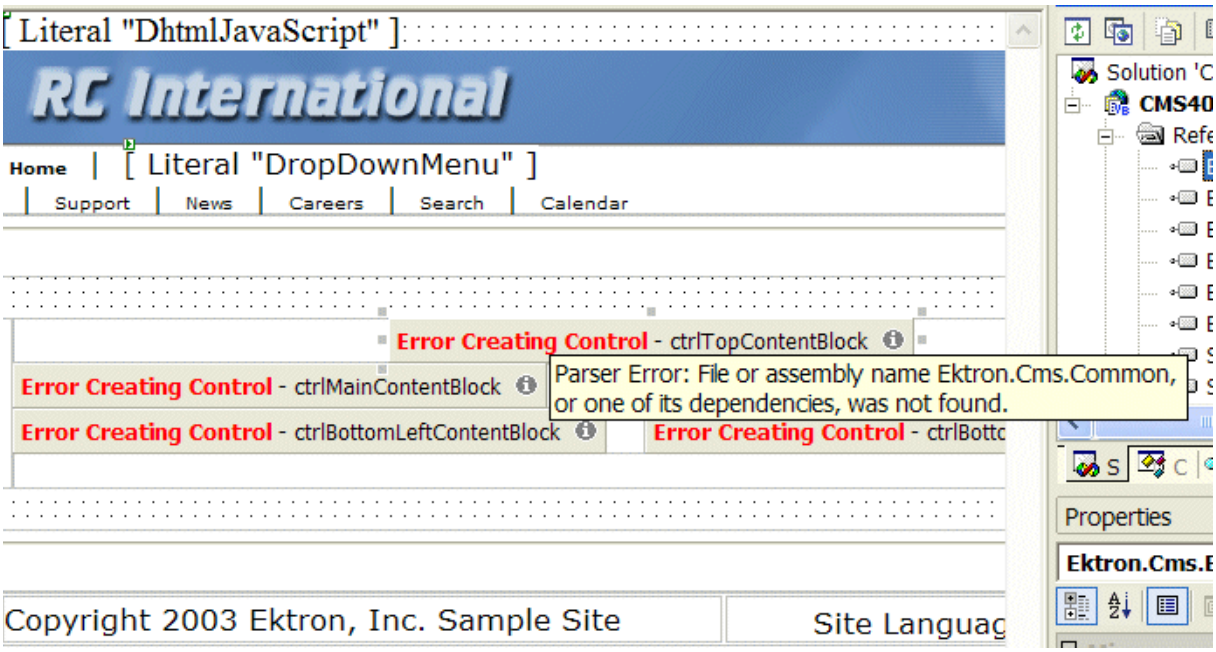
```
Dim MyObj As New Ektron.Cms.Controls.ContentBlock

MyObj.DefaultContentID = 24
```

Now, the content block can be specified dynamically in the URL of the hyperlink that calls it. If not, content block 24 displays.

This is just an example of programmatically applying property values to content blocks. For a complete list of properties available to **Ektron CMS400.NET** objects, see "List of Server Controls" on page 41.

Troubleshooting Error Creating Control Message

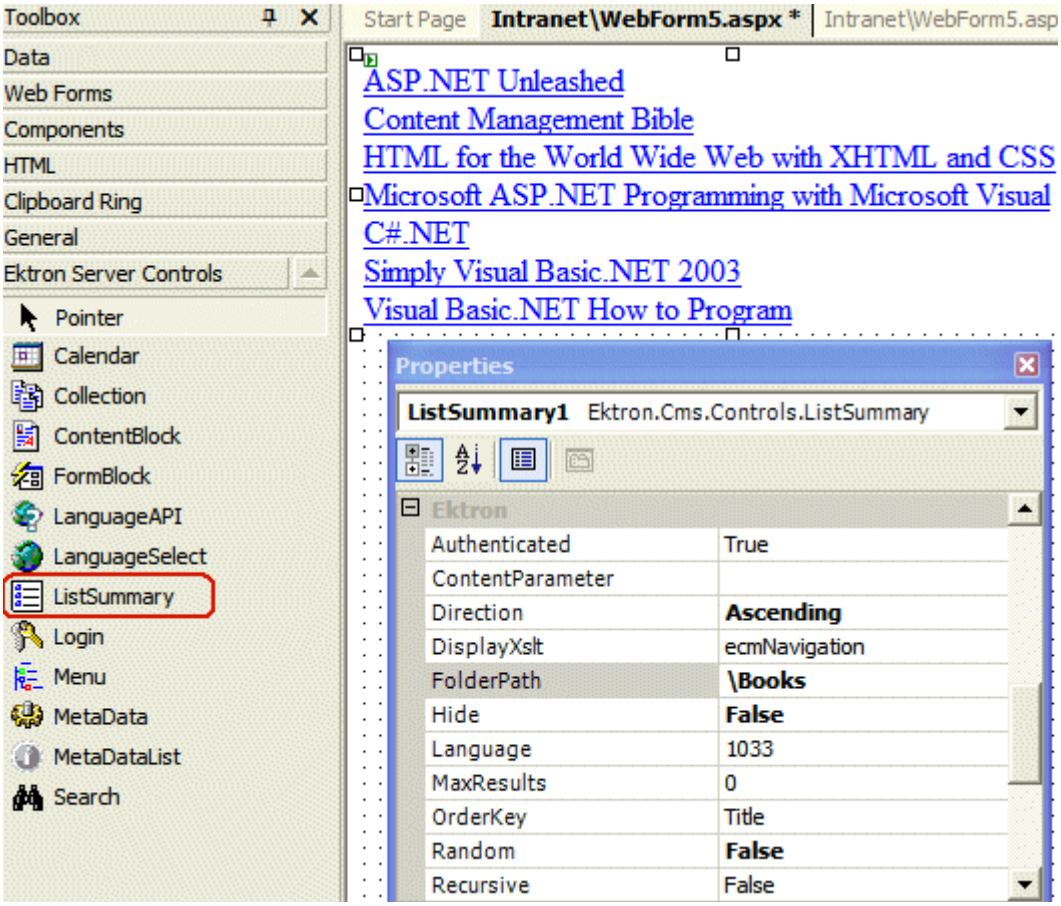
If you get an Error Creating Control message while trying to use a server control, you can view the text of the error message by hovering the mouse over the control. See illustration below.



Using Drag & Drop and Programmatically Together

All server controls can be further manipulated after dragging and dropping them on a web form by using the code behind. Here is an example of using a drag and drop ListSummary Server Control then modifying it programmatically.

First, drag and drop a List Summary on your Web form and set your properties.



Then add the following code to the code behind.

```
Dim myString As String
Dim i
For i = LBound(ListSummary1.EkItems) To UBound(ListSummary1.EkItems)
myString &= "<a href=""" & ListSummary1.EkItems(i).QuickLink & """>" &
ListSummary1.EkItems(i).DateCreated & "</a><br>"
            ListSummary1.Text = myString
Next
```

An Explanation of the code:

ListSummary1 is the ID of the object. It is used to get access to its properties.

Create a string that contains the output (myString).

```
Dim myString As String
```

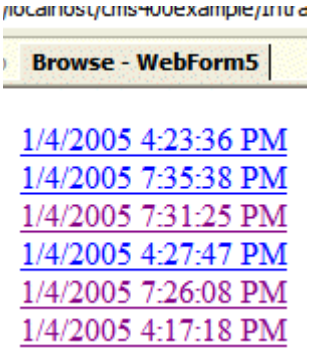
Set the object's text property to that string.

```
myString &= "<a href=""" & ListSummary1.EkItems(i).QuickLink & """>"
& ListSummary1.EkItems(i).DateCreated & "</a><br>"
```

Wrap in a loop so it loops through each of the items.

```
Dim i

For i = LBound(ListSummary1.EkItems) To UBound(ListSummary1.EkItems)

Next
```

This example outputs the date created for each content block in a List Summary.



Data Binding with Server Controls

With data binding, you can bind Ektron server controls to a DataGrid Control, DataList Control, or Repeater Control. This gives more flexibility when you use data from the Ektron server controls. Some benefits of data binding are ease of data manipulation and the ability to format data.

Title	ID	Teaser
Plastic Molder #123	13	RC International is looking for an experienced plastics molder.
Servo Control Engineer #124	14	RC International is looked for a highly skilled servo control eng

The following are data bindable Ektron Server Controls.

- Collection
- Content Block
- Form Block
- List Summary
- Search

The following is an example of code behind that uses a DataGrid to display a Collection:

```
Dim myCol As New Ektron.Cms.Controls.Collection

myCol.DefaultCollectionID = 1

myCol.Fill()

DataGrid1.DataSource = myCol

DataGrid1.DataBind()
```

NOTE When using drag and drop server controls for the Collection and DataGrid, you need to call the DataBind() method in the code behind. An example line of code would be: `DataGrid1.DataBind()` .

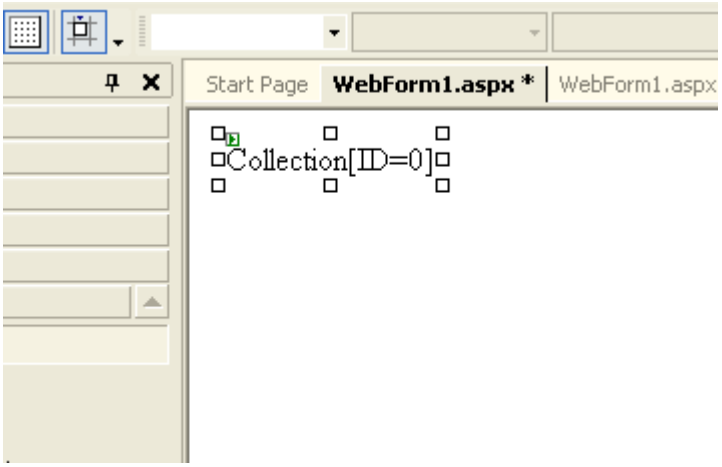
This subsection contains the following topics:

- ”Steps to Data Binding Using Drag and Drop Server Controls” on page 32
- ”Customizing the DataGrid Table View” on page 35

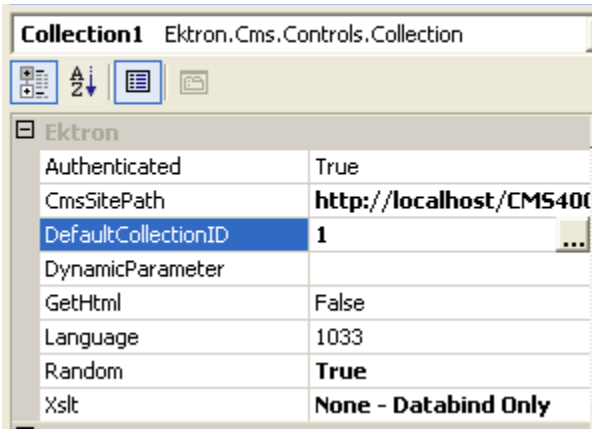
Steps to Data Binding Using Drag and Drop Server Controls

The following example uses a Collection.

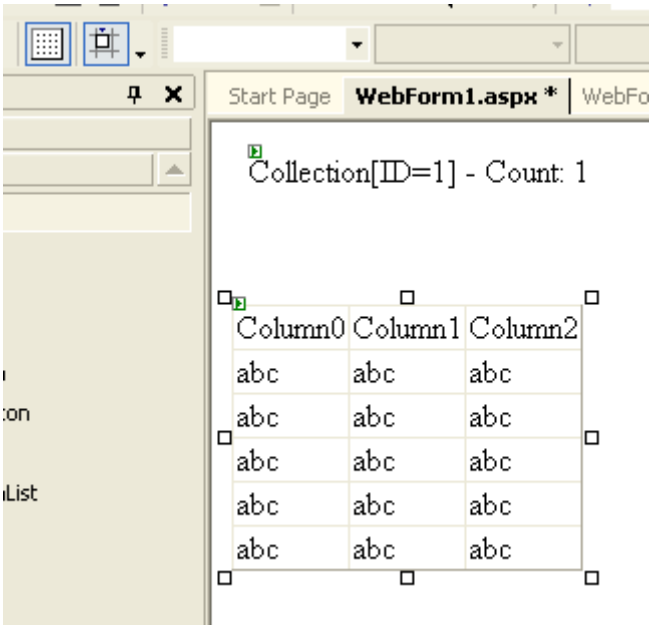
1. Drag and drop a data bindable Server Control on the Web form.



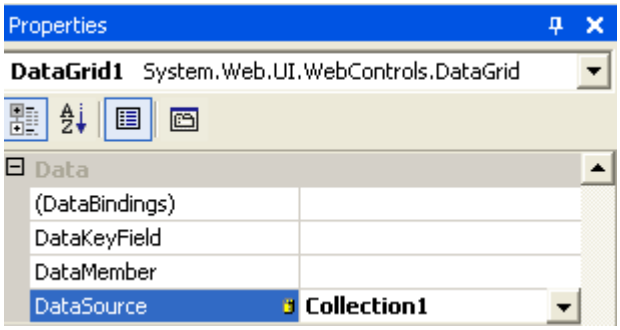
2. In properties, choose the Ektron default ID for the Server Control and make sure the XSLT is set to **None- DataBind Only**.



3. Drag and drop a DataGrid on the web form.



4. In the DataGrid properties, choose the DataSource.



5. The DataGrid is populated with column headers and the type of information that will be in the table.
6. Go to code behind and add `DataGrid1.DataBind()` to the Page_Load event.

```
Private Sub Page_Load(ByVal sender As System.Object
    'Put user code to initialize the page here
    DataGrid1.DataBind()
End Sub
```

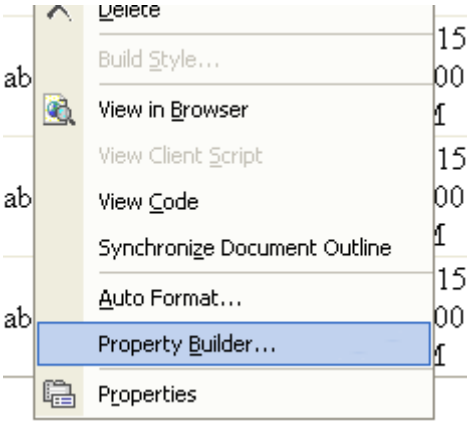
7. Switch to Design mode and right-click on the DataGrid. Then, choose **View in Browser**. The table appears.

LastEditorFname	DateModified	Html	QuickLink	HasErrors	Teaser	ID
Application	2/17/2004 4:12:11 PM		/CMS400Sample/products.aspx?id=5	False	RC Internationals most successful entry in to the competitive RC racing world is the RC Cheetah.	5

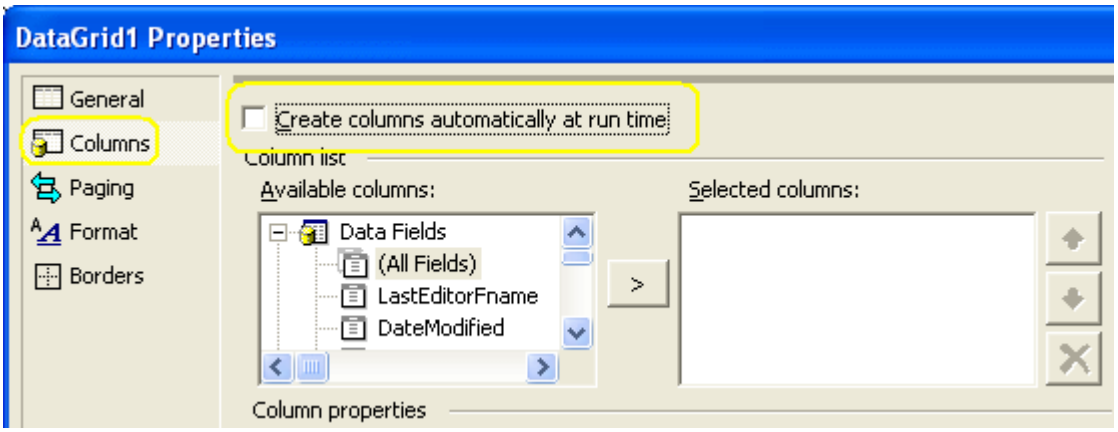
Customizing the DataGrid Table View

Customizing the DataGrid is a great way to display and format specific information from a data source. You can select formatting for columns, paging and borders, plus other general formatting for the table. In the following example, we choose table columns.

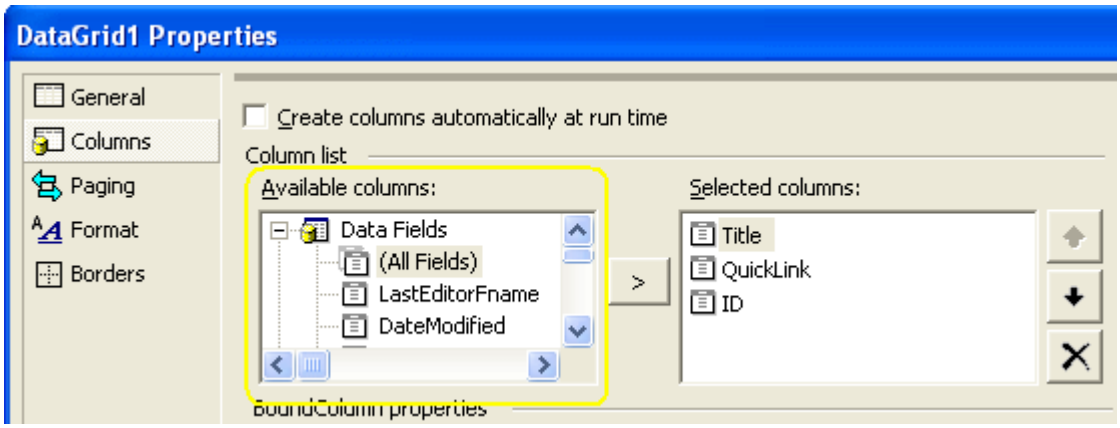
1. In Web form design, right click the DataGrid and choose **Property Builder**.



2. Click **Columns** and uncheck the **Create columns automatically at run time** checkbox.



3. Choose the fields for the custom table from the available columns.



4. Click **OK**.
5. Right click the DataGrid and choose **View in Browser**.
6. The custom table appears.

Title	QuickLink	ID
RC Lilly	/CMS400Sample/products.aspx?id=6	6

For more information on working with DataBind() or DataGrid, review the help topics in Visual Studio.

Caching with Server Controls

One of the most important factors in building high-performance, scalable Web applications is the ability to store items in memory the first time they are requested. These items include data objects, pages, or parts of a page.

Known as caching, this process allows you to use a number of techniques to store page output or application data across HTTP requests and reuse it. You can store items on the Web server or other software in the request stream, such as the proxy server or browser.

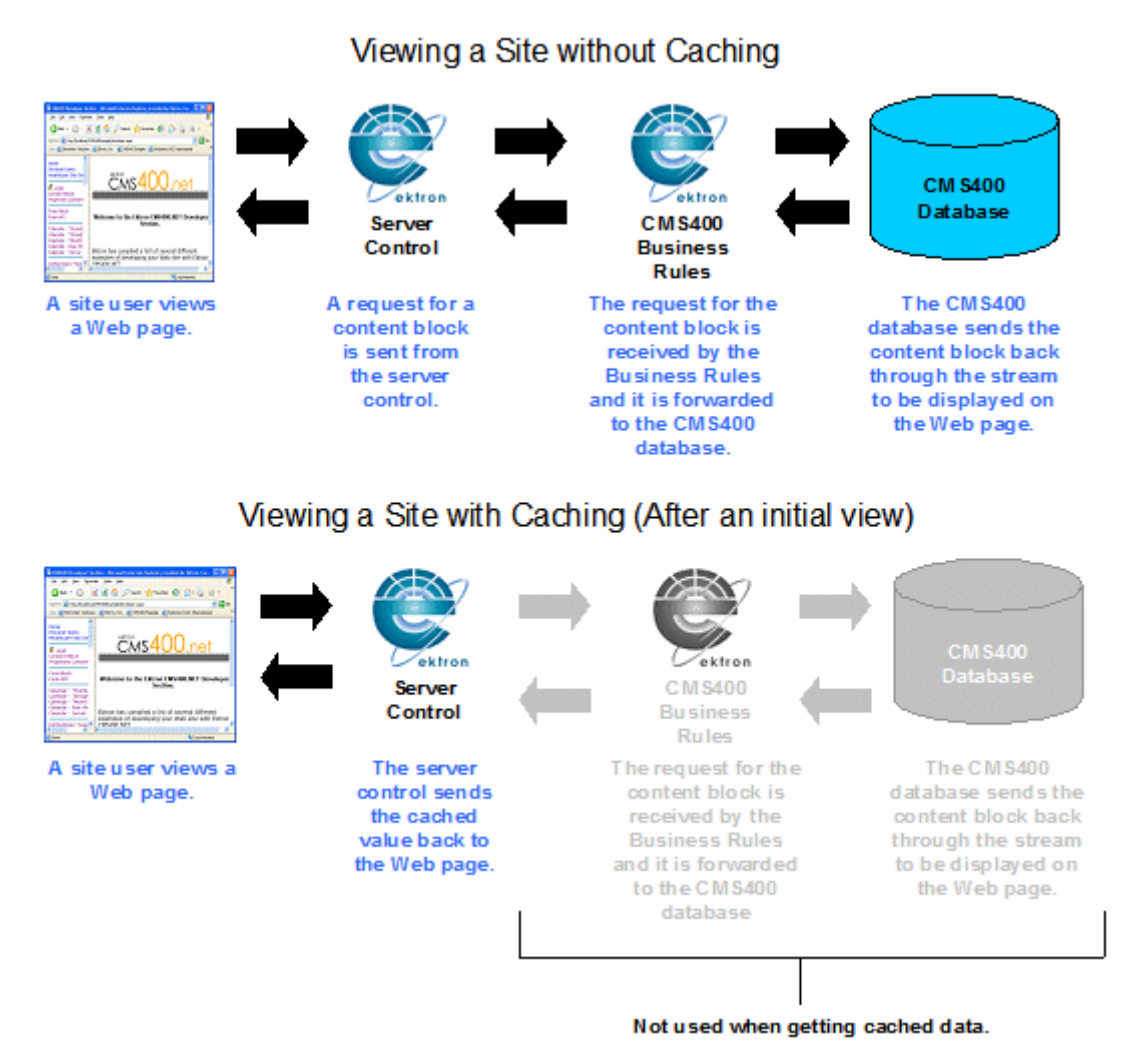
With caching, the server does not have to recreate information, particularly information that demands significant processor time or other resources.This saves valuable time and resources.

There are two types of caching available:

- Caching Individual Content
- Page Level Caching

Caching individual content allows you to cache a part of a Web page and is only available on some server controls. With Page Level Caching, you can use any server control and the entire page is cached.

The following is a visual example of how data flows in non-cached and cached enviroments.



IMPORTANT! Controls serve data from the cache only if the user is not logged in.

Caching Individual Content

Caching individual content is available on the following server controls:

- ContentBlock
- FormBlock
- Collections
- ListSummary
- MetadataList
- Menu
- DhtmlMenu

Before caching individual content, you must make sure the `ek_CacheControls value = "1"` (one). This is the default. This setting is found in the Web.config file and allows you to control whether each server control can utilize caching.

Once caching is turned on, use the `CacheInterval` property in the server control to set the amount of time the server control's data is cached. The default is 0 (zero). This is the amount of time, in seconds, a control's data is cached. For example, if you want to cache the data for five minutes, you set the `CacheInterval` property to 300 (three hundred).

WARNING! If you change `ek_CacheControls` value to 0 (zero), you will not be able cache individual content. This also causes the `CacheInterval` property of each server control to be set to 0 (zero) the next time that server control is called on your site. This is done so you can globally reset caching of Web site pages to 0 (zero). For example, if your site has five hundred Web pages and half of them use caching, you can set the `ek_CacheControls` to 0 (zero) and then none of them will have caching.

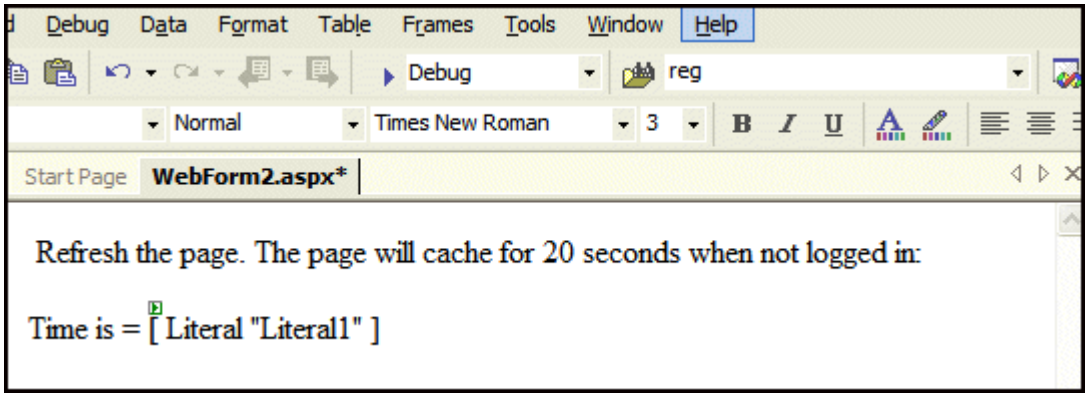
To set up caching of individual content for any Web page, follow these steps.

1. Make sure the `ek_CacheControls value = "1"` in the Web.config file.
2. Add a server control that supports caching of individual content to a Web form.
3. Set the `CacheInterval` property in the server control to the length of time for the data to be cached.

Page Level Caching

Below is an example of a page level cache for non logged in users. In this example, you use a Server Control to define whether a user is logged in or not. Then, further define if the time is cached, based on the user's status.

1. Create a new Web form in your Ektron CMS400.NET project.
2. Add the following text to the Web form:
   ```
   Refresh the page. The page will cache for 20 seconds when not
   logged in:
   Time is =
   ```
3. Next to **Time is =**, add a literal.



4. Below that, add a ContentBlock Server Control.
5. Set the DefaultContentID. For this example, `DefaultContentID = 20`.
6. Add the following code to the code behind.

```
If Not ContentBlock1.IsLoggedIn Then

            Response.Cache.SetExpires(DateTime.Now.AddSeconds(20))

            Response.Cache.SetCacheability(HttpCacheability.Public)

            Response.Cache.SetValidUntilExpires(True)
```

```
        Response.Cache.VaryByParams("id") = True

        Response.Cache.SetVaryByCustom("cmsCache")

    End If

    Literal1.Text = Now()
```

7. Build your Web form.
8. Browse to your Web form with your browser.
9. Press the refresh button on your browser.

If you are not logged in to CMS400.NET, the time remains the same for twenty seconds. After twenty seconds, when you refresh, you get the new time.

NOTE The same code can be employed in a user control to cache the output in a particular region of the page.

List of Server Controls

To customize your Ektron CMS400.NET application, you can use these server controls.

Server Control Names	Description	More Information
BreadCrumb	Creates a breadcrumb trail of where site visitors have been and allows them to navigate back to previous pages.	”BreadCrumb Server Control” on page 57
Calendar	Displays an event calendar.	”Calendar Server Control” on page 68
Collection	Displays links generated by collections.	”Collection Server Control” on page 103
ContentBlock	Displays and manages a content block.	”ContentBlock Server Control” on page 113
DhtmlMenu	Displays a drop down menu with submenus	”DhtmlMenu Server Control” on page 170

Server Control Names	Description	More Information
FormBlock	Displays a content block that is associated with a form.	”FormBlock Server Control” on page 129
IndexSearch	Perform XML index searches on XML content.	”IndexSearch Server Control” on page 234 For information on using XHTML Search, see ”Search Server Control” on page 200
LanguageAPI	Allows developers to force a language for a Web site.	”LanguageAPI Server Control” on page 134
LanguageSelect	Lets user select language of site.	”LanguageSelect Server Control” on page 137
ListSummary	Displays a list of content block summaries.	”ListSummary Server Control” on page 139
Login	Displays the login or logout button, which lets the user log in and use Ektron CMS400.NET.	”Login Server Control” on page 188
Membership	Creates a form for site visitors to add themselves to your Membership users list.	”Membership Server Control” on page 146
Menu	Calls a defined menu to be used with an Xslt to display the menu.	”Menu Server Control” on page 153
MetaData	Adds metadata contained in content blocks to a Web page.	”MetaData Server Control” on page 178
MetaDataList	Creates a list of content blocks based on the metadata contained in the content blocks.	”MetaDataList Server Control” on page 183
RssAggregator	Aggregates an RSS feed from a RSS Web site.	”RssAggregator Server Control” on page 196
Search	Inserts a text box with a search button in the template. The site visitor enters criteria and launches a search of the XHTML content on your Web site.	”Search Server Control” on page 200 For information on using XML Indexing search, see ”IndexSearch Server Control” on page 234

Blog Server Controls

IMPORTANT! In addition to reading this section on the Blog server controls, reading the User Manual section “Blogs” is recommend.

Ektron provides five server controls that relate to displaying a Blog on a Web site. One of these, the Blog Server Control, allows you to display all of the pieces that typically make up a blog site on a form. These items are blog entries, blog posts, a blog calendar and a blog roll. The other four Blog server controls allow you to display each of these items individually. The four individual Blog server controls offer additional ways to customize the page layout and appearance of the items.

The five Blog server controls provided by Ektron are

- ”Blog Server Control” on page 45
- ”BlogEntries Server Control” on page 46
- ”BlogPost Server Control” on page 49
- ”BlogCalendar Server Control” on page 51
- ”BlogRoll Server Control” on page 55

Using the Blog Server Control vs. Using the Individual Blog Server Controls

When you use the Blog Server Control to display a blog, you only have to deal with one server control and its settings. This makes adding a blog to your site quick and easy.

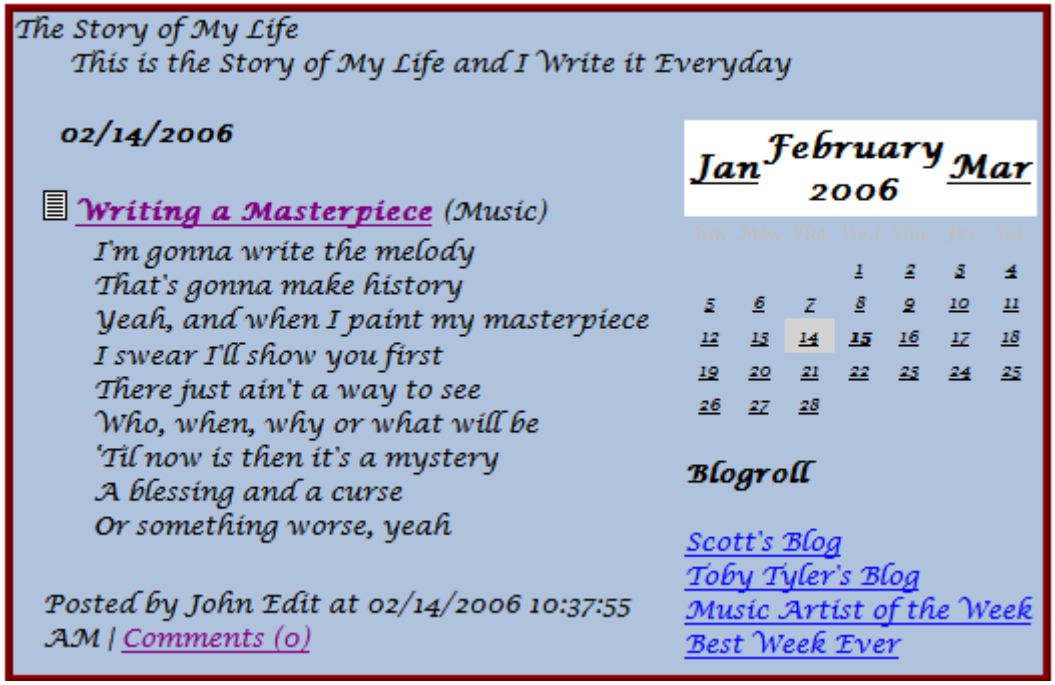
Using the Individual Blog server controls to display a Blog, allows you the flexibility to display the parts of a Blog you want to display. For example, you might to display the Blog entries and a Blog roll, but not display a Blog calendar. In addition, using the individual Blog server controls allows you to further define the display details of each control.

Blog Server Control

The Blog Server Control allows you to quickly add a blog to a Web form. It has all of the items commonly found on a blog page in one server control. These items include blog posts, a comments link, a blog roll and a blog calendar. If you want to learn more about each item, see the User Manual section on “Blogs”

Using the Blog Server Control allows you to easily maintain the overall look of the blog. When you change display properties, such as the background color or the font, they change for the whole blog not just a piece of it.

While you can change display properties, you can not change the location of each item in the server control. The title always appears on the top, with the tagline below it. The blog posts always appear to the left and the calendar and blog roll to the right.



The Blog Server Control is easy to use. Simply add it a Web form and set the `BlogPath` property to the folder location in the CMS400.NET. Next, set the BlogTagline and BlogTitle properties.

If you want to further customize the look of the blog, you can change the colors and fonts by using the appearance properties associated with the server control.

Blog Server Control Properties

The following table explains the properties of the Blog control.

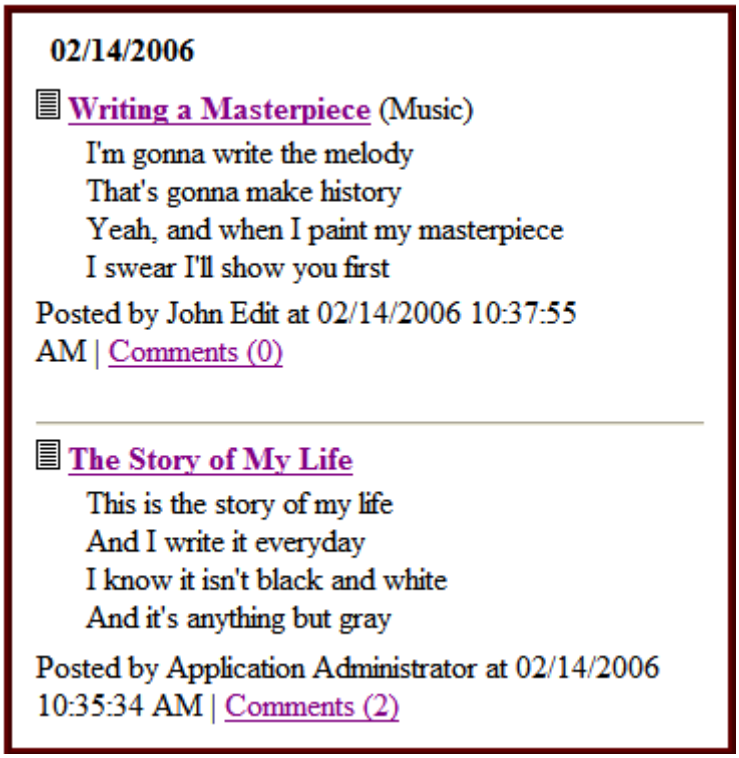
NOTE The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

Property	Value	Data Type
BlogPath	The path to the blog in CMS400.NET. For example: \Blog\Ektron Blog. If you don't know the path to the blog, use the CMS Explorer to browse to it. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	String
BlogTagline	This is a line of additional information that describes the blog to blog readers. For example, the title of a blog might be "John's Blog Page" with a tag line of "A place to learn about John's past, present, future, and his current state of mind!".	String
BlogTitle	The title of the blog.	String

BlogEntries Server Control

The BlogEntries Server Control allows you to display a list of multiple blog posts on a Web form. By using this server control, you can add a blog posts to a Web form without having to add items normally contained in a blog.

Using this server control in combination with the other individual blog server controls, allows you the flexibility to change the appearance of this control without changing other blog controls. In addition, you can layout the controls any way you like, unlike the Blog Server Control. Below is an example of a BlogEntries Serve



You can use the BlogCalendar and BlogEntries Server Controls together to allow blog readers to choose which day's blog posts they see. For information on using these server controls together, see "Using the BlogCalendar and BlogEntries Server Controls Together" on page 52.

BlogEntries Server Control Properties

The following table explains the properties of the LanguageSelect control.

NOTE The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

Property	Value	Data Type
Authenticated	Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	String
BlogPath	The path to the blog in CMS400.NET. For example: \Blog\Ektron Blog. If you don't know the path to the blog, use the CMS Explorer to browse to it. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	String
BlogPostParameter	Works like the `DynamicParameter` for content blocks. When `id` is selected, this server control passes the blog post ID as a URL parameter. If you do not set this parameter to id, you will not be forwarded to the blog post's page when you click on any links in the post. The default setting is **id**. **Blank** - the list of blog posts is static. The links in the blog posts are inactive. **id** - the id of the blog post is passed to the URL as a parameter and **None use default** - the list of blog posts is static. The links in the blog posts are inactive.	String
CacheInterval	Sets the amount of time the server control's data is cached. The default is 0 (zero). This is the amount of time, in seconds, a control's data is cached. For example, if you want to cache the data for five minutes, you set the `CacheInterval` property to 300 (three hundred).	Integer
DateToStart	The date of the last blog entries you want to appear. For example, if you want to display blog entries for January 1, 2006 and before, you enter 1/1/2006. Clicking the dropdown box provides you with a calendar.	DateTime
DisplayXslt	Specify an external XSLT file. *See Also:* "The OverrideXslt and DisplayXslt Properties" on page 119.	String

Property	Value	Data Type
Hide	Used to hide language select dropdown box and label in design time and run time. **True** = Hide language select dropdown box and label **False** = Show language select dropdown box and label	Boolean
Language	Set a language for the Language Select Box. The language property shows results in design-time (in Visual Studio) and at run-time (in a browser).	Integer
MaxResults	The maximum number of entries displayed. 0 (zero) = the default set in the Workarea blog property # Post Visible. The default value is 0 (zero).	Integer
WrapTag	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code. **Custom** - Allows you to use a custom tag.	String

BlogPost Server Control

The BlogPost Server Control allows you to display an individual blog post on a page.

There are two ways this server control displays a blog post. First, if a user is logged in as a CMS400.NET user or a Membership user, the control displays the blog post, comments and comments form on the web page. The second way is if a user is not logged in, the control displays only the blog post.

NOTE the Enable Comments property in the workarea must be enabled for comments and the comments form to appear.

BlogPost Server Control Properties

The following table explains the properties of the LanguageSelect control.

NOTE The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

Property	Value	Data Type
Authenticated	Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	String
CacheInterval	Sets the amount of time the server control's data is cached. The default is 0 (zero). This is the amount of time, in seconds, a control's data is cached. For example, if you want to cache the data for five minutes, you set the `CacheInterval` property to 300 (three hundred).	Integer
DefaultContentID	The ID of a blog post that appears where you inserted this server control if no other content block is identified, or is not available. If you don't know the ID number of the blog post, use the CMS Explorer to browse to it. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	Integer
DisplayXslt	Specify an external XSLT file. *See Also:* "The OverrideXslt and DisplayXslt Properties" on page 119.	String
DynamicParameter	To make this blog post dynamic, select **id**. When you do, this server control uses the blog post passed as a URL parameter.	String
Hide	Used to hide language select dropdown box and label in design time and run time. **True** = Hide language select dropdown box and label **False** = Show language select dropdown box and label	Boolean
Language	Set a language for the Language Select Box. The language property shows results in design-time (in Visual Studio) and at run-time (in a browser).	Integer

Property	Value	Data Type
OverrideXslt	An internal Xslt file used to display XML content if one is not identified in the Xslt property. *See Also:* "The OverrideXslt and DisplayXslt Properties" on page 119	must be one of these values: default, one, two, three
ShowType	This property is for future use.	
WrapTag	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code. **Custom** - Allows you to use a custom tag.	String

BlogCalendar Server Control

The BlogCalendar Server Control allows a user to display a calendar on a Web page and associate it with a blog. When a Blog calendar is associated with a blog, the days that have blog posts are highlighted on the calendar.

You can use a BlogCalendar Server Control with a BlogEntries Server Control. This gives site visitors a way to display blog posts for a given day. For an example of using the BlogCalendar and BlogEntries Server Controls together, see "Using the BlogCalendar and BlogEntries Server Controls Together" on page 52

BlogCalendar Server Control Properties

The following table explains the properties of the LanguageSelect control.

NOTE The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

Property	Value	Data Type
BlogPath	The path to the blog in CMS400.NET. For example: \Blog\Ektron Blog. If you don't know the path to the blog, use the CMS Explorer to browse to it. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	String
Language	Set a language for the Language Select Box. The language property shows results in design-time (in Visual Studio) and at run-time (in a browser).	Integer

Using the BlogCalendar and BlogEntries Server Controls Together

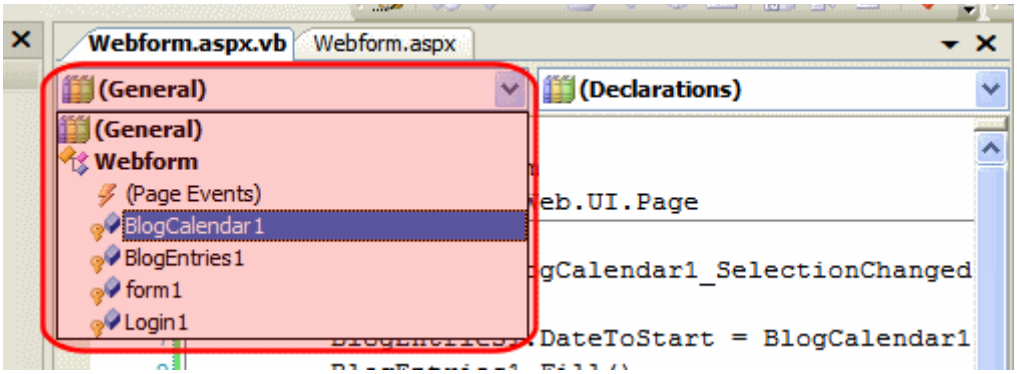
You can use the BlogCalendar and BlogEntries Server Controls together to allow blog readers to choose which day's blog posts they see.

Steps to Use BlogCalendar and BlogEntries Together.

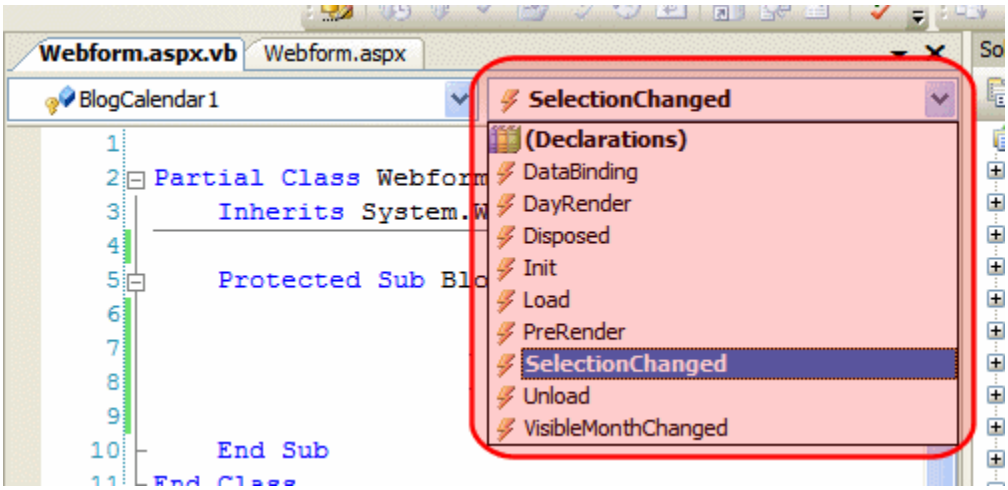
1. On a Visual Studio Web form, add a BlogCalendar Server Control.
2. In the Ektron properties, set the BlogPath to the blog you want to use.

3. If you want, change the appearance properties for the calendar. (This step is optional.)
4. Add a BlogEntries Server Control.
5. If you want, change the appearance properties for the BlogEntries Server Control. (This step is optional.)

6. On the code behind page, set the **Class Name** list to **BlogCalendar1**.



7. Set the **Method Name** list to **SelectionChanged**.



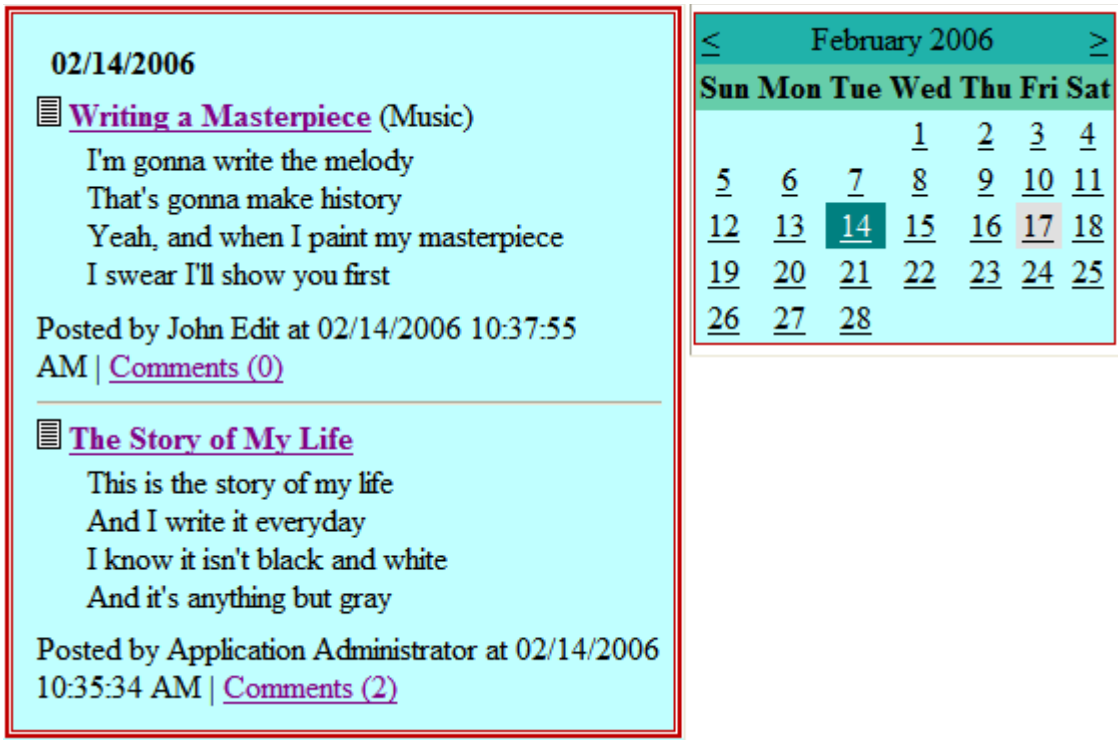
8. In the Protected Sub **BlogCalendar1_SelectionChanged**, Add the following line of code.

```
BlogEntries1.DateToStart = BlogCalendar1.SelectedDate
```

NOTE This line of code equates the DateToStart property for the BlogEntry control to the SelectedDate property of the BlogCalendar control.

```
Webform.aspx.vb   Webform.aspx
BlogCalendar1                    SelectionChanged

1
2  Partial Class Webform
3      Inherits System.Web.UI.Page
4
5      Protected Sub BlogCalendar1_SelectionChanged(ByVal sender As Object,
6      ByVal e As System.EventArgs) Handles BlogCalendar1.SelectionChanged
7
8        BlogEntries1.DateToStart = BlogCalendar1.SelectedDate
9
10     End Sub
11 End Class
```

9. Click the **Build Menu > Build Page**.
10. When you view page in a browser and click on a date in the calendar that has blog posts, you see the posts for that day.



BlogRoll Server Control

The BlogRoll Server Control displays the blog roll for a blog on a Web form. Blog rolls are created in the Workarea for each blog. This control reads that list and displays a clickable list of links. For more information on the blog roll, see the User Manual Section “Blogs” > “Blog Properties” > “The Blog Roll”.

Typically this control is displayed along side other Individual Blog server controls. Using the individual server controls, such as the Blog Roll Server Control, allows you to have more flexibility for layout and appearance than if you use Blog Server Control.

BlogRoll Server Control Properties

The following table explains the properties of the LanguageSelect control.

NOTE The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

Property	Value	Data Type
Authenticated	Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* ”Using CMS Explorer to Browse Your Ektron CMS400.NET Site” on page 10	String
BlogPath	The path to the blog in CMS400.NET. For example: \Blog\Ektron Blog. If you don’t know the path to the blog, use the CMS Explorer to browse to it. *See Also:* ”Using CMS Explorer to Browse Your Ektron CMS400.NET Site” on page 10	String
CacheInterval	Sets the amount of time the server control’s data is cached. The default is 0 (zero). This is the amount of time, in seconds, a control’s data is cached. For example, if you want to cache the data for five minutes, you set the `CacheInterval` property to 300 (three hundred).	Integer

Property	Value	Data Type
Hide	Used to hide language select dropdown box and label in design time and run time. **True** = Hide language select dropdown box and label **False** = Show language select dropdown box and label	Boolean
Language	Set a language for the Language Select Box. The language property shows results in design-time (in Visual Studio) and at run-time (in a browser).	Integer
WrapTag	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code. **Custom** - Allows you to use a custom tag.	String

BreadCrumb Server Control

There are two kinds of breadcrumbs: sitemap breadcrumbs and history breadcrumbs.

Sitemap breadcrumbs show you the path through the sitemap to the current page, like the path shown at the top of the page of a Yahoo! directory listing. Sitemap breadcrumbs always show the same path for the same page, regardless of how that page is reached.

History breadcrumbs more accurately fulfill the breadcrumb trail metaphor. They display an edited version of the browser's history - links to each of the most recently visited pages in the order that they were visited.

The BreadCrumb server control displays history breadcrumbs. Your site visitors can use the breadcrumb trail to jump to previously visited pages.

The breadcrumb trail can appear vertically or horizontally on a Web page. In addition, you can define how many breadcrumbs are left for site visitors to navigate back. You can also use an image and Alt text instead of, or in addition to, text to display the breadcrumb trail.

NOTE If a site visitor revisits a page that is in the breadcrumb trail already, the breadcrumb trail reverts back to that point in the trail. For example, if you visit the following pages Dev > FAQ > WebForm > PR, and then return to FAQ, the breadcrumb trail looks like this: **Dev > FAQ**.

Breadcrumb Trail Type	Example
Horizontal; text only	Dev > WebForm > pr > FAQ > Meta*

Breadcrumb Trail Type	Example
Horizontal with images	DEV > PR > FAQ > Search > Meta
Horizontal with images and Alt text	DEV > PR > FAQ > Search > Meta Meta
Vertical; text only	> Dev > WebForm > pr > FAQ > Meta*
Vertical with images	DEV > PR > FAQ > Search > Meta
Vertical with images and Alt text	DEV > PR > FAQ > Search > Meta Meta

You should add a BreadCrumb Server Control to every Web form in your site. Doing this ensures that a breadcrumb trail is created for each Web page.

This subsection contains the following topics:

- ”BreadCrumb Properties Apply to Web Forms, Not Web Pages” on page 59
- ”Types of BreadCrumb Server Control Properties” on page 59
- ”BreadCrumb Server Control Properties” on page 61
- ”Using the BreadCrumb Server Control” on page 65
- ”Making a Content Block’s Title Appear in the Breadcrumb Trail” on page 66

BreadCrumb Properties Apply to Web Forms, Not Web Pages

Keep in mind that your Web site is made up of Web forms. Each page depends on a form to determine much of its appearance. Forms and pages have a parent > child relationship. That is, any form can be associated with any number of pages.

When defining breadcrumb properties, you define them for a *form*. All pages that use that form inherit its breadcrumb properties.

Assume, for example, that a Web form is used for the Human Resources section of your Web site. You might use the title **Human Resources** to identify the form in the BreadCrumb Server Control. Whenever a page is visited that uses that form, **Human Resources** appears on the breadcrumb trail -- that is, the form title appears, not the individual page.

Types of BreadCrumb Server Control Properties

BreadCrumb Server Control properties are divided into two categories:

- ”Properties that Determine a Breadcrumb Trail’s Appearance” on page 60
- ”Properties that Determine How Pages Associated with the Form Appear on a Breadcrumb Trail” on page 60

Properties that Determine a Breadcrumb Trail's Appearance

For each Web form, you can customize the breadcrumb trail. For example, you might want the breadcrumb trail to appear horizontally on one Web form and vertically on another.

Property	Determines this aspect of the breadcrumb trail's appearance on this form	For more information, see
CurrentPageIndicator	Symbols or characters to identify the current page	"CurrentPageIndicator" on page 61
DisplayStyle	Whether it appears horizontally or vertically	"DisplayStyle" on page 61
LinkLastItem	Whether the last item is a hyperlink	"LinkLastItem" on page 62
LinkTarget	The type of window that appears when a user clicks an item	"LinkTarget" on page 63
MaxItems	The maximum number of items	"MaxItems" on page 63
Mode	whether the breadcrumb trail appears as hyperlinked text or plain text.	"Mode" on page 63
Separator	Symbols or characters used to separate items	"Separator" on page 63

Properties that Determine How Pages Associated with the Form Appear on a Breadcrumb Trail

Use these properties define how any page that uses this form appears within a breadcrumb trail. It does not matter where the trail appears.

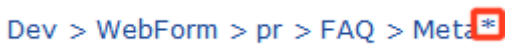
Note that you can use text, an image, or both to describe the form within the breadcrumb trail. If you use both, the image appears first, followed by the text.

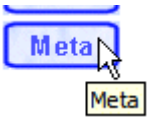
Property	Determines this aspect of Web form when it appears on breadcrumb trail	For more information, see
DisplayTitle	Text to describe it	”DisplayTitle” on page 62
IconAlt	“Alt” text associated with image specified in IconPath property	”IconAlt” on page 62
IconPath	Image to define it	”IconPath” on page 62

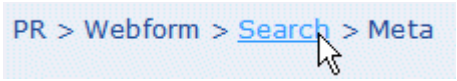
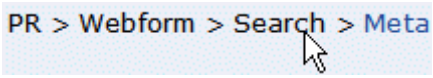
BreadCrumb Server Control Properties

The BreadCrumb Server Control properties are described in this table.

NOTE The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

Property	Response	Data Type
Authenticated	Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* ”Using CMS Explorer to Browse Your Ektron CMS400.NET Site” on page 10	String
CurrentPageIndicator	If desired, enter one or more symbols or characters that identify the current page in the breadcrumb trail. See example below.  These characters appear *after* the image or title that identifies the current page in the breadcrumb trail.	String
DisplayStyle	Indicate how to display the breadcrumb trail: horizontally or vertically. The default is Horizontal. *See Also:* ”Breadcrumb Trail Type” on page 57	String

Property	Response	Data Type
DisplayTitle	Enter text to describe this Web form when it appears in the breadcrumb trail. For example, if the Web form whose properties you are defining is used for all Human Resources pages on your site, enter **Human Resources**. *See Also:* "BreadCrumb Properties Apply to Web Forms, Not Web Pages" on page 59 The default is **Title**. If you define an image in the **IconPath** property below, the image appears in the trail, followed by this text.	String
DynamicParameter	Gets or sets the QueryString parameter to read a content ID dynamically.	String
Hide	Used to hide login button in design time and run time. **True** = Hide login button **False** = Show login button	Boolean
IconAlt	If you define an image in the **IconPath** property, enter any "Alt" text that should appear when a site visitor hovers the cursor over that image. Here is an example. 	String
IconPath	If you want the breadcrumb trail to display an image to identify this Web form, enter the path to the image. **Important:** The image location must be relative to the Web root. For example: `\CMS400Example\Workarea\Images\bc_meta_icon.gif` On the breadcrumb trail, the image precedes any text defined in the **DisplayTitle** property.	String
LinkLastItem	Use this property to determine whether the last breadcrumb item appears as a hyperlink on this Web form. If this property is set to **true**, and a user clicks the item, the current page reappears. **True** = last item is a hyperlink **False** = last item is an image and/or text only; the user cannot click on it	Boolean

Property	Response	Data Type
LinkTarget	Determines the type of window that appears on this Web form when a user clicks an item in the breadcrumb trail. The default is **_self**. **_Self -**opens in same window **_Top** - opens in parent window **_Blank** - opens in new window **_Parent** - opens in the parent frame	String
MaxItems	Enter the maximum number of items in the breadcrumb trail on this Web form. The default is **5**. If you set a value of 1 or greater and the user navigates beyond that number of pages, only the most recent pages appear. The older pages disappear from the trail. To allows an unlimited number of breadcrumb trail items on this Web form, enter zero (**0**).	Integer
MetadataName	Specify the name of a Metadata Type that you want to associate with the page. For more information, see "BreadCrumb Metadata Type" on page 64	String
Mode	Allows you to make the breadcrumb trail appear as non-hyperlinked plain text. The default is **Normal**. **Normal** = breadcrumb trail is hyperlinked  **DisplayOnly** = breadcrumb trail is plain text 	String
Separator	Enter one or more characters to separate the items in a breadcrumb trail on this Web form. The default character is the greater than sign (**>**). 	String

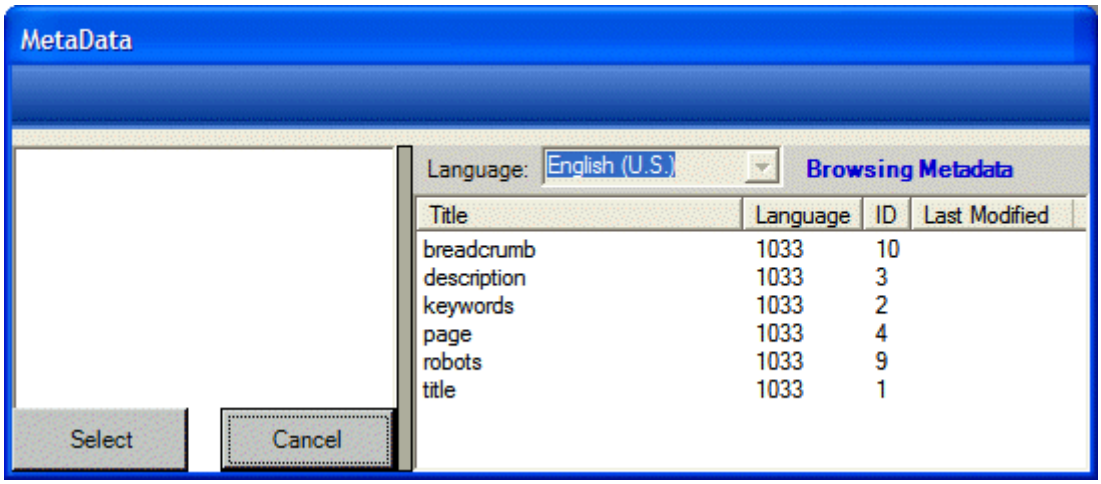
Property	Response	Data Type
WrapTag	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code. **Custom** - Allows you to use a custom tag.	String

BreadCrumb Metadata Type

To associate a Metadata Type with the BreadCrumb Server Control, follow these steps.

1. In the properties window for the BreadCrumb Server Control, click on the MetadataName property.
2. Click the Ellipses button (...).

   If you are not logged in, log in now using the CMS Explorer window.
3. The Browsing Metadata screen appears.



4. Select a Metadata Type to apply to the BreadCrumb Server Control.
5. Set the DynamicParameter to ID. This allows the metadata to reflect what ever content block appears on the page.

Using the BreadCrumb Server Control

Add the BreadCrumb Server Control to each Web form for which you want to create a breadcrumb trail. Follow these steps to use the BreadCrumb Server Control.

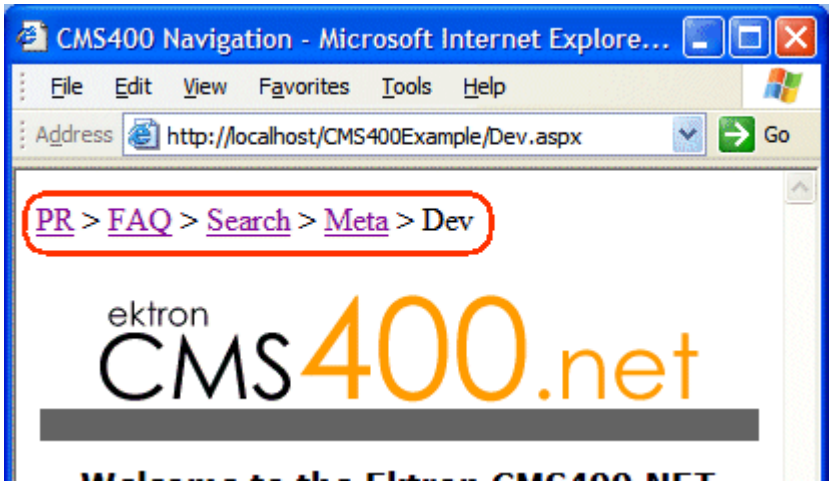
WARNING! It is recommended that you do not add a BreadCrumb Server Control to a Web form that contains a Calendar Server Control. When both server controls appear on the same page and a site visitor clicks on different months, the BreadCrumb trail could look like this: Calendar > Calendar > Calendar > Calendar > Calendar. This happens because each time a site visitor clicks on a month, you are in effect opening a new Web form.

1. Open a Web form for which you want to create a breadcrumb trail.
2. Drag and drop the BreadCrumb Server Control onto an appropriate location of the Web form.
3. Set the **DisplayTitle** property. See ”DisplayTitle” on page 62.

NOTE To customize the look of your breadcrumb trail, configure the other properties. For more information, see”BreadCrumb Server Control Properties” on page 61.

4. Save the Web form.
5. Repeat steps one through four for all Web forms to be included in the breadcrumb trail.
6. Open a browser.
7. View a Web page with a BreadCrumb Server Control in it.
8. Navigate to the rest of the Web pages that contain BreadCrumb Server Controls.

9. As you move around, the pages are added to the breadcrumb trail.



Making a Content Block's Title Appear in the Breadcrumb Trail

By adding a line or two of code, the breadcrumb trail can display the title of a content block. You can add a line of code to each Web form or if your content is dynamic, add the code once and each time a new piece of content is called dynamically a new breadcrumb is created.

Only one content block per Web form can be used with the breadcrumb. If you have multiple content blocks on a page, select the one that describes the overall page best.

Steps to Use a Content Block's Title as a Breadcrumb

1. Drag and drop a **ContentBlock Server Control** on a Web form.
2. Choose a **DefaultContentID** for the content block.
3. Drag and drop a **BreadCrumb Server Control** on the same form.
4. Remove the word **Title** from the **DisplayTitle** property.
5. Add the following line code to the page load event code behind:

```
BreadCrumb1.DisplayTitle = ContentBlock1.EkItem.Title
```

6. Build the project.
7. View the Web form in a Web browser.

The following is an example content block titles used as breadcrumbs:

The word Title appears in the breadcrumb trail when the DisplayTitle property is left empty. Even with the added code, if the breadcrumb loads before the content block, the breadcrumb has no information in the DisplayTitle property and therefore displays the word title by default.

If the word Title appears in the breadcrumb trail, check to make sure the word Title does not appear in the DisplayTitle property. Next, if Title still appears, add the following line of code to the page load event of the code behind:

```
ContentBlock1.Fill()
```

The code in the page load event should now look like this:

```
ContentBlock1.Fill()

BreadCrumb1.DisplayTitle = ContentBlock1.EkItem.Title
```

This ensures that the content block information is loaded first.

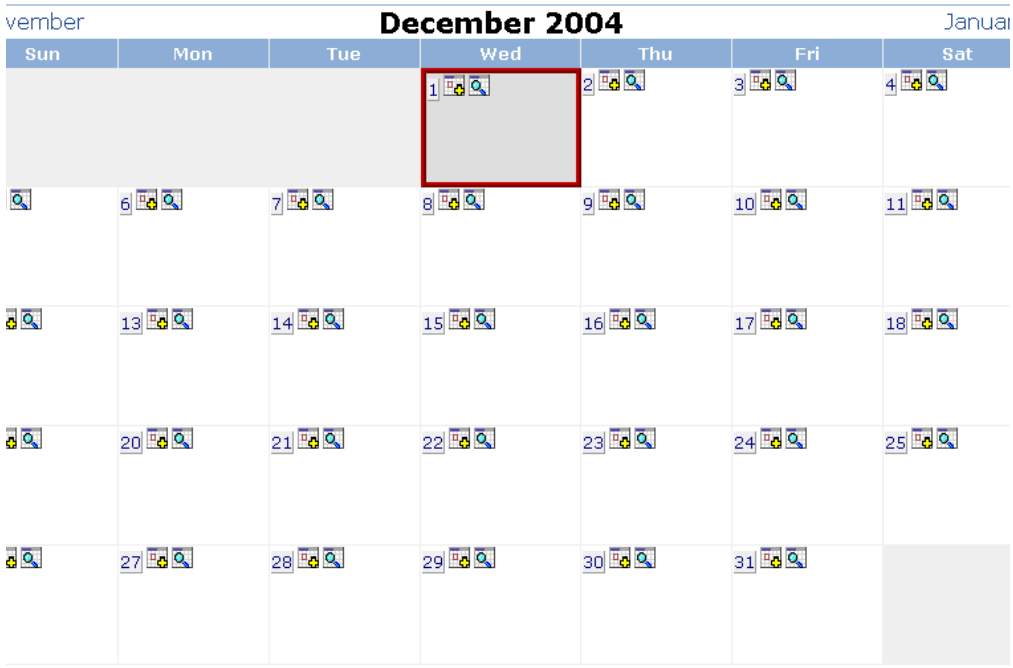
Calendar Server Control

The Calendar Server Control displays a calendar that was created in the Workarea. By using the Calendar Server Control's properties, you can select multiple views of the calendar to display.
See Also: "DisplayType" on page 72

You can adjust a calendar's look and feel by using a style sheet.
See Also: "Calendar Style Sheet" on page 74 and "calendarStyles.css Class Description" on page 75.

The Calendar Server Control can output the raw XML data of the calendar. You can use this XML data to create a completely customized calendar.

Below is a Calendar displayed on the Web Page. When logged in, you can add and view calendar events. When logged out, calendar events only appear.

Login Status	Calendar display
Logged in	

Login Status	Calendar display
Logged Out	

A calendar can be *static* or *dynamic*.

- A static calendar is selected in the server control properties.
- A dynamic calendar is passed to the page by a URL parameter.

This subsection contains the following topics:

- ”Calendar Server Control Properties” on page 70
- ”Calendar Style Sheet” on page 74
- ”calendarStyles.css Class Description” on page 75

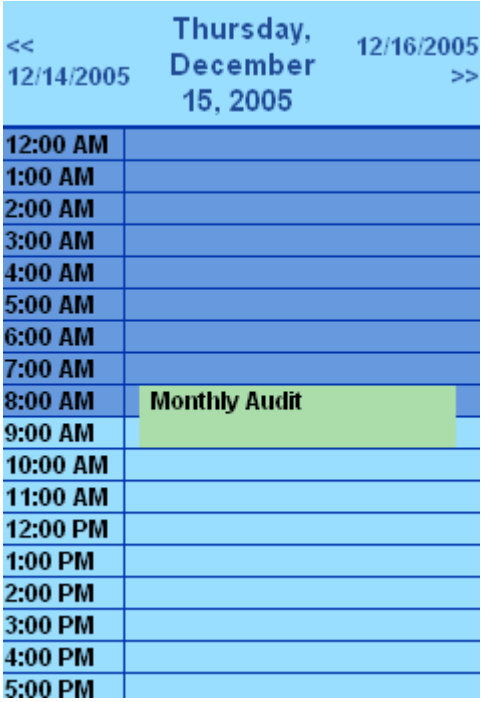
Calendar Server Control Properties

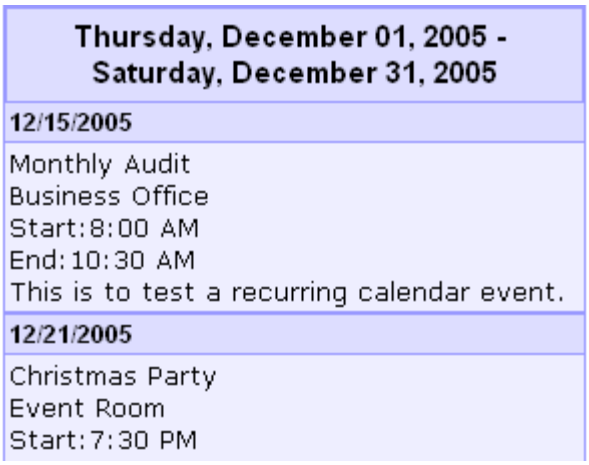
The Calendar Server Control properties are described in this table.

NOTE There are additional settings to help define the calendar in the Ektron CMS400.NET Workarea. These include display weekends, forward only, show all label, etc. For information on these properties, see Ektron CMS400.NET **Administrator's Manual** > **Calendars** > **Managing Calendar Modules** > **Viewing Calendar Modules**.

NOTE The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

Property	Value	Data Type
Authenticated	Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	String
DefaultCalendarID	The ID number of the calendar that appears where you inserted this server control. If you don't know the ID number of the calendar, use the CMS Explorer to browse to it. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	Integer

Property	Value	Data Type
DisplayType	Specify the Calendar view type. The default is **Month**. • **Month** - A 30 day calendar month view. • **Day** - Single day view.  	String

Property	Value	Data Type
DisplayType (Con't)	• **Inline** - Display events that occur from the start date of the calendar through the end date.  • **MonthUpWithEvent** - Displays a month calendar. Events appear to the right of the calendar when you click on a day. Note: The URL parameters `hevtype` and `evhighlight` do not work. They are reserved for future use.	
DynamicParameter	To make this calendar dynamic, select **calendar_id**. When you do, this server control uses the calendar passed as a URL parameter.	String
EndDate	Specify the end date the calendar displays. Used in conjunction with the StartDate property. For example, you have a calendar that shows only classes for the next three month semester.	Date

Property	Value	Data Type
Hide	Used to hide output of calendar in design time and run time. **True** = Hide calendar **False** = Display calendar	Boolean
Language	Set a language for viewing calendar. The language property shows results in design-time (in Visual Studio) and at run-time (in a browser).	Integer
ShowXML	At runtime, output the raw XML data instead of the calendar. With this XML data, you can apply your own XSL to create your own custom display formats. Default is **False.** **True** = Show raw XML data **False** = Show Calendar	Boolean
StartDate	Specify the start date the calendar displays. Used in conjunction with the EndDate property. For example, you have a calendar that shows only classes for the next three month semester.	Date
WrapTag	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an inline portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code. **Custom** - Allows you to use a custom tag.	String

Calendar Style Sheet

Before you insert a Calendar into your Ektron CMS400.NET template, Ektron recommends referencing the style sheet `\webroot\cms400example\calendarStyles.css` in your template. Copy the style sheet to your webroot folder then add it to your project. Next create a reference to calenderStyles.css. For example:

```
html>

 <head>

   <title>Company Calendar</title>
```

```
    <link href="calendarStyles.css" rel="stylesheet" type="text/
css" />

 </head>
```

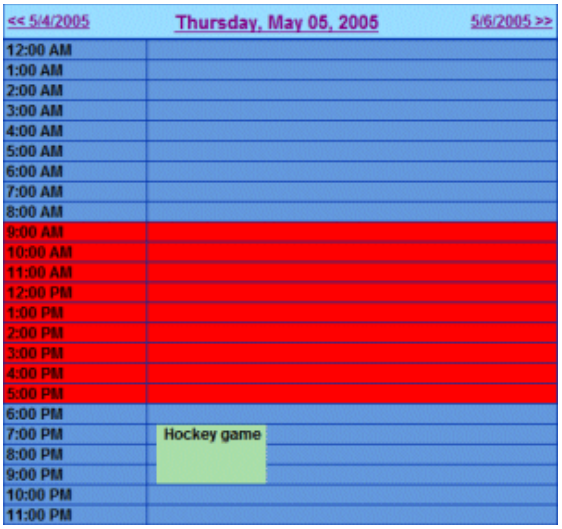
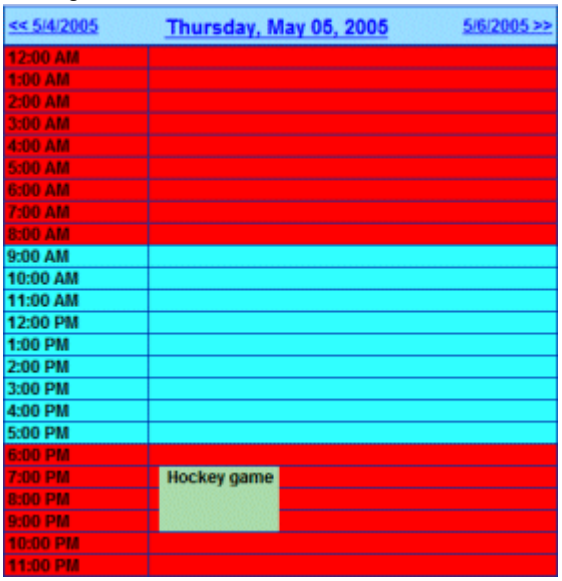
This can also be accomplished via drag and drop. If you drag and drop the calendarStyles.css from the Solution Explorer to your Web form, the reference is created in the HTML automatically.

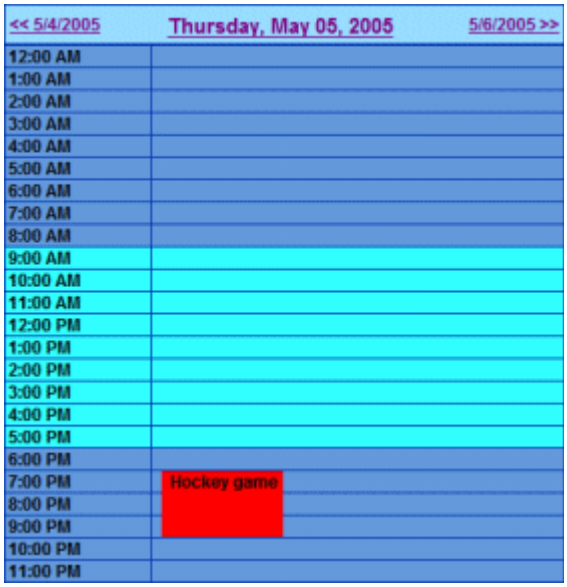
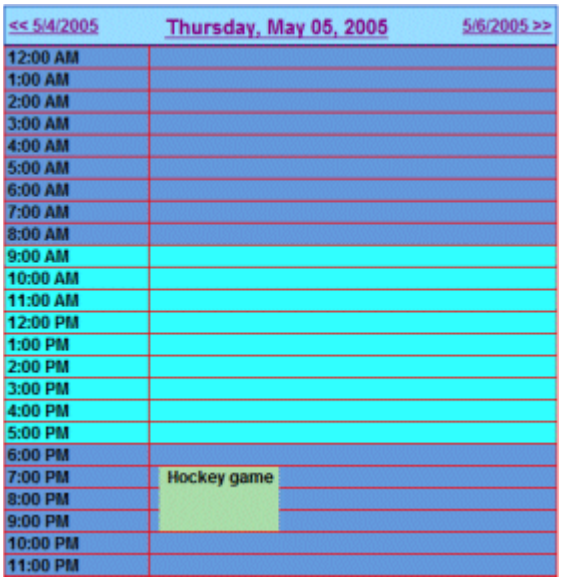
calendarStyles.css Class Description

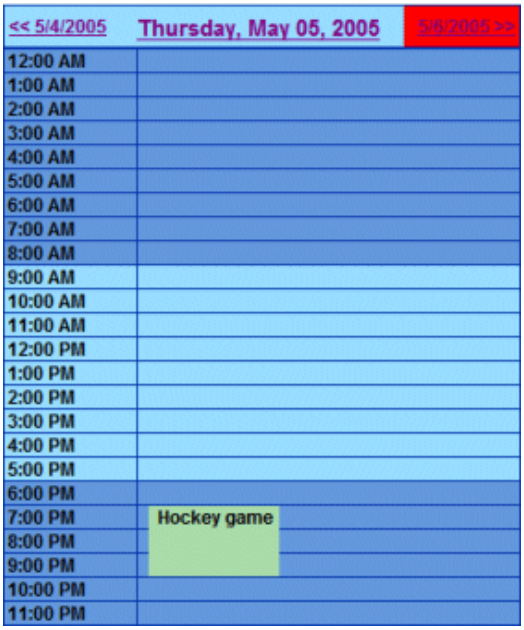
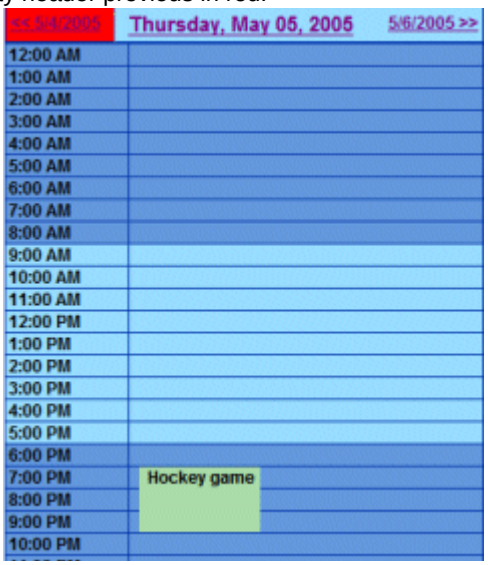
The following table describes the classes found in the calendarStyles.css file. This table contains the following sections.

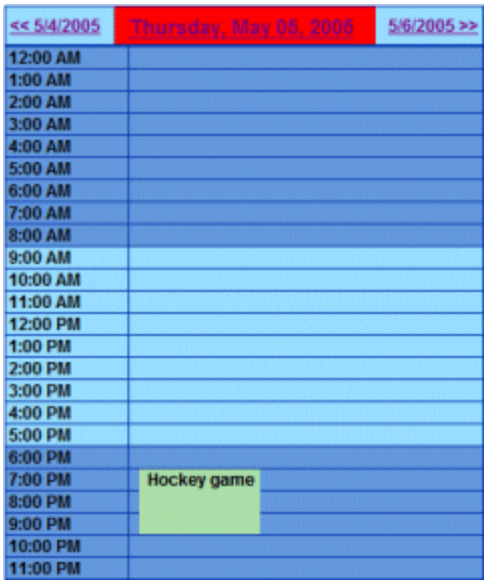
- ”Single Day View (Day)” on page 75
- ”Month View (month)” on page 79
- ”Inline View (inline)” on page 91
- ”Month-Up with Event (monthupwithevent)” on page 93
- ”Event Type Control (used in month display)” on page 100

Style Class	Description	Attributes and Default Settings
Single Day View (Day)		

Style Class	Description	Attributes and Default Settings
.dv_HourCell	The workday hour cells. The following example shows these cells with a red background.  	• **background**: #99DDFF • **font-family**: Arial, Arial Narrow, Times New Roman • **font-size**: 9pt • **font-weight**: bold
.dv_HourCellEve	The evening and morning hour cells. The following example shows these cells with a red background.  	• **background**: #6699DD • **font-family**: Arial, Arial Narrow, Times New Roman • **font-size**: 9pt • **font-weight**: bold

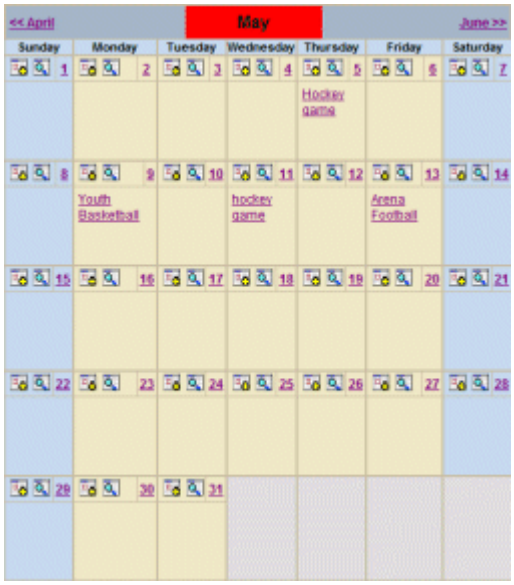
Style Class	Description	Attributes and Default Settings
.dv_EventTitle	An event listed on the daily calendar. The following example shows the event with a red background.  	• **background**: #aaddaa • **font-family**: Arial, Arial Narrow, Times New Roman • **font-size**: 9pt • **font-weight**: bold • **cursor**: pointer
.dv_BorderCell	The borders of the hour cells. The following example shows the borders in red.  	• **background**: #0033AA

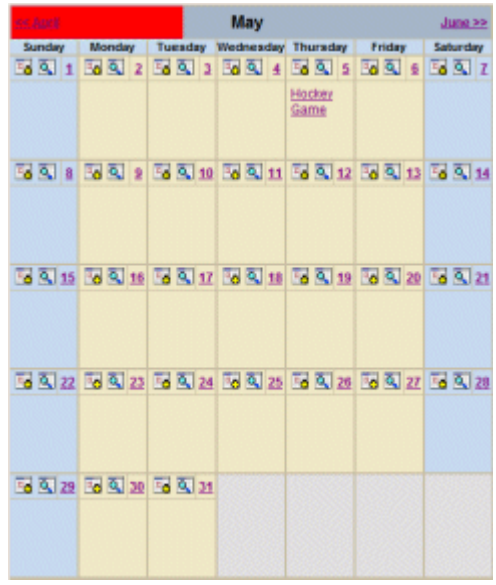
Style Class	Description	Attributes and Default Settings
.dv_DayHeaderNext	The button for moving the calendar to the next day. The following example shows the day header next in red.  	• **background**: #99DDFF • **color**: #0033AA • **font-family**: Arial • **font-size**: 9pt • **font-weight**: bold • **text-align**: right
.dv_DayHeaderPrev	The button for moving the calendar to the previous day. The following example shows the day header previous in red.  	• **background**: #99DDFF • **color**: #0033AA • **font-family**: Arial • **font-size**: 9pt • **font-weight**: bold • **text-align**: right

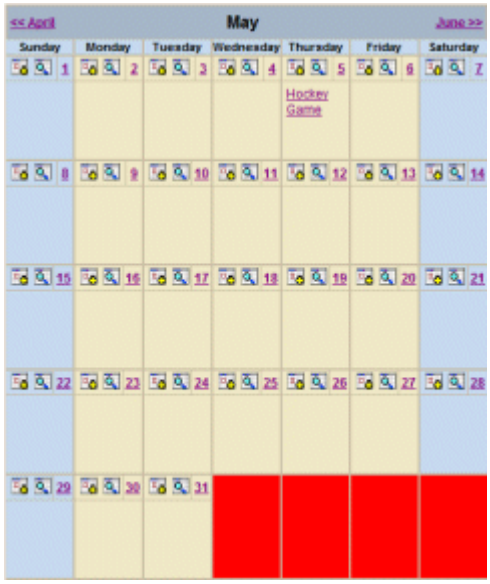
Style Class	Description	Attributes and Default Settings
.dv_DayHeader	The day header for the calendar. The following example shows the day header in red.  	• **background**: #99DDFF • **color**: #0033AA • **font-family**: Arial • **font-size**: 11pt • **font-weight**: bold • **text-align**: center
Month View (month)		

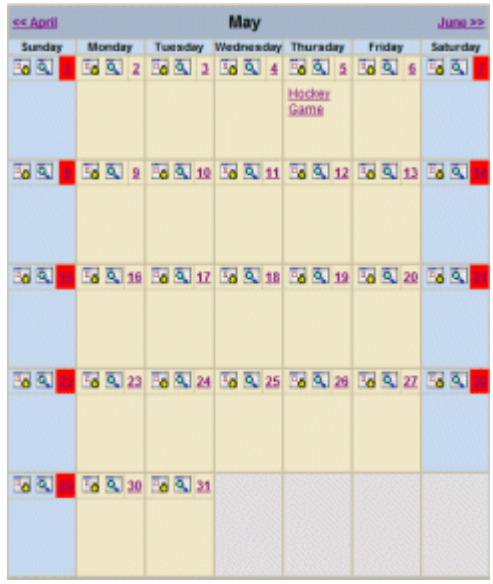
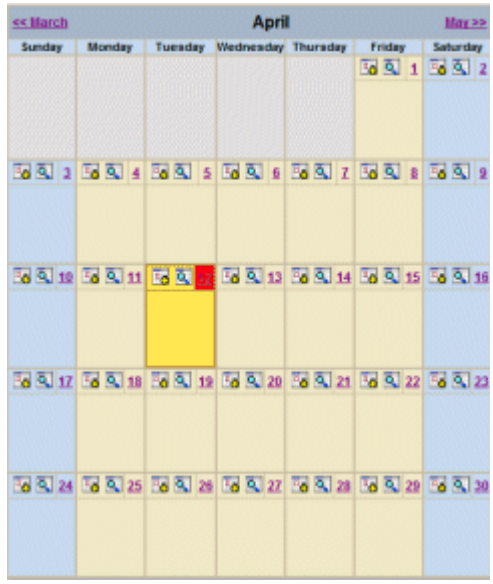
Style Class	Description	Attributes and Default Settings
.mv_weekendDOWHeader	Weekend day of week header. The following example shows the weekend day of week header in red. 	• **background**: #c5daef • **font-family**: arial • **font-weight**: bold • **font-size**: 8pt • **border-top**: #c9bda5 1px solid • **border-bottom**: #c9bda5 1px solid • **border-right**: #c9bda5 1px solid • **border-left**: #c9bda5 1px solid • **text-align**: center

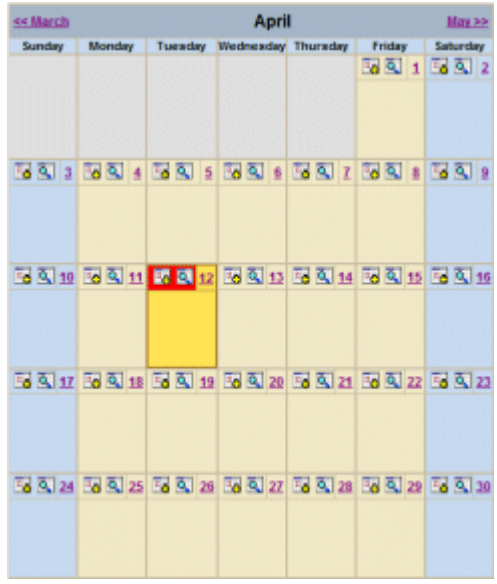
Style Class	Description	Attributes and Default Settings
.mv_weekdayDOWHeader	Weekday day of week header. The following example shows the weekday day of week header in red. 	• **background**: #c5daef • **font-family**: arial • **font-weight**: bold • **font-size**: 8pt • **border-top**: #c9bda5 1px solid • **border-bottom**: #c9bda5 1px solid • **border-right**: #c9bda5 1px solid • **border-left**: #c9bda5 1px solid • **text-align**: center

Style Class	Description	Attributes and Default Settings
.mv_MonthHeader	Month header. The following example shows the month header background in red.  	• **font-weight**: bold • **font-size**: 12pt • **background**: transparent • **font-family**: Arial
.mv_MonthHeaderBkg	Month header background. The following example shows the month header background in red. 	• **border-right**: #c9bda5 2px solid • **border-top**: #c9bda5 2px solid • **border-left**: #c9bda5 2px solid • **font-weight**: bold • **font-size**: 9pt • **background**: #A5B7C9 • **font-family**: arial

Style Class	Description	Attributes and Default Settings
.mv_MonthHeaderNext	The button for moving the calendar to the next month. The following example shows the month header next background in red. 	• **font-weight**: bold • **fontsize**: 9pt • **background**: transparent • **font-family**: Arial • **text-align**: right
.mv_MonthHeaderPrev	The button for moving the calendar to the previous month. The following example shows the month header previous background in red.  	• **font-weight**: bold • **fontsize**: 9pt • **background**: transparent • **font-family**: Arial • **text-align**: left

Style Class	Description	Attributes and Default Settings
.mv_NonMonthBorder	The cells at the beginning or end of the month that do not contain days. The following example shows the non month border in red.  	• **border-right**: #c9bda5 1px solid • **border-top**: #c9bda5 1px solid • **background**: #e0e0e0 • **border-left**: #c9bda5 1px solid • **border-bottom**: #c9bda5 1px solid
.mv_DateCell	The cells where the numerical weekdays appear. The following example shows the date cell in red. 	• **font-weight**: bold • **font-size**: 9pt • **background**: #F0E9C5 • **border-left**: #c9bda5 1px solid • **border-bottom**: #c9bda5 1px solid • **font-family**: arial

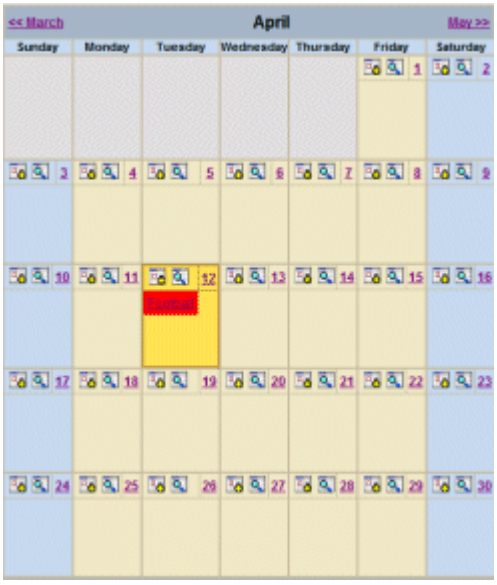
Style Class	Description	Attributes and Default Settings
.mv_DateCellWeekend	The cells where the numerical weekend days appear. The following example shows the date cell weekend in red.  	• **font-weight**: bold • **font-size**: 9pt • **background**: #C5DAEF • **border-left**: #c9bda5 1px solid • **border-bottom**: #c9bda5 1px solid • **font-family**: arial
.mv_DateCellToday	The cells where today's numerical day appears. The following example shows the date cell today in red.  	• **background**: #FFE354 • **border-left**: #b38b3b 1px solid • **border-bottom**: #b38b3b 1px solid • **font-family**: Arial • **font-size**: 9pt • **font-weight**: bold

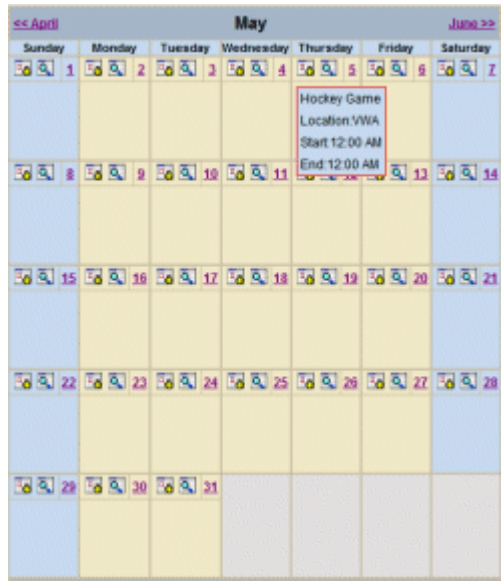
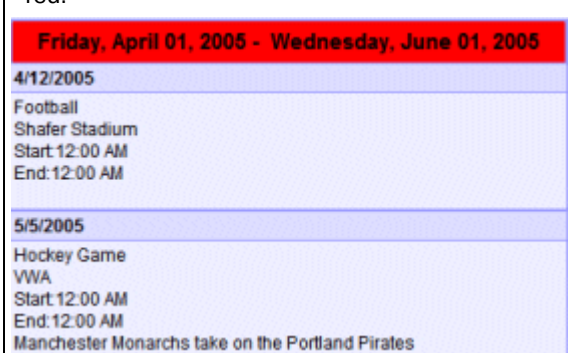
Style Class	Description	Attributes and Default Settings
.mv_DateCellEmptyWeekend **Note**: These cells appear empty when you are not logged in to Ektron CMS400.NET.	The weekend cells that hold the Add Event and View Event buttons. The following example shows the date cell empty weekend in red. 	• **background**: #C5DAEF • **border-bottom**: #c9bda5 1px solid • **font-family**: Arial • **font-size**: 9pt • **font-weight**: bold
.mv_DateCellEmptyToday **Note**: This cell appears empty when you are not logged in to Ektron CMS400.NET.	The today cell that holds the Add Event and View Event buttons. The following example shows the date cell empty today in red.  	• **background**: #FFE354 • **border-bottom**: #B38B3B 1px solid • **font-family**: Arial • **font-size**: 9pt • **font-weight**: bold

Style Class	Description	Attributes and Default Settings
.mv_DateCellEmpty **Note**: These cells appear empty when you are not logged in to Ektron CMS400.NET	The weekday cells that holds the Add Event and View Event buttons. The following example shows the date cell empty in red. 	• **background**: #f0e9c5 • **border-bottom**: #c9bda5 1px solid • **font-family**: Arial • **font-size**: 9pt • **font-weight**: bold
.mv_eventContainer	The popup window that appears when the event is hovered over. The following example shows the event container circled in red. 	• **visibility**: hidden • **position**: absolute Note: When the visibility property is set to visible, the event container appears upon page load.

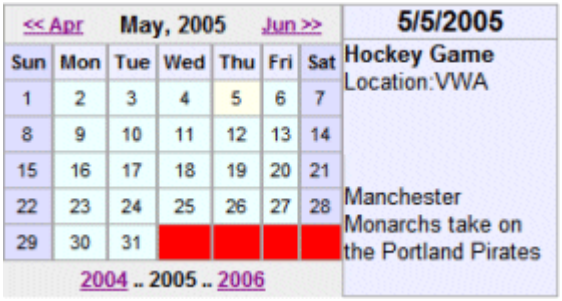
Style Class	Description	Attributes and Default Settings
.mv_dayBorderToday	The today cell that holds listed events. The following example shows the day border today in red. 	• **border-right**: #b38b3b 2px solid • **border-top**: #b38b3b 2px solid • **background**: #ffe354 • **border-left**: #b38b3b 2px solid • **border-bottom**: #b38b3b 2px solid
.mv_dayBorder This class effects all weekday cells, except the today cell.	The weekday cell that holds listed events. The following example shows the day border in red. 	• **border-right**: #c9bda5 1px solid • **border-top**: #c9bda5 1px solid • **background**: #f0e9c5 • **border-left**: #c9bda5 1px solid • **border-bottom**: #c9bda5 1px solid

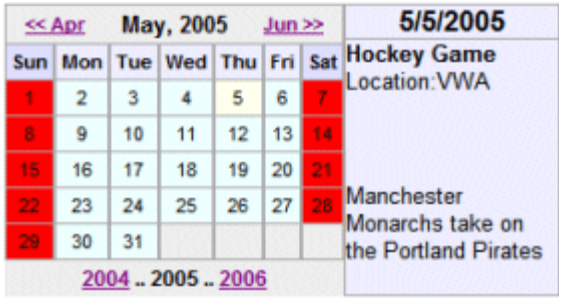
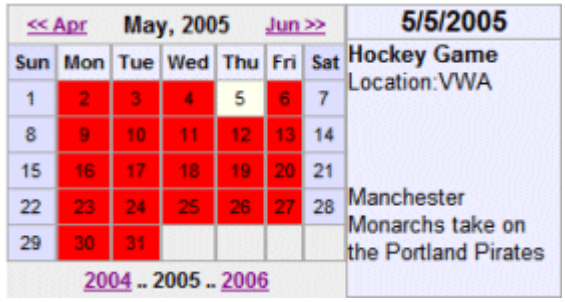
Style Class	Description	Attributes and Default Settings
.mv_dayBorderWeekend	The Weekend day cell that holds listed events. The following example shows the day border weekend in red. 	• **border-right**: #c9bda5 1px solid • **border-top**: #c9bda5 1px solid • **background**: c5daef • **border-left**: #c9bda5 1px solid • **border-bottom**: #c9bda5 1px solid
.mv_eventTitle	The event title cell. The following example shows the event title cell in red. 	• **font-weight**: normal • **font-size**: 9pt • **background**: transparent • **font-family**: Arial

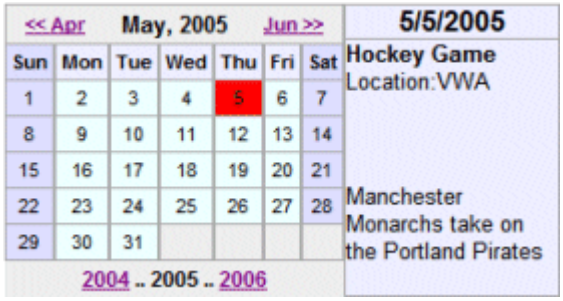
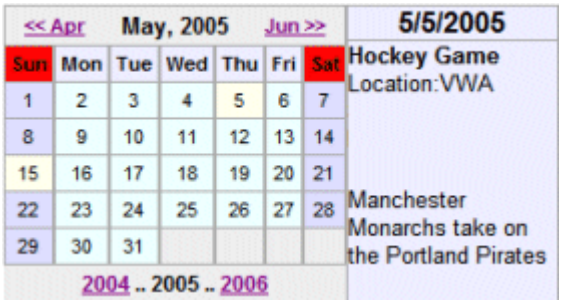
Style Class	Description	Attributes and Default Settings
.mv_DayCellToday	Today's event title cell. The following example shows the day cell today in red.  	• **background**: transparent • **font-family**: Arial • **font-weight**: bold • **font-size**: 9pt
.mv_TodayCell	The font for the event pop-up. The following example shows the today cell font in red. 	• **font-weight**: normal • **font-size**: 9pt • **background**: #C5DAEF • **font-family**: Arial

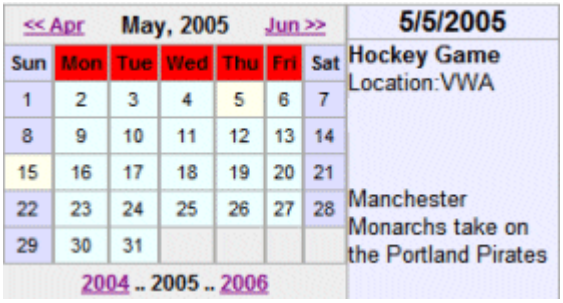
Style Class	Description	Attributes and Default Settings
.mv_eventPopup	The border for the event pop-up. The following example shows the event pop-up border in red.  	• **border-right**: #000000 1px solid • **border-top**: #000000 1px solid • **border-left** #000000 1px solid • **border-bottom**: #000000 1px solid
Inline View (inline)		
.ILViewHeaderBkg	The inline view header background. The following example shows the view header background in red.  	• **border-right**: #9999ff 2px solid • **border-top**: #9999ff 2px solid • **font-weight**: bold • **font-size**: 11pt • **background**: #ddddff • **border-left**: #9999ff 2px solid • **font-family**: Arial • **text-align**: center

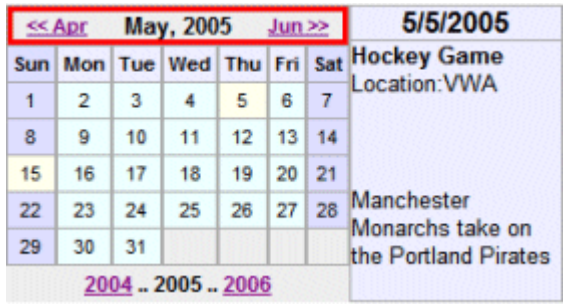
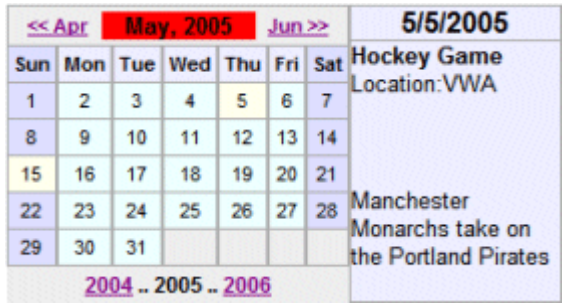
Style Class	Description	Attributes and Default Settings
.ILViewEvent	Font for the event description.The following example shows the view event font in red. **Friday, April 01, 2005 - Wednesday, June 01, 2005** 4/12/2005 Football Shafer Stadium Start:12:00 AM End:12:00 AM 5/5/2005 Hockey Game VWA Start:12:00 AM End:12:00 AM Manchester Monarchs take on the Portland Pirates	• **font-size**: 9pt • **font-family**: Arial
.ILViewDate	The date for the inline view. The following example shows the view date background in red. **Friday, April 01, 2005 - Wednesday, June 01, 2005** 4/12/2005 Football Shafer Stadium Start:12:00 AM End:12:00 AM 5/5/2005 Hockey Game VWA Start:12:00 AM End:12:00 AM Manchester Monarchs take on the Portland Pirates	• **border-right**: #9999ff 1px solid • **border-top**: #9999ff 1px solid • **font-weight**: bold • **font-size**: 9pt • **background**: #ddddff • **border-left**: #9999ff 1px solid • **font-family**: Arial

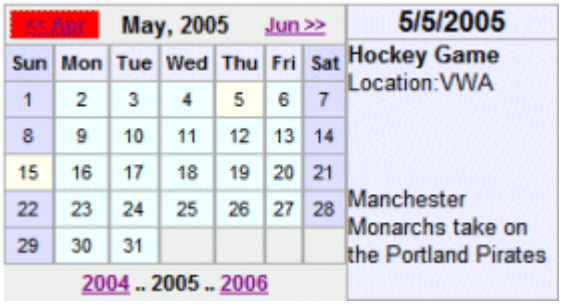
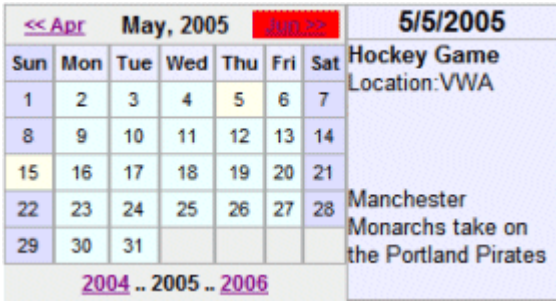
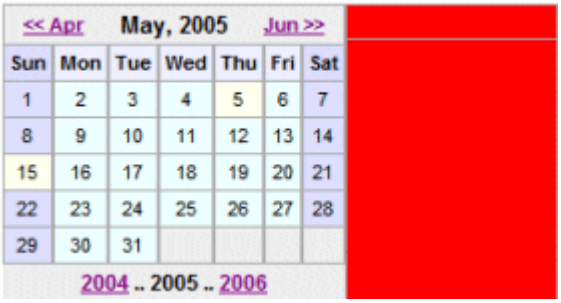
Style Class	Description	Attributes and Default Settings
.ILViewDayCell	The cell the events are listed in. The following example shows the view day cell in red. 	• **border-right**: #9999ff 1px solid • **border-top**: #9999ff 1px solid • **font-weight**: bold • **font-size**: 9pt • **background**: #eeeeff • **border-left**: #9999ff 1px solid • **border-bottom**: #9999ff 1px solid • **font-family**: Arial
Month-Up with Event (monthupwithevent)		
.mup_nonMonthCell	The cells at the beginning or end of the month that do not contain days. The following example shows the non-month cells in red.  	• **background:** #eeeeee • **font-family**: Arial • **font-weight:** normal • **font-size**: 8pt • **text-align**: center • **border-right**: #aaaaaa 1px solid • **border-bottom**: #aaaaaa 1px solid • **border-left**: #aaaaaa 1px solid

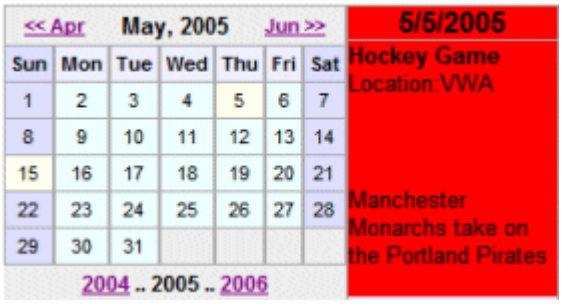
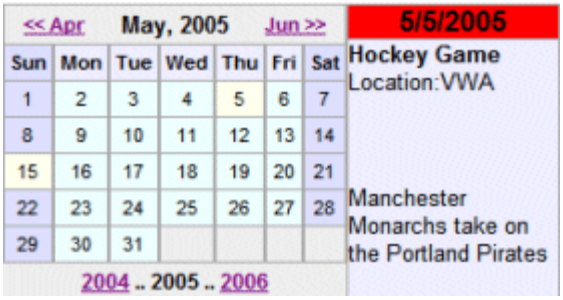
Style Class	Description	Attributes and Default Settings
.mup_weekendCell	The weekend day cell. The following example shows the weekend cell in red.  	• **background:** #dedeff • **font-family**: Arial • **font-weight:** normal • **font-size**: 8pt • **text-align**: center • **border-right**: #aaaaaa 1px solid • **border-bottom**: #aaaaaa 1px solid • **border-left**: #aaaaaa 1px solid
.mup_weekdayCell	The week day cell. The following example shows the week day cell in red.  	• **background:** #eeffff • **font-family**: Arial • **font-weight:** normal • **font-size**: 8pt • **text-align**: center • **border-right**: #aaaaaa 1px solid • **border-bottom**: #aaaaaa 1px solid • **border-left**: #aaaaaa 1px solid

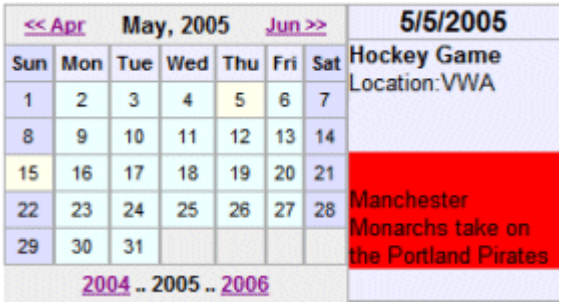
Style Class	Description	Attributes and Default Settings
.mup_weekdayEventCell	Highlights the weekdays that have events. The following example shows the weekday event cell in red.  	• **background:** #ffffee • **font-family**: Arial • **font-weight:** normal • **font-size**: 8pt • **text-align**: center • **border-right**: #aaaaaa 1px solid • **border-bottom**: #aaaaaa 1px solid • **border-left**: #aaaaaa 1px solid • **cursor**: pointer
.mup_weekendDOWHeader	Weekend day of week header. The following example shows the weekend day of week header in red.  	• **background:** #ddddff • **font-family**: Arial • **font-weight:** Bold • **font-size**: 8pt • **border-right**: #aaaaaa 1px solid • **border-bottom**: #aaaaaa 1px solid • **border-left**: #aaaaaa 1px solid • **text-align**: center

Style Class	Description	Attributes and Default Settings
.mup_weekdayDOWHeader	Weekday day of week header. The following example shows the weekday day of week header in red.  	• **background:** #eeeeff • **font-family**: Arial • **font-weight:** Bold • **font-size**: 8pt • **border-right**: #aaaaaa 1px solid • **border-bottom**: #aaaaaa 1px solid • **border-left**: #aaaaaa 1px solid • **text-align**: center
.mup_yearFooter	The year selection cell. The following example shows the year footer in red. 	• **background**: #eeeeee • **font-family**: Arial • **font-weight**: Bold • **font-size**: 9pt • **text-align**: center

Style Class	Description	Attributes and Default Settings
.mup_monthHeaderBorder	The border around the month header. The following example shows the month header border in red.  	• **background**: #eeeeee • **font-family**: Arial • **font-weight**: Bold • **font-size**: 10pt • **border-right**: #aaaaaa 1px solid • **border-top**: #aaaaaa 1px solid • **border-bottom**: #aaaaaa 1px solid • **border-left**: #aaaaaa 1px solid • **text-align**: center
.mup_monthHeader	The month header. The following example shows the month header in red.  	• **background**: #eeeeee • **font-family**: Arial • **font-weight**: Bold • **font-size**: 10pt • **text-align**: center

Style Class	Description	Attributes and Default Settings
,mup_monthHeaderPrev	The previous button, on the month header. The following example shows the month header previous back ground in red.  	• **background**: #eeeeee • **font-family**: Arial • **font-weight**: Bold • **font-size**: 8pt • **text-align**: center
.mup_monthHeaderNext	The next button, on the month header. The following example shows the month header next background in red.  	• **background**: #eeeeee • **font-family**: Arial • **font-weight**: Bold • **font-size**: 8pt • **text-align**: center
.mup_eventDispBkg Note: .mup_eventDispBkg deals with the background layer of the event display. .mup_eventDisplay deals with the layer the event is displayed on.	The background for the event display. The following example shows the event display background in red.  	• **background**: #eeeeff • **border-right**: #aaaaaa 1px solid • **border-top**: #aaaaaa 1px solid • **border-bottom**: #aaaaaa 1px solid • **border-left**: #aaaaaa 1px solid

Style Class	Description	Attributes and Default Settings
.mup_eventDisplay Note: .mup_eventDispBkg deals with the background layer of the event display. .mup_eventDisplay deals with the layer the event is displayed on.	The event display. The following example shows the event display in red.  	**background**: transparent
.mup_eventDisplayDate	The event display date header. The following example shows the event display date in red.  	• **font-size**: 12pt • **font-weight**: bold • **background**: transparent • **font-family**: Arial • **text-align**: center • **border-bottom**: #aaaaaa 1px solid
.mup_eventDisplayEventTitle	Display for the event title. The following example shows the event display event title in red. 	• **font-weight**: bold • **background**: transparent • **font-family**: Arial • **font-size**: 10pt • **text-align**: left

Style Class	Description	Attributes and Default Settings
.mup_eventDisplayLocation	Display for the event location. The following example shows the event display location in red. 	• **background**: transparent • **font-family**: Arial • **font-weight**: normal • **font-size**: 10pt • **text-align**: left
.mup_eventDisplayTimes	Display for event times. The following example shows the event display times in red. 	• **background**: transparent • **font-family**: Arial • **font-weight**: normal • **font-size**: 10pt • **text-align**: left
.mup_eventDisplayLongDesc	Displays the event's long description. The following example shows the event display long description in red.  	• **background**: transparent • **font-family**: Arial • **font-weight**: normal • **font-size**: 10pt • **text-align**: left
Event Type Control (used in month display)		

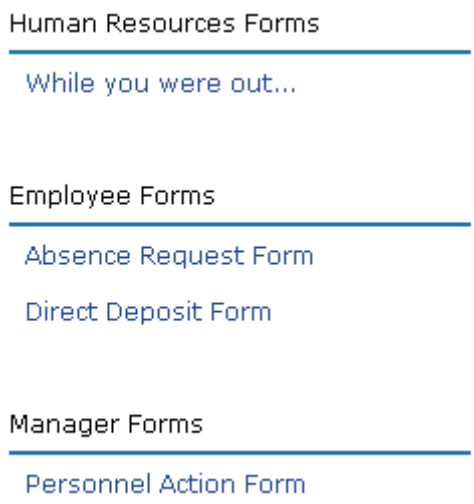
Style Class	Description	Attributes and Default Settings
.etCtrl_Background	The background of the event type control cell. The following example shows the event type control background in red. 	• **border-right**: #c9bda5 2px solid • **border-top**: #c9bda5 2px solid • **border-left**: #c9bda5 2px solid • **border-bottom**: #c9bda5 2px solid • **font-size**: 9pt • **background**: #A5B7C9 • **font-family**: Arial
.etCtrl_InstructCell	The instructions cell for event type control. The following example shows the instruction cell in red. 	• **background**: transparent • **font-family**: Arial • **font-size**: 9pt • **font-weight**: bold • **text-align**: right • **vertical-align**: top

Style Class	Description	Attributes and Default Settings
.etCtrl_SelCell	The cell the selection box is displayed in. The instructions cell for event type control. The following example shows the selection cell in red. 	• **background**: transparent • **font-family**: Arial • **font-size**: 9pt • **text-align**: left • **vertical-align**: top
.etCtrl_selectBox	The event type selection box. The instructions cell for event type control. The following example shows the selection box in red. 	• **background**: #c5daef

Collection Server Control

The Collection Server Control is used to display a list of content block links that were created as a collection in the Ektron CMS400.NET Workarea. The control is highly customizable, allowing you to define the display of the collection on a Web page.

When added to a template and visited, collections can look like this (The following illustration shows three collections on the same page of a sample site.) You can modify the display by editing its properties.



This subsection contains the following topics:

- ”Collection Server Control Properties” on page 104
- ”Example of ecmNavigation Display” on page 105
- ”Example of ecmTeaser Display” on page 106
- ”Retrieving the XML Structure of a Collection” on page 107
- ”Using the Collection Server Control Programmatically Example” on page 108

Collection Server Control Properties

The Collection Server Control properties are described in this table.

NOTE The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

Property	Description	Data Type
Authenticated	Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	String
ContentParameter	Checks the QueryString for this value and replaces the collection with a content block when specified. Leave blank to always display the Collection or List Summary.	String
DefaultCollectionID	The ID of a collection that appears where you inserted this server control if no other collection is identified, or is not available. If you don't know the ID number of the collection, use the CMS Explorer to browse to it. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	Integer
DisplayXslt	Determines how the information on the page is displayed **None**-databind only **ecmNavigation** - lists the title of every content block in the collection *See Also:* "Example of ecmNavigation Display" on page 105 **ecmTeaser** - lists the title of every content block in the collection plus the content summary *See Also:* "Example of ecmTeaser Display" on page 106 **Path to Custom Xslt** - Enter the path to an Xslt that determines the display of the page	String

Property	Description	Data Type
DynamicParameter	To make this collection dynamic, select **coll_id**. When you do, this server control uses the collection passed as a URL parameter.	String
GetHtml	Set to **True** if you want to retrieve and display content (html body) for all content blocks in the collection. For example, to display content inside a web server control such as a DataGrid.	Boolean
Hide	Used to hide output of collection in design time and run time. **True** = Hide collection **False** = Display collection	Boolean
Language	Set a language for viewing the collection. The language property shows results in design-time (in Visual Studio) and at run-time (in a browser).	Integer
Random	Set to **True** if you want to randomly display one collection item. The item changes each time a user views the page.	Boolean
WrapTag	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an inline portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code. **Custom** - Allows you to use a custom tag.	String

Example of ecmNavigation Display

The following is an example of a collection being used as a navigation menu.

ecmNavigation XSL code

Below is the XSL code that is used to create ecmNavigation Display. You can use this code as the basis to design your own XSLT.

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/
XSL/Transform">

<xsl:template match="/">

<table border="0" cellspacing="0" cellpadding="0" width="100%">

<xsl:for-each select="Collection/Content">

<tr><td><a><xsl:attribute name="href">

<xsl:value-of select="QuickLink"/></xsl:attribute><xsl:value-of
select="Title"/>

</a></td></tr></xsl:for-each></table></xsl:template></
xsl:stylesheet>
```

Example of ecmTeaser Display

The following is an example of a collection using the ecmTeaser display style.

ecmTeaser XSL code

Below is the XSL code that is used to create ecmTeaser Display. You can use this code as the basis to design your own XSLT.

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/
XSL/Transform">

<xsl:template match="/"><table border="0" cellspacing="0"
cellpadding="0" width="100%">

<xsl:for-each select="Collection/Content"><tr><td><a>

<xsl:attribute name="href"><xsl:value-of select="QuickLink"/></
xsl:attribute>

<xsl:value-of select="Title"/></a> 

   // xml.Append("<xsl:value-of
select="substring(DisplayDateModified, 0, string-
length(DisplayDateModified)-11)" />

</td></tr><tr><td><xsl:value-of select="Teaser" /></td></
tr><tr><td> </td></tr>

</xsl:for-each></table></xsl:template></xsl:stylesheet>
```

Retrieving the XML Structure of a Collection

Retrieving the XML structure of XML content allows for greater control over developing XSLs. The following is an example of how to retrieve the XML structure:

1. Open a new Web form.
2. Drag and drop a Collection Server Control onto it.

3. Set the DefaultCollectionID property.
4. Drag and drop a Textbox on the Web form.
5. Set the TextMode property to MultiLine.

NOTE It is also recommended that you set the width of the text box to at least 400px.

6. On the code behind page, add the following line.

```
Textbox1.Text = Collection1.XmlDoc.InnerXml
```

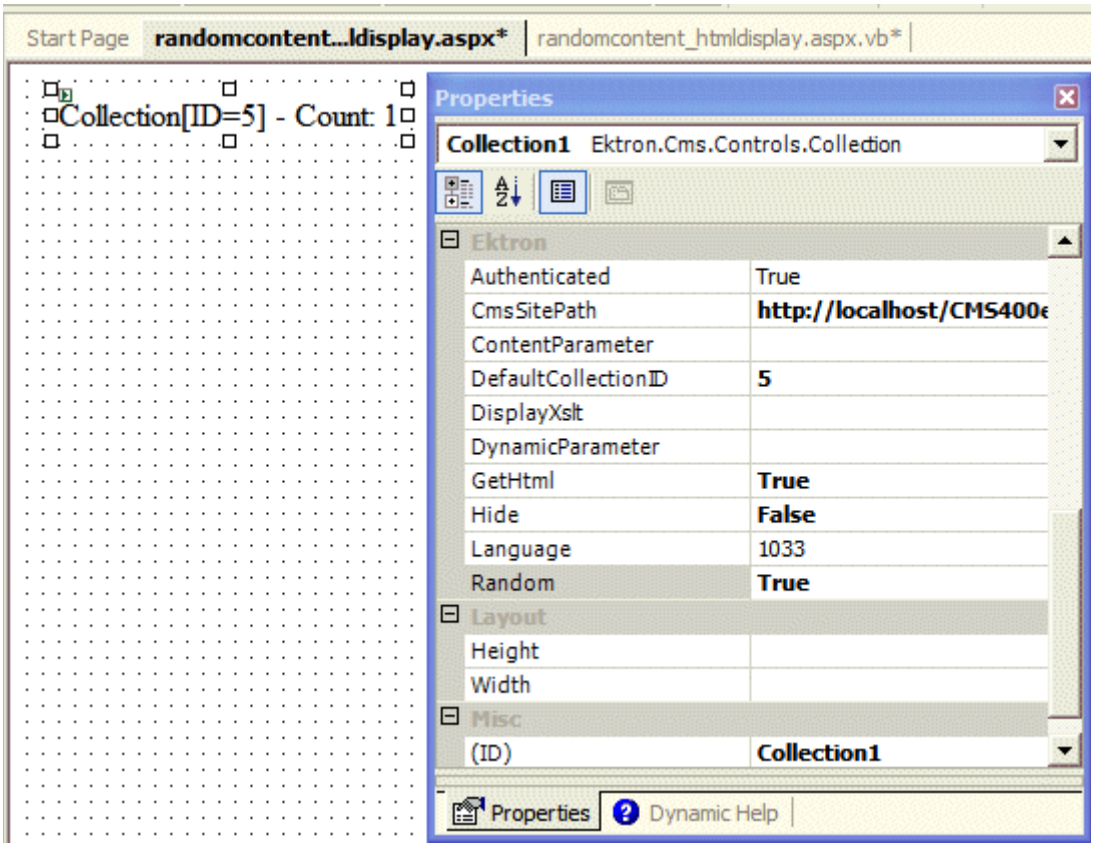
7. Build the project.
8. View the Web form in a browser.
9. The XML structure of the collection appears in the textbox.

Using the Collection Server Control Programmatically Example

The following is an example of using code behind, and a drag and drop Collection Server Control to display random content from a collection.

1. Drag and Drop a Collection Server Control on your Web form.
2. Set the properties in the properties window.

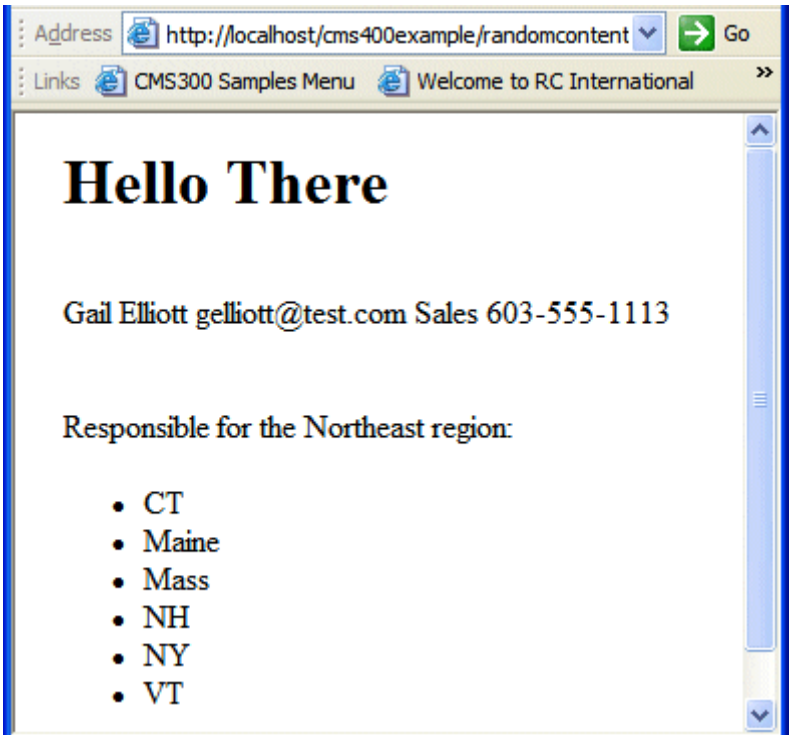
NOTE In this example, the Random property and the GetHtml property must be set to **True**.



3. Add the following code to the code behind.

```
Dim str As String

str = "<h1>Hello There</h1><br>"

str &= Collection1.EkItems(0).Html

Collection1.Text = str
```

4. Build and browse your web form.


Address
http://localhost/cms400example/randomcontent
Go
Links
CMS300 Samples Menu
Welcome to RC International
Hello There
Gail Elliott gelliott@test.com Sales 603-555-1113
Responsible for the Northeast region:
CT
Maine
Mass
NH
NY
VT

Content Analytics Server Control

Use the AnalyticsTracker server control to track statistics about visits to your Web site, such as:

- how often your content was viewed
- how many site visitors viewed for first time, and how many returned
- the most popular referral pages (that is, the Web page from which visitors clicked to arrive on your site)

For more information about Content Analytics, see the **Ektron CMS400.NET** Administrator manual chapter “Content Analytics Feature.”

Content Analytics Server Control Properties

NOTE The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

Ektron Properties

Property	Value	Data Type
Authenticated	Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* ”Using CMS Explorer to Browse Your Ektron CMS400.NET Site” on page 10	Boolean

Property	Value	Data Type
DefaultContentID	If desired, enter the ID of a content item being tracked by this server control. If you do, all visits to this content item are tracked for content analytic statistics. If you don't know the ID number of the content block, use the CMS Explorer to browse to it. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	Integer
DynamicParameter	To make this content rating control dynamic, select **id**. When you do, this server control is attached to the content block passed as a URL parameter.	String
Hide	Select **False** to display this server control on the page. Select **True** to suppress it.	Boolean
Language	Set a language for viewing content. The language property shows results in design-time (in Visual Studio) and at run-time (in a browser).	Integer
WrapTag	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The `<span>` tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The `<div>` tag is used when you want to apply attributes to a block of code. **Custom** - Lets you use custom tag.	String
EnableAnalytics	Determines if the Analytics feature is enabled for this page and (if identified) content item. **ConfigSpecified -** Take the value from the web.config file's `enableAnalytics` property. **True -** Enable Analytic tracking **False -** Disable Analytic tracking *See Also:* **Ektron CMS400.NET** Administrator Manual Section "Content Analytics Feature" > "Enabling the Content Analytics Feature."	

ContentBlock Server Control

The ContentBlock Server Control displays a content block on an Ektron CMS400.NET Web page.

Ektron CMS400.NET has two types of content blocks:

- Static - displays one specified content block
- Dynamic - displays the content block of the ID passed through a URL parameter

In addition, a content block can contain XHTML or XML content.

For an example of using the ContentBlock Server Control programmatically, see ”Using the ContentBlock Server Control Programmatically Example” on page 122

This subsection contains the following topics:

- ”Static Content Block” on page 113
- ”ContentBlock Server Control Properties” on page 114
- ”Dynamic Content Block” on page 115
- ”XML Content Block” on page 118
- ”The OverrideXslt and DisplayXslt Properties” on page 119
- ”Using the ContentBlock Server Control Programmatically Example” on page 122

Static Content Block

A static content block displays one identified content block on a Web page. The following table explains how to complete the server control properties to insert a static content block on a Web page.

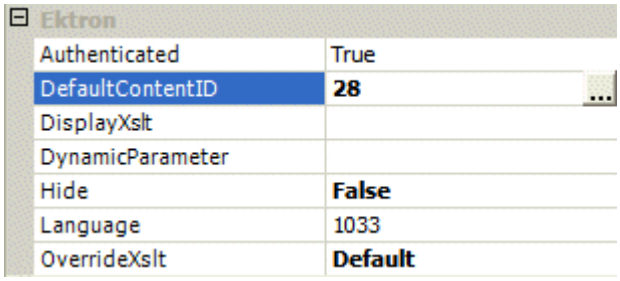
ContentBlock Server Control Properties

The following table explains the properties of the ContentBlock Server Control.

NOTE The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

Property	Value	Data Type
Authenticated	Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	String
DefaultContentID	The ID of a content block that appears where you insert this server control. If you don't know the ID number of the content block, use the CMS Explorer to browse to it. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	Integer
DynamicParameter	Select **None - Use Default**. This parameter is used for dynamic content blocks.	String
Hide	Used to hide a content block in design time and run time. **True** = Hide content block **False** = Show content block	Boolean
Language	Set a language for viewing content. The language property shows results in design-time (in Visual Studio) and at run-time (in a browser).	Integer
OverrideXslt	Ignore for a non-XML content block.	must be one of these values: default, one, two, three
DisplayXslt	Ignore for a non-XML content block.	String

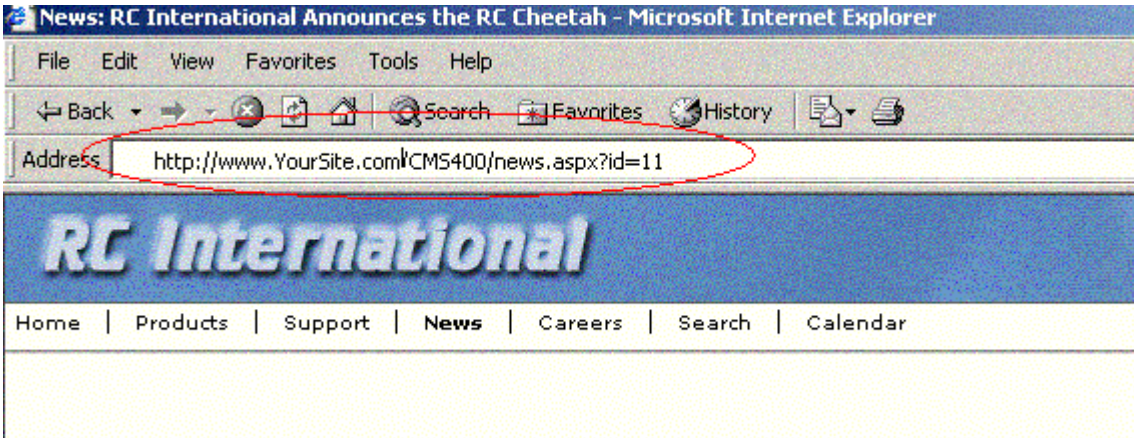
Property	Value	Data Type
WrapTag	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code. **Custom** - Allows you to use a custom tag.	String



In the above example, the server control retrieves content block id=28 and displays it in the browser.

Dynamic Content Block

Use a dynamic content block to display a content block whose ID is passed through a URL parameter. You would use this server control with a dynamic template.



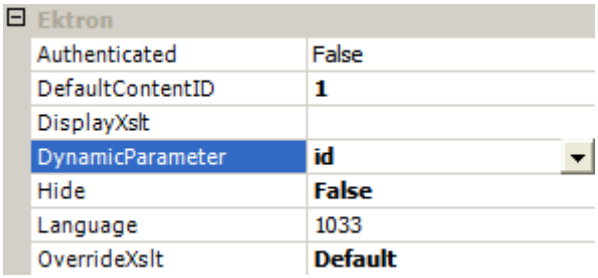
The following table explains how to complete the ContentBlock Server Control's properties to insert a dynamic content block on a template.

NOTE The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

Property	Value	Data Type
Authenticated	Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	String
DefaultContentID	The ID of a content block that appears where you inserted this server control if no other content block is identified, or is not available. If you don't know the ID number of the content block, use the CMS Explorer to browse to it. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	Integer
DynamicParameter	Select **id**. When you do, this server control uses the content block passed as a URL parameter.	String

Property	Value	Data Type
Hide	Used to hide a content block in design time and run time. **True** = Hide content block **False** = Show content block	Boolean
Language	Set a language for viewing content. The language property shows results in design-time (in Visual Studio) and at run-time (in a browser).	Integer
OverrideXslt	Ignore for a non-XML content block.	must be one of these values: default, one, two, three
DisplayXslt	Ignore for a non-XML content block.	String
WrapTag	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code. **Custom** - Allows you to use a custom tag.	String

The following example shows how the ContentBlock Server Control can be used to pass the id as a URL parameter.



When a user clicks a link that passes the content block ID as a URL parameter, that content block appears. If that content block is not available, content block 1 appears.

XML Content Block

Use an XML content block to display an XML content block on an Ektron CMS400.NET Web page.

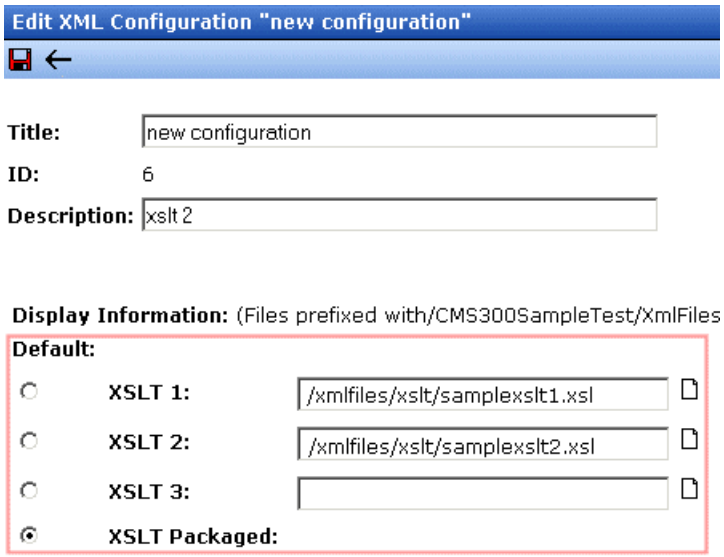
NOTE The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

Property	Value	Data Type
Authenticated	Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	String
DefaultContentID	The ID of a content block that appears where you inserted this server control if no other content block is identified, or is not available. If you don't know the ID number of the content block, use the CMS Explorer to browse to it. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	Integer
DynamicParameter	To make this content block dynamic, select **id**. When you do, this server control uses the content block passed as a URL parameter.	String
Hide	Used to hide a content block in design time and run time. **True** = Hide content block **False** = Show content block	Boolean
Language	Set a language for viewing content. The language property shows results in design-time (in Visual Studio) and at run-time (in a browser).	Integer
OverrideXslt	An internal Xslt file used to display XML content if one is not identified in the Xslt property. *See Also:* "The OverrideXslt and DisplayXslt Properties" on page 119	must be one of these values: default, one, two, three
DisplayXslt	Specify an external XSLT file. *See Also:* "The OverrideXslt and DisplayXslt Properties" on page 119	String

Property	Value	Data Type
WrapTag	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code. **Custom** - Allows you to use a custom tag.	String

The OverrideXslt and DisplayXslt Properties

The DisplayXslt property is optional. If used, it specifies an external XSLT file. If the DisplayXslt property is not defined, the OverrideXslt property specifies an XSLT identified in the Edit XML Configuration screen. The following table provides more information about these properties.

To use this display XSLT	Enter this value for XSLT	Enter this value for Override XSLT
The default XSLT specified in the Edit XML Configuration screen (illustrated below). Note that, in this example, the XSLT Packaged option is the default XSLT, since it is selected. XSLT Packaged is the XSLT from the Edit XML configuration screen (that is, the XSLT created in the Data Designer).  		Default
XSLT 1 from the Edit XML configuration screen		1
XSLT 2 from the Edit XML configuration screen		2
XSLT 3 from the Edit XML configuration screen		3

To use this display XSLT	Enter this value for XSLT	Enter this value for Override XSLT
An absolute or relative path to an XSLT file.	An external XSLT file not specified in the Edit XML configuration screen. For example `sample.xslt`	does not matter - if XSLT value exists, **OverrideXSLT** is ignored

See Also: The “Managing XML” chapter of the **Ektron CMS400.NET** Administrator manual

Retrieving the XML Structure of an XML Content Block

Retrieving the XML structure of XML content allows for greater control over developing XSLs. The following is an example of how to retrieve the XML structure:

1. Open a new Web form.
2. Drag and drop a Content Block Server Control onto it.
3. Set the DefaultContentID to an XML content block.

WARNING! This does not work with HTML content blocks, as there is no XML structure to output.

4. Drag and drop a Text box on the Web form.
5. Set the TextMode property to MultiLine.

NOTE It is also recommended that you set the width of the text box to at least 400px.

6. On the code behind page, add the following line.

```
TextBox1.Text = ContentBlock1.EkItem.Html
```

7. Build the project.
8. View the form in a browser.
9. The XML structure of the content block appears in the text box.

Using the ContentBlock Server Control Programmatically Example

The following code will display a content block:

NOTE Before adding these lines of code, you need to drag and drop a literal box on your Web form.

```
Dim MyContentBlock As New ContentBlock

MyContentBlock.DefaultContentID = 8

MyContentBlock.Fill()

Literal1.Text = MyContentBlock.EkItem.Html
```

To display a content block with the content block title, do the following:

NOTE Before adding these lines of code, you need to drag and drop a two literal boxes on your Web form.

```
Dim MyContentBlock As New ContentBlock

MyContentBlock.DefaultContentID = 8

MyContentBlock.Fill()

Literal1.Text = MyContentBlock.EkItem.Title

Literal2.Text = MyContentBlock.EkItem.Html
```

ContentRating Server Control

Use the ContentRating server control to place a rating scale on any page of your Web site. The scale lets Ektron CMS400.NET collect feedback by giving site visitors the opportunity to rate a particular content item. A visitor can rank the content on a numerical scale of one through nine, as well as submit a text comment.

The control lets a site visitor rate a content item only once. On subsequent visits to the page, a message informs the visitor that he has previously rated the page, which can optionally display a summary of ratings for it. As a result, the control has two appearance modes: *initial* and *complete*.

Ektron CMS400.NET maintains and reports content ratings data. For more information, see the **Ektron CMS400.NET** Administrator Manual section “Managing Content” > “Content Rating.”

Content Rating Server Control Properties

The Ektron-specific Content Rating server control properties are divided into two groups:

Ektron - settings that affect properties, such as the display

Ektron labels - settings that affect the text labels within the control

These properties are described below.

NOTE The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

Ektron Properties

Property	Value	Data Type
Authenticated	Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	Boolean
CompleteGraph Display	Specify a value to indicate how the ratings graph appears if a site visitor has already rated content. • **horizontal** - graph appears to the right of the AlreadyRatedMessage • **vertical** - graph appears below the AlreadyRated-Message • **none** - graph does not appear	String
DefaultContentID	The ID of a content block being rated by this server control. It typically would be the content directly above the rating scale if no other content block is identified, or is not available. If you don't know the ID number of the content block, use the CMS Explorer to browse to it. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	Integer
DynamicParameter	To make this content rating control dynamic, select **id**. When you do, this server control is attached to the content block passed as a URL parameter.	String
Hide	Select **False** to display this server control on the page. Select **True** to suppress it.	Boolean
InitialGraphDisplay	Specify a value to indicate how the ratings graph appears if a site visitor has not yet rated content. • **horizontal** - graph appears to the right of the content rating scale • **vertical** - graph appears below the content rating scale • **none** - graph does not appear	String

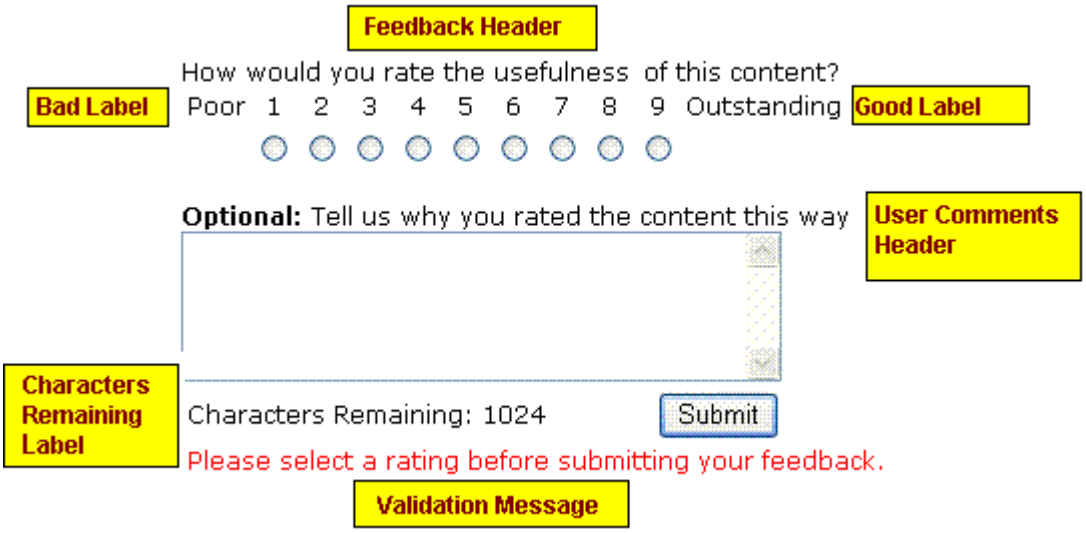
Property	Value	Data Type
Language	Set a language for viewing content. The language property shows results in design-time (in Visual Studio) and at run-time (in a browser).	Integer
Padding	Enter the number of pixels used to create space between the rating scale and the graph. Note: If you do not display the graph, this setting has no effect.	Integer
WrapTag	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The `<span>` tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The `<div>` tag is used when you want to apply attributes to a block of code. **Custom** - Lets you use custom tag.	String

Ektron Label Properties

The label properties determine the text that appears in the server control. Most of them are illustrated below.

Note that the first illustration and table describe the initial graph, while the second illustration and table describe the complete graph.

Initial Graph Display

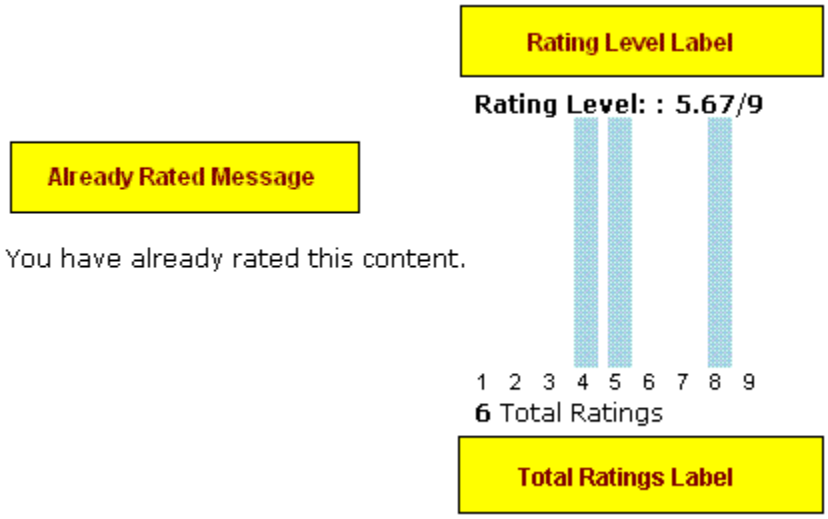


Initial Graph Properties

Property	Description	Default Text
BadLabel	The low end of the rating scale	Poor
CharsRemainingLabel	Number of additional characters user may enter into comment	Characters Remaining:
CookiesRequired Message	Appears if browser cookies have been disabled	We're sorry, but you must enable cookies in your browser to rate this content.
FeedbackHeader	Introduces rating scale	How would you rate the usefulness of this content?
GoodLabel	The high end of the rating scale	Outstanding
RatingLevelLabel	Appears at the top of the graph	Rating Level:
ThankYouMessage	Appears after site visitor selects a number, optionally enters a comment, then presses **Submit**.	Thank you for your feedback.

Property	Description	Default Text
TotalRatingsLabel	Appears below the graph, and lists the number of rating submissions received.	Total Ratings
UserCommentsHeader	Introduces free-text comment area below rating scale	Tell us why you rated the content this way
ValidationMessage	Appears on initial display if user clicks **Submit** button before selecting rating number	Please select a rating before submitting your feedback.

Complete Graph Display



Complete Graph Properties

Property	Description	Default Text
AlreadyRatedMessage	Appears in place of the rating scale if the user previously rated the page	You have already rated this content.
RatingLevelLabel	Appears at the top of the graph	Rating Level:

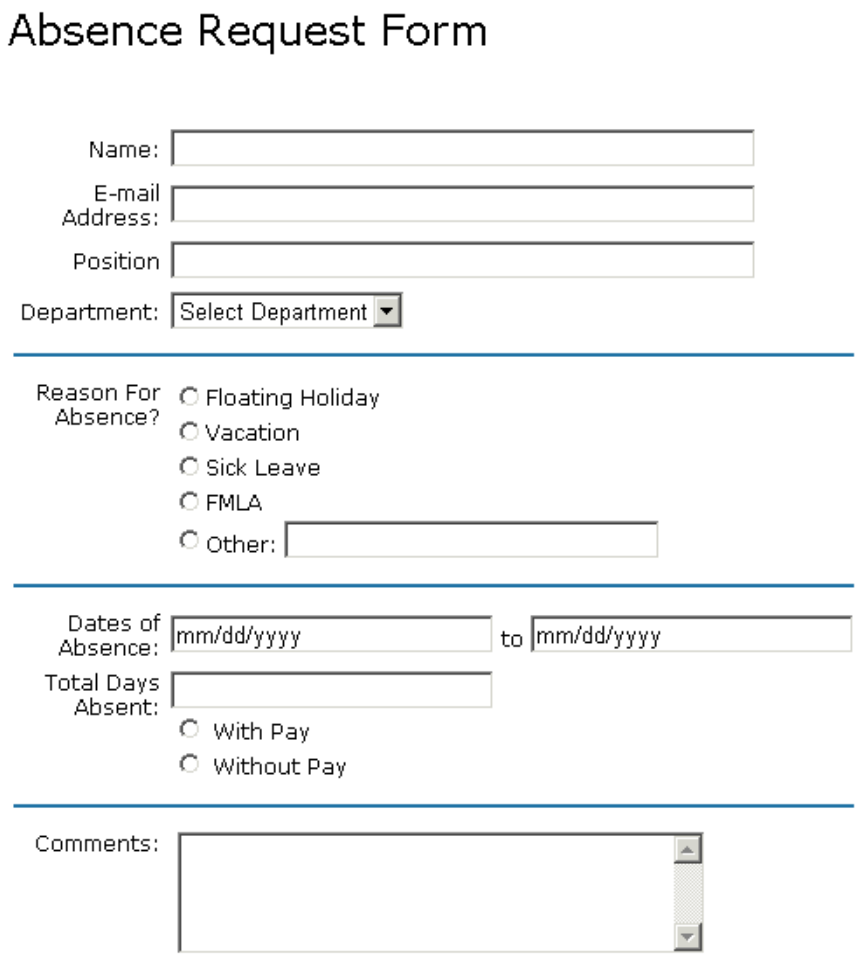
Property	Description	Default Text
TotalRatingsLabel	Appears below the graph, and lists the number of rating submissions received.	Total Ratings

FormBlock Server Control

The FormBlock Server Control displays a content block associated with a form.

IMPORTANT! If you create a template for an existing form content block, you must manually change its quicklink to point to the new template. This change does not occur automatically. This procedure is described in the Ektron CMS400.NET Administrator manual “Managing Library Assets” > “Updating Default Template for Multiple Quicklinks.”

When added to a template and visited, the form content block might look like this. You can change the display to suit your needs by modifying its properties.



This subsection contains the following topics:

- ”FormBlock Server Control Properties” on page 130
- ”Automatic versus Manual Generation of Form Tags” on page 132

FormBlock Server Control Properties

The following table explains the properties of the FormBlock Server Control.

NOTE The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

Property	Value	Data Type
AddValidation	Set to **True** to add Validation to your main runat=server form.	Boolean
Authenticated	Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10.	String
DefaultFormID	The ID of a FormBlock that appears where you inserted this server control if no other form block is identified. If you don't know the ID number of the form block, use the CMS Explorer to browse to it. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10.	Integer
DynamicParameter	To make this form block dynamic, select **id**. When you do, this server control uses the form block passed as a URL parameter.	String
FormName	Set this value to the name of your main runat=server form.	String
Hide	Used to hide a form block in design time and run time. **True** = Hide form block **False** = Show form block	Boolean
Include Tags	Determines if tags are generated automatically of manually. *See Also:* "Automatic versus Manual Generation of Form Tags" on page 132	Boolean
Language	Set a language for viewing form content. The language property shows results in design-time (in Visual Studio) and at run-time (in a browser).	Integer

Property	Value	Data Type
PostResultsAt	Enter the URL that a site user is forwarded to after he fills out the form and clicks the submit button. For example, you might have the site user fill out a form before he is allowed to see private content. Once the form is submitted, the user is forwarded to the private content. The default is blank. When left blank, the post back message associated with the form is displayed. Usage example: http://www.ektron.com **Important:** A Formblock Server Control must exist on the Web form to which you are posting.	String
WrapTag	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code. **Custom** - Allows you to use a custom tag.	String

Automatic versus Manual Generation of Form Tags

When using the FormBlock Server control's Include tags property, you have two options.

Option	Recommended if you are developing	How to Enable
Let **Ektron CMS400.NET** generate form tags automatically	A pure script .aspx page	Set Include Tags property to **True**.

Option	Recommended if you are developing	How to Enable

Modify HTML form tags in the .aspx file	An .aspx page and associated code-behind web form	Set Include Tags property to **False**.
		Here is the default .NET generated form tag:
		`<form id="Form1" method="post" runat="server">`
		Modify the form tag as indicated in red:
		`<form id="Form1" method="post" runat="server" OnSubmit="EkFmValidate(this);">`

LanguageAPI Server Control

The LanguageAPI Server Control lets a developer force a particular language for a Web site. You can do this simply by dropping the server control on the page and then choosing a language in the SiteLanguage property box. You can also override site language logic by programmatically using the LanguageAPI Server Control to detect the browser's language, and display the site in that language.

This subsection contains the following topics:

- "LanguageSelect Server Control vs. LanguageAPI Server Control" on page 134
- "LanguageAPI Server Control Properties" on page 134
- "Code Behind Only Properties" on page 135

LanguageSelect Server Control vs. LanguageAPI Server Control

The LanguageSelect Server Control lets users view your site in the language they choose. The LanguageAPI Server Control, on the other hand, forces users to view a site in a particular language. For information on LanguageSelect, see "LanguageSelect Server Control" on page 137.

LanguageAPI Server Control Properties

The LanguageAPI Server Control properties are described in this table.

NOTE The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

Property	Value	Data Type
SiteLanguage	Sets the site language. Runs through the IsValid function to verify that the language is an active language in the system.	String
SiteLanguageID	Numeric value of the site language. This is the property you will use if you are using code behind to set the sites language ID. Example: 1036 = French For a list of supported languages, see The Administrator Manual > “Appendix A: List of Supported Languages”	Integer

Code Behind Only Properties

The following is a list of LanguageAPI specific properties that can only be used programmatically.

Property	Value	Data Type
DefaultLanguage	**Read only**. What the default language of the site is set to. For example, the demo site is “English (Standard)”.	String
DefaultLanguageID	**Read only**. Returns the value that is the default language id of the site. For example, the demo site is “1033” for English.	Integer
GetLanguage	This method returns a string. Pass in a valid language ID and it will return the language name.	Integer Argument
GetLanguageID	This method returns an integer. Pass in a valid language name and it will return the language ID.	String Argument
IsValid	This method returns a boolean. You can pass in a language ID or a string and it will tell you if the system is supporting it.	Argument

Property	Value	Data Type
LanguageIdList	**Read only**. Lists all the language IDs that are activated in **Ektron CMS400.NET**. For information on how to enable languages, see The Administrator Manual > “Multi-Language Support” > “Working with Multi-Language Content” > “Determining Which Languages are Available”.	Array of Integers
LanguageTitleList	**Read only**. Lists all the languages that are activated in **Ektron CMS400.NET**.	Array of Strings
MultiLanguageEnabled	**Read only**. Tells if the site supports multi-language mode. **True** = Multi-language enabled **False** = Multi-language not enabled Multi-language is enable by default in **Ektron CMS400.NET**. To disable Multi-language, see Administrator Manual > “Multi-Language Support” > “Working with Multi-Language Content” > “Disabling Support for Multiple Language Content”.	Boolean

Using the LanguageAPI Server Control Programmatically

This example uses a logo that is not managed through Ektron CMS400.NET. It retrieves the current language from the LanguageAPI control, and uses that information to choose the logo version to display. The codebehind looks like this.

```
Select Case LanguageAPI1.CurrentLanguageID

            Case 1031

                Image1.ImageUrl = "germanlogo.gif"

            Case 1033

                Image1.ImageUrl = "englishlogo.gif"

            Case 1036

                Image1.ImageUrl = "frenchlogo.gif"

End Select
```

LanguageSelect Server Control

The LanguageSelect Server Control displays a language selection dropdown list on an Ektron CMS400.NET Web page.

This control lets a visitor to your Web site select a language in which to view the site. Here is what the control looks like when published on a Web page.

It lists all languages selected on the **Settings > Configuration > Language settings** screen. (For more information, see the “Multi-Language Support “chapter of the **Ektron CMS400.NET** Administrator manual.)

You can place this control in any location of any page on your site. On the sample site, it appears in the lower right corner of the home page.

This subsection contains the following topics:

- ”LanguageSelect Server Control vs. LanguageAPI Server Control” on page 137
- ”LanguageSelect Server Control Properties” on page 138

LanguageSelect Server Control vs. LanguageAPI Server Control

The LanguageSelect Server Control lets users view your site in the language they choose. The LanguageAPI Server Control, on the other hand, forces users to view a site in a particular language. For

information on LanguageAPI, see "LanguageAPI Server Control" on page 134.

LanguageSelect Server Control Properties

The following table explains the properties of the LanguageSelect control.

NOTE The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

Property	Value	Data Type
Authenticated	Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	String
Hide	Used to hide language select dropdown box and label in design time and run time. **True** = Hide language select dropdown box and label **False** = Show language select dropdown box and label	Boolean
LabelName	Lets you define the label next to the language select dropdown box. The default label is **Language**	String
Language	Set a language for the Language Select Box. The language property shows results in design-time (in Visual Studio) and at run-time (in a browser).	Integer
WrapTag	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code. **Custom** - Allows you to use a custom tag.	String

ListSummary Server Control

The ListSummary Server Control is used to display a list of content blocks on a Web page.

When added to a template and visited, a List Summary looks like this.

It displays each content block's title and summary information. You can modify the display to suit your needs by modifying its properties.

This subsection contains the following topics:

- "ListSummary Server Control Properties" on page 139
- "OrderKey Property" on page 143
- "List Summary Example" on page 144
- "Retrieving the XML Structure of a List Summary" on page 145

ListSummary Server Control Properties

The ListSummary Server Control properties are described in this table.

NOTE The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

Property	Description	Data Type	Values
Authenticated	Indicates if you are logged in to the CMS Explorer and can use it to browse to the folder needed for the **FolderPath** property. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	String	
ContentParameter	Checks the QueryString for this value and replaces the list summary with a content block when specified. Leave blank to always display the list summary.	String	
ContentType	Determines the type of content that appears in the list summary. The default is Content.	String	**AllTypes** - displays all of the content types for the given folder. **Content** - displays a list of content blocks. **Forms** - forms appear in the list summary. **Archive_Content** - archived content blocks appear in the list summary. **Archive_Forms** - archived forms appear in the list summary. **Assets** - assets, such as offices documents, appear in the list summary. **Archive_Assets** - archived assets appear in the list summary. **LibraryItem** - library items appear in the list summary. **NonLibraryContent** - all types of content appear in the list summary except for library items.

Property	Description	Data Type	Values
Direction	How to order the hyperlinks on the list. The sort field is determined by the OrderKey property.	either ascending or descending	**ascending** - they are arranged A, B, C or 1,2,3. **descending** - they are arranged. Z,Y,X or 3,2,1 If sorting by date, descending puts the most recent first.
DisplayXslt	Determines how the information on the page is displayed	String	**None**-databind only **ecmNavigation** - lists the title of every content block in the folder *See Also:* ”Example of ecmNavigation Display” on page 105 **ecmTeaser** - lists the title of every content block in the folder plus the content summary *See Also:* ”Example of ecmTeaser Display” on page 106 **Path to Custom Xslt** - Enter the path to an Xslt that determines the display of the page
FolderPath	The folder that contains the content blocks which appear in the list summary. *See Also:* ”Using CMS Explorer to Browse Your Ektron CMS400.NET Site” on page 10 The Recursive property determines whether content blocks in this folder’s child folders also appear.	String	Any folder
GetHtml	Set to **True** if you want to retrieve and display content (html body) for all content blocks in the list summary. For example, to display content inside a web server control such as a DataGrid.	Boolean	**True** = Get and display HTML for each content block in the list summary **False** = Do not get and display HTML.
Hide	Used to hide output of the list summary in design time and run time.	Boolean	**True** = Hide results **False** = show results

Property	Description	Data Type	Values
Language	Set a language for viewing the list summary. The language property shows results in design-time (in Visual Studio) and at run-time (in a browser).	Integer	Integer
MaxResults	Maximum number of summaries displayed on a Web page.	Integer	any integer **0** = unlimited
OrderKey	Sort the list by one of the values.	must be one of the values	Title DateModified DateCreated LastEditorFname LastEditorLname *See Also:* "OrderKey Property" on page 143
Random	Set to **True** if you want to randomly display one content block in the specified folder. The content changes each time a user views the page.	Boolean	True False
Recursive	Determines if the display includes content blocks in child folders of the selected folder.	Boolean	True False
WrapTag	Allows a developer to specify a server control's tag. The default is **Span**.	String	**Span** - The tag is used to designate an inline portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code. **Custom** - Allows you to use a custom tag.

OrderKey Property

The OrderKey property determines what information appears to the right of the hyperlinked title. The values are described below.

Value	Description	Web Page View
DateModified	Displays the date the content was last modified.	, RC Sportster. The RC Sportster is design mid level RC racer currently on the market nference *(8/27/2002 2:44:44 PM)* growing company in the RC host the 3rd annual RC conference in Atla
DateCreated	Displays the date when the content was first created.	d level RC racer currently on the mark erence *(4/25/2002 4:42:55 PM)* owing company in the RC ost the 3rd annual RC conference in A
LastEditorFname	Displays the first name of the user who last edited the content block.	mid level RC racer currentl nference *(John)* growing company in the R host the 3rd annual RC co
LastEditorLnames	Displays the last name of the user who last edited the content block.	ct, RC Sportster. The RC Sp d mid level RC racer currentl Conference *(Doe)* st growing company in the R to host the 3rd annual RC co

List Summary Example

Below is an example of the ListSummary Server Control properties.

Ektron	
Authenticated	False
ContentParameter	
ContentType	**Content**
Direction	**Ascending**
DisplayXslt	ecmNavigation
FolderPath	**\Marketing\News**
GetHtml	False
Hide	**False**
Language	**0**
MaxResults	0
OrderKey	Title
Random	**False**
Recursive	False
WrapTag	Span

The example has the following properties:

- Content Type - The results are a list of content blocks.
- Direction - The results are displayed in ascending order by title
- DisplayXslt - will use the navigation display format
- FolderPath - Displays summaries from the `\Marketing\News` folder
- GetHtml - Does not display the HTML from the content blocks in the list summary
- MaxResults - Unlimited results appear
- OrderKey - Summaries sorted by content block title
- Random - Displays summary for all content blocks in folder, not just one random one
- Recursive - Includes only selected folder, not its child folders

Retrieving the XML Structure of a List Summary

Retrieving the XML structure of XML content allows for greater control over developing XSLs. The following is an example of how to retrieve the XML structure:

1. Open a new Web form.
2. Drag and drop a ListSummary Server Control onto it.
3. Set the FolderPath property.
4. Drag and drop a Textbox on the Web form.
5. Set the TextMode property to MultiLine.

NOTE It is also recommended that you set the width of the text box to at least 400px.

6. On the code behind page, add the following line.

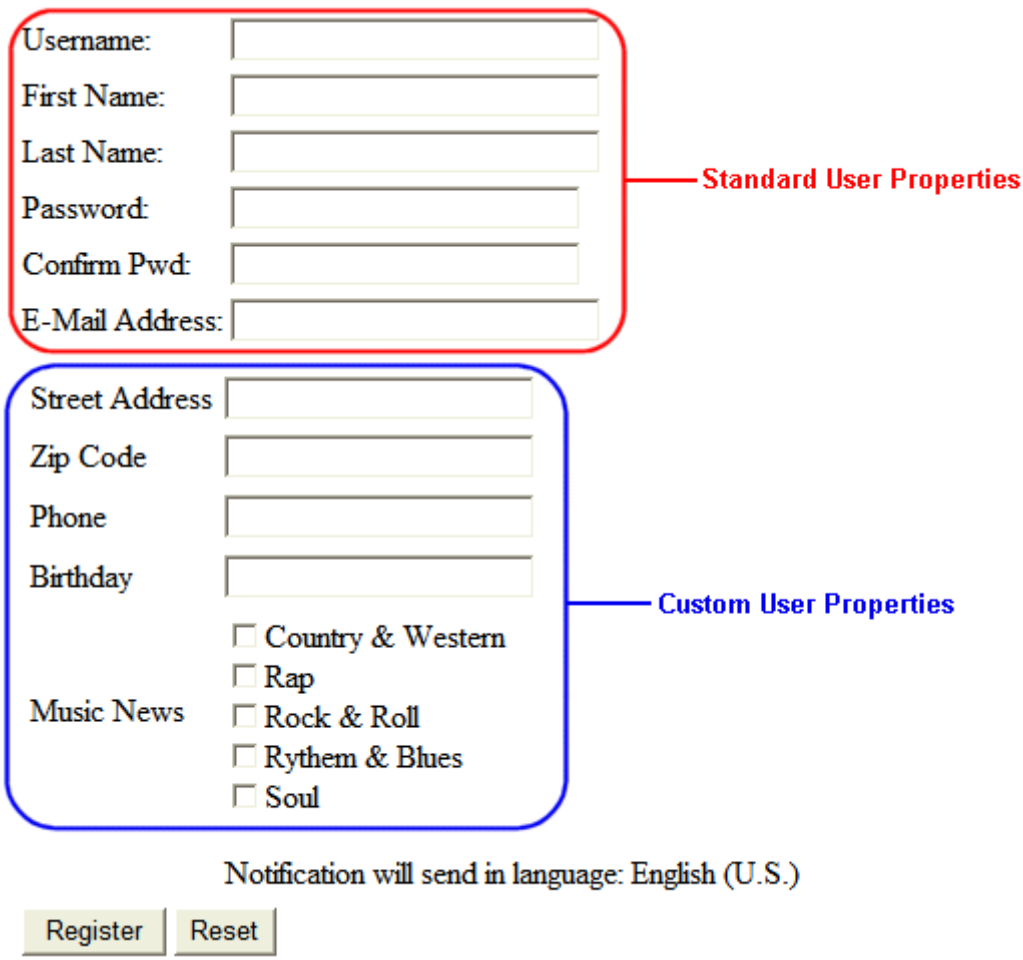
```
Textbox1.Text = ListSummary1.XmlDoc.InnerXml
```

7. Build the project.
8. View the Web form in a browser.
9. The XML structure of the list summary appears in the textbox.

Membership Server Control

The Membership Server Control creates a form on a Web page that allows a user to create or update a profile of themselves in your membership list. The control creates the form based on the standard Ektron user properties and any custom user properties that exist.

Below is an example of the Membership Server Control when viewed on a Web page.



When you create custom user properties, they appear for both membership users and **Ektron CMS400.NET** users. If you create a property such as Employee ID number, the property appears when a membership user views this form on a web page. The Membership Server Control allows you to decide if want the custom user properties to appear when the form is viewed.

It is recommended that you review the following documentation when using the Membership Server Control.

- The Administration Manual section “Managing Users & User Groups”
- The Administration Manual section “Custom User Properties”
- The Administration Manual section “Web Alert Feature”

Understanding these topics along with using this server control extends the most functionality to your membership users.

Membership Server Control Properties

The following table explains the properties of the Membership Server Control.

NOTE The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

Property	Value	Data Type
Authenticated	Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* ”Using CMS Explorer to Browse Your Ektron CMS400.NET Site” on page 10	String

Property	Value	Data Type
DisplayMode	The type of membership form displayed. Choose from the following. • **UserRegistration** - Lets a site visitor register as a membership user. • **ResetPassword** - Lets a membership user reset his password. • **UnsubscribeSecured** - Lets a membership user unsubscribe. He needs to enter his user-name and password. • **UnsubscribeUnsecured** - Lets a member-ship user unsubscribe by entering his mem-bership username only. • **AccountActivate** - Let a site visitor activate his membership account by entering the ID number in the account verification email. For more information, see ”DisplayMode” on page 151.	String
Hide	Used to hide login button in design time and run time. **True** = Hide login button **False** = Show login button	Boolean
Language	Set a language for the Server Control. The language property shows results in design-time (in Visual Studio) and at run-time (in a browser).	Integer
RedirectFailedURL	The URL where a membership user is sent if the registration fails. • If the page resides in the same folder as the registration page, enter the name of the page. For example, **RegFailed.aspx**. • If the redirect page is located in a subfolder, add the folder path. For example, **mem-bers\RegFailed.aspx**.	String

Property	Value	Data Type
RedirectSucessURL	The URL where a membership user is sent when the registration succeeds. • If the page resides in the same folder as the registration page, enter the name of the page. For example, **RegSucceed.aspx**. • If the redirect page is located in a subfolder, add the folder path. For example, **members\RegSucceed.aspx**.	String

RegisterButtonImg	Lets you add an image in place of the text on the register button. Enter a path to the image. For example:
	`http://www.ektron.com/buttons/registerbutton.gif`
	If the image is located in the site root, you do not need to enter the full path. Only enter the subfolder path and image name. For example:
	`/buttons/registerbutton.gif`
		String

RegisterButtonText	Text that appears on the Register button. The default is **Register**. If you use a register button image, you do not see this text.	String

ResetButtonImg	Lets you add an image in place of the text on the reset button. Enter a path to the image. For example:
	`http://www.ektron.com/buttons/resetbutton.gif`
	If the image is located in the site root, you do not need to enter the full path. Only enter the subfolder path and image name. For example:
	`/buttons/resetbutton.gif`
		String

ResetButtonText	Text that appears on the reset button. The default is **Reset**. If you use a reset button image, you do not see this text.	String

Property	Value	Data Type
ShowExtended	Decide if the Custom User Properties are available when using this control. The default setting is **True**. **True** = Show Custom User Properties **False** = Hide Custom User Properties **Note:** For more information on Custom User Properties, see the Administration Manual section "Custom User Properties".	String
UserExistsMessage	The message that appears when a membership user already exists. The default message is: **Username(email) already exists!**	String
UserSuccessMessage	The message that appears when a membership user successfully registers. The default message is: **You have registered successfully.**	String
UserUpdateSucessMessage	The message that appears when a membership user successfully updates their information. The default message is: **You have successfully updated your information.**	String
WrapTag	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code. **Custom** - Allows you to use a custom tag.	String

DisplayMode

Lets a developer decide what type of membership form is added to the Web form. The following table describes settings for the DisplayMode property.

Setting	Description	Example
UserRegistration	Lets a site visitor register as a membership user.	Username: First Name: Last Name: Password: Confirm Pwd: E-Mail Address: Register Reset
ResetPassword	Lets a membership user reset his password.	Username: Register Reset
UnsubscribeSecured	Lets a membership user unsubscribe. The membership user needs to enter his username and password to unsubscribe.	Username: Password: Register Reset
UnsubscribeUnsecured	Lets a membership user unsubscribe by entering his membership username only.	Username: Register Reset

Setting	Description	Example
AccountActivate	Let a site visitor activate his membership account by entering the ID number sent him in the account verification email.	Username: Account ID: Register Reset

Menu Server Control

The Menu Server Control calls a menu to be displayed on a page. Using the Menu Server Control, you can manipulate a menu by using the DisplayXslt property. Below is a menu display with the SampleMenu Xslt.

- Products
  - RC Cars
    - RC Cheetah
    - RC Sportster
  - RC Planes
    - RC Lilly
    - RC Redstar
  - Products Page
  - Visit Ektron.com

Before you can use the Menu Server Control, you must create a menu in the **Ektron CMS400.NET** work area. To learn about creating menus, see “Working with Menus” in the **Ektron CMS400.NET** User Manual.

This subsection contains the following topics:

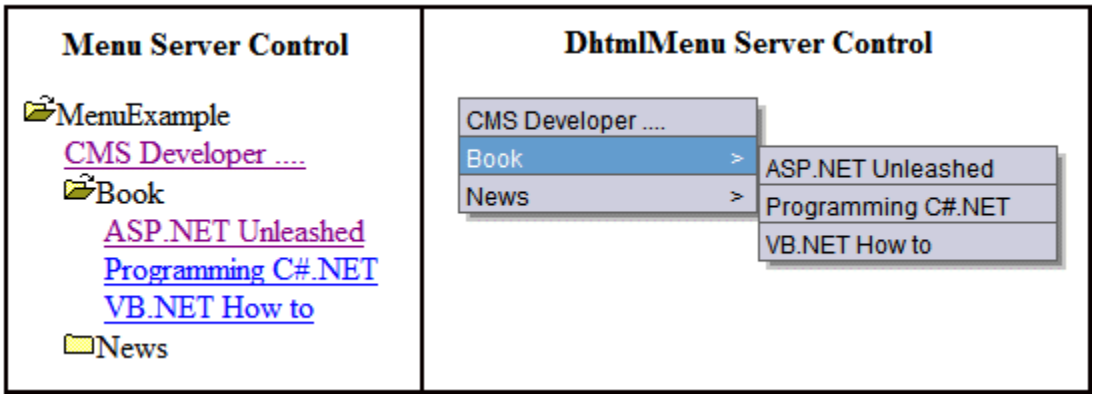
- ”Menu Server Control vs. DhtmlMenu Server Control” on page 153
- ”Menu Server Control Properties” on page 154
- ”Using DisplayXslt Samples” on page 156
- ”Retrieving the XML Structure of a Menu” on page 169

Menu Server Control vs. DhtmlMenu Server Control

The Menu Server Control calls a menu to be displayed on a page. You can manipulate the menu by using the DisplayXslt property.

With the DhtmlMenu Server Control, you display a dynamic menu with drop down boxes and submenus. You manipulate DhtmlMenus with a style sheet. DhtmlMenus save space on the Web page by having the drop down menus and submenus float over content.

WARNING! You cannot create a DHTML menu using the code behind -- you must drag and drop it. Once the menu is on the template, you can manipulate its properties using the code behind.



Menu Server Control Properties

The Menu Server Control properties are described in this table.

NOTE The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

Property	Value	Data Type
Authenticated	Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	String

Property	Value	Data Type
DefaultMenuID	The ID of a menu that appears where you insert this server control if no other menu is identified or available. If you don't know the ID number of the menu, use the CMS Explorer to browse to it. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	Integer
DisplayXslt	The Xslt to use to render the menu. **None** - databind only SampleMenu - A sample display, formatted as a bulleted menu list TreeMenu - A sample display, formatted as a folder tree. You can expand the tree by clicking on the folder icon. **Path to Custom Xslt** - Enter the path to an Xslt that determines the display of the page For more information on using the SampleMenu and TreeMenu DisplayXslt, see "Using DisplayXslt Samples" on page 156.	String
DynamicParameter	Select **menu_id**. When you do, this server control uses the menu passed as a URL parameter.	String
Hide	Used to hide a menu in design time and run time. **True** = Hide menu **False** = Show menu	Boolean
Language	Set a language for viewing menus. The language property shows results in design-time (in Visual Studio) and at run-time (in a browser).	Integer
WrapTag	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an inline portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code. **Custom** - Allows you to use a custom tag.	String

Using DisplayXslt Samples

Two DisplayXslt samples are provided with the Menu Server Control, SampleMenu and TreeMenu. This section explains how to use them.

SampleMenu

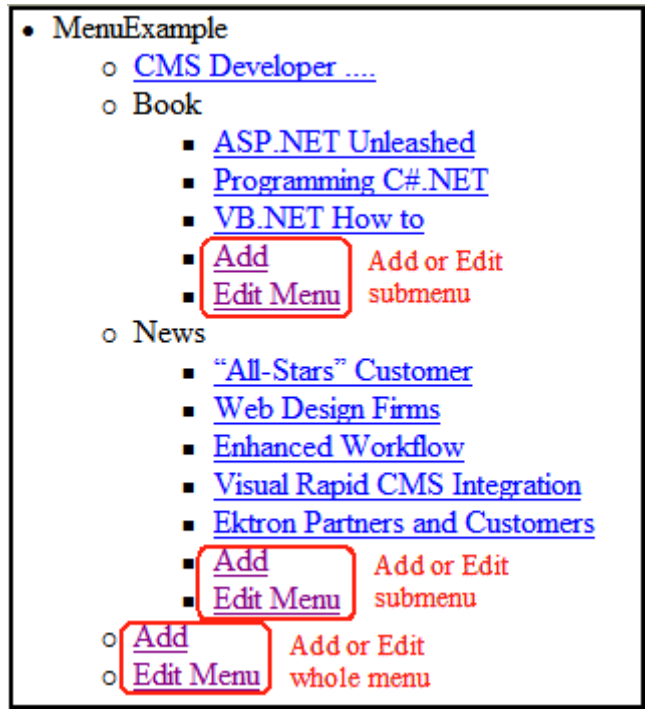
The SampleMenu DisplayXslt lets you display an **Ektron CMS400.NET** menu as a bulleted item list. See below for an example.

- MenuExample
  - CMS Developer ....
  - Book
    - ASP.NET Unleashed
    - Programming C#.NET
    - VB.NET How to
  - News
    - "All-Stars" Customer
    - Web Design Firms
    - Enhanced Workflow
    - Visual Rapid CMS Integration
    - Ektron Partners and Customers

When you are logged in to your **Ektron CMS400.NET** site, two menu items are added to each section of the menu: Add and Edit Menu. The user can use these to add a new menu item or edit an existing one. See example below.

For Information on a adding a menu Item, see **Ektron CMS400.NET** Users Manual > "Working with Menus" > "Managing Menus" > "Adding a New Content Block to a Menu via Navigation Link on a Web Page".

For Information on a editing a menu, see **Ektron CMS400.NET** Users Manual > "Working with Menus" > "Managing Menus" > "Editing a Menu via Navigation Link on a Web Page".



SampleMenu XSL code

Below is the XSL code that is used to create SampleMenu Display. You can use this code as the basis to design your own XSLT.

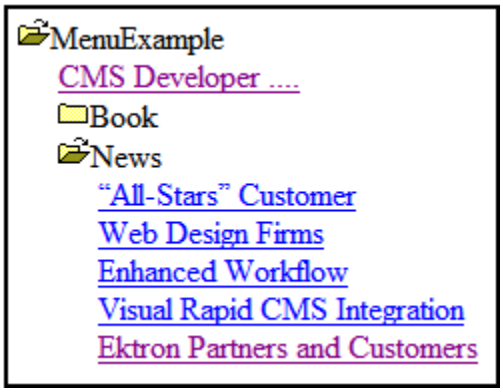
```
<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/
XSL/Transform">

 <xsl:output method="xml" version="1.0" encoding="UTF-8"
indent="yes" omit-xml-declaration="yes"/>

 <xsl:template match="/">

  <xsl:apply-templates select="MenuDataResult/Item"/>

 </xsl:template>

  <xsl:template match="Item[parent::MenuDataResult]">

  <xsl:call-template name="makelist"/>

 </xsl:template>

 <xsl:template match="Item[child::Menu]">
```

```
  <LI>

  <xsl:apply-templates select="Menu"/>

  </LI>

 </xsl:template>

 <xsl:template name="makelist" match="Menu">

  <xsl:choose>

  <xsl:whentest="Link='']><xsl:value-of select="Title"/></
xsl:when><xsl:otherwise>

  <A><xsl:attribute name="href"><xsl:value-of
select="concat('',Link)"/></xsl:attribute>

  <xsl:value-of select="Title"/></A></xsl:otherwise></xsl:choose>

  <UL><xsl:apply-templates select="Item"/></UL>

 </xsl:template>

 <xsl:template match="Item">

  <xsl:param name="pItemLink"><xsl:value-of select="ItemLink"/></
xsl:param>

  <xsl:choose><xsl:whentest="ItemLink=''">

  <LI>

  <xsl:value-of select="ItemTitle"/>

  <xsl:apply-templates select="Item|Menu"/></LI>

  </xsl:when>

  <xsl:otherwise>

  <LI><A>

  <xsl:choose>

   <xsl:whentest="ItemType='ExternalLink'">

      <xsl:attribute name="href"><xsl:value-of select="ItemLink"/></
xsl:attribute>

    <xsl:value-of select="ItemTitle"/>

   </xsl:when>

   <xsl:whentest="ItemType='Javascript'">
```

```
    <xsl:attribute name="onClick">Javascript:<xsl:value-of
select="ItemLink"/></xsl:attribute>

    <xsl:attribute name="href">#</xsl:attribute>

    <xsl:value-of select="ItemTitle"/>

   </xsl:when>

   <xsl:otherwise>

    <xsl:attribute name="href">

    <xsl:value-of select="concat('',ItemLink)"/>

    </xsl:attribute>

    <xsl:value-of select="ItemTitle"/>

   </xsl:otherwise>

  </xsl:choose></A><xsl:apply-templates select="Item|Menu"/></LI></
xsl:otherwise></xsl:choose>

 </xsl:template>

 <xsl:template name="maketitle"><xsl:choose>

  <xsl:whentest="ItemLink=''"><xsl:value-of select="Title"/></
xsl:when><xsl:otherwise><A>

  <xsl:attribute name="href"><xsl:value-of
select="concat('',Link)"/></xsl:attribute>

  <xsl:value-of select="Title"/></A></xsl:otherwise></xsl:choose>

  <UL><xsl:apply-templates select="Item"/></UL>

 </xsl:template>

 <!-- identity template -->

 <xsl:template match="@*|node()">

  <xsl:copy><xsl:apply-templates select="@*|node()"/></xsl:copy>

 </xsl:template>

</xsl:stylesheet>
```

TreeMenu

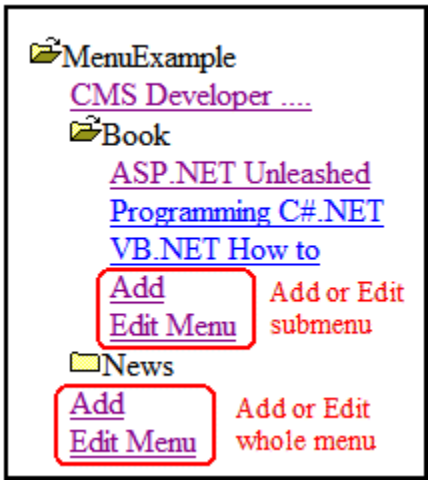
The TreeMenu DisplayXslt lets you display an **Ektron CMS400.NET** menu as a clickable folder list. You can expand and collapse the menu by clicking on the folder icons. See below for an example.



When you are logged in to your **Ektron CMS400.NET** site, two menu items are added to each section of the menu: Add and Edit Menu. The user can use these to add a new menu item, or edit an existing menu item. See example below.

For Information on a adding a menu Item, see **Ektron CMS400.NET** Users Manual > "Working with Menus" > "Managing Menus" > "Adding a New Content Block to a Menu via Navigation Link on a Web Page".

For Information on a editing a menu, see **Ektron CMS400.NET** Users Manual > "Working with Menus" > "Managing Menus" > "Editing a Menu via Navigation Link on a Web Page".



NOTE When using the TreeMenu Server Control with Visual Studio and Windows 2003, the folder images do not display during design-time. They do, however, display correctly during run-time.

TreeMenu XSL code

Below is the XSL code that is used to create TreeMenu Display. You can use this code as the basis to design your own XSLT.

```
<?xml version="1.0" encoding="ISO-8859-1"?>

    <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/
1999/XSL/Transform">

    <xsl:output method= "xml" version="1.0" encoding="UTF-8"
indent="yes" omit-xml-declaration= "yes"/>

<!-- Change sitepath to match you environment -->

<xsl:variable name="sitepath">/CMS400Example/</xsl:variable>

<!--

menuId must be different for each menu on the page.

This does mean that you need to use a different copy

of this file (each with their own menuId) for each

menu that you will display.
```

```
-->

<xsl:variable name="menuId">Menu1</xsl:variable>

<xsl:template match="/">

<xsl:apply-templates select="MenuDataResult/Item"/>

</xsl:template>

<xsl:template match="Item[parent::MenuDataResult]">

<xsl:apply-templates select="Item/Menu"/>

</xsl:template>

<xsl:template match="Item[child::Menu]">

<xsl:apply-templates select="Menu"/>

</xsl:template>

<xsl:template match="Menu">

<xsl:choose>

<xsl:whentest="Link='">

<span>

<xsl:attribute name="class">handcur</xsl:attribute>

<xsl:attribute name="onclick">showBranch('<xsl:value-of
select="$menuId" />_<xsl:value-of select="ID"/>')</xsl:attribute>

<img>

<xsl:attribute name="src"><xsl:value-of select="$sitepath" />
>workarea/images/application/ico_menu-closed.gif</xsl:attribute>

<xsl:attribute name="id">I<xsl:value-of select="$menuId" />
>_<xsl:value-of select="ID"/></xsl:attribute>

</img>

<xsl:choose>

<xsl:whentest="ImageOverride='True'">

<img>

<xsl:attribute name="border">0</xsl:attribute>

<xsl:attribute name="src"><xsl:value-of select="Image"/></
xsl:attribute>

</img>
```

```
</xsl:when>

<xsl:otherwise>

<xsl:iftest="not(Image='')">

<img>

<xsl:attribute name="border">0</xsl:attribute>

<xsl:attribute name="src"><xsl:value-of select="Image"/></
xsl:attribute>

</img><xsl:text> </xsl:text>

</xsl:if>

<xsl:value-of select="Title"/>

</xsl:otherwise>

</xsl:choose>

</span>

<br/>

</xsl:when>

<xsl:when test="not(Link)">

</xsl:when>

<xsl:otherwise>

<span>

<xsl:attribute name="class">handcur</xsl:attribute>

<xsl:attribute name="onclick">showBranch('<xsl:value-of
select="$menuId" />_<xsl:value-of select="ID"/>')</xsl:attribute>

<img>

<xsl:attribute name="src"><xsl:value-of select="$sitepath" /
>Workarea/images/application/ico_menu-closed.gif</xsl:attribute>

<xsl:attribute name="id">I<xsl:value-of select="$menuId" /
>_<xsl:value-of select="ID"/></xsl:attribute>

</img>

<xsl:choose>

<xsl:whentest="ImageOverride='True'">
```

```
<A><xsl:attribute name="href"><xsl:value-of
select="concat('',Link)"/></xsl:attribute>

<img>

<xsl:attribute name="border">0</xsl:attribute>

<xsl:attribute name="src"><xsl:value-of select="Image"/></
xsl:attribute>

</img>

</A>

</xsl:when>

<xsl:otherwise>

<xsl:iftest="not(Image='')">

<A><xsl:attribute name="href"><xsl:value-of
select="concat('',Link)"/></xsl:attribute>

<img>

<xsl:attribute name="border">0</xsl:attribute>

<xsl:attribute name="src"><xsl:value-of select="Image"/></
xsl:attribute>

</img>

</A><xsl:text> </xsl:text>

</xsl:if>

<A><xsl:attribute name="href"><xsl:value-of
select="concat('',Link)"/></xsl:attribute>

<xsl:value-of select="Title"/>

</A>

</xsl:otherwise>

</xsl:choose>

</span>

<br/>

</xsl:otherwise>

</xsl:choose>

<span class="branch">
```

```
<xsl:attribute name="id"><xsl:value-of select="$menuId" /
>_<xsl:value-of select="ID"/></xsl:attribute>

<xsl:apply-templates select="Item"/>

</span>

</xsl:template>

<xsl:template match="Item">

<xsl:choose>

<xsl:whentest="ItemLink='">

<span class="handcur">

<xsl:choose>

<xsl:whentest="ItemImageOverride='True'">

<img>

<xsl:attribute name="border">0</xsl:attribute>

<xsl:attribute name="src"><xsl:value-of select="ItemImage"/></
xsl:attribute>

</img>

</xsl:when>

<xsl:otherwise>

<xsl:iftest="not(ItemImage='')">

<img>

<xsl:attribute name="border">0</xsl:attribute>

<xsl:attribute name="src"><xsl:value-of select="ItemImage"/></
xsl:attribute>

</img><xsl:text> </xsl:text>

</xsl:if>

<xsl:value-of select="ItemTitle"/>

</xsl:otherwise>

</xsl:choose>

</span><br/>

<xsl:apply-templates select="Item|Menu"/>
```

```
</xsl:when>

<xsl:otherwise>

<span>

<xsl:attribute name="class">handcur</xsl:attribute>

<xsl:choose>

<xsl:whentest="ItemType='ExternalLink'">

<xsl:choose>

<xsl:whentest="ItemImageOverride='True'">

<A><xsl:attribute name="href"><xsl:value-of select="ItemLink"/></
xsl:attribute>

<img>

<xsl:attribute name="border">0</xsl:attribute>

<xsl:attribute name="src"><xsl:value-of select="ItemImage"/></
xsl:attribute>

</img>

</A>

</xsl:when>

<xsl:otherwise>

<xsl:iftest="not(ItemImage='')">

<A><xsl:attribute name="href"><xsl:value-of select="ItemLink"/></
xsl:attribute>

<img>

<xsl:attribute name="border">0</xsl:attribute>

<xsl:attribute name="src"><xsl:value-of select="ItemImage"/></
xsl:attribute>

</img>

</A><xsl:text> </xsl:text>

</xsl:if>

<A><xsl:attribute name="href"><xsl:value-of select="ItemLink"/></
xsl:attribute>

<xsl:value-of select="ItemTitle"/>
```

```
</A>

</xsl:otherwise>

</xsl:choose>

</xsl:when>

<xsl:whentest="ItemType='Javascript'">

<xsl:choose>

<xsl:whentest="ItemImageOverride='True'">

<A><xsl:attribute name="onClick">Javascript:<xsl:value-of
select="ItemLink"/></xsl:attribute><xsl:attribute name="href">#</
xsl:attribute>

<img>

<xsl:attribute name="border">0</xsl:attribute>

<xsl:attribute name="src"><xsl:value-of select="ItemImage"/></
xsl:attribute>

</img>

</A>

</xsl:when>

<xsl:otherwise>

<xsl:iftest="not(ItemImage='')">

<A><xsl:attribute name="onClick">Javascript:<xsl:value-of
select="ItemLink"/></xsl:attribute><xsl:attribute name="href">#</
xsl:attribute>

<img>

<xsl:attribute name="border">0</xsl:attribute>

<xsl:attribute name="src"><xsl:value-of select="ItemImage"/></
xsl:attribute>

</img>

</A><xsl:text> </xsl:text>

</xsl:if>

<A><xsl:attribute name="onClick">Javascript:<xsl:value-of
select="ItemLink"/></xsl:attribute><xsl:attribute name="href">#</
xsl:attribute>

<xsl:value-of select="ItemTitle"/>
```

```
</A>

</xsl:otherwise>

</xsl:choose>

</xsl:when>

<xsl:otherwise>

<xsl:choose>

<xsl:whentest="ItemImageOverride='True'">

<A><xsl:attribute name="href"><xsl:value-of
select="concat('',ItemLink)"/></xsl:attribute>

<img>

<xsl:attribute name="border">0</xsl:attribute>

<xsl:attribute name="src"><xsl:value-of select="ItemImage"/></
xsl:attribute>

</img>

</A>

</xsl:when>

<xsl:otherwise>

<xsl:iftest="not(ItemImage='')">

<A><xsl:attribute name="href"><xsl:value-of
select="concat('',ItemLink)"/></xsl:attribute>

<img>

<xsl:attribute name="border">0</xsl:attribute>

<xsl:attribute name="src"><xsl:value-of select="ItemImage"/></
xsl:attribute>

</img>

</A><xsl:text> </xsl:text>

</xsl:if>

<A><xsl:attribute name="href"><xsl:value-of
select="concat('',ItemLink)"/></xsl:attribute>

<xsl:value-of select="ItemTitle"/>

</A>
```

```
</xsl:otherwise>

</xsl:choose>

</xsl:otherwise>

</xsl:choose></span><br/>

<xsl:apply-templates select="Item|Menu"/>

</xsl:otherwise>

</xsl:choose>

</xsl:template>

<!-- identity template -->

<xsl:template match="@*|node()">

<xsl:copy><xsl:apply-templates select="@*|node()"/></xsl:copy>

</xsl:template>

</xsl:stylesheet>
```

Retrieving the XML Structure of a Menu

Retrieving the XML structure of XML content allows for greater control over developing XSLs. The following is an example of how to retrieve the XML structure:

1. Open a new Web form.
2. Drag and drop a Menu Server Control onto it.
3. Set the DefaultMenuID properties.
4. Drag and drop a Textbox on the Web form.
5. Set the TextMode property to MultiLine.

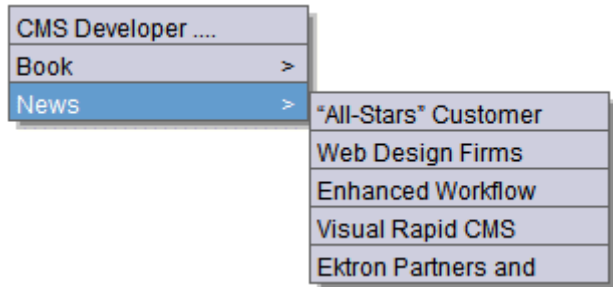
NOTE It is also recommended that you set the width of the text box to at least 400px.

6. On the code behind page, add the following line.
   `Textbox1.Text = Menu1.XmlDoc.InnerXml`
7. Build the project.
8. View the Web form in a browser.
9. The XML structure of the menu appears in the textbox.

DhtmlMenu Server Control

NOTE For optimal performance, Ektron recommends using at minimum Internet Explorer 5.5, Netscape 6.0, or Mozilla Firefox 1.0.

The DhtmlMenu Server Control lets you display a menu with drop down boxes and submenus. This lets developers add a menu to a page without taking up a lot space. You can modify the menu's style via a style sheet.

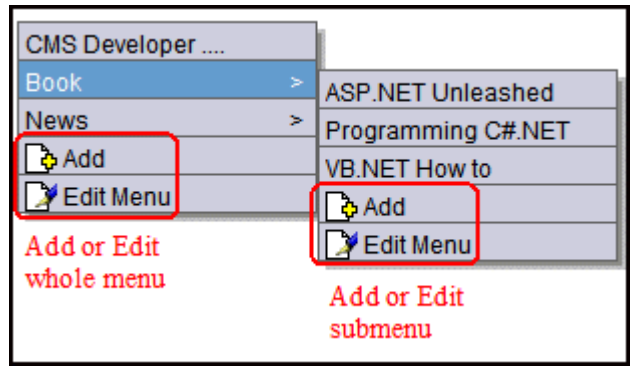


Before using the DhtmlMenu Server Control, you must add items to a menu in the Ektron CMS400.NET workarea. To learn about creating menus, see "Working with Menus" in the CMS400.NET User Manual.

The DhtmlMenu Server Control also lets developers add content to a menu. When you are logged in to your **Ektron CMS400.NET** site, two menu items are added to each menu: Add and Edit Menu. The user can use these to add a new menu item or edit an existing menu item. See example below.

For Information on a adding a menu Item, see **Ektron CMS400.NET** Users Manual > "Working with Menus" > "Managing Menus" > "Adding a New Content Block to a Menu via Navigation Link on a Web Page".

For Information on a editing a menu, see **Ektron CMS400.NET** Users Manual > "Working with Menus" > "Managing Menus" > "Editing a Menu via Navigation Link on a Web Page".



This subsection contains the following topics:

- "Menu Server Control vs. DhtmlMenu Server Control" on page 171
- "DhtmlMenu Server Control Properties" on page 171
- "Using a Style Sheet" on page 175

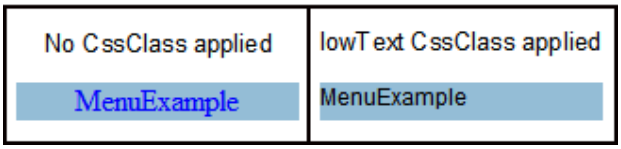
Menu Server Control vs. DhtmlMenu Server Control

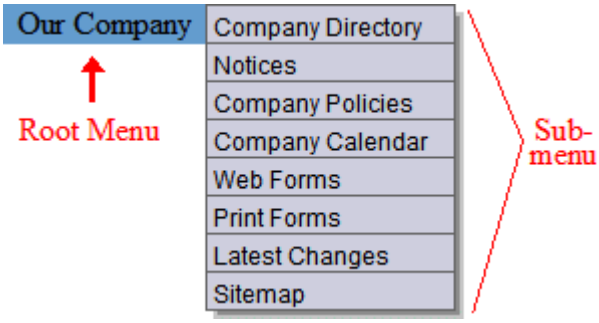
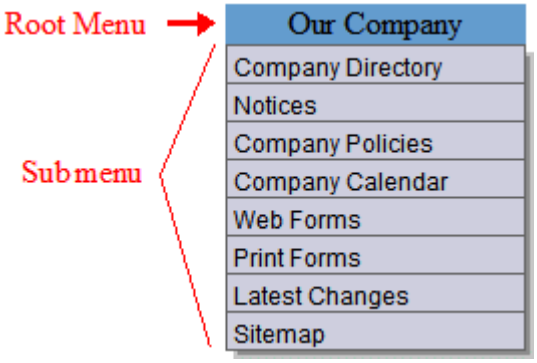
See "Menu Server Control vs. DhtmlMenu Server Control" on page 153.

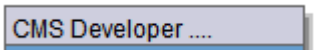
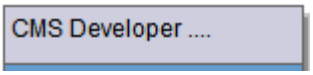
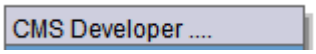
DhtmlMenu Server Control Properties

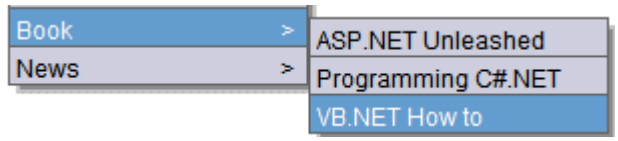
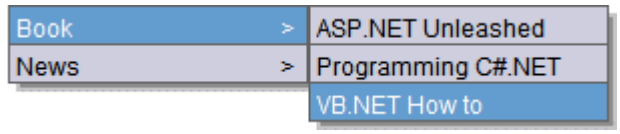
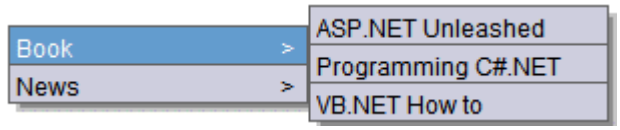
The DhtmlMenu Server Control properties are described in this table.

NOTE The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

Property	Value	Data Type
Align	Specifies the horizontal alignment of the root menu text. The default is **Center.** **NotSet** - Use the default setting. **Left** - Aligns text to the left. **Center** - Aligns text to the center. **Right** - Aligns text to the right. **Justify** - Aligns text justified to both margins.	String
Authenticated	Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc.... *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	String
BackColor	Sets the background color for the root menu. This property appears in the **Appearance** category.	String
CssClass	Apply a .css class to the root menu. Enter the class name from the .css defined in the Stylesheet property. *See Also:* "Stylesheet" on page 174 Since pop_style.css is the default style sheet, you can use its classes without defining it in the Stylesheet property. For example, if you enter lowtext in the property, the root menu changes as shown below.  	String
DefaultMenuID	The ID of a menu that appears where you inserted this server control if no other content block is identified, or is not available. If you don't know the ID number of the menu, use the CMS Explorer to browse to it. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	Integer
DynamicParameter	Select **menu_id**. When you do, this server control uses the menu passed as a URL parameter.	String

Property	Value	Data Type
ForeColor	Sets the color of the font for the root menu. This property appears in the **Appearance** category.	
Hide	Used to hide a menu in design time and run time. **True** = Hide menu **False** = Show menu	Boolean
HorizontalDisplay	Displays submenu to the right of the root menu. This only affects the first submenu of the root menu. All subsequent submenus appear to the right of each submenu. The default is **False** **True** = Submenu appears to the right of the root menu  **False** = Submenu appears below the root menu  	Boolean
Language	Set a language for viewing Dhtml Menus. The language property shows results in design-time (in Visual Studio) and at run-time (in a browser).	Integer

Property	Value	Data Type
MenuHoverOverColor	Sets the background color for the root menu when hovering over it.	String
MenuStyle	A specified inline style that defines the root menu. For example, if you enter the following in the MenuStyle property box: `font: bold 18px Arial;color:royalblue; background-color:slateblue` 	String
Stylesheet	Specify the location and style sheet for the submenu. For example: `\Workarea\csslib\mytest.css.` Leave blank to use the default. `pop_style.css` is the default style sheet. It is located in `\webroot\CMS400Example\Workarea\csslib`. For more information on style sheets, see "Using a Style Sheet" on page 175 The Stylesheet property can be used in conjunction with the CssClass property to define the root menu. *See Also:* "CssClass" on page 172	String
SubMenuBackColor	Sets the background color for the submenu.	String
SubMenuHoverOverColor	Sets the background color for the submenu when hovering over it.	String
SubMenuItemHeight	Sets the height of each submenu item in pixels. The Default is 20 Example set to 20:  Example set to 30: 	Integer
SubMenuItemWidth	Sets the width of each submenu item in pixels. The Default is 150 Example set to 150:  Example set to 130:	Integer

Property	Value	Data Type
SubMenuTopOffset	Sets the location of the top of the submenu. This number is in pixels. The default is 5 Example set to 5:  Example set to 0 (zero):  Example set to -10:  	Integer
WrapTag	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an inline portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code. **Custom** - Allows you to use a custom tag.	String

Using a Style Sheet

You can create your own style sheet, or modify the existing default one, `pop_style.css`. It is located in `\webroot\CMS400Example\Workarea\csslib`.

Style Sheets let you choose different text colors, text alignment, text fonts, and text size to create a customized menu.

> Best Practice
>
> Ektron recommends copying and renaming the default style sheet. This insures you always have a clean file to start with. This also gives you something to reference if you are not getting the results you expect.

The style sheet must be located in the site root.

You can only define one menu style sheet for each Web page. If you have more than one menu on a single Web page, they share the style sheet.

The following is a list of style sheet properties that can be modified in the `pop_style.css`.

Style Class	Description	Attribute	Default Value
.lowText	Text of the floating menu as displayed on the menu.	• color • font • text-align • text-decoration	• #000000 (Black) • 12px Arial, sans-serif • left • none
.highText	Text of the floating menu when highlighted.	• color • font • text-align • text-decoration	• #000000 (Black) • 12px Arial, sans-serif • left • none

Style Class	Description	Attribute	Default Value
.roothighText Note: This class only works when called by the CssClass property. Use the Font property in the Appearance category to define root menu text.	Text of the root menu as displayed on a Web page.	• color • font • text-align • text-decoration	• #000000 (Black) • 16px Arial, sans-serif • center • none
.itemBorder	Border color and size of the menu.	• border	• 1px solid #666666
.itemBorderBlank	Separator border color and size of the menu.	• border	• 1px solid #CCCCDD

For more information on style sheets, visit http://www.w3.org/Style/CSS/.

MetaData Server Control

The Metadata Server Control lets you add the metadata from content blocks to a Web page. This lets developers add metadata quickly without having to type it in. You can add metadata from a single content block, multiple content blocks, or by dynamically passing a content ID from a URL.

This subsection contains the following topics:

- ”MetaData Server Control vs. MetaDataList Server Control” on page 178
- ”MetaData Server Control Properties” on page 178
- ”Using the Simple Dublin Core Metadata Standard” on page 180
- ”Using the MetaData Server Control” on page 181

MetaData Server Control vs. MetaDataList Server Control

With the MetaData Server Control, you add metadata from content blocks to your Web page. With the MetaDataList Server Control, you create a list of content blocks to display on your site, based on the Metadata in each content block. For Information on the MetaDataList Server Control, see ”MetaDataList Server Control” on page 183.

MetaData Server Control Properties

The Metadata Server Control properties are described in this table.

NOTE The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

Property	Value	Data Type
Authenticated	Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	String
DefaultContentID	The ID of the content block that the server control will get the metadata from. If you want to add metadata from several content blocks, set this property to zero (0) and use the DefaultItemList property to identify the content blocks. If you don't know the ID number of the content block, use the CMS Explorer to browse to it. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	Integer
DefaultItemList	A comma delimited list of content block IDs to get metadata from. DefaultContentID must be set to zero (0) so you can add IDs to the list. DefaultItemList [12,4,7]	Integer
DynamicParameter	Gets or sets the QueryString parameter to read a content ID dynamically.	String
GenerateDublinCore	When enabled, this property automatically creates seven of the Simple Dublin Core metadata fields from standard CMS400.NET system properties. The default is false. **True** = Generate Simple Dublin Core metadata fields **False** = Do not generate Simple Dublin Core metadata fields The seven fields and how they are associated with the CMS400.NET properties is explained in "Using the Simple Dublin Core Metadata Standard" on page 180	Boolean
Hide	Used to hide a Metadata Server Control in design time and run time. **True** = Hide Metadata Server Control **False** = Show Metadata Server Control	boolean

Using the Simple Dublin Core Metadata Standard

Simple Dublin Core is a set of fifteen standard names for metadata fields designed to cover the most useful items of information on a document. From the Dublin Core FAQ: Dublin Core metadata provides card catalog-like definitions for defining the properties of objects for Web-based resource discovery systems.

For more information on the Dublin Core standard refer to the Usage Guide: http://www.dublincore.org/documents/usageguide/

To generate Dublin Core metadata, set the GenerateDublinCore property to True. This creates seven of the fifteen Dublin Core metadata fields. These fields are automatically filled with the information from the equivalent Ektron CMS400.NET property. Below is a list of the seven fields that are created and their Ektron CMS400.NET equivalent property.

IMPORTANT! These seven Dublin Core fields are automatically populated with information from their equivalent Ektron CMS400.NET property.

Dublin Core Field Name	Ektron CMS400.NET Property
DC.title	Content block title
DC.description	Plain text version of a content block teaser (summary)
DC.contributor	Content block last editor name
DC.date	Content block last edit date
DC.format	"text/html"
DC.identifier	URL of current page (from ASP.NET Server.Request object)
DC.language	CMS language cookie / current site language, expressed as a .NET System.Globalization Culture Name

To fully comply with the Simple Dublin Core metadata element set, the administrator must create the remaining eight Dublin Core

fields as standard CMS400.NET Metadata fields and apply them to all CMS400.NET folders. Next, the CMS users must fill in the appropriate values for each content block.

For information on the remaining eight Simple Dublin Core fields, see the Administrator Manual section “Managing Content Folders” > “Metadata” > “Simple Dublin Core Metadata”.

Using the MetaData Server Control

These steps show how to use the MetaData Server Control.

1. Drag a MetaData Server Control into a template.
2. Set the properties of the Metadata Server Control. This will create the following HTML in the HTML body.

```
        <cms:MetaData id="MetaData1" runat="server"
DefaultContentID="12"></cms:MetaData>
```

   Or if you are using multiple content block IDs In the DefaultItemList, the following HTML will be created.

```
        <cms:metadata id="MetaData1" runat="server"
DefaultItemList="[12,7,4]"></cms:metadata>
```

3. Click on the HTML tab and copy that line from the <body> tag into the <head> tag.

```
<HEAD>
    <title>WebForm5</title>
    <cms:MetaData id="Metadata2" runat="server" DefaultContentID="12"></cms:MetaData>
    <meta name="GENERATOR" content="Microsoft Visual Studio .NET 7.1">
    <meta name="CODE_LANGUAGE" content="Visual Basic .NET 7.1">
    <meta name="vs_defaultClientScript" content="JavaScript">
    <meta name="vs_targetSchema" content="http://schemas.microsoft.com/intellisense/i
</HEAD>
<body>
    <form id="Form1" method="post" runat="server">
    </form>
</body>
```

4. Save the Web form and rebuild the solution.
5. Open the Web page in the browser.
6. Right click on the Web page and click **View Source**. Look in the head tag. The meta tags from the content block are added to the page.

The metadata information added is shown below.

WebForm5[1] - Notepad

File Edit Format View Help

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
  <HEAD>
    <title>WebForm5</title>
    <title>CMS Developer</title>

    <meta name="keywords" content="CMS400; .NET; CMS">
    <meta name="description" content="CMS400 developer section">

    <meta content="Microsoft Visual Studio .NET 7.1" name="GENERATOR">
    <meta content="Visual Basic .NET 7.1" name="CODE_LANGUAGE">
    <meta content="JavaScript" name="vs_defaultClientScript">
    <meta content="http://schemas.microsoft.com/intellisense/ie5"
name="vs_targetSchema">
  </HEAD>
```

MetaDataList Server Control

Use the MetaDataList Server Control to create lists based on Keyword Names and Keyword Values contained within the metadata of a content block. In the example below the KeyWordNames are circled in red and the KeyWordValues are circled in green.

The list can display the information as a list of hyperlinks. You can choose, based on properties you set, if you would like to have a summary displayed. You can also choose how to order the display. For information on adding Metadata to a content block, see the Users Manual > “Adding or Editing Metadata”.

This subsection contains the following topics:

- ”MetaData Server Control vs. MetaDataList Server Control” on page 184
- ”MetaDataList Server Control Properties” on page 184
- ”Retrieving the XML Structure of a MetadataList” on page 187

MetaData Server Control vs. MetaDataList Server Control

With the MetaData Server Control, you add metadata from content blocks to your Web page. With the MetaDataList Server Control, you create a list of content blocks to display on your site based on the Metadata in each content block. For Information on the MetaData Server Control, see "MetaData Server Control" on page 178

MetaDataList Server Control Properties

The MetaDataList Server Control properties are described in this table.

NOTE The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

Property	Value	Data Type
Authenticated	Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	String
ContentType	One of the following • all types • archived content • archived forms • content • forms To learn about archived content, see the **Ektron CMS400.NET** User Manual, "Scheduling Content to Begin and End" > "Setting Archive Options."	String

Property	Value	Data Type
DisplayXslt	Determines how the information on the page is displayed **None**-databind only **ecmNavigation** - lists the title of every content block in the folder *See Also:* "Example of ecmNavigation Display" on page 105 **ecmTeaser** - lists the title of every content block in the folder plus the content summary. *See Also:* "Example of ecmTeaser Display" on page 106 **Path to Custom Xslt** - Enter the path to an Xslt that determines the display of the page	String
ExactPhrase	Determines whether the KeyWordValue needs to match the phrase exactly. For example, if "site" is the KeyWordValue and the title of a content block is "Welcome to the site" and ExactPhrase is set to true, you would not see the content block in the metadata list. This is because "site" does not equal "Welcome to the site". **True** = Match the exact phrase **False** = Doesn't need to match exact phrase	Boolean
FolderID	The root folder content will be pulled from.	Integer
GetHtml	Set to **True** if you want to retrieve and display content (html body) for all content blocks in the collection. For example, display content inside a web server control such as a DataGrid.	Boolean
Hide	Used to hide a metadata list in design time and run time. **True** = Hide metadata list **False** = Show metadata list	Boolean

Property	Value	Data Type
KeyWordName	The KeyWordName is the container where the KeyWordValues are located. Some examples of a KeyWordName might be “Keywords or Title” For information on creating KeyWordNames for Metadata, see the **Ektron CMS400.NET** Administrators Manual > “Managing Content” > “Metadata” > “Managing Metadata Definitions” > “Adding a Metadata Definitions”	String
KeyWordValue	The value that is associated with the KeyWordName Some examples of a KeyWordValue might be “Home; page; or company”	String
Language	Set a language for viewing the MetaDataList. The language property shows results in design-time (in Visual Studio) and at run-time (in a browser).	Integer
MaxNumber	The maximum number of results returned for display. **0** = unlimited	Integer
OrderBy	The order of the list to be returned. • **Title** - The title of the content block • **ID** - The content block ID number • **Date Created** - The date the content block was created • **Date Modified** - The date the content block was last modified • **LastEditorLname** - The last editor’s last name • **LastEditorFname** - The last editor’s first name	String
Recursive	Whether to search sub-folders of the identified root folder. The root folder is identified in the Folderpath property.	Boolean
SortOrder	Ascending or Descending	String

Property	Value	Data Type
WrapTag	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an inline portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code. **Custom** - Allows you to use a custom tag.	String

Retrieving the XML Structure of a MetadataList

Retrieving the XML structure of XML content allows for greater control over developing XSLs. The following is an example of how to retrieve the XML structure:

1. Open a new Web form.
2. Drag and drop a MetadataList Server Control onto it.
3. Set the KeyWordName and KeyWordValue properties.
4. Drag and drop a Textbox on the Web form.
5. Set the TextMode property to MultiLine.

NOTE It is also recommended that you set the width of the text box to at least 400px.

6. On the code behind page, add the following line.

```
Textbox1.Text = Metadata1.XmlDoc.InnerXml
```

7. Build the project.
8. View the Web form in a browser.
9. The XML structure of the MetadataList appears in the textbox.

Login Server Control

The Login Server Control paints a login button on the template when displayed in a browser.

When the Login Server Control is inserted and the project is built, the control displays the following buttons on a Web page.

Button	Description
Login	When user is not logged in, this button appears. Clicking the button opens the login window, where a user can enter a username and password. Upon authentication, the user is logged in to the Ektron CMS400 Web site.
Logout	After a user logs in, this button replaces the login button to let the user log out.
Workarea	When logged in, this button appears under the logout button, allowing the user to access the Workarea.
Preview	Lets the user preview the entire Web site as if all checked-in content were published. For more information, see the **Ektron CMS400.NET** User Manual section “Logging In and Out” > “Site Preview.”
Preview	Turns off site preview mode.
Help	Launches online, context sensitive help for **Ektron CMS400.NET**.

This subsection contains the following topics:

- ”Placing a Login Button” on page 189
- ”Login Server Control Properties” on page 189

Placing a Login Button

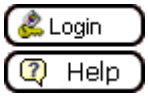
You can add any number of login buttons to a template. You can insert a login button on each template, or set up a special Web page, called login.aspx, from which users can log into the CMS400 site without the public being able to access the page.

Login Server Control Properties

The following table explains the properties of the Login Server Control.

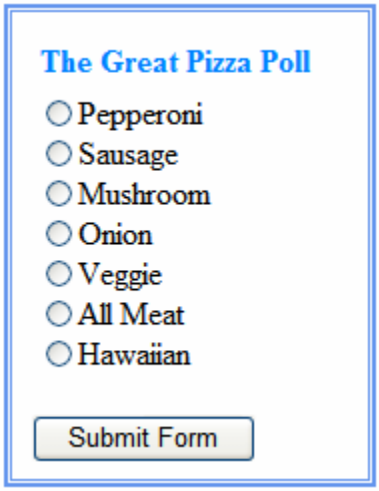
NOTE The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

Property	Value	Data Type
Authenticated	Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	String
AutoLogin	When set to true, users are automatically logged in using Active Directory authentication. They do not need to enter a username or password. The default is False. **True** = Use Active Directory authentication when logging in. **False** = Do not use Active Directory authentication when logging in. **Important!**: For this property to function properly, you must be using Active Directory authentication with your CMS400.NET Site.	Boolean
Hide	Used to hide login button in design time and run time. **True** = Hide login button **False** = Show login button	Boolean

Property	Value	Data Type
Language	Set a language for the Login Server Control. The language property shows results in design-time (in Visual Studio) and at run-time (in a browser).	Integer
OnlyAllowMemberLogin	This changes the Login button so only membership users can log in. This prevents someone from trying to login as an Ektron CMS400.NET user and gain access to the Ektron CMS400.NET Workarea. The default is False. **True** = Allow only membership users to login. **False** = Allow Ektron CMS400.NET users and membership users to login.	Boolean
SuppressHelpButton	Hides the Help button that appears below the Login button when set to true. The default is False. **True** = Do not display the Help button. **False** = Display Help button. 	Boolean
WrapTag	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code. **Custom** - Allows you to use a custom tag.	String

Poll Server Control

The Poll Server Control displays a poll or survey created from a form on a Web page. When added to a template and visited, the poll might look like this. You can change the way a poll is displayed by modifying its properties.



While you can use a FormBlock Server Control to add a poll or survey, it is recommended that you use the Poll Server Control. Using the Poll Server Control allows you greater flexibility on how the poll appears on your Web page.

Typically developers or site administrators want a poll or survey to be a small part of a Web page not the main content. They also want to display the results in the same area as the poll or survey without disturbing the surrounding content. The following properties allow you to do this:

- IsolatedDisplay - displays the poll in an iFrame on the web page
- Scale - changes the display size of the poll
- Border - creates a border around the poll to add emphasis or further isolate the poll

For a detailed description of these properties, click on them or see ”Poll Server Control Properties” on page 192.

This subsection contains the following topics:

- ”Poll Server Control Properties” on page 192
- ”Automatic versus Manual Generation of Form Tags” on page 195

Poll Server Control Properties

The following table explains the properties of the Poll Server Control.

NOTE The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

Property	Value	Data Type
AddValidation	Set to **True** to add Validation to your main runat=server form.	Boolean
Authenticated	Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* ”Using CMS Explorer to Browse Your Ektron CMS400.NET Site” on page 10.	String
Border	Specifies whether a border should be drawn around the poll or survey. The default setting is true. **True** = Draw border around the poll or survey. **False** = Do not draw border around the poll or survey. The look of the border can be changed using the following properties: BorderColor - Changes the color of the border BorderStyle - Changes the border style. BorderWidth - Changes the thickness of the border line.	Boolean

Property	Value	Data Type
DefaultFormID	The FormBlock ID of the poll that appears where you inserted this server control. If you don't know the ID number of the form block, use the CMS Explorer to browse to it. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10.	Integer
DynamicParameter	To make this form block dynamic, select **id**. When you do, this server control uses the form block passed as a URL parameter.	String
FormName	Set this value to the name of your main runat=server form.	String
Hide	Used to hide a form block in design time and run time. **True** = Hide form block **False** = Show form block	Boolean
Include Tags	Determines if tags are generated automatically of manually. *See Also:* "Automatic versus Manual Generation of Form Tags" on page 195	Boolean
IsolatedDisplay	Displays the poll or results, using an iFrame, in the area of the page that contains the poll without disturbing its surrounding contents. The default is true. **True** = Polls and results are shown in an iFrame without any modification or refresh of the surrounding contents. **False** = The poll and the results are shown in place of the Web page's content. **Note**: Setting this property to true activates the Scale property. *See Also:* "Scale" on page 194.	
Language	Set a language for viewing form content. The language property shows results in design-time (in Visual Studio) and at run-time (in a browser).	Integer

Property	Value	Data Type
PostResultsAt	Enter the URL that a site user is forwarded to after he fills out the form and clicks the submit button. For example, you might have the site user fill out a form before he is allowed to see private content. Once the form is submitted, the user is forwarded to the private content. The default is blank. When left blank, the post back message associated with the form is displayed. Syntax example: http://www.ektron.com **Important:** A Formblock Server Control must exist on the Web form to which you are posting.	String
Scale	The percentage of scaling of the form and results when displayed on the same page. This allows for scaling down a survey to fit in a tight area of a content template. The value is a decimal, where 1 = 100%. The default value is 0.75 **Note**: This is only active if the IsolatedDisplay property is set to True. *See Also:* ”IsolatedDisplay” on page 193.	Double
WrapTag	Allows a developer to specify a server control’s tag. The default is **Span**. **Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code. **Custom** - Allows you to use a custom tag.	String

Automatic versus Manual Generation of Form Tags

When using the FormBlock Server Control's Include tags property, you have two options.

Option	Recommended if you are developing	How to Enable
Let **Ektron CMS400.NET** generate form tags automatically	A pure script .aspx page	Set Include Tags property to **True**.

Modify HTML form tags in the .aspx file	An .aspx page and associated code-behind web form	Set Include Tags property to **False**.
		Here is the default .NET generated form tag:
		`<form id="Form1" method="post" runat="server">`
		Modify the form tag as indicated in red:
		`<form id="Form1" method="post" runat="server" OnSubmit="EkFmValidate(this);">`

RssAggregator Server Control

An RSS Aggregator or reader takes an RSS feed and displays the information in a readable format. The RssAggregator Server Control lets you do the same by processing and displaying an RSS feed from any Web site. This allows you to create an information Web page for news, stories, images, lists of music, or for whatever purpose someone creates an RSS feed.

You can create a multi-level information Web page by using multiple RssAggregator Server Controls on the same page. For example, you could create a world news Web page by adding RSS feeds from the NY Times, The BBC, and AFP (Agence France-Presse).

The process of adding an RSS Aggregator to your Web site is simple.

1. Choose a Web form where the RSS Aggregator will reside
2. Drag and drop the RssAggregator Server Control on that Web form
3. Set the URL property to point at the RSS feed. For example, `http://msdn.microsoft.com/rss.xml`
4. Save the Web form

When a visitor to your site views the Web form in a browser, he sees the RSS feed displayed properly. When he refreshes the Web page, any information that the RSS feed provider has updated is now reflected. *See Also:* ”Using the RssAggregator Server Control” on page 198

By creating a custom XSLT, you can change the way an RSS feed is viewed by your site visitors.

This subsection contains the following topics:

- ”RSSAggregator Server Control Properties” on page 197
- ”Using the RssAggregator Server Control” on page 198

RSSAggregator Server Control Properties

The RSSAggregator Server Control properties are described in this table.

NOTE The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

Property	Value	Data Type
Authenticated	Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	String
DisplayXslt	Determines how the information on the page is displayed **None**-databind only **ecmNavigation** - lists the title of every RSS feed item *See Also:* "Example of ecmNavigation Display" on page 105 **ecmTeaser** - lists a title and a description of every RSS feed item *See Also:* "Example of ecmTeaser Display" on page 106 **Path to Custom Xslt** - Enter the path to an Xslt that determines the display of the page	String
Hide	Used to hide a RssAggregator Server Control in design time and run time. **True** = Hide Metadata Server Control **False** = Show Metadata Server Control	boolean
Language	Set a language for viewing the RssAggregator. The language property shows results in design-time (in Visual Studio) and at run-time (in a browser).	Integer
MaxResults	The Maximum number of items from an RSS feed that are returned (0=unlimited).	Integer
URL	The RSS feed path for the server control. For example: `http://msdn.microsoft.com/rss.xml`	String

Property	Value	Data Type
WrapTag	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an inline portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code. **Custom** - Allows you to use a custom tag.	String

Using the RssAggregator Server Control

These steps show how to use the RssAggregator Server Control.

1. Drag a RssAggregator Server Control into a template.
2. Set the URL property with the path of the RSS feed you want to display. This example use the following RSS feed:

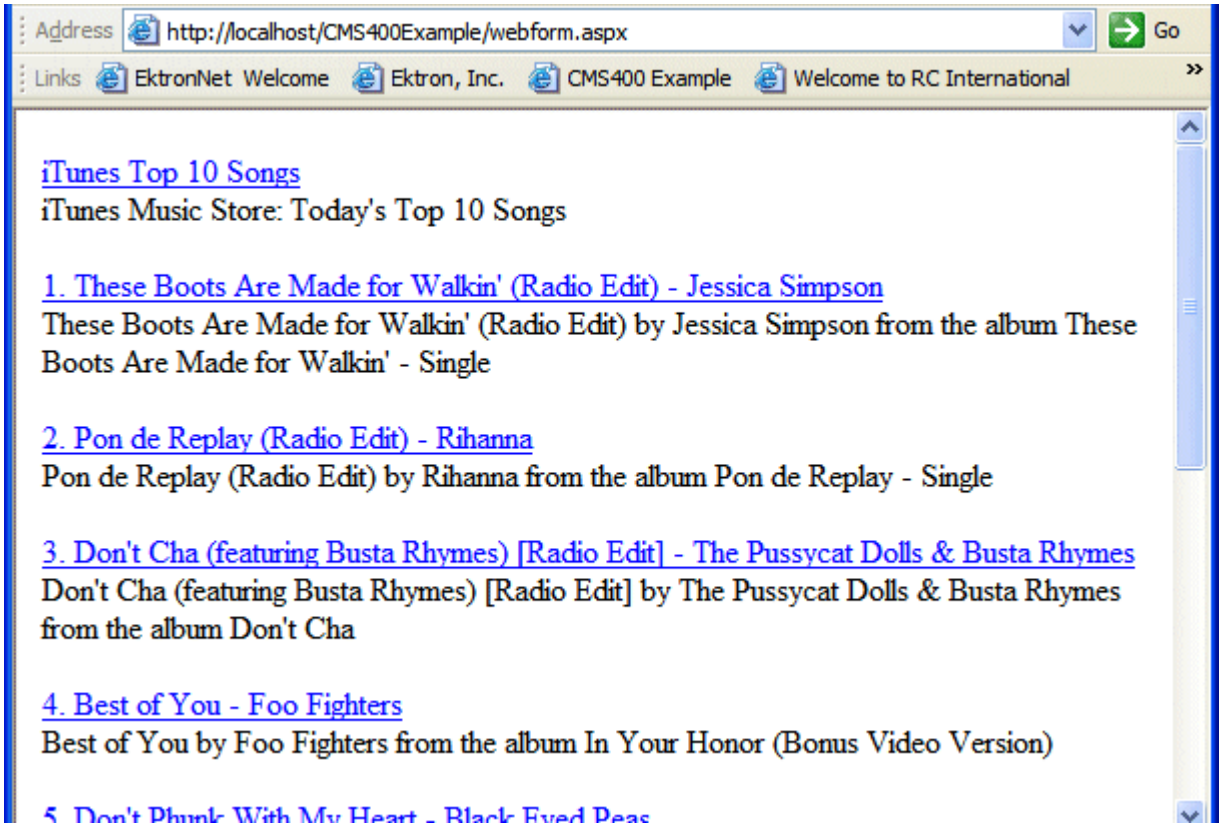
```
http://ax.phobos.apple.com.edgesuite.net/WebObjects/MZStore.woa/wpa/MRSS/topsongs/limit=10/
rss.xml
```

   This feed displays the top 10 songs according to the iTunes Music Store.

NOTE In addition to the URL property, other Ektron properties are available to customize this server control. For more information on these properties, see "RSSAggregator Server Control Properties" on page 197

3. Build the Web form.

4. Browse to the Web page in your browser.

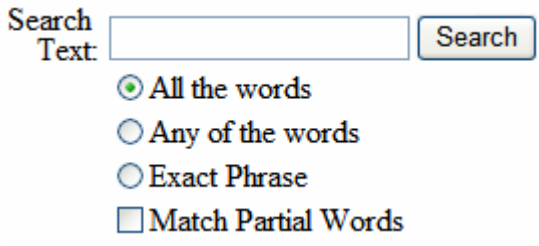


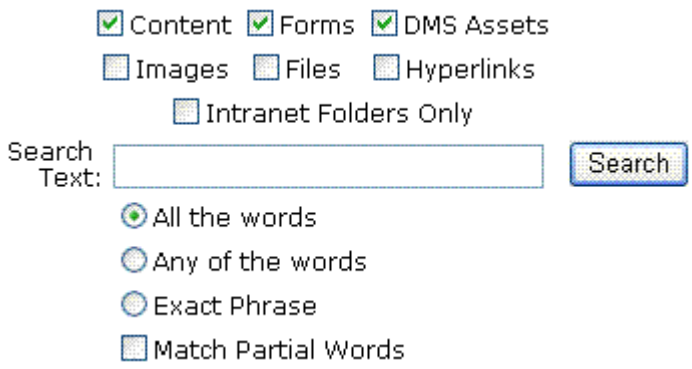
5. The page appears with an RSS list of information.

Search Server Control

Introduction

The Search Server Control displays a text box with a search button on the template. The search is highly customizable and allows you to provide a basic search, a standard search, or advanced search. The different types of searches are explained in "Using the Advanced Search Properties" on page 221. Below is a visual example of the different searches.

Description	Web Page View
A Basic Search	
A Standard Search	

Description	Web Page View
An Advanced or Mixed Search	

When users perform a search using the Search Server Control, they can be looking through Ektron CMS400.NET content, DMS assets, forms and library items. To learn more about configuring the Search Server Control to search different content types, see "Using the Advanced Search Properties" on page 221.

Searches can also be performed on content that utilizes the Searchable Property metadata type. You can make this search parameter available to users performing a search or hide it. *See Also:* "ShowExtendedSearch" on page 218

To search XML content, Use the IndexSearch Server Control. *See Also:* "Search Server Control vs. IndexSearch Server Control" on page 202.

This subsection contains the following topics.

- "Search Server Control vs. IndexSearch Server Control" on page 202
- "Search Server Control Properties" on page 202
- "Property Usage Table" on page 202
- "Property Descriptions" on page 204
- "Text Box Examples" on page 214
- "Button Image Source Examples" on page 215
- "Search Button Text Examples" on page 216

- ”Display Examples” on page 217
- ”ShowDate Examples” on page 217
- ”ShowExtendedSearch” on page 218
- ”Using the Advanced Search Properties” on page 221
- ”Fieldset Properties” on page 229
- ”Retrieving the XML Structure of a Search” on page 230
- ”Displaying Search Results on a Separate Page” on page 231

Search Server Control vs. IndexSearch Server Control

The Search Server Control lets users search your Web site for XHTML Content, archive content, DMS assets, forms and library items. The IndexSearch Server Control lets users search XML content on your Web site. For more information on the IndexSearch Server Control, see ”IndexSearch Server Control” on page 234.

Search Server Control Properties

The following tables explain the properties of the Search Server Control. The first table shows whether the properties affect the search display, the search criteria or the search results display. The second table contains a list of all the properties, their descriptions, data type and links to more information. The rest of the tables in this section provide further information on the various Search Server Control Properties.

Property Usage Table

Properties in the Search Server Control generally affect the control in one of three ways:

- **Site Search Display** - how the Search Server Control is displayed on your site
- **Site Search Criteria** - the criteria the control uses to search the site

- **Search Results Display** - how the search results are displayed on your site

The following table shows which properties are associated with each category. *See Also:* "Property Descriptions" on page 204

Site Search Display	Site Search Criteria	Search Results Display
• ButtonImgSrc • ButtonText • ContentFieldsetLegend • Display • EnableAdvancedLink • EnableArchivedBtn • EnableContentBtn • EnableContentFieldset • EnableDMSAssetsBtn • EnableFolderFieldset • EnableFormsBtn • EnableLibFilesBtn • EnableLibHyperlinksBtn • EnableLibImageBtn • EnableLibraryFieldset • FolderFieldsetLegend • Hide • Language • LibraryFieldsetLegend • MaxCharacters • ShowExtendedSearch • ShowSearchOptions • Text Box Size • WrapTag	• EnableArchived • EnableContent • EnableDMSAssets • EnableForms • EnableLibFiles • EnableLibHyperlinks • EnableLibImages • FolderPath • Language • Recursive • ShowSearchOptions • TargetFolderList	• DisplayXslt • EmptyResultMsg • Hide • Language • MaxResults • OrderDirection • PostResultsAt • ShowDate • ShowSearchBoxAlways

Property Descriptions

The following table describes the Search Server Control Properties. *See Also:* "Property Usage Table" on page 202

NOTE The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

Property	Description	Data Type	For more information, see
Authenticated	Indicates if you are logged in to the CMS Explorer and can use it to browse to the folder needed for the **FolderPath** property.	String	"Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10
ButtonImgSrc	If you want to display an image on the submit button, enter the server path to it.	String	"Button Image Source Examples" on page 215
ButtonText	The text used for the button if no image source is identified. If an image source is identified, this is alternative text for the button.	String	"Search Button Text Examples" on page 216
ContentFieldsetLegend	Text that appears in the fieldset border around the content check boxes. This property is used only when the EnableContentFieldset property is set to True. The default is: Content Items: *See Also:* "EnableContentFieldset" on page 206	String	"Fieldset Properties" on page 229
Display	Determines if the search options are in a row (**Horizontal**) or a column (**Vertical**).	String	"Display Examples" on page 217

Property	Description	Data Type	For more information, see
DisplayXslt	Determines how the information on the page is displayed **None**-databind only **ecmNavigation** - lists the title of every content block in the folder **ecmTeaser** - lists the title of every content block in the folder plus the content summary **Path to Custom Xslt** - Enter the path to an Xslt that determines the display of the page	String	**ecmNavigation -** "Example of ecmNavigation Display" on page 105 **ecmTeaser** - "Example of ecmTeaser Display" on page 106
EmptyResultMsg	The return message displayed if the search returns no hits. The default is: Search returned zero results	String	
EnableAdvancedLink	Set to true to display the link that makes the advanced properties available. **True** = Display Advanced link **False** = Hide Advanced link	Boolean	"Using the Advanced Search Properties" on page 221
EnableArchived	Enables users to search archived content. The archived content must be archived as Archived Remain On Site.The default is False. **True** = Searching archived content is enabled. **False** = Searching archived content is disabled, unless Include Archived is checked in the Advanced search. *See Also:* "EnableArchivedBtn" on page 206	Boolean	"Using the Advanced Search Properties" on page 221 For more information on archiving content, see the User Manual section "Scheduling Content to Begin and End" > "Setting Archive Options".

Property	Description	Data Type	For more information, see
EnableArchivedBtn	Provides a check box so users can decide if they want to perform an archive search. The default is False. **True** = The Include Archive check box is enabled in the advanced search properties. If used with EnableArchived, the check box is checked by default. **False** = The Include Archive check box is disabled in the advanced search properties.	Boolean	”Using the Advanced Search Properties” on page 221
EnableContent	Enables users to search content. The default is True. **True** = Searching content is enabled. **False** = Searching content is disabled, unless Content is checked in the Advanced search. *See Also:* ”EnableContentBtn” on page 206	Boolean	”Using the Advanced Search Properties” on page 221
EnableContentBtn	Provides a check box so users can decide if they want to perform a content search. The default is True. **True** = The Content check box is enabled in the advanced search properties. If used with EnableContent, the check box is checked by default. **False** = The Content check box is disabled in the advanced search properties.	Boolean	”Using the Advanced Search Properties” on page 221
EnableContentFieldset	Creates a border around the group of content items in an advanced search. These content items include Content, Forms and DMS Assets. The default is False. **True** = Show border around content items. **False** = No border around content items. *See Also:* ”ContentFieldsetLegend” on page 204	Boolean	”Fieldset Properties” on page 229

Property	Description	Data Type	For more information, see
EnableDMSAssets	Enables users to search DMS Assets. The default is True. **True** = Searching DMS Assets is enabled. **False** = Searching DMS Assets is disabled, unless DMS Assets is checked in the Advanced search. *See Also:* ”EnableDMSAssetsBtn” on page 207	Boolean	”Using the Advanced Search Properties” on page 221
EnableDMSAssetsBtn	Provides a check box so users can decide if they want to perform a DMS Asset search. The default is True. **True** = The DMS Asset check box is enabled in the advanced search properties. If used with EnableContent, the check box is checked by default. **False** = The DMS Asset check box is disabled in the advanced search properties.	Boolean	”Using the Advanced Search Properties” on page 221
EnableFolderFieldset	Creates a border around the group of folders in an advanced search. These folders include any folders listed in the TargetFolderList property. The default is False. **True** = Show border around folders. **False** = No border around folders. *See Also:* ”FolderFieldsetLegend” on page 210 and ”TargetFolderList” on page 213	Boolean	”Fieldset Properties” on page 229
EnableForms	Enables users to search Forms. The default is True. **True** = Searching Forms is enabled. **False** = Searching Forms is disabled, unless Forms is checked in the Advanced search. *See Also:* ”EnableFormsBtn” on page 208	Boolean	”Using the Advanced Search Properties” on page 221

Property	Description	Data Type	For more information, see
EnableFormsBtn	Provides a check box so users can decide if they want to perform a Forms search. The default is True. **True** = The Forms check box is enabled in the advanced search properties. If used with EnableContent, the check box is checked by default. **False** = The Forms check box is disabled in the advanced search properties.	Boolean	”Using the Advanced Search Properties” on page 221
EnableLibFiles	Enables users to search Library Files. The default is True. **True** = Searching Library Files is enabled. **False** = Searching Library Files is disabled, unless Files is checked in the Advanced search. *See Also:* ”EnableLibFilesBtn” on page 208	Boolean	”Using the Advanced Search Properties” on page 221
EnableLibFilesBtn	Provides a check box so users can decide if they want to perform a Library Files search. The default is True. **True** = The Files check box is enabled in the advanced search properties. If used with EnableLibFiles, the check box is checked by default. **False** = The Files check box is disabled in the advanced search properties.	Boolean	”Using the Advanced Search Properties” on page 221
EnableLibHyperlinks	Enables users to search Library Hyperlinks. The default is True. **True** = Searching Library Hyperlinks is enabled. **False** = Searching Library Hyperlinks is disabled, unless Hyperlinks is checked in the Advanced search. *See Also:* ”EnableLibHyperlinksBtn” on page 209	Boolean	”Using the Advanced Search Properties” on page 221

Property	Description	Data Type	For more information, see
EnableLibHyperlinksBtn	Provides a check box so users can decide if they want to perform a Library Hyperlinks search. The default is True. **True** = The Hyperlinks check box is enabled in the advanced search properties. If used with EnableLibHyperlinks, the check box is checked by default. **False** = The Hyperlinks check box is disabled in the advanced search properties.	Boolean	”Using the Advanced Search Properties” on page 221
EnableLibImages	Enables users to search Library Images. The default is True. **True** = Searching Library Images is enabled. **False** = Searching Library Images is disabled, unless Images is checked in the Advanced search. *See Also:* ”EnableLibImageBtn” on page 209	Boolean	”Using the Advanced Search Properties” on page 221
EnableLibImageBtn	Provides a check box so users can decide if they want to perform a Library Images search. The default is True. **True** = The Images check box is enabled in the advanced search properties. If used with EnableLibImages, the check box is checked by default. **False** = The Images check box is disabled in the advanced search properties.	Boolean	”Using the Advanced Search Properties” on page 221

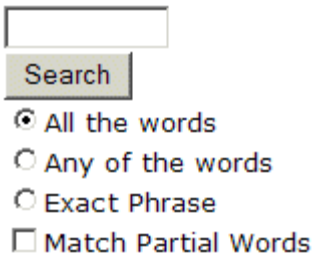
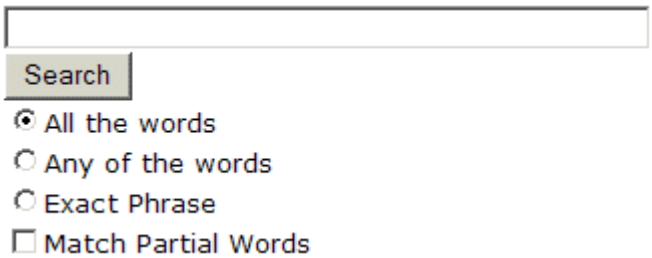
Property	Description	Data Type	For more information, see
EnableLibraryFieldset	Creates a border around the group of library items in an advanced search. These items include: Images, Files and Hyperlinks. The default is False. **True** = Show border around library items. **False** = No border around library items. *See Also:* ”LibraryFieldsetLegend” on page 211	Boolean	”Fieldset Properties” on page 229
FolderFieldsetLegend	Text that appears in the fieldset border around the Folder check boxes. This property is used only when the EnableFolderFieldset property is set to True. The default is: Folder Descriptions: *See Also:* ”EnableFolderFieldset” on page 207	String	”Fieldset Properties” on page 229
FolderPath	The folder at which the search begins. The starting folder need not be the root folder. The Recursive property determines if the search examines this folder’s subfolders.	String	”Using CMS Explorer to Browse Your Ektron CMS400.NET Site” on page 10
Hide	Used to hide Search Server Control in design time and runtime. **True** = Hide control **False** = Show control	Boolean	
Language	Set a language for viewing search content. The language property shows results in design-time (in Visual Studio) and at run-time (in a browser).	Integer	

Property	Description	Data Type	For more information, see
LibraryFieldsetLegend	Text that appears in the fieldset border around the Library items check boxes. This property is used only when the EnableLibraryFieldset property is set to True. The default is: Library items: *See Also:* ”EnableLibraryFieldset” on page 210	String	”Fieldset Properties” on page 229
MaxCharacters	The maximum number of characters the input text box accepts when doing a search.	Integer	
MaxResults	The maximum number of content blocks returned (0=unlimited).	Integer	
OrderDirection	The direction the search results will be sorted in. The default is **Ascending** • **Ascending** - Results from A to Z • **Descending** - Results from Z to A	String	
PostResultsAt	Enter the Web form where the search results will be posted. When left blank, the results are posted to the Web form from which the search was initiated. **Important:** A search control must exist on the posting page.	String	”Displaying Search Results on a Separate Page” on page 231
Recursive	Whether to search sub-folders of the identified root folder. The root folder is identified in the `Folderpath` property.	Boolean	
ShowDate	If **True**, displays the content block’s last modified date. This is useful for determining the most recent content block.	Boolean	”ShowDate Examples” on page 217

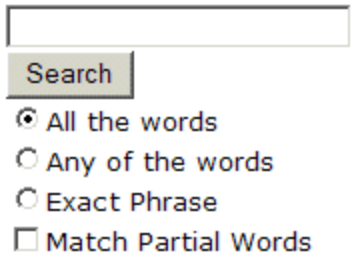
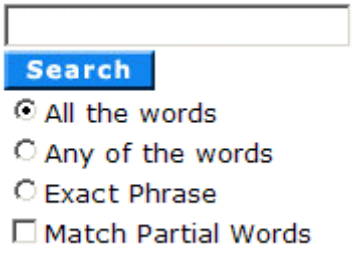
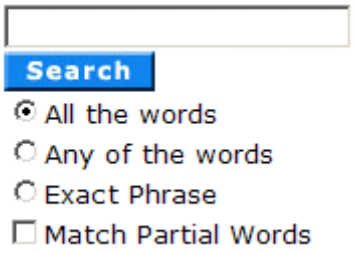
Property	Description	Data Type	For more information, see
ShowExtendedSearch	Allows a user to search via the metadata type: Searchable Property. **True** = Show extended search. **False** = Hide extended search.	Boolean	”ShowExtendedSearch” on page 218 For more information on Metadata and Searchable Property, see the Administrator Manual section “Metadata”.
ShowSearchBoxAlways	If set to **false**, the search box does not appear on PostBack. The default is True. **True** = Display search box on PostBack **False** = Do not display search box on PostBack	Boolean	
ShowSearchOptions	If set to true, the following choices appear along with the search box. This type of search is also known as a Standard search. The default setting is True. • All the words • Any of the words • Exact phrase • Match Partial words **True** = Show search options **False** = Hide search options	Boolean	”Basic and Standard Searches” on page 223

Property	Description	Data Type	For more information, see
TargetFolderList	A comma delineated list of folders to search. The target folder list uses the Description property of a folder to define which folders are searched. This is an optional property. If you use this property, it will supersede the FolderPath property. The default is blank. Below is an example of the syntax used. Football=1,News\Headlines=0 Football, News and Headlines are the folder descriptions. The ones (1) and zeros (0) define whether a site visitor is forced to search the folder or not. 1 = Automatically search this folder 0 = Do not automatically search this folder	String	”TargetFolderList” on page 220
Text Box Size	The size of the display text box for user input. The width is based on the number of characters.	Integer	”Text Box Examples” on page 214
WrapTag	Allows a developer to specify a server control’s tag. The default is **Span**. **Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code. **Custom** - Allows you to use a custom tag.	String	

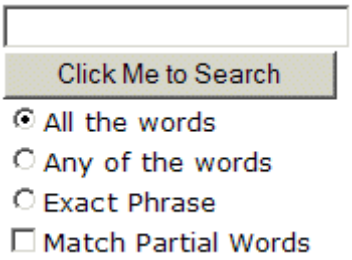
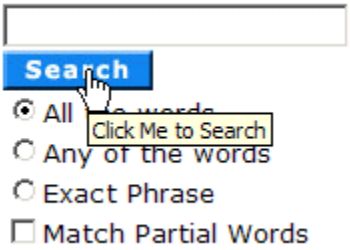
Text Box Examples

Value	Web Page View
10	
50	

Button Image Source Examples

Value	Web Page View
(blank)	
search.jpg	
http://www.ektron.com/images/search.jpg	

Search Button Text Examples

Value	Web Page View
“Click Me to Search”	
“Click Me to Search”	

Display Examples

Value	Web Page View
Vertical	Search All the words Any of the words Exact Phrase Match Partial Words
Horizontal	Search All the words Any of the words Exact Phrase Match Partial Words

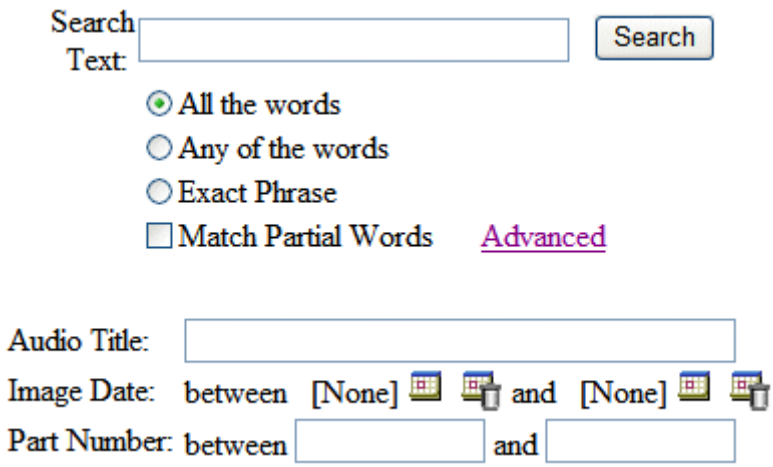
ShowDate Examples

Value	Web Page View
False	**Search Results** RC Sportster RC Redstar RC Lilly RC Cheetah Products Page RC International Ships RC Sportster RC International Hosts RC Conference RC International Announces the RC Cheetah

Value	Web Page View
True	**Search Results** RC Sportster *(8/27/2002 7:52:40 PM)* RC Redstar *(8/27/2002 7:52:01 PM)* RC Lilly *(8/27/2002 7:51:26 PM)* RC Cheetah *(8/27/2002 7:50:53 PM)* Products Page *(8/27/2002 7:50:09 PM)* RC International Ships RC Sportster *(8/27/2002 2:46:12 PM)* RC International Hosts RC Conference *(8/27/2002 2:44:44 PM)* RC International Announces the RC Cheetah *(8/27/2002 2:43:01 PM)*

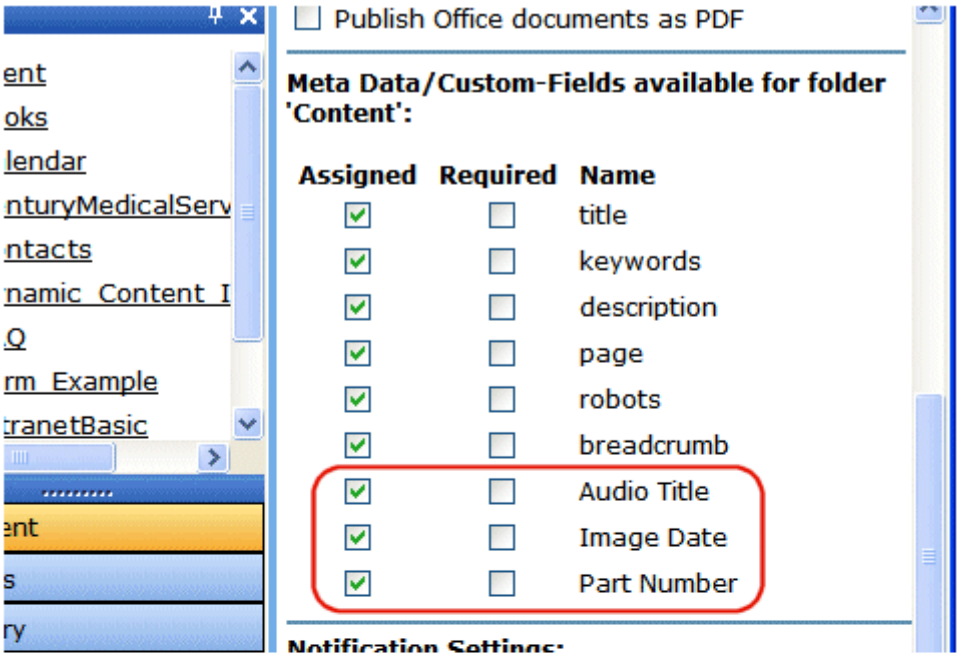
ShowExtendedSearch

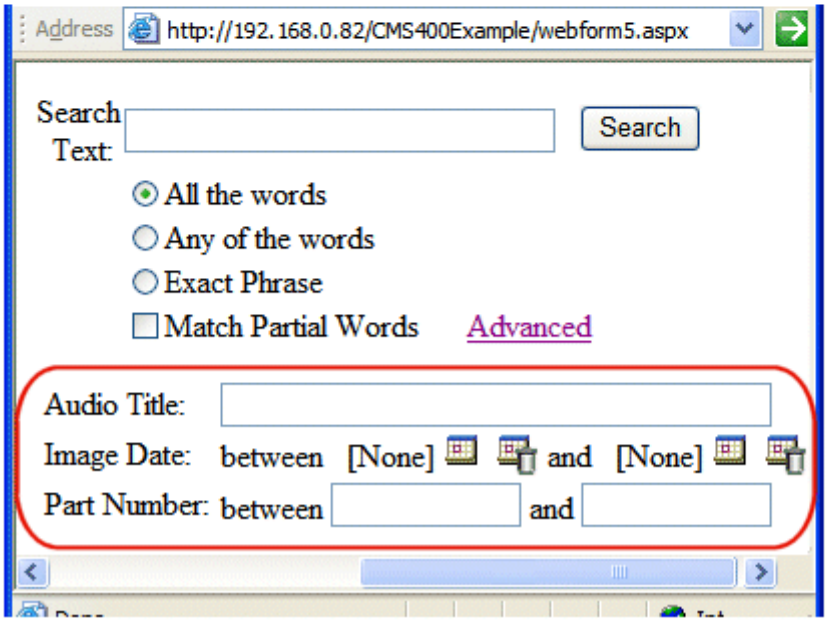
ShowExtendedSearch lets you decide if the user can search the Searchable Property contained in the metadata of a content block. By setting the ShowExtendedSearch property to **true**, when a site visitor views the search, the extended search parameters appear at the bottom of the search form.



The Searchable Properties for a folder must be turned on for them to appear in a search. For more information on using Searchable Properties, see the Administrator Manual section “Metadata”.

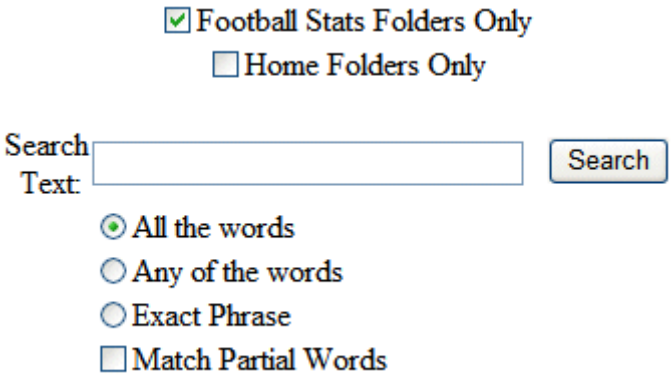
The example below shows the Searchable Properties being activated in the folder properties of the Workarea and the Searchable Properties appearing in a search form.





TargetFolderList

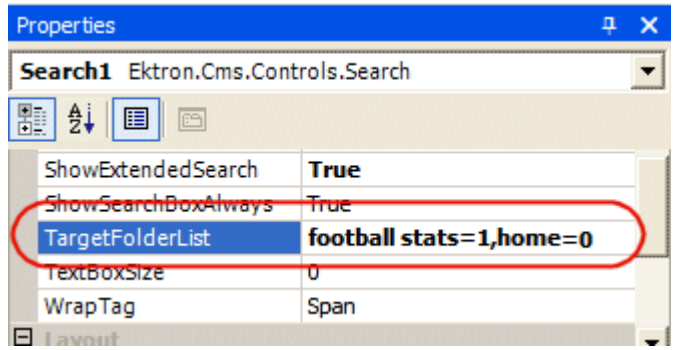
The TargetFolderList property allows a developer to create a custom folder list for site visitors to search. Using this property is optional. If the property is used, it overrides the settings in the FolderPath property.

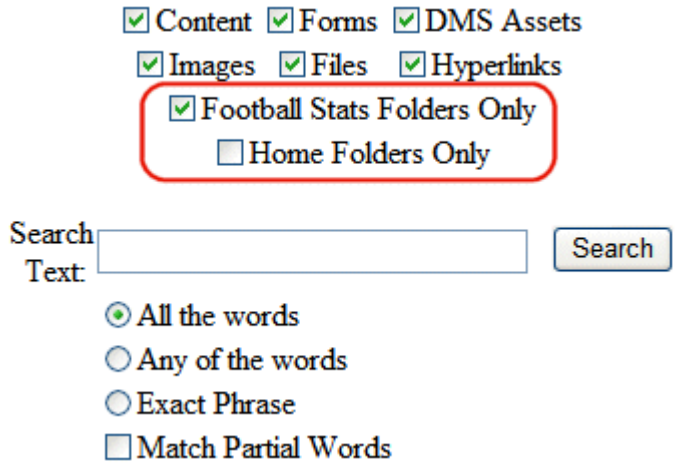


You can force site visitors to search only the specific folders you set in the TargetFolderList. This is accomplished by creating a comma delineated folder list in the TargetFolderList property and setting the `EnableAdvancedLink` property to False. When a visitor to your site performs a search, they only receive results from folders in the TargetFolderList.

In contrast, you can give site visitors the ability to decide which folders they want to search. This is accomplished by creating a comma delimited folder list in the `TargetFolderList` property and setting the `EnableAdvancedLink` property to True. This allows the site visitor to click the **Advanced** link and then choose from the folders you allow. The options appear as check boxes below the advanced properties

Here is an example of the TargetFolderList in Visual Studio. If the Advanced link is disabled, a site user is forced to search only the Football Stats folder. If the Advanced link is enabled and a site visitor clicks it, he has the choice to search the Football Stats and Home. The **Football Stats** check box is checked by default, while the **Home** check box is not.





Using the Advanced Search Properties

There are three ways to use the advanced search properties. One way is to create a basic or standard search, in which the user must search by properties you choose. Another way is to give users the ability to decide which properties they want use for the search. The last way is a mix of both. *See Also:* "Basic and Standard Searches" on page 223, "Advanced Search" on page 225 and "Using Standard and Advanced Searches Together" on page 226

The following is a list of the advanced properties. These properties appear in the list of properties for the server control. For descriptions of these properties, see "Search Server Control Properties" on page 202.

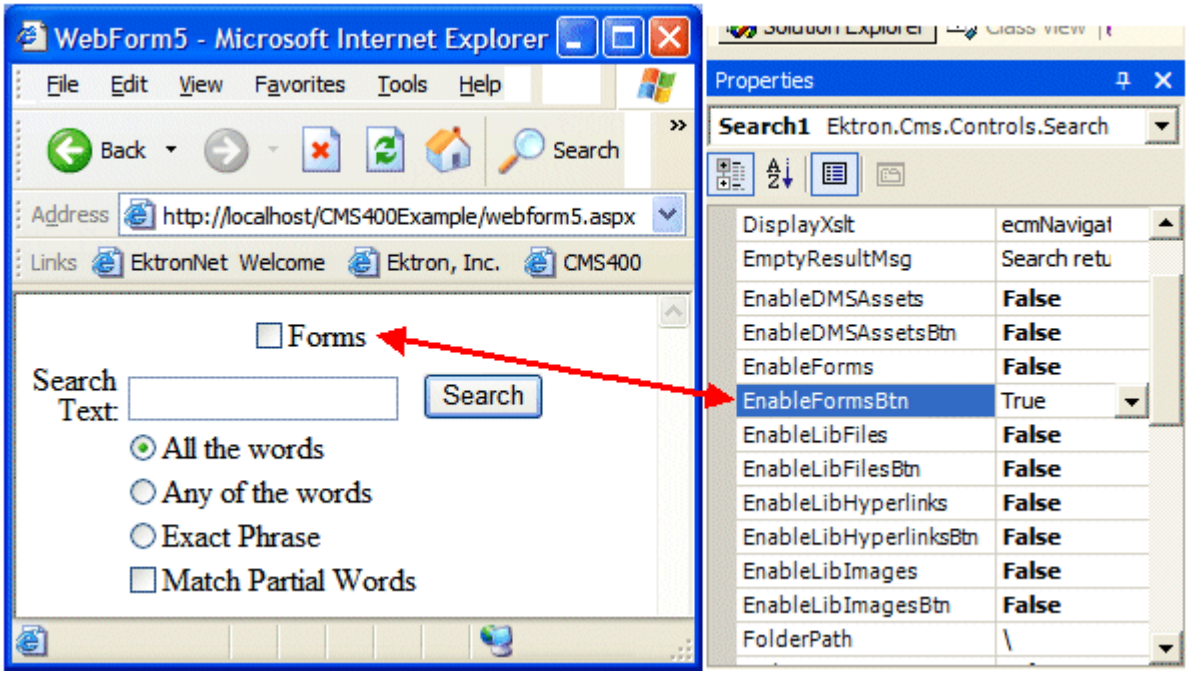
- EnableArchived
- EnableArchivedBtn
- EnableContent
- EnableContentBtn
- EnableDMSAssets
- EnableDMSAssetsBtn
- EnableForms
- EnableFormsBtn
- EnableLibFiles
- EnableLibFilesBtn
- EnableLibHyperlinks
- EnableLibHyperlinksBtn
- EnableLibImages
- EnableLibImagesBtn

WARNING! If you set all above properties to **false**, the search returns no results. This happens because there are no parameters to search against.

These properties can be used individually or in combination. You could have a search that only searches Forms, or a search that searches Forms and DMS Assets.

Each advanced property has an **enable** check box property associated with it. These properties have **Btn** at the end of their name. They create a check box that appears when the Advanced link on a search is clicked. The check boxes only appear when the properties are set to true. Using the Advanced link and check boxes to search is known as an Advanced Search. *See Also:* ”Advanced Search” on page 225

The following example shows the Forms check box enabled on the site search and the EnableFormsBtn property set to True in Visual Studio.



You can force a user to search using only properties you define, preventing them from changing any properties. For example, to create a forms-only search, you would set all properties with **Enable** in the name to **false** except for EnableForms. This is known as a standard search. *See Also:* "Basic and Standard Searches" on page 223

NOTE When setting up a server control for a standard search, you must choose the actual property to search with, not the property that enables the check box. For example, to search forms, use the property EnableForms, not the EnableFormsBtn property.

Basic and Standard Searches

A basic search contains only three items when displayed on a page:

- text to describe the text box
- a search text box

- a submit or “search” button

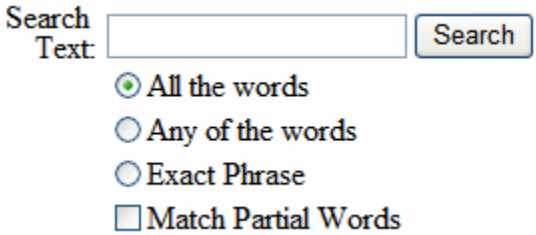
Utilizing a basic search takes the least amount of real estate on your site and provides the least amount of options for a site user. Below is an example of a basic search. This handy when you want to have a search perform specified functions with little user control. To make your search control appear as a basic search, set the ShowSearchOptions property to **false**.

A standard search allows a site user a few more options when searching your site. These options, listed below, allow a site user to decide how the text should match the information they are searching for.

- All the words
- Any of the words
- Exact phrase
- Match Partial words

With basic and standard searches, you decide which properties are used in the search. At least one advanced search property must be set to **true** in the server control. Otherwise, the search returns no matches.

When added to a template and visited, a standard search looks like this.



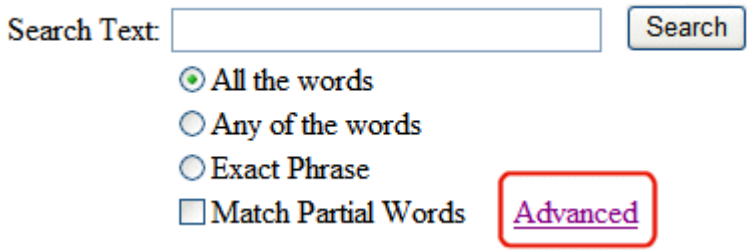
Users enter text, change the way the text matches the items being searched, and click the search button to search for the types of content you allow. For examples of setting up a standard search, see "Using Standard and Advanced Searches Together" on page 226

Advanced Search

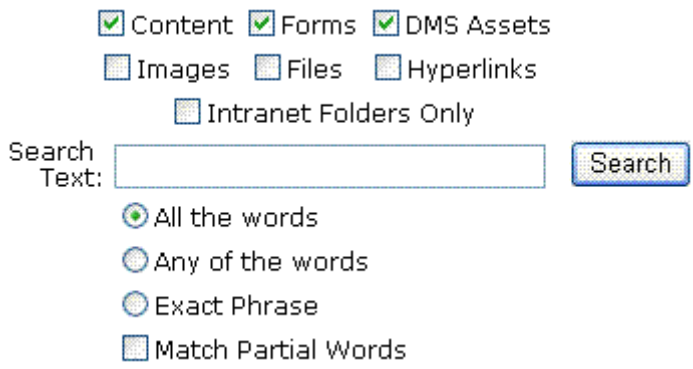
In an advanced search, you decide which content types are available. From those choices, users decide which content types they want and perform a search.

To let a site visitor choose which content types to search, the Advanced Hyperlink must be turned on. Otherwise, users can only search properties you have enabled. *See Also:* "EnableAdvancedLink" on page 205

When the Advanced hyperlink is enabled, this is how the search appears.



When the Advanced hyperlink is clicked, this is how the search appears. This assumes all choices are enabled and automatically checked. For examples of setting up an advanced search, see "Using Standard and Advanced Searches Together" on page 226



Using Standard and Advanced Searches Together

You can use a mix of the standard search and advanced search to force people to search by certain properties while giving them a choice on others. For example, you want users to always search Content, yet give them an option to search forms, files, and images.

The following examples illustrate how to configure the Search Server Control using the advanced properties. If a property is not mentioned in the examples below, use its default setting.

Description	How the Search Appears to Site Visitors	Property Settings
Search content only. Do not let the site user change advanced settings. No standard options appear. This is a **Basic search**. *See Also:* "Basic and Standard Searches" on page 223	. 	EnableAdvancedLink = False EnableContent = True Set all other properties with the word **Enable** to **False**.

Description	How the Search Appears to Site Visitors	Property Settings
Search content, forms, and library images only. Do not let the user change advanced settings. No standard options appear. This is a **Basic search**. *See Also:* "Basic and Standard Searches" on page 223	Search Text: Search	EnableAdvancedLink = False EnableContent = True EnableForms = True EnableLibImages = True Set all other properties with the word **Enable** to **False**.
Search content only. Do not let the site user change advanced settings. Standard options appear. This is a **Standard search**. *See Also:* "Basic and Standard Searches" on page 223	Search Text: Search All the words Any of the words Exact Phrase Match Partial Words	EnableAdvancedLink = False EnableContent = True ShowSearchOptions = True Set all other properties with the word **Enable** to **False**.
Let the user choose whether to search content, forms, library images, or all of them together. The user can also mix and match properties. Standard options appear. For example, search content and library images, but not forms. This is an **Advanced search**. *See Also:* "Advanced Search" on page 225 **Warning:** If a user does not check at least one box, the search returns no results.	. Content Forms Images Search Text: Search All the words Any of the words Exact Phrase Match Partial Words	EnableAdvancedLink = True EnableContentBtn = True EnableFormsBtn = True EnableLibImagesBtn = True ShowSearchOptions = True Set all other properties with the word **Enable** to **False**.

Description	How the Search Appears to Site Visitors	Property Settings
The check boxes are checked by default. The user can change the check boxes. Standard options appear. This is an **Advanced search**. *See Also:* "Advanced Search" on page 225 **Warning:** If a user unchecks all of the boxes, the search returns no results.	☑ Content ☑ Forms ☑ Images Search Text: [ ] Search ◉ All the words ○ Any of the words ○ Exact Phrase ☐ Match Partial Words	EnableAdvancedLink = True EnableContent = True EnableContentBtn = True EnableForms = True EnableFormsBtn = True EnableLibImages = True EnableLibImagesBtn = True ShowSearchOptions = True Set all other properties with the word **Enable** to **False**.
The user is forced to search Content but can choose to search Forms or Images. The Forms and Images check boxes are checked by default. Standard options appear. This is a mix of Standard and Advanced searches. *See Also:* "Using Standard and Advanced Searches Together" on page 226 **Warning:** If a user unchecks all of the boxes, the search still search content and will return any matching results.	☑ Forms ☑ Images Search Text: [ ] Search ◉ All the words ○ Any of the words ○ Exact Phrase ☐ Match Partial Words	EnableAdvancedLink = True EnableContent = True EnableForms = True EnableFormsBtn = True EnableLibImages = True EnableLibImagesBtn = True ShowSearchOptions = True Set all other properties with the word **Enable** to **False**.

Fieldset Properties

NOTE The properties described below can be seen by a site visitor only when the Advanced link is enabled and clicked.

The fieldset properties let you refine the look and feel of your site search by separating items used in the Advanced search into groups. Each group can have its own legend. The legends makes it easier for site visitors to understand their choices and how those choices affect the search.

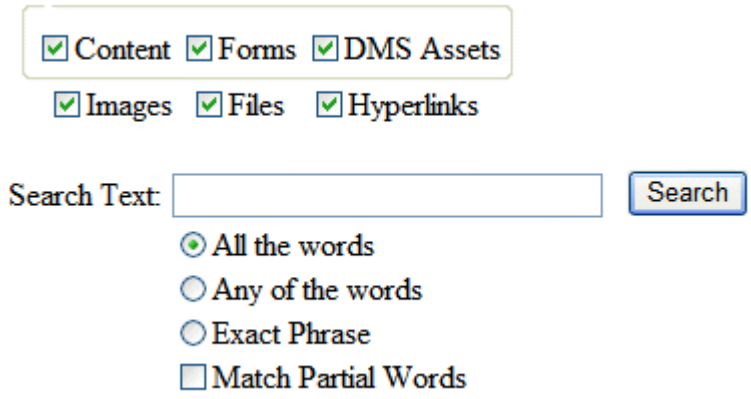
The fieldset properties are:

- ContentFieldsetLegend
- EnableContentFieldset
- FolderFieldsetLegend
- EnableFolderFieldset
- LibraryFieldsetLegend
- EnableLibraryFieldset

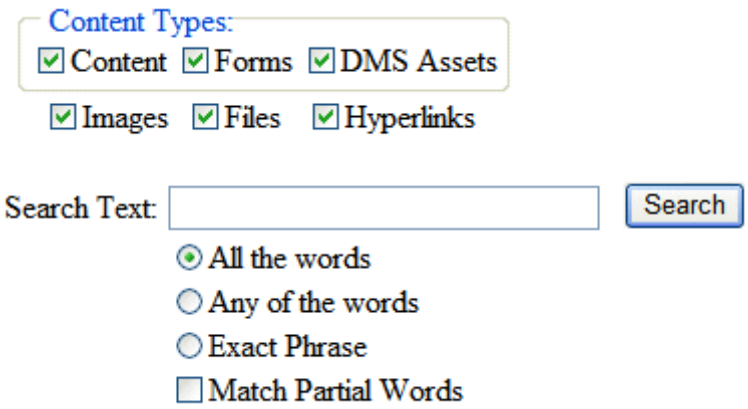
These properties can be broken down into two types:

- enabling fieldsets
- enabling fieldset legends

When you enable a fieldset, a border is created around a group of items. For example, if the `EnableContentFieldset` property is set to **true**, a border surrounds the different types of content.



You can add a legend to the fieldset to describe its content. For example, by adding **Content Types:** to the ContentFieldsetLegend property, you create a legend for the fieldset around the different types of content.



Retrieving the XML Structure of a Search

Retrieving the XML structure of XML content provides greater control over developing XSLs. The following is an example of how to retrieve the XML structure:

1. Open a new Web form.
2. Drag and drop a Search Server Control onto it.

3. Set the FolderPath property.
4. Drag and drop a Textbox on the Web form.
5. Set the TextMode property to MultiLine.

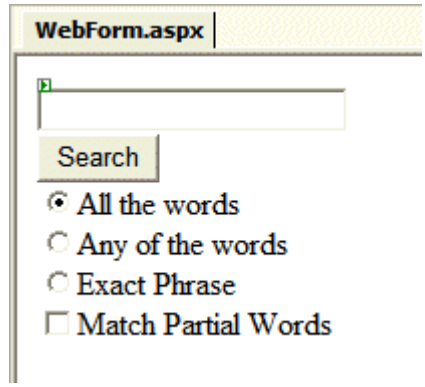
NOTE It is also recommended that you set the width of the text box to at least 400px.

6. On the code behind page, add the following line.
   ```
   Textbox1.Text = Search1.XmlDoc.InnerXml
   ```
7. Build the project.
8. View the Web form in a browser.
9. Search for something.
10. When the search returns any items, the XML structure of those items appear in the textbox.

Displaying Search Results on a Separate Page

The following example shows a search dialog box on a Web form, and the results displayed on separate Web form. This is done when you have the search dialog box in a navigation pane, and display the results on a separate Web form.

1. Drag and drop a Search Server Control on a Web form.



2. Set the following Ektron properties for the Search Server Control.

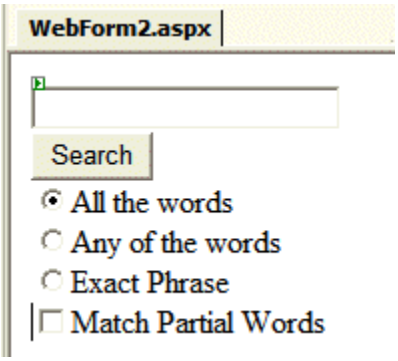
   DefaultResultsDisplay = False

   DisplayXslt = None - Databind Only

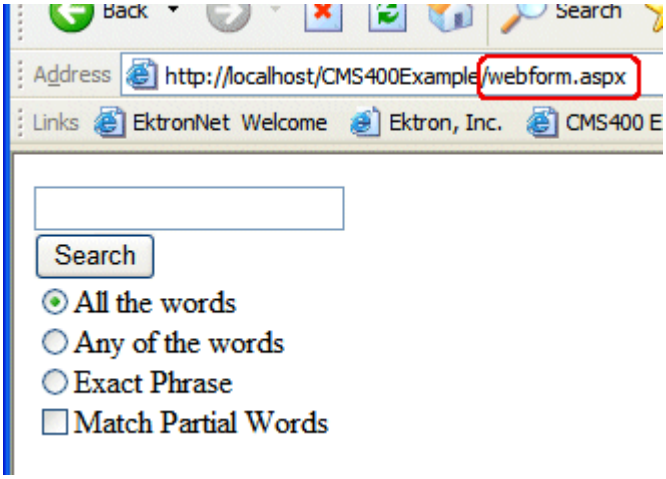
   PostResultAt = The Web form name where the results are posted. For example, `Webform2.aspx`

NOTE webform2.aspx represents the name of the second Web form.

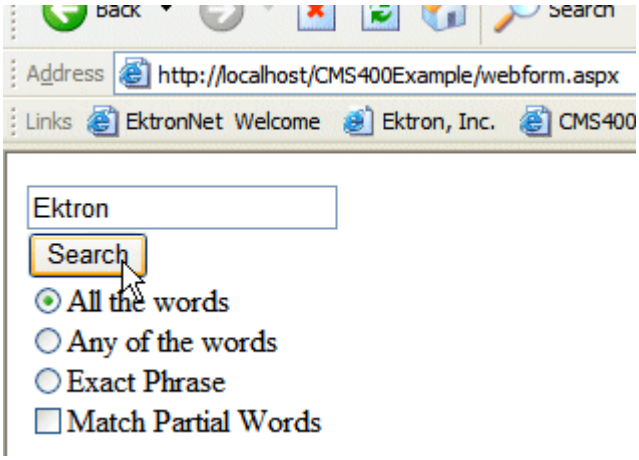
3. Add a second Web form to the project.
4. Drag and drop a Search Server Control on it.



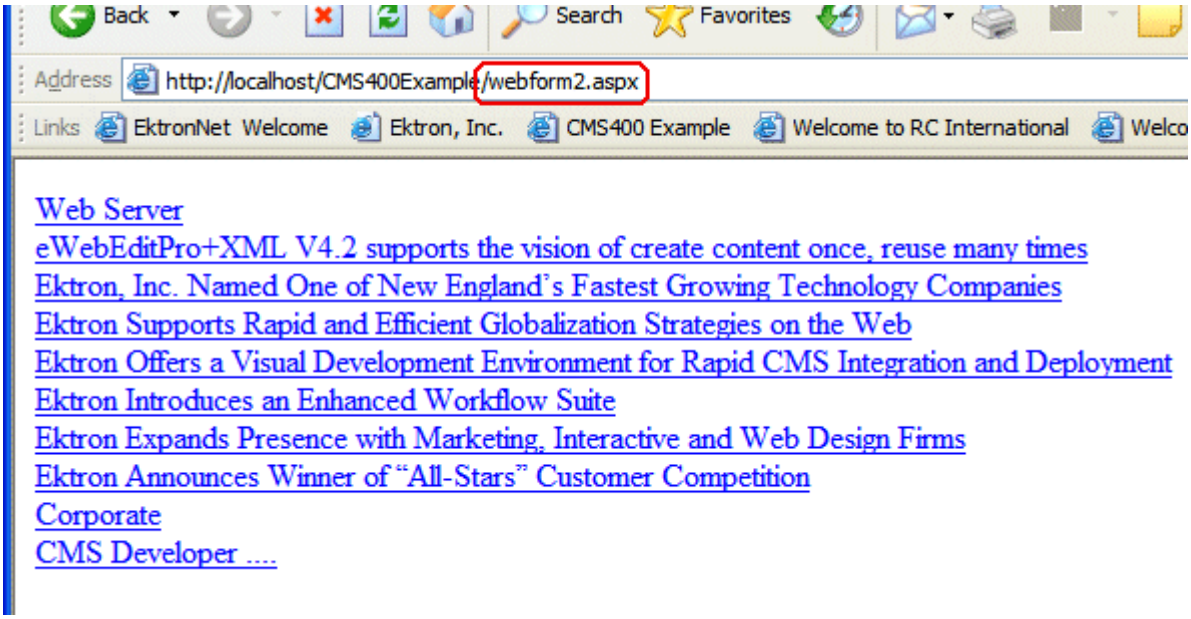
5. Set the Ektron property **ShowSearchBoxAlways** to **false**.
6. Build the solution.
7. In your browser, go to the first Web form.



8. Use the search dialog.



9. The search results display on the second Web form.

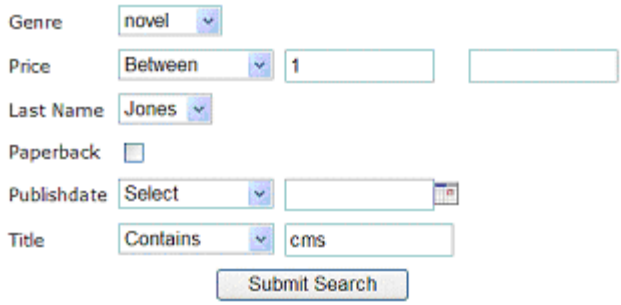


IndexSearch Server Control

Use the IndexSearch Server Control to display an XML configuration's search screen on any Web form. This lets a user search your site based on XML parameters.

See Also: The **Ektron CMS400.NET** Administrator Manual section "XML Indexing".

Below is a sample screen that can be generated via the IndexSearch Server Control.



You can also predefine a search and have the results appear on a Web form template by using the SearchParmXML property.

This subsection contains the following topics:

- "IndexSearch Server Control vs. Search Server Control" on page 235
- "IndexSearch Server Control Properties" on page 235
- "SearchParmXML Property" on page 240
- "Using the IndexSearch Server Control Programmatically Example" on page 243

- "Programmatically Predefined General Search Results Replaced By Specific Search Results" on page 246
- "Turn Off Caching In Netscape Navigator and Mozilla Firefox" on page 250

IndexSearch Server Control vs. Search Server Control

See, "Search Server Control vs. IndexSearch Server Control" on page 202.

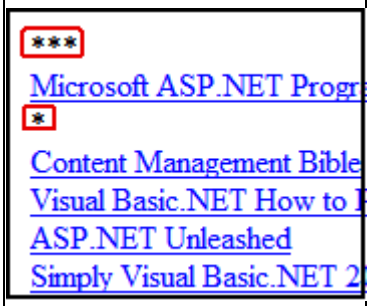
IndexSearch Server Control Properties

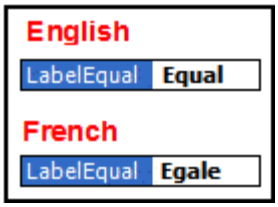
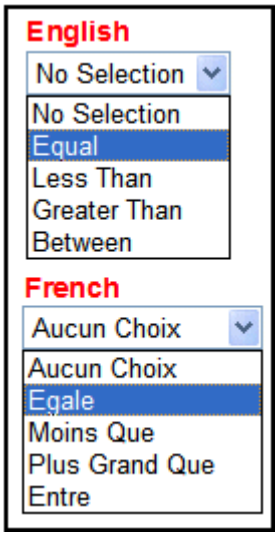
The IndexSearch Server Control properties are described in this table.

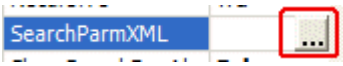
NOTE The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

Property	Description	Data Type	Values
Authenticated	Indicates if you are logged in to the CMS Explorer and can use it to browse to the folder needed for the **FolderID** property. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10	String	
ButtonText	The text that appears on the button.	String	The default is: Search

Property	Description	Data Type	Values
ContentParameter	Checks the QueryString for the content block ID value and replaces the search with content when ID is specified. Leave blank to always display the search.	String	None - Use Default ID - The server control uses the content block passed as a URL parameter.
DisplayXslt	Determines how the information on the page is displayed	String	**None**-databind only **ecmNavigation** - lists the title of every content block in the folder *See Also:* "Example of ecmNavigation Display" on page 105 **ecmTeaser** - lists the title of every content block in the folder plus the content summary *See Also:* "Example of ecmTeaser Display" on page 106 **Path to Custom Xslt** - Enter the path to an Xslt that determines the display of the page
EmptyResultMsg	The message that displays if the search returns no hits.	String	The default value is: Your search did not match any documents.
FolderId	The numeric id of the folder that you want to search. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10 The Recursive property determines whether the IndexSearch Server Control will use this folder's child folders in the search.	Integer	Any folder's numeric ID **0** (zero) represents the root folder.

Property	Description	Data Type	Values
Hide	Used to hide output of the index search in design time and run time. For example, you want to pass the results to code behind for further manipulation.	Boolean	**True** = Hide results **False** = show results
HitsMarker	This symbol shows how many search criteria each hit matched. The following example shows the first hit met three of the criteria. The second group met one criterion.  This property is used when the weighted property is set to true. *See Also:* "Weighted" on page 240.	String	* # + **(Custom)** Custom lets you choose the marker to use. The custom marker can be any text, numeric, or symbol character except **&** and **<**. For more Information, see The Ektron CMS400.NET Administrator Manual section "XML Indexing" > "Displaying Search Results"

Property	Description	Data Type	Values
Labelafter Labelbefore LabelBetween LabelContains LabeldateBetween LabelEqual LabelExactPhrase LabelGreaterThan Labelinputdate Labelinputnumber LabelLessThan Labelon LabelSelect	All of these properties let you change their label names. This is useful for Multi-language issues. For example, if you have a French site, you change **Equal** to **Égale.**  Note: These label properties will not change based on the **Language** property's setting. For example, setting the Language property to 1036 will not change these labels to French.	String	Examples of what the user will see on the site in French and English. . 
Language	Set a language for viewing index search results. The language property shows results in design-time (in Visual Studio) and at run-time (in a browser).	Integer	
MaxResults	The Maximum number of content blocks returned (0=unlimited).	Integer	

Property	Description	Data Type	Values
OrderBy	The order of the list to be returned. For example, you want to sort search results by last modified date.	String	• **Title** - The title of the content block • **ID** - The XML content block ID number • **Date Created** - The date the content block was created • **Date Modified** - The date the content block was last modified • **AuthorLName** - The last author's last name • **AuthorFname** - The last author's first name • **StartDate** - The GoLive Date
OrderByDirection	The direction the results will be sorted in. This property works with the **OrderBy** property only.	String	Ascending Descending
Recursive	Determines whether the IndexSearch Server Control will use this folder's child folders in the search.	Boolean	True False
SearchParmXML	Opens a separate window that lets you predefine a search. For example, you want a predefined list of books to appear on a web page. To access the window click the gray button. 		For more information on SearchParmXML, see "SearchParmXML Property" on page 240
ShowSearchBoxAlways	If set to **false**, the search box does not appear on PostBack	Boolean	True False

Property	Description	Data Type	Values
Weighted	If you set this value to **true**, content blocks are ranked by the number of criteria matches. For example, if a content block matches all search criteria, it appears at the top of the results page. A hit marker appears for each match. If a second content block only matches one criterion, it also appears on the results page underneath the first content block. One asterisk appears above the page's title to indicate the single match.	Boolean	**True** - display every content block that matches *at least one* search criteria **False** - show only content blocks that match *all* search criteria
WrapTag	Allows a developer to specify a server control's tag. The default is **Span**.	String	**Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code. **Custom** - Allows you to use a custom tag.
xmlConfigID	ID of the XML configuration whose indexed fields will appear on the search screen.	Integer	

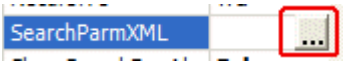
SearchParmXML Property

The SearchParmXML property is used to create a predefined XML search, whose results appear upon page load. Clicking on the gray box in this property opens a window, where you define the search. The parameters in this window are defined by the xmlConfigID you choose. Using this property launches the CMS Explorer login, if you are not already logged in. For more information on the CMS

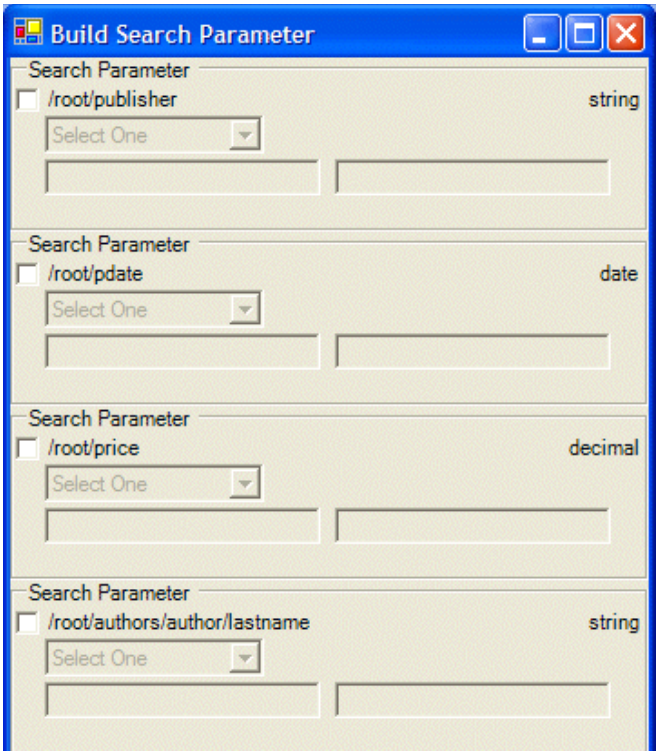
Explorer login, see "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 10

The following example uses the SearchParmXML property to create a list of books, based on publishers with "press" in their name.

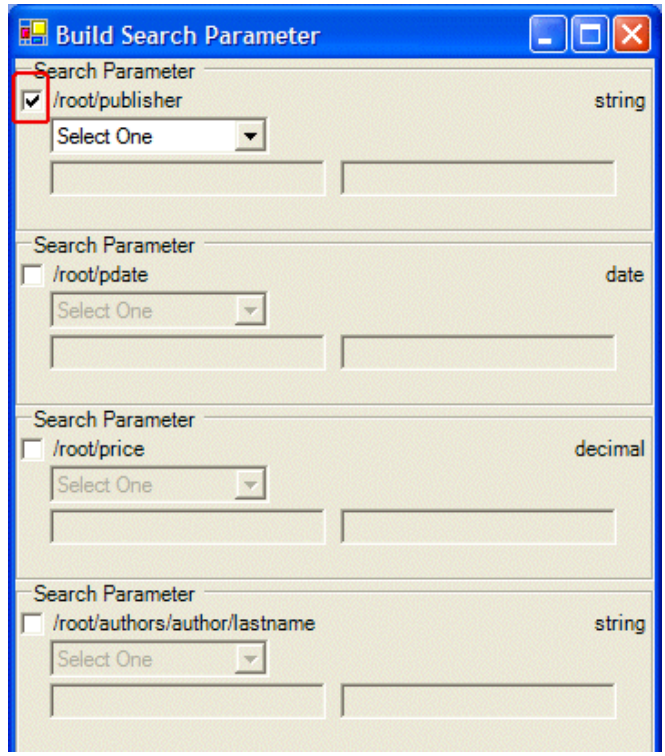
1. To activate the window, click the gray box in the SearchParmXML property.



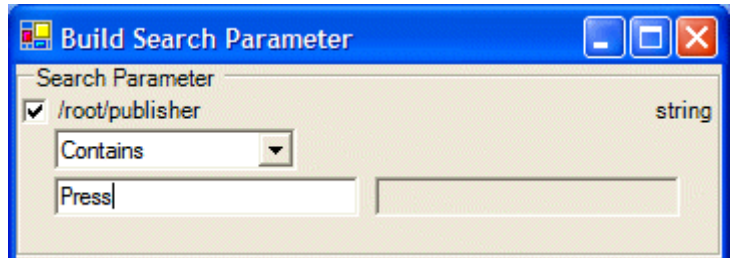
2. The Build Search Parameter screen appears.



3. Click the check boxes for the parameters you want to search with.



4. Define the parameter.



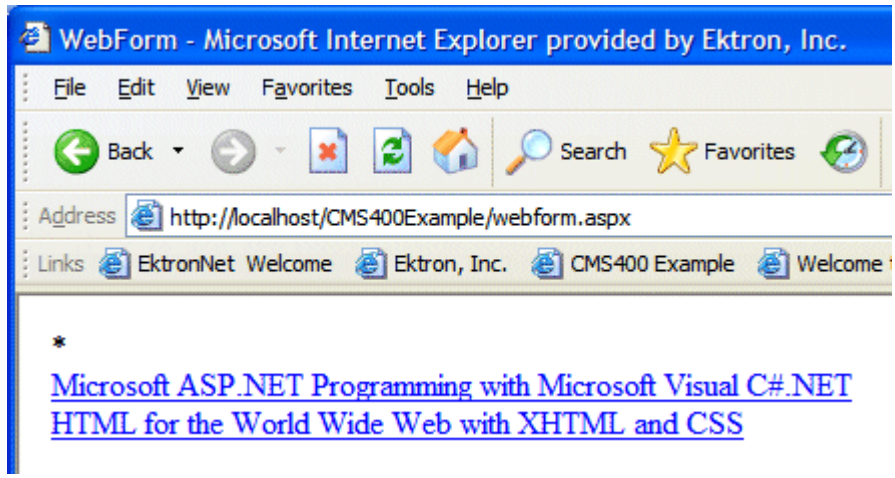
The drop down box in this example contains the following variables:

Select One - Prompts you to choose a variable.

Exact Phrase - You enter a phrase and search for that exact phrase, contained within the XML content.

Contains - You enter a word or phrase and search for any XML content, that has that word or phrase contained within it.

5. Click OK.
6. Build the Web form.
7. Browse to the Web page in your browser.



8. The page appears with a list of XML content blocks that match the parameters you set.

Using the IndexSearch Server Control Programmatically Example

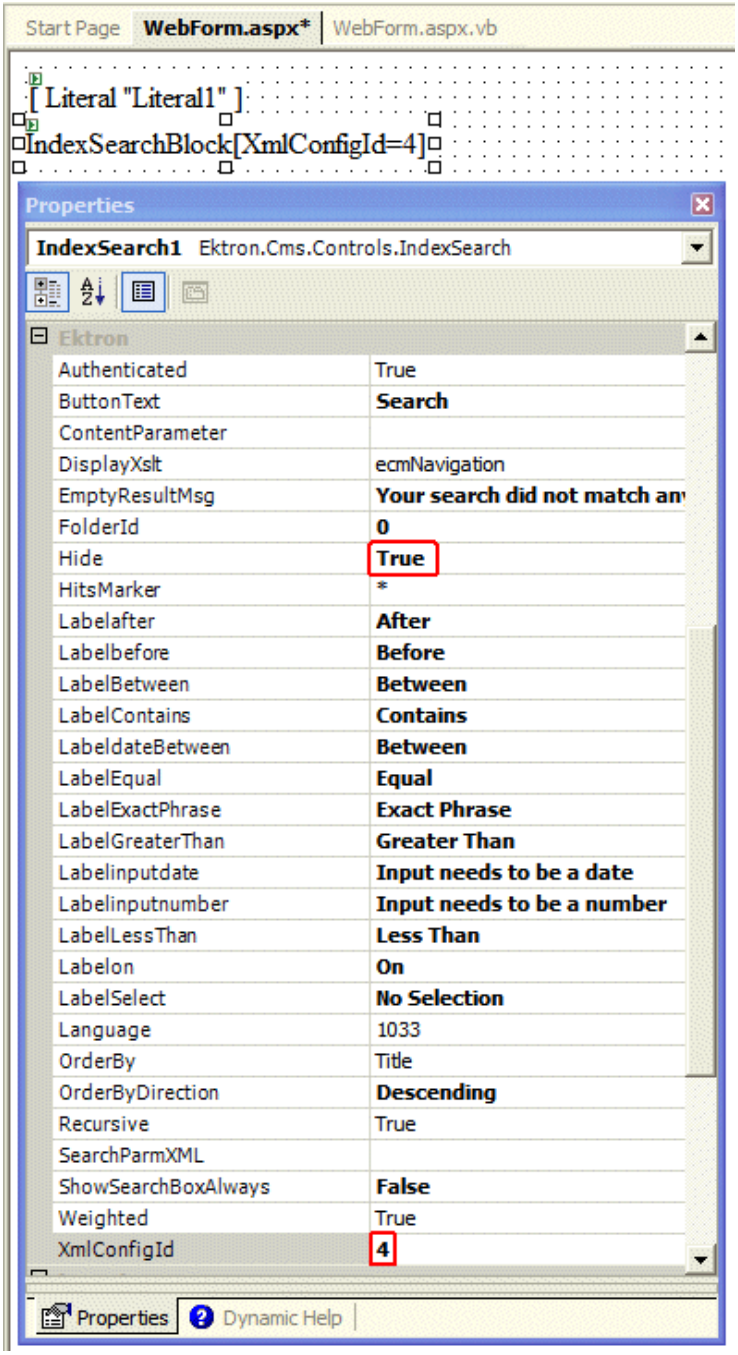
NOTE When creating an IndexSearch Server Control form user interface using code behind only, you need the IndexSearch Server Control to be connected with the page events. Otherwise you will not be able to submit your search parameters.

You can use the IndexSearch Server Control, in code behind, to manipulate the way users view the search results. The following example, using code behind, shows how to return values associated with an XPath. This example shows a list of book subject titles.

1. Open a Web form.
2. Drag and drop an IndexSearch Server Control on your Web form.
3. Set the properties in the properties window.

NOTE In this example, the hide setting must be set to **true**. Remember to set the XmlConfigID. This example uses ID number **4**.

4. Drag and drop a Literal on the Web form.



5. Add the following code to the code behind.

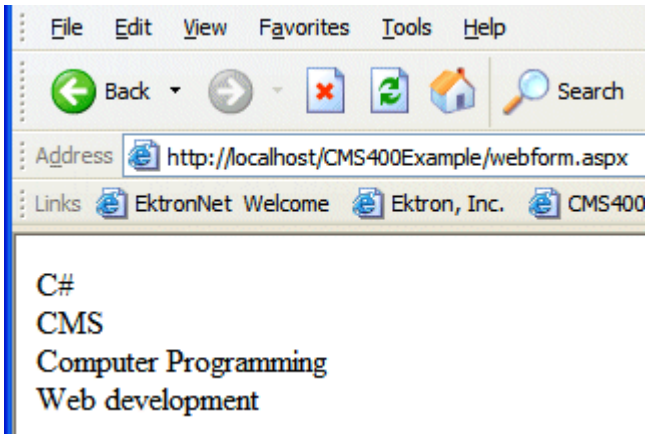
```
IndexSearch1.Parm.XPath = "/root/subject"

Dim arStr As String()

Dim Str As String

Dim strRet As String

arStr = IndexSearch1.GetXPathValues("/root/subject")

For Each Str In arStr

    strRet += Str & "<br/>"

Next

Literal1.Text = strRet
```

Here is a description of what is happening in the code.

- This line defines the parameter XPath location:
  `IndexSearch1.Parm.XPath = "/root/subject"`
- The next three lines create variables:
  ```
  Dim arStr As String()
  Dim Str As String
  Dim strRet As String
  ```
- This line set the variable arStr as an array of the XPathValues:
  ```
  arStr = IndexSearch1.GetXPathValues("/root/
  subject")
  ```
- The for next loop iterates through the array of XPathValues:

  ```
  For Each Str In arStr
      strRet += Str & "<br/>"
  Next
  ```
- The last line displays the values on your web form:
  `Literal1.Text = strRet`

6. Build the project.
7. Browse to your web form.
8. The list of book subject titles appears.



Programmatically Predefined General Search Results Replaced By Specific Search Results

This Code behind example shows a predefined search appearing on a Web page, then being replaced by more specific search results. This example creates a hyperlink list of book publishers. When you click a specific publisher, it replaces the publisher list with a book list from that publisher.

Follow these steps.

1. Add an IndexSearch Server Control to a Web form.
2. Set the following properties:

- DisplayXslt = ecmTeaser
- Hide = True
- HitsMarker = (blank)
- ShowSearchBoxAlways = False
- Weighted = True
- XmlConfigId = 4

3. Add two Literals.
4. Add the following code to the Page Load Event in code behind.

```
Dim bRet As Boolean
   If Request.QueryString("publisher") <> "" Then
```

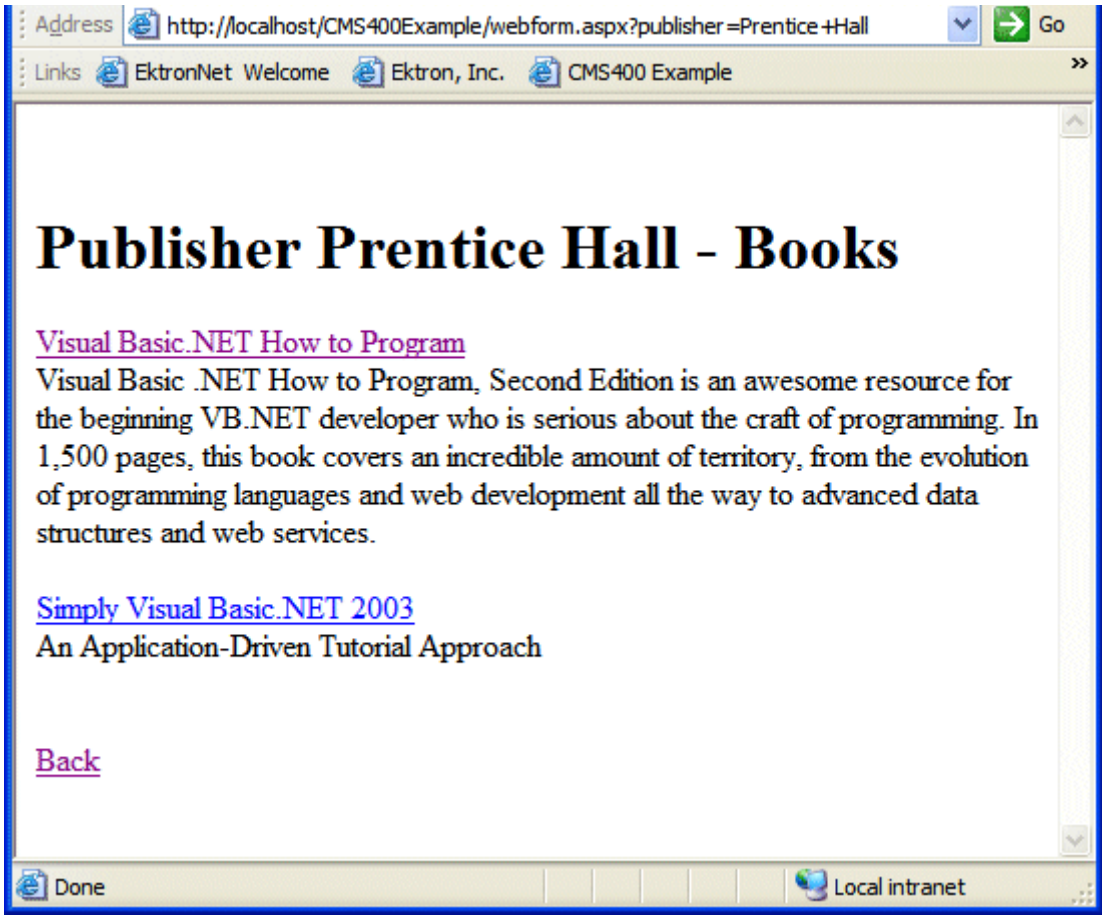
```
      IndexSearch1.Hide = False
      IndexSearch1.Parm.XPath = "/root/publisher"
      IndexSearch1.Parm.DataType =
Ektron.Cms.Common.EkSearch.XMLDataType.String
      IndexSearch1.Parm.SearchType =
Ektron.Cms.Common.EkSearch.XMLSearchRangeType.ExactPhrase
      IndexSearch1.Parm.Value1 = Request.QueryString("publisher")
      bRet = IndexSearch1.AddParm()
      IndexSearch1.Search()

      Literal1.Text = "<h1>Publisher " & Request.QueryString("publisher") &
" - Books</h1>"
      Literal2.Text = "<a href=""webform.aspx"">Back</a>"
   Else
      Literal1.Text = "<h1>Publishers</h1>"
      Dim arStr As String()
      Dim Str As String
      Dim strRet As String
      arStr = IndexSearch1.GetXPathValues("/root/publisher")
      For Each Str In arStr
        strRet += "<a href=""webform.aspx?publisher=" &
Server.UrlEncode(Str) & """>" & Str & "</a><br/>"
      Next
      Literal1.Text += strRet
   End If
```

5. Save your Web form.
6. Build your Web form.
7. Browse to the Web form in your browser.

8. The list of publishers appears.

9. Click on a publisher.

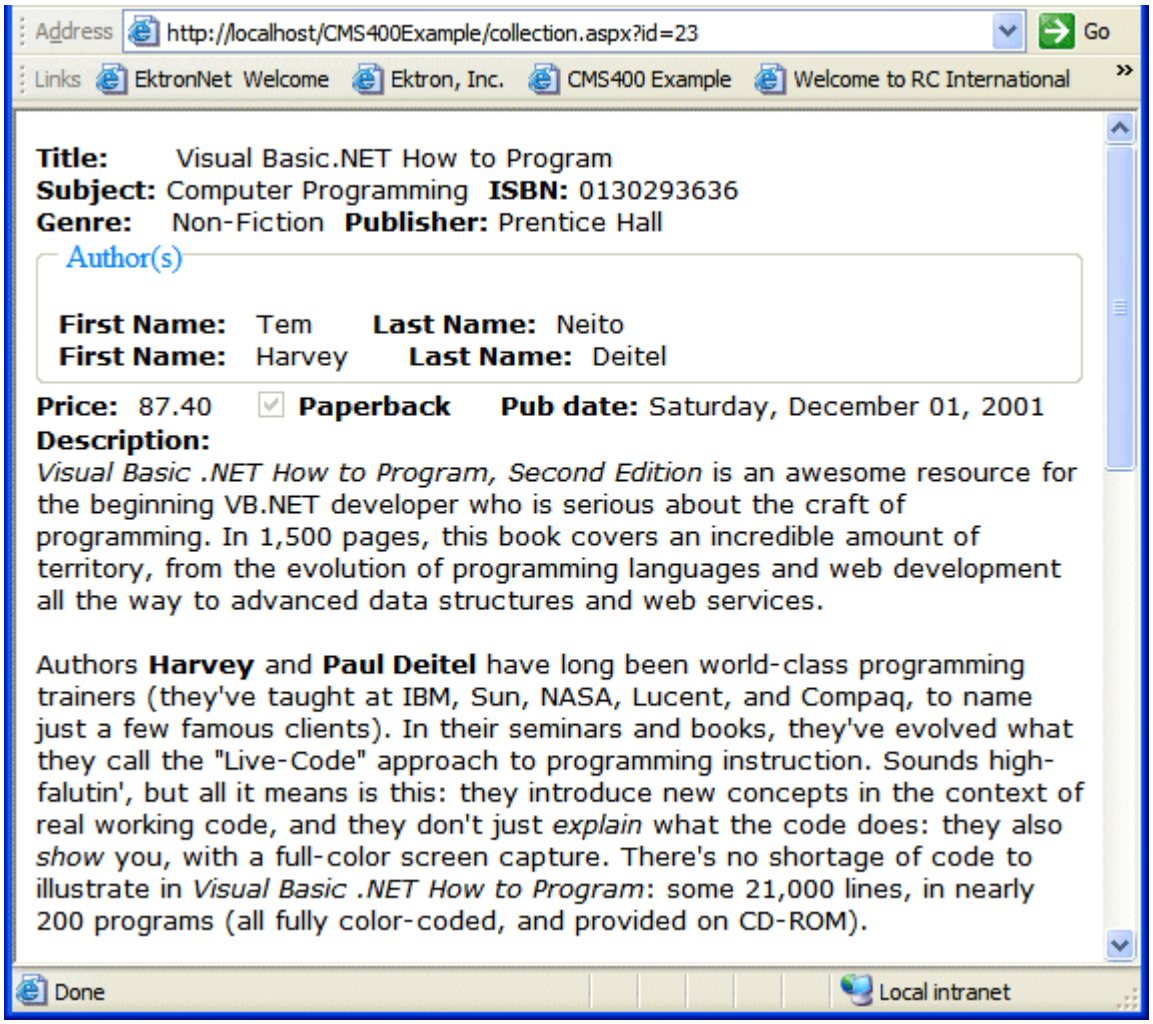
10. A list of books with teasers by that publisher appears.



Notice the second Literal now contains a hyperlink to go back.

11. Click on a book.

12. The XML Content Block for that book appears.



Turn Off Caching In Netscape Navigator and Mozilla Firefox

When using Netscape Navigator or Mozilla Firefox to perform an index search some parameters become cached. To prevent caching in Netscape Navigator and Mozilla Firefox, add the following line to the Page Load Event in code behind:

```
Response.Cache.SetNoStore()
```

Dreamweaver Extension Support

Ektron CMS400.NET has a custom Macromedia® Extension Package, which allows you to not only create your templates in Dreamweaver® or UltraDev™, but also to insert the custom functions through Dreamweaver®.

To do this, all you need is Ektron CMS400.NET, Dreamweaver®/ UltraDev™, and to keep reading.

Ektron supports Dreamweaver extensions for the following scripting languages:

- ASP.NET - for .NET pages
- ASP - for Active Server Pages
- CFM - for ColdFusion pages
- JSP - Java Server Pages
- PHP - Hypertext Preprocessor

The following sections are contained in this chapter:

- ”Requirements” on page 252
- ”Preparing Dreamweaver for Ektron CMS400.NET Functions” on page 253
- ”Installing the Extensions” on page 253
- ”Setup” on page 258
- ”Using the Ektron CMS400.NET Extensions” on page 262
- ”Inserting Custom Functions” on page 271

Requirements

To use a Dreamweaver extension supplied by Ektron, you need to have Dreamweaver 7.0.1 or higher installed, as well as Ektron CMS400.NET.

Preparing Dreamweaver for Ektron CMS400.NET Functions

To have **Ektron CMS400.NET** function symbols appear in Dreamweaver, while editing a Web page, you must edit two `PreviewModeCustomTags.edml` Dreamweaver files. The steps below explain how to edit the files.

1. In Windows Explorer navigate to the following location

```
C:\Program Files\Macromedia\Dreamweaver MX 2004\Configuration\Translators\ASP.NET_VB\
```

2. Open the `PreviewModeCustomTags.edml` file using Notepad.
3. Search for the following line of code in the file.

```
<searchPattern requiredLocation="openTag"><![CDATA[/runat\s*=\s*"?server"?/i]]></
searchPattern>
```

4. Add the following line after it.

```
<searchPattern requiredLocation="tagName"><![CDATA[/^(?!cms:)/i]]></searchPattern>
```

WARNING! This line must be inserted after the line above. Order is important.

5. Save and close the file.
6. Navigate to the following location.

```
C:\Program Files\Macromedia\Dreamweaver MX 2004\Configuration\Translators\ASP.NET_Csharp\
```

7. Open the `PreviewModeCustomTags.edml` file using Notepad.
8. Search for the following line of code in the file.

```
<searchPattern requiredLocation="openTag"><![CDATA[/runat\s*=\s*"?server"?/i]]></
searchPattern>
```

9. Add the following line after it.

```
<searchPattern requiredLocation="tagName"><![CDATA[/^(?!cms:)/i]]></searchPattern>
```

WARNING! This line must be inserted after the line above. Order is important.

10. Save the file.

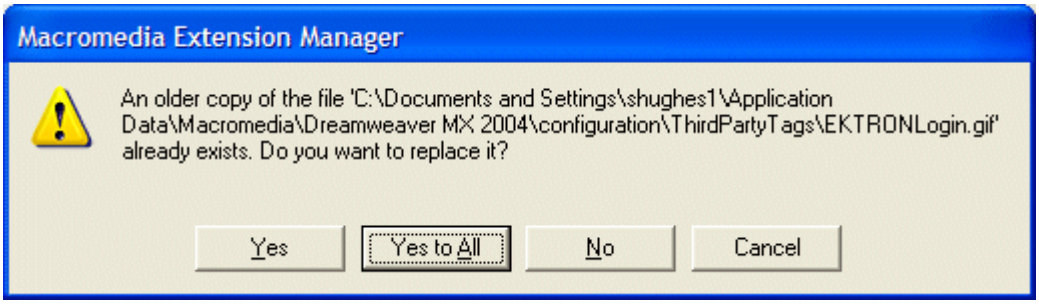
You are now ready to Install the **Ektron CMS400.NET** extension.
See Also: ”Installing the Extensions” on page 253

Installing the Extensions

There are two extensions that can be installed on your server:

- Ektroncms400.mxp - Adds ASP.NET extensions
- C400_ASP_PHP_CF_JSP.mxp - adds ASP, PHP, CF and JSP extensions

You can have both extensions installed at the same time. The extensions do not affect each other. When you install the second extension, you might receive the following message:



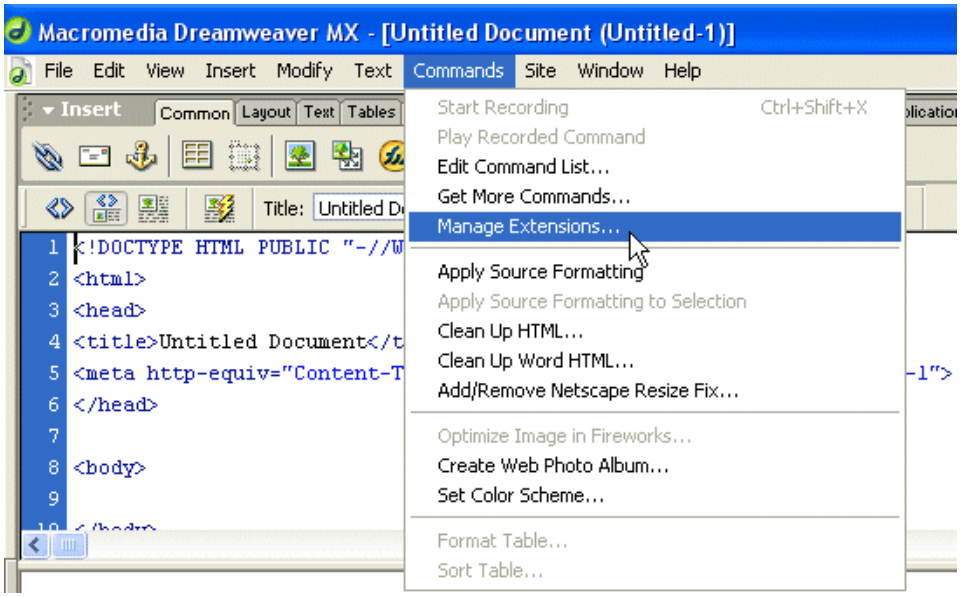
If you see this message or one that say the files are newer, click **Yes to All** to continue. Overwriting the files will not affect the extensions.

Before you can insert the Ektron CMS400.NET custom functions through Dreamweaver®, you must install at least one of the extensions. Listed below are the steps to install an extension.

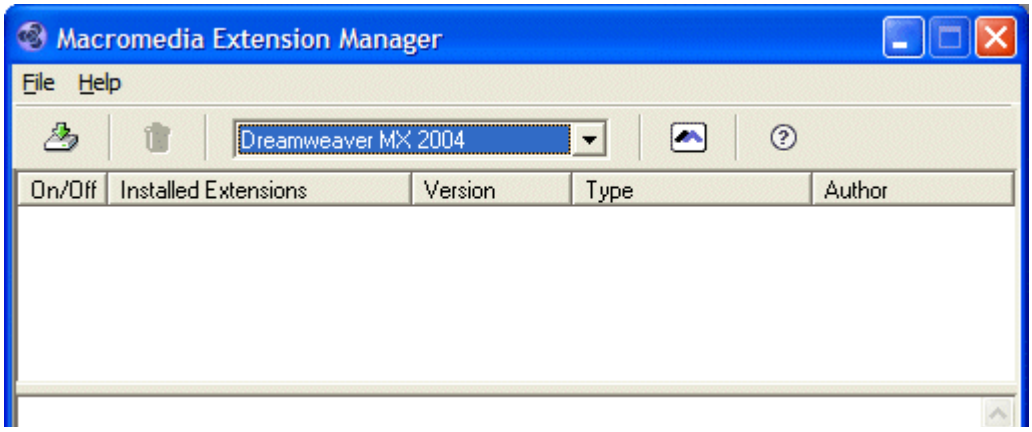
NOTE For additional information about installing an extension, refer to your Dreamweaver® Manual.

To install an extension, follow these steps.

1. Begin by following the path in Dreamweaver:
   Commands > Manage Extensions...

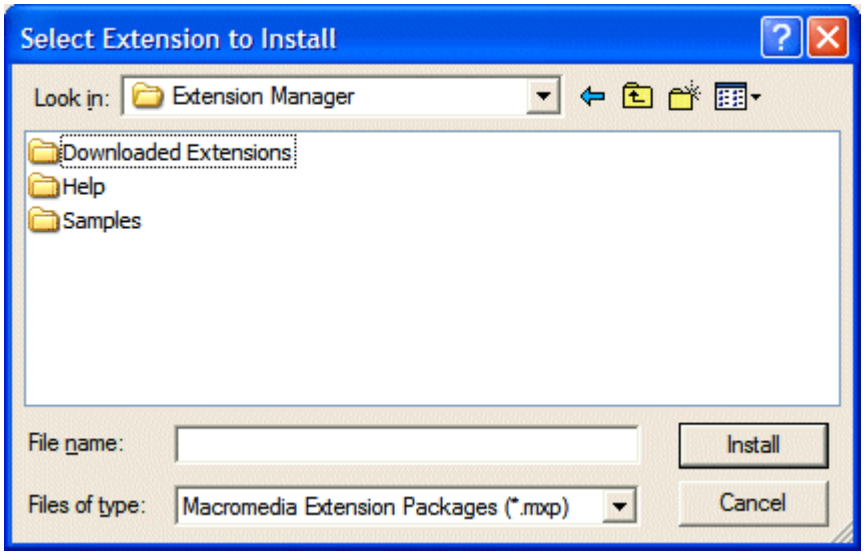


2. The Macromedia Extension Manager is displayed.



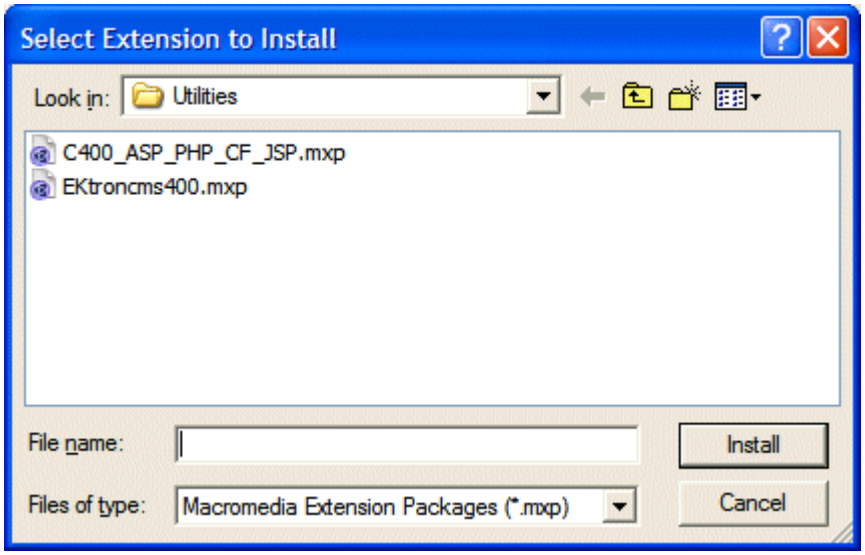
3. From this dialog box, follow the path:
   File > Install Extension...

4. The Select Extension to Install window opens.



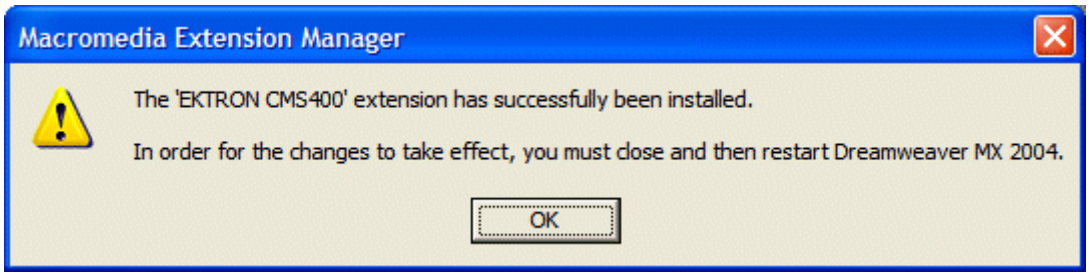
5. Locate the Macromedia® Extension Package (.mxp) file that corresponds to Ektron CMS400.NET. By default, the file gets installed to the following directory:

   C:\Program Files\Ektron\CMS400v50\Utilities

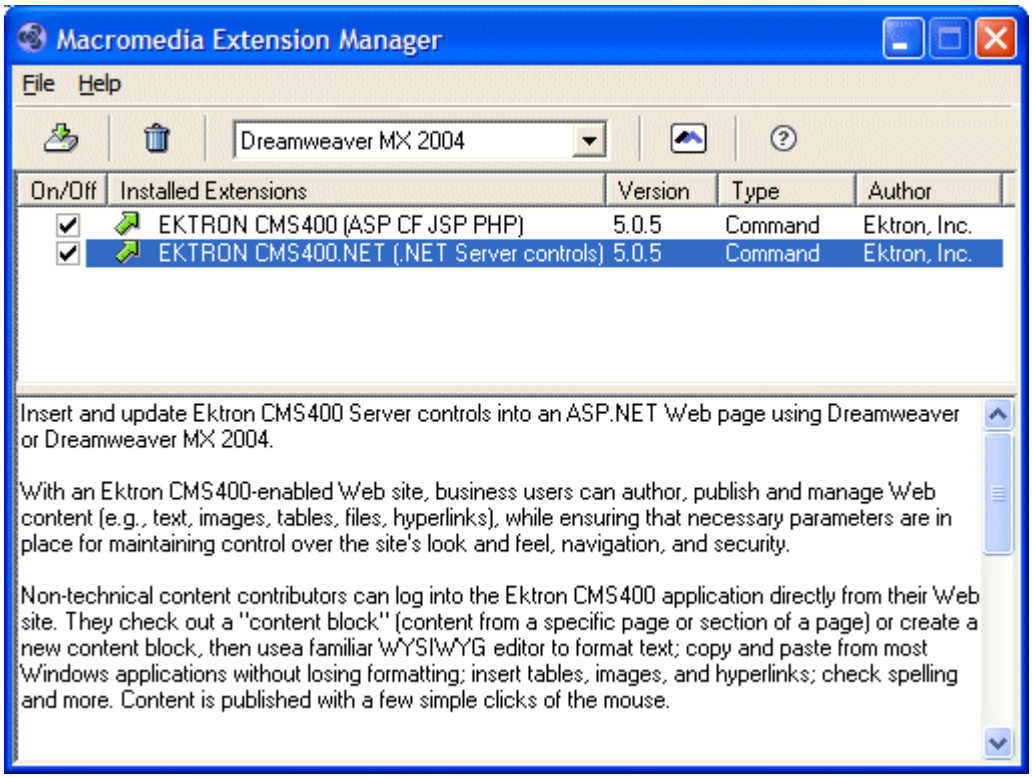


6. After you select the file, click the **Install** button.
7. The installation begins with the Macromedia® Extensions Disclaimer. Click **Accept**.

8. The extension begins to install. Once installed, you see a confirmation box.



9. Click **OK** to continue.
10. The Macromedia Extension Manager is again displayed, but with the added extension.



11. If needed, exit Dreamweaver®, then start it back up.

Now you can use the Ektron CMS400.NET Extension Pack.

Refer to "Using the Ektron CMS400.NET Extensions" on page 262 for more information on using Dreamweaver® to create and edit templates for Ektron CMS400.NET.

Setup

Before you can use Dreamweaver®/UltraDev™ to create and edit Ektron CMS400.NET templates, you should configure the Ektron CMS400.NET Extension. Listed below are the configurative options for the extension in Dreamweaver:

- Username
- Password
- Domain
- Server File
- Server
- Work Online
- Include
- Check for Include

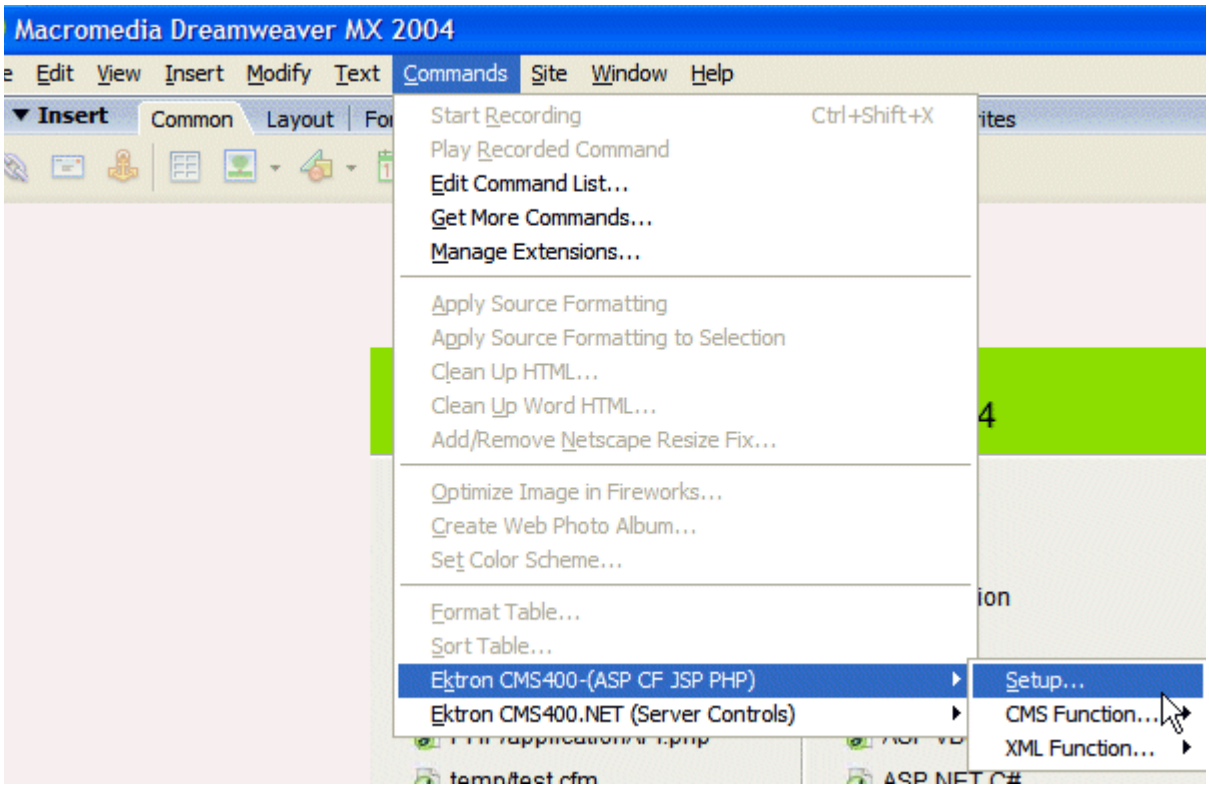
To change these options, follow these steps.

1. If you use ASP, ColdFusion, JSP or PHP, follow this path:

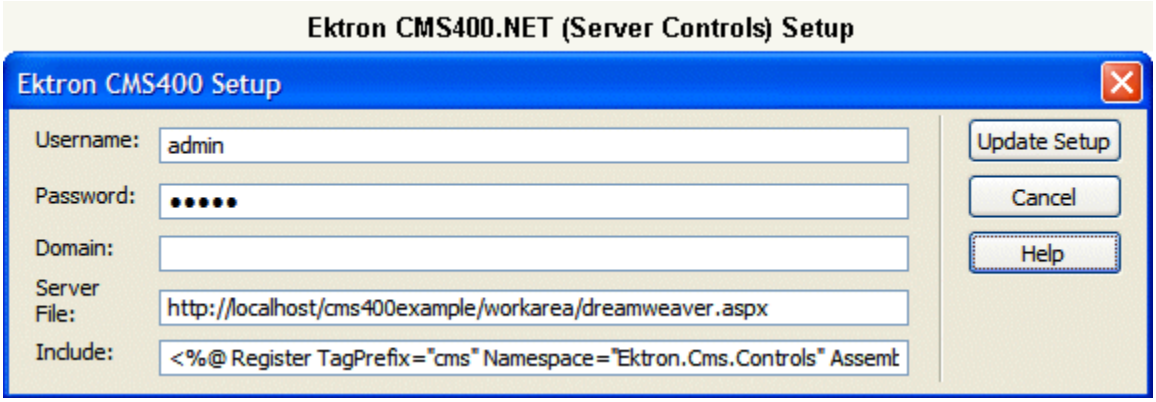
   Commands > Ektron CMS400 - (ASP CF JSP PHP) > Setup...

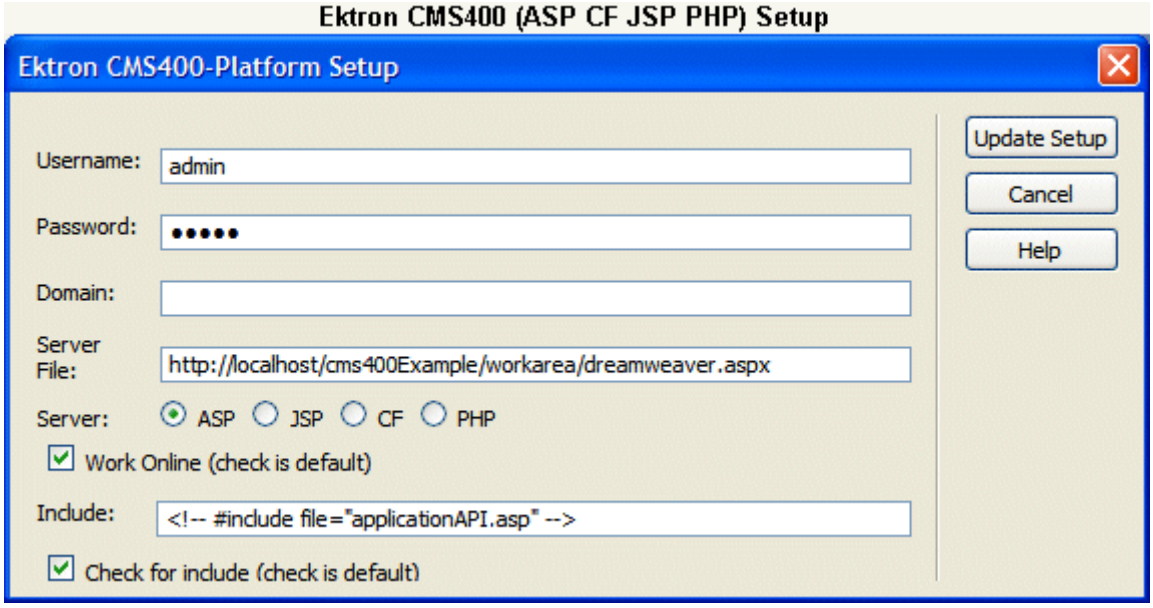
   If you use ASP.NET, follow this path:

   Commands > Ektron CMS400.NET (Server Controls) > Setup...



2. The Setup dialog box is displayed on the screen.





3. Use the following table to help you update your setup information. This table refers to Ektron CMS400.NET (Server Controls) setup. For Ektron CMS400 - (ASP CF JSP PHP), see "Use the following table to help you update your setup information. This table refers to Ektron CMS400 - (ASP CF JSP PHP) Setup." on page 261

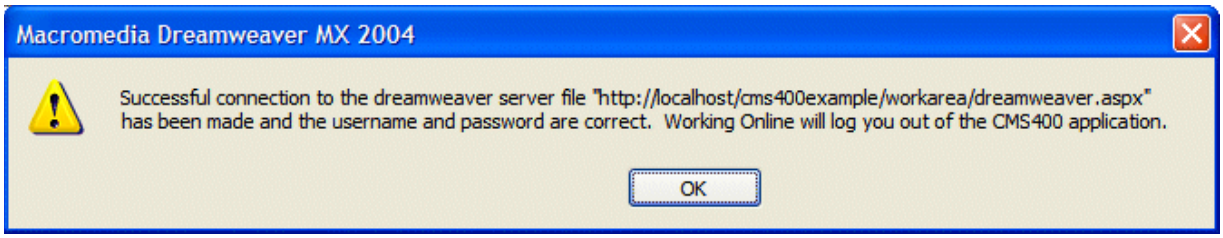
Field	Description
Username	Valid username of a user of your Ektron CMS400.NET Web site. When using the extension, the username is used to log in to the database to retrieve information about content blocks and other Web site information
Password	Enter the password for the user.
Domain	Only required if Active Directory has been enabled for your Web site. Enter the domain corresponding to the username entered.
Server File	Specify the location and file name for the dreamweaver.aspx file needed for the extension to properly work and make connection with your Ektron CMS400.NET database.
Include	Each Ektron CMS400.NET template requires an include file to point to the API with the definitions for the custom functions. Here, specify the include file.

Use the following table to help you update your setup information. This table refers to Ektron CMS400 - (ASP CF JSP PHP) Setup.

Field	Description
Username	Valid username of a user of your Ektron CMS400.NET Web site. When using the extension, the username is used to log in to the database to retrieve information about content blocks and other Web site information
Password	Enter the password for the user.
Domain	Only required if Active Directory has been enabled for your Web site. Enter the domain corresponding to the username entered.
Server File	Specify the location and file name for the dreamweaver.aspx file needed for the extension to properly work and make connection with your Ektron CMS400.NET database.
Server	Choose the language of the server. • **ASP** - for Active Server Pages • **CFM** - for ColdFusion pages • **JSP** - Java Server Pages • **PHP** - Hypertext Preprocessor Note: If you want to use a different language, you need to run this setup again and choose a different language.
Work Online	By checking this box, you will have access to the database to obtain the correct folder and content information. For example, when adding a content tag, you are able to see the content titles in a drop down list. This makes it easy to insert the right content tag. Checked is the default. ☑ **Checked** - work offline ☐ **Unchecked** - cannot work offline
Include	Each template requires an include file to point to the API with the definitions for the custom functions. Here, specify the include file. • For ASP use: `<!-- #include file="applicationAPI.asp" -->` • For CFM use: Leave blank • For JSP use: `<%@ include file="applicationAPI.jsp" %>` • For PHP use: `<?php include "applicationAPI.php"; ?>`

Field	Description
Check for Include	If checked the Include statement is added to your page. Checked is the default. ☑ **Checked** - include statement added to your page ☐ **Unchecked** - include statement is not added to your page

4. Click Update Setup.
5. A message appears.



NOTE A message other than this is probably due to an incorrect server file, username, or password. Recheck your settings and try again.

6. Click **OK** to continue.
7. The setup dialog box closes. You are ready to create templates using Macromedia Dreamweaver.

Using the Ektron CMS400.NET Extensions

Now that the Macromedia® Extension Package has been installed for Dreamweaver, you can create and edit templates for Ektron CMS400.NET.

To explain how to use this, we use templates that are in the Ektron CMS400.NET installation, but with the custom functions stripped out.

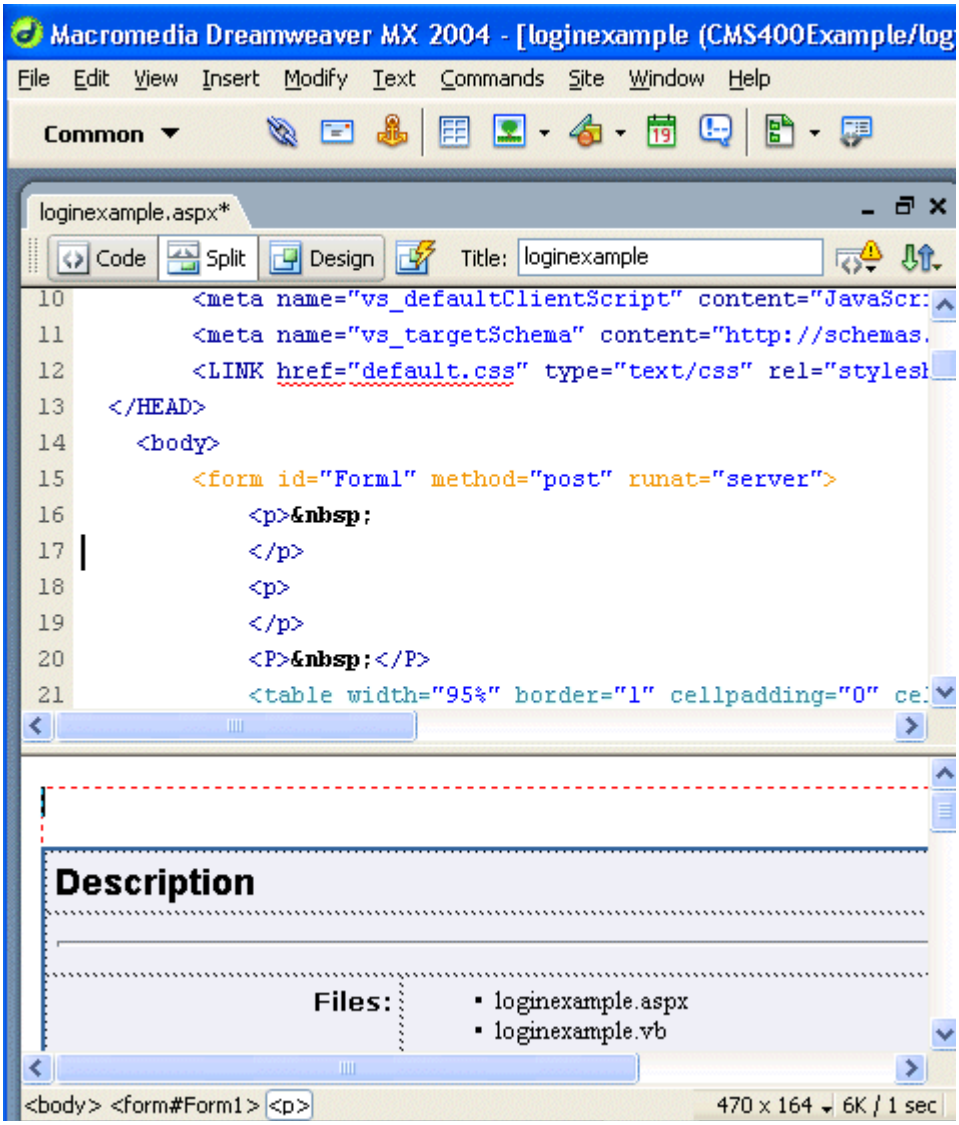
NOTE You can also create your own templates instead of using the ones in the Ektron CMS400.NET installation.

The following subsections are contained in this section:

- ”Custom Function Symbols” on page 264
- ”Ektron CMS400.NET Command Menus” on page 266

In Dreamweaver, begin by opening the file:

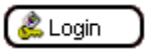
\webroot\CMS400Example\loginexample.aspx



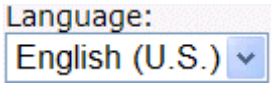
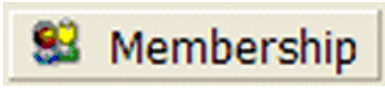
NOTE The custom function that is inserted into this template by default has been stripped out.

Custom Function Symbols

During the process of inserting functions, symbols are used to represent the custom display functions. The following table explains each symbol.

Symbol	Represents	More Information
	Login/Logout button	"Login" on page 272
	Static content block.	"Inserting a Content Block" on page 275
	Dynamic content block	"Inserting a Dynamic Content Block" on page 279
	Collection	"Inserting a Collection" on page 281
	List Summary	"Inserting a List Summary" on page 285
	Search	"Inserting a Search Function" on page 296
	Search display function	"Inserting a Search Display Function" on page 305
No symbol appears when Metadata is inserted in a Web page.	Metadata	"Inserting a Metadata Function for ASP.NET pages" on page 290

Symbol	Represents	More Information
	Single Summary	”Inserting a Single Summary Function” on page 307
	Random Content	”Inserting a Random Content Function” on page 310
	Random Summary	”Inserting a Random Summary Function” on page 312
	Event Calendar	”Inserting an Event Calendar Function” on page 314
	Form Content Block	”Inserting a Form” on page 317
	XML Index Search	”Inserting an XML Index Search” on page 324
	Display By Meta Value	”Inserting Display By Meta Value” on page 330
	Multipurpose content block.	”Inserting a Multipurpose content block” on page 321
	Advanced Calendar Tag	”Inserting an Advanced Calendar Tag” on page 322

Symbol	Represents	More Information
	Standard Menu	"Inserting a Standard Menu" on page 335
	DHTML Menu	"Inserting a DHTML Menu" on page 338
	Language Select	"Inserting a Language Select Box" on page 344
	Language API	"Inserting a Language API" on page 346
	RSS Aggregator	"Inserting an RSS Aggregator" on page 349
	BreadCrumb	"Inserting a BreadCrumb Trail" on page 351
	Membership	"Inserting a Membership Control" on page 355

Ektron CMS400.NET Command Menus

The following section explains how to locate Ektron CMS400.NET commands in Dreamweaver.

There are two Ektron command menus available. One menu is for ASP.NET pages and the other is used for ASP, ColdFusion, JSP and PHP pages:

- **Commands > Ektron CMS400.NET (Server Controls) > CMS Function...**

- **Commands > Ektron CMS400 - (ASP CF JSP PHP) > CMS Function...**

To locate a command menu in Dreamweaver, perform the following steps:

1. Open Dreamweaver.
2. In the top menu, follow one of these paths:

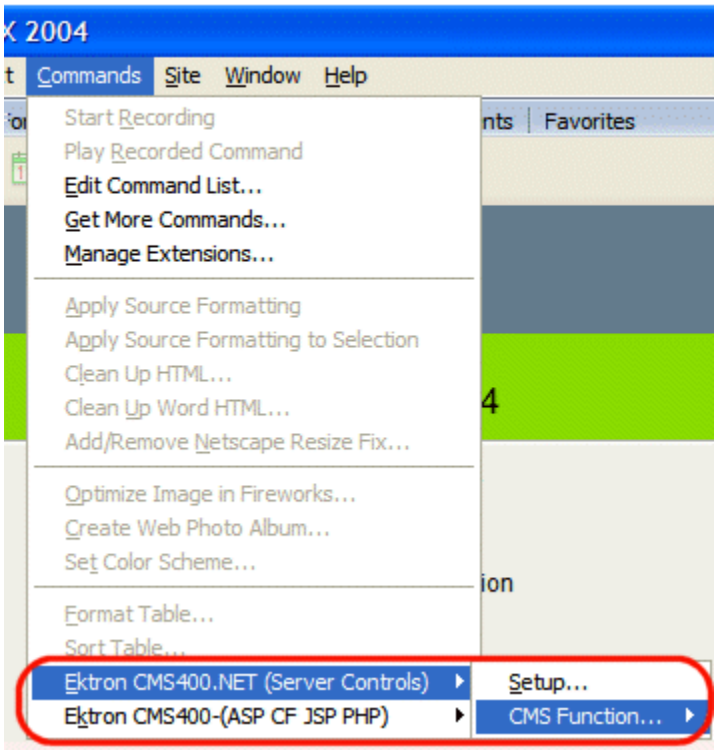
   If you are using ASP.NET pages, click

   Commands > Ektron CMS400.NET (Server Controls) > CMS Function...

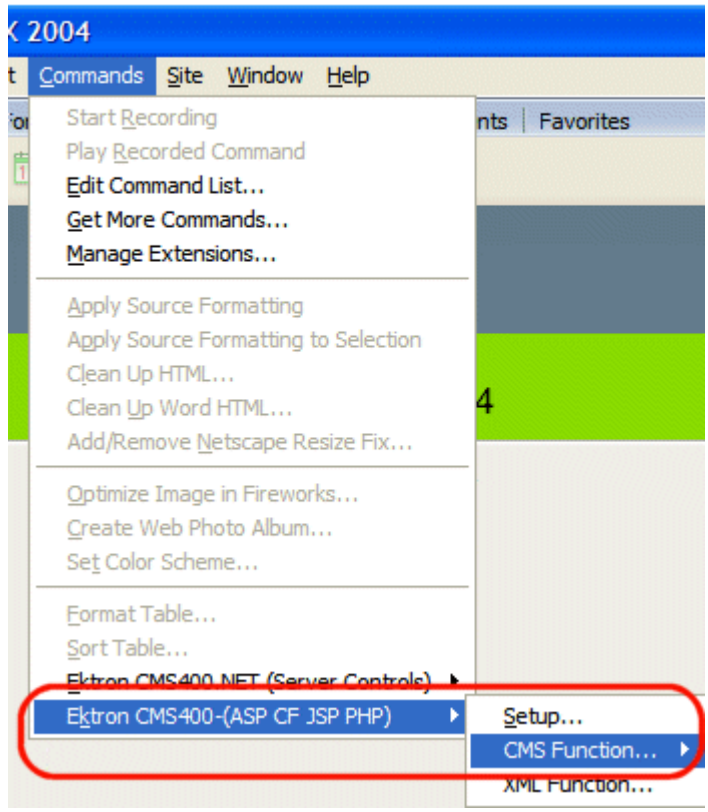
   If you are using ASP, ColdFusion, JSP or PHP pages, click

   Commands > Ektron CMS400 - (ASP CF JSP PHP) > CMS Function...
3. The Ektron CMS400.NET commands menu is displayed.

If you use ASP.NET pages:



If you use ASP, CF, JSP or PHP pages:



4. Click on the command for the task you would like to perform. The following table explains each command.

Command	Description	ASP.NET	ASP - CF JSP - PHP	More Information
Setup	Opens Ektron CMS400.NET Setup information dialog for Dreamweaver.	✓	✓	"Setup" on page 258
CMS Function...				
Login	Inserts an Ektron CMS400.NET login function into the template.	✓	✓	"Login" on page 272

Command	Description	ASP.NET	ASP - CF JSP - PHP	More Information
Content	Inserts an Ektron CMS400.NET content block function into the template.	✓	✓	”Inserting a Content Block” on page 275
Dynamic	Inserts an Ektron CMS400.NET dynamic content block placeholder function into the template.		✓	”Inserting a Dynamic Content Block” on page 279
Collection	Inserts an Ektron CMS400.NET collection function into the template.	✓	✓	”Inserting a Collection” on page 281
List Summary	Inserts an Ektron CMS400.NET list summary function into the template.	✓	✓	”Inserting a List Summary” on page 285
Metadata	Inserts an Ektron CMS400.NET metadata function into the template.	✓	✓	”Inserting a Metadata Function for ASP.NET pages” on page 290
Search	Inserts an Ektron CMS400.NET search function into the template.	✓	✓	”Inserting a Search Function” on page 296
Search Display Tag	Inserts an Ektron CMS400.NET search display function into the template.		✓	”Inserting a Search Display Function” on page 305
Single Summary Tag	Inserts an Ektron CMS400.NET single summary function into the template.		✓	”Inserting a Single Summary Function” on page 307
Random Content Tag	Inserts an Ektron CMS400.NET random content function into the template.		✓	”Inserting a Random Content Function” on page 310

Command	Description	ASP.NET	ASP - CF JSP - PHP	More Information
Random Summary Tag	Inserts an Ektron CMS400.NET random summary function into the template.		✓	”Inserting a Random Summary Function” on page 312
Calendar	Inserts an Ektron CMS400.NET event calendar function into the template.	✓	✓	”Inserting an Event Calendar Function” on page 314
Form	Inserts an Ektron CMS400.NET form content block into the template.	✓	✓	”Inserting a Form” on page 317
Multipurpose Tag	Insert a content block or form content block into the template.		✓	”Inserting a Multipurpose content block” on page 321
Advance Calendar Tag	Inserts a calendar into the template.		✓	”Inserting an Advanced Calendar Tag” on page 322
XML Index Search	Inserts an Ektron CMS400.NET XML Index Search into the template.	✓		”Inserting an XML Index Search” on page 324
Display By Meta Value	Inserts an Ektron CMS400.NET display by meta value block into the template.	✓		”Inserting Display By Meta Value” on page 330
Standard Menu	Inserts an Ektron CMS400.NET standard menu into the template.	✓		”Inserting a Standard Menu” on page 335
DHTML Menu	Inserts an Ektron CMS400.NET DHTML menu into the template.	✓		”Inserting a DHTML Menu” on page 338
Language Select	Inserts an Ektron CMS400.NET language select box into the template.	✓		”Inserting a Language Select Box” on page 344

Command	Description	ASP.NET	ASP - CF JSP - PHP	More Information
Language API	Inserts an Ektron CMS400.NET Language API into the template.	✓		”Inserting a Language API” on page 346
RssAggregator	Aggregates an RSS feed from a RSS Web site.	✓		”Inserting an RSS Aggregator” on page 349
BreadCrumb	Creates a breadcrumb trail of where site visitors have been and allows them to navigate back to previous pages.	✓		”Inserting a BreadCrumb Trail” on page 351
Membership	Adds a Membership form to your Web form. This allows site visitors to self-register themselves to your site.	✓		”Inserting a Membership Control” on page 355
XML Function...				
Content XSLT Tag	Inserts an XML content block with your choice of XSLT in the template. NOTE: Used to insert an XML content block with an XSLT other than the default applied to it.		✓	”Insert Content XSLT Tag...” on page 360

Inserting Custom Functions

CMS Function

The CMS Function commands let you insert standard Ektron CMS400.NET display functions into your template.

Accessing CMS Content Functions

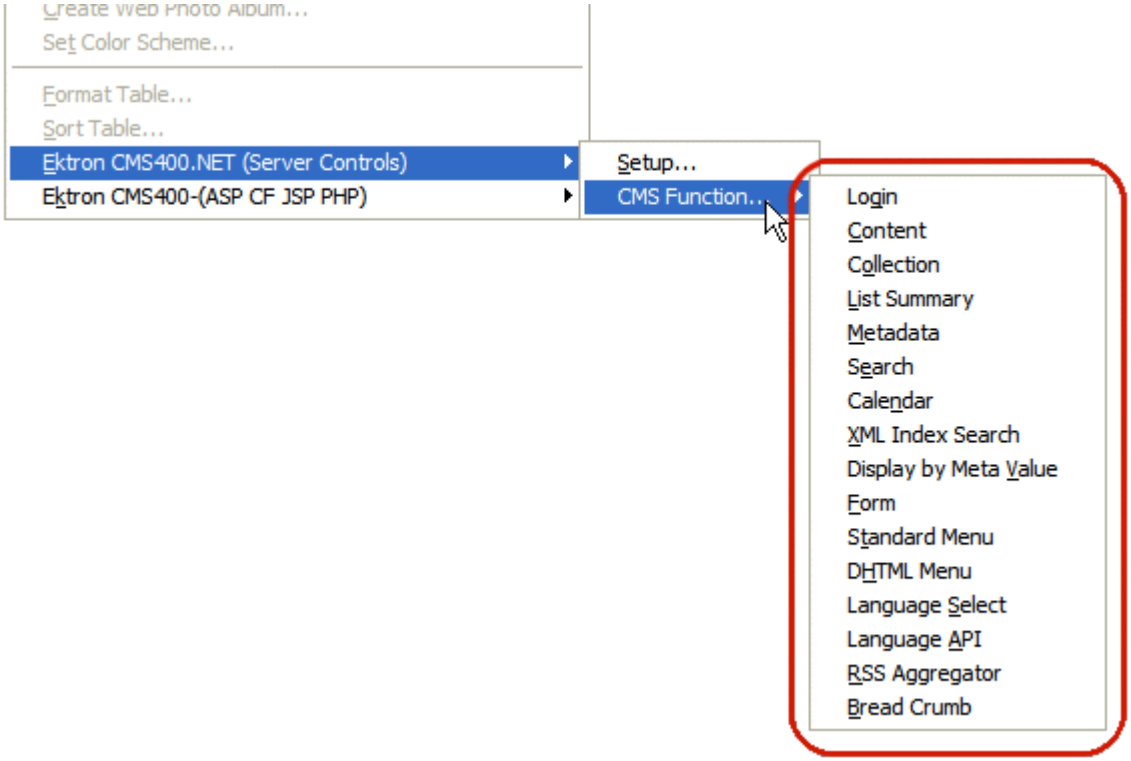
To access the CMS Content commands menu, follow one of these paths:

- **Commands > Ektron CMS400 - (ASP CF JSP PHP) > CMS Function...**

  or

- **Commands > Ektron CMS400.NET (Server Controls) > CMS Function...**

A list of all CMS Content functions that can be inserted is displayed.



The following sections explain how to insert each function.

Login

For a description of the Login button, see "Login Server Control" on page 188.

This tag appears on the following menu(s):

- **Commands > Ektron CMS400 - (ASP CF JSP PHP) > CMS Function...**

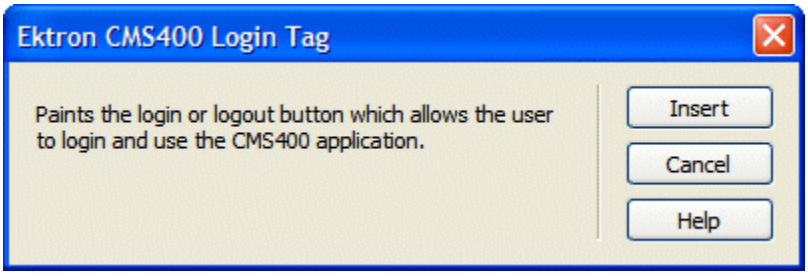
  or

- **Commands > Ektron CMS400.NET (Server Controls) > CMS Function...**

To insert a login button:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET functions, click **Login**.
3. The Insert Ektron CMS400.NET Login box appears.

IMPORTANT! When inserting a login function using ASP, CF, JSP or PHP, the dialog box that appears has no settings.

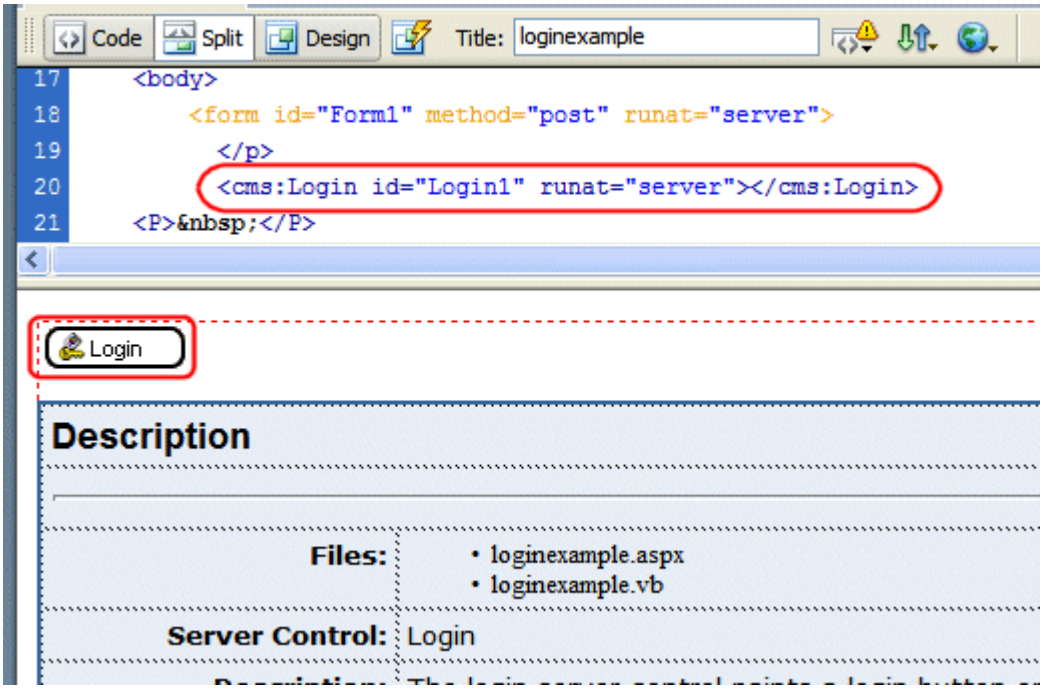


4. Use the following table to assist you with inserting a Login button.

Field	Description
ID Used in ASP.NET only.	A unique name for the function you are inserting.
Hide Used in ASP.NET only.	Used to hide a Login button at run time. ☑ **Checked** = Hide Login button ☐ **Unchecked** = Show Login button Note: If Hide and Visible are both checked, the property is hidden.
Visible Used in ASP.NET only.	Used to show or hide the function at run time. ☑ **Checked** = Hide function ☐ **Unchecked** = Show function Note: If Hide and Visible are both checked, the property is hidden.
Style (Positioning Info - GridLayout) Used in ASP.NET only.	Set position information of a control as absolute, when using a GridLayout Web page. For example: style="Z-INDEX: 101; LEFT: 328px; POSITION: absolute; TOP: 240px"
Wrap Tag Used in ASP.NET only.	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code.

5. Click **insert**.

   A login icon indicates where the Login button appears on the template.



Shown below is an example of the code that is inserted:

Platform	Code View
.NET	<cms:Login id="Login1" runat="server"></cms:Login>
ASP	<% ecmlogin %>
ColdFusion	<cfmodule template="#request.ecm.AppPath#ecmLogin.cfm”>
JSP	<%= ecmLogin() %>
PHP	<?php ecmlogin() ?>

Inserting a Content Block

This tag appears on the following menu(s):

- **Commands > Ektron CMS400 - (ASP CF JSP PHP) > CMS Function...**

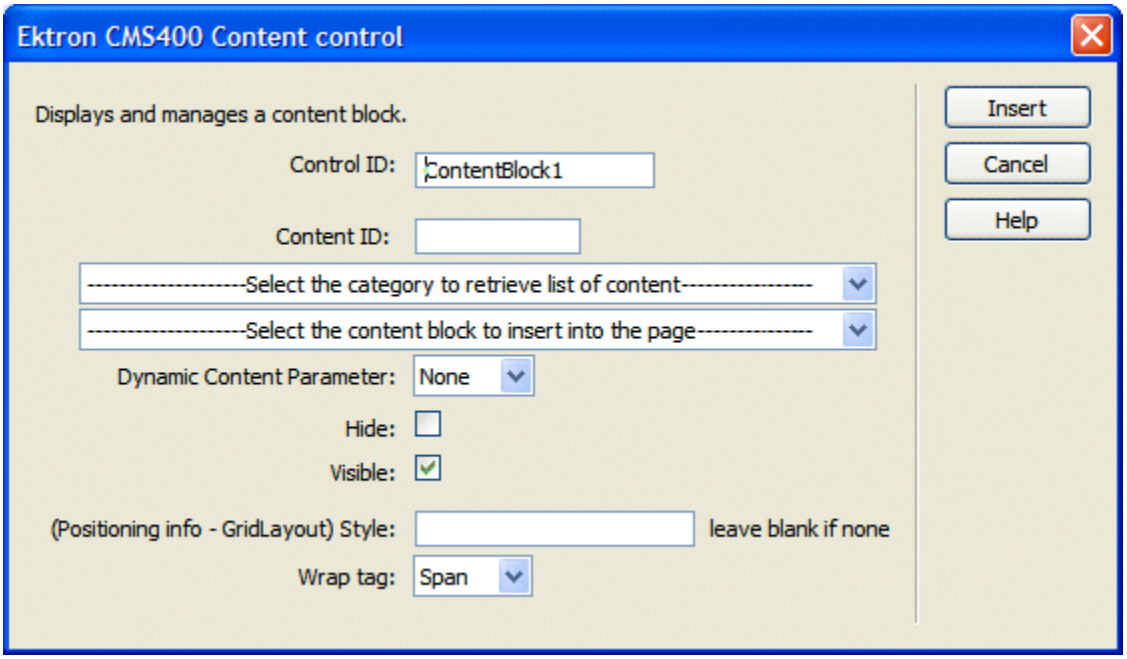
  or

- **Commands > Ektron CMS400.NET (Server Controls) > CMS Function...**

For a description of the Content Block function, see "ContentBlock Server Control" on page 113.

To insert a content block:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Functions, click **Content**.
3. The Insert Content box is displayed



4. Use the following table to help you insert a content block.

Field	Description
Control ID Used in ASP.NET only.	A unique name for the function you are inserting.

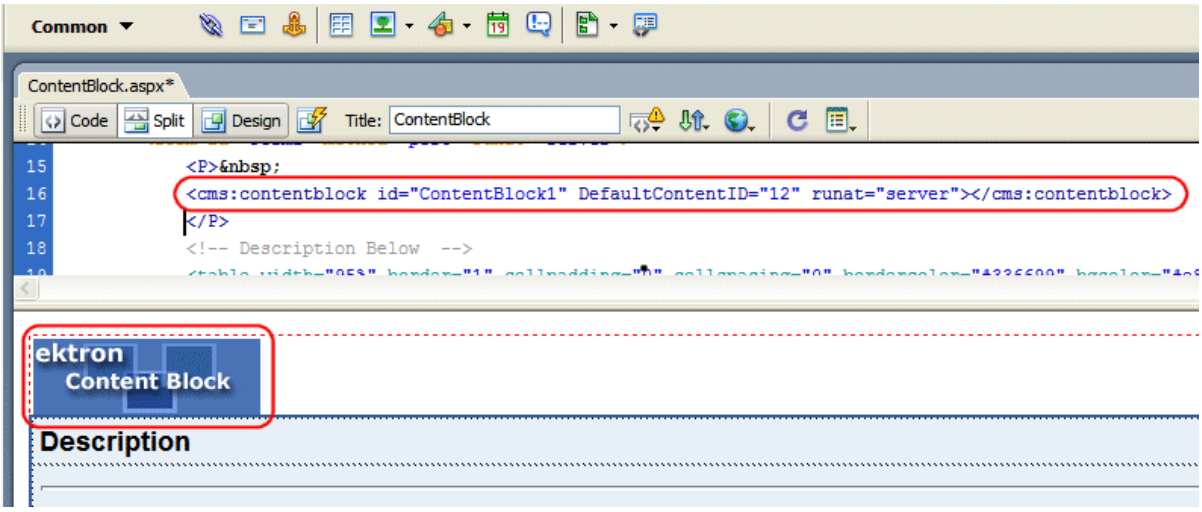
Field	Description
Content ID	Specify the ID number of the content block that you wish to use as a content block on the Web page. If you do not know the Content ID, you can navigate to it using the Select Category and Select Content drop down boxes.
Select Category	Choose the folder where the content is located. If you insert the content ID, you do not need to select a category.
Select Content	Select the content block to use from the list. If you insert the content ID, you do not need to select content.
Dynamic Content Parameter Used in ASP.NET only.	Determines if content is displayed as static or passed dynamically. By making content dynamic, when the content displayed on a Web page contains a link to another piece of content, you can display the second piece of content in the same Web page space. **None**: Displays a static content block. **id**: Content block is passed as a URL parameter. **Important**: If you are using an ASP, CF, JSP or PHP page, you must use the Dynamic Content Block Tag to create a dynamic content block. See "Inserting a Dynamic Content Block" on page 279
Hide Used in ASP.NET only.	Used to hide a content block in design time and run time. **True** = Hide content block **False** = Show content block Note: If Hide and Visible are both checked, the property is hidden.
Visible Used in ASP.NET only.	Used to show or hide the function at run time. ☑ **Checked** = Hide function ☐ **Unchecked** = Show function Note: If Hide and Visible are both checked, the property is hidden.
Style (Positioning Info - GridLayout) Used in ASP.NET only.	Set position information of a control as absolute, when using a GridLayout Web page. For example: style="Z-INDEX: 101; LEFT: 328px; POSITION: absolute; TOP: 240px"

Field	Description
Wrap Tag Used in ASP.NET only.	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code.

NOTE You see only the list of Ektron CMS400.NET content blocks if you are working online.

5. Click **insert**.

   A graphic indicates the static content block's location on the template.



Shown below is an example of the code that is inserted.

Platform	Code View
.NET	<cms:contentblock id="ContentBlock1" DefaultContentID="12" runat="server"></cms:contentblock>
ASP	<% ecmContentBlock(1) %>
ColdFusion	<cfmodule id="1" template="#request.ecm.AppPath#ecmContentBlock.cfm">
JSP	<%= ecmContentBlock(12) %>

Platform	Code View
PHP	<?php ecmContentBlock(1); ?>

Inserting a Dynamic Content Block

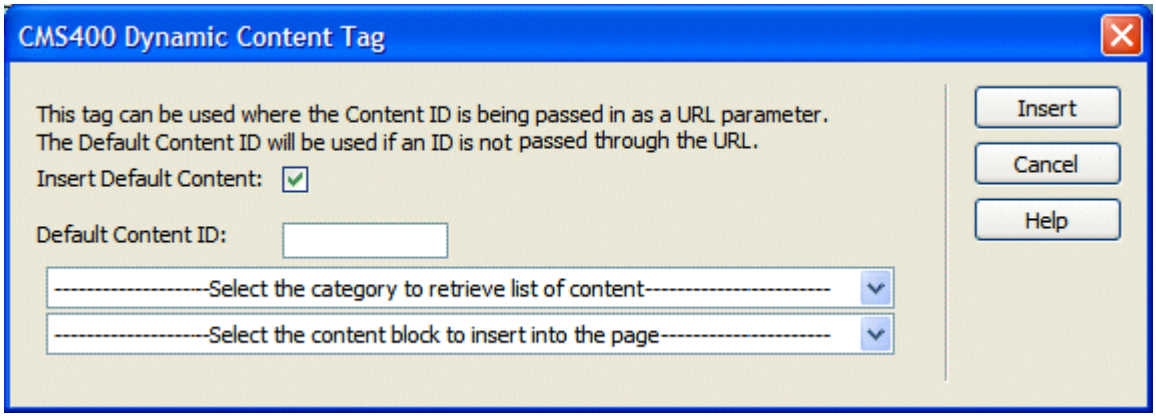
This tag appears on the following menu(s):

- **Commands > Ektron CMS400 - (ASP CF JSP PHP) > CMS Function...**

For a description of a dynamic content block, see "Dynamic Content Block" on page 380

To insert a dynamic content block tag:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Content functions, click on **Insert Dynamic Tag...**
3. The Insert Ektron Dynamic Tag box is displayed



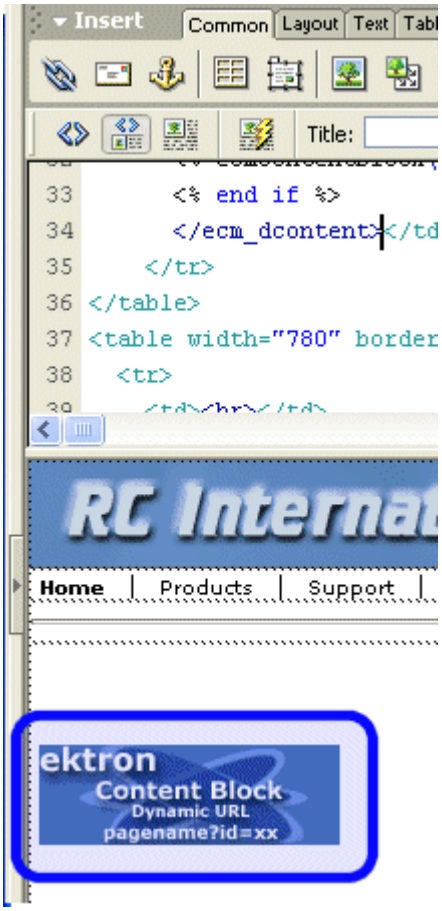
4. Use the following table to assist you with inserting a dynamic content block function.

Field	Description
Insert Default Content	If you want a static content block to be displayed if no ID is passed through the URL, check off this box.
Default Content ID	Specify the ID number of the content block that you wish to use as a default content block on the Web page.

Field	Description
Select Content	If you are working online, you may select the content block to use as a default content block from the list.

5. When completed, click **insert**.

   A graphic to define where the dynamic content block has been placed on your template is inserted.



Shown below is the code that will be inserted according to the platform that has been specified in the setup:

Platform	Code View
.NET	For ASP.NET pages, use the CMS400.NET (Server Control) Content Block tag and set the dynamic parameter to id. See "Inserting a Content Block" on page 275

Platform	Code View
ASP	`<% if request.QueryString(“id”)<> “” then ecmContentBlock(request.QueryString(“id”)) else ecmContentBlock(1) end if %>`

ColdFusion	`<cfif isdefined(“url.id”)>`
	`<cfmodule id=”#url.id#” template=”#request.ecm.AppPath#ecmContentBlock.cfm”>`
	`<cfelse>`
	`<cfmodule id=”1” template=”#request.ecm.AppPath#ecmContentBlock.cfm”>`
	`</cfif>`

JSP	Inserting a Dynamic Content Block using Ektron’s Dreamweaver Extensions is not supported in JSP.
PHP	`<?php $id =GetURL_Id(1); ecmContentBlock($id); ?>`

Inserting a Collection

For a description of the Collection function, see ”Collection Server Control” on page 103

This tag appears on the following menu(s):

- **Commands > Ektron CMS400 - (ASP CF JSP PHP) > CMS Function...**

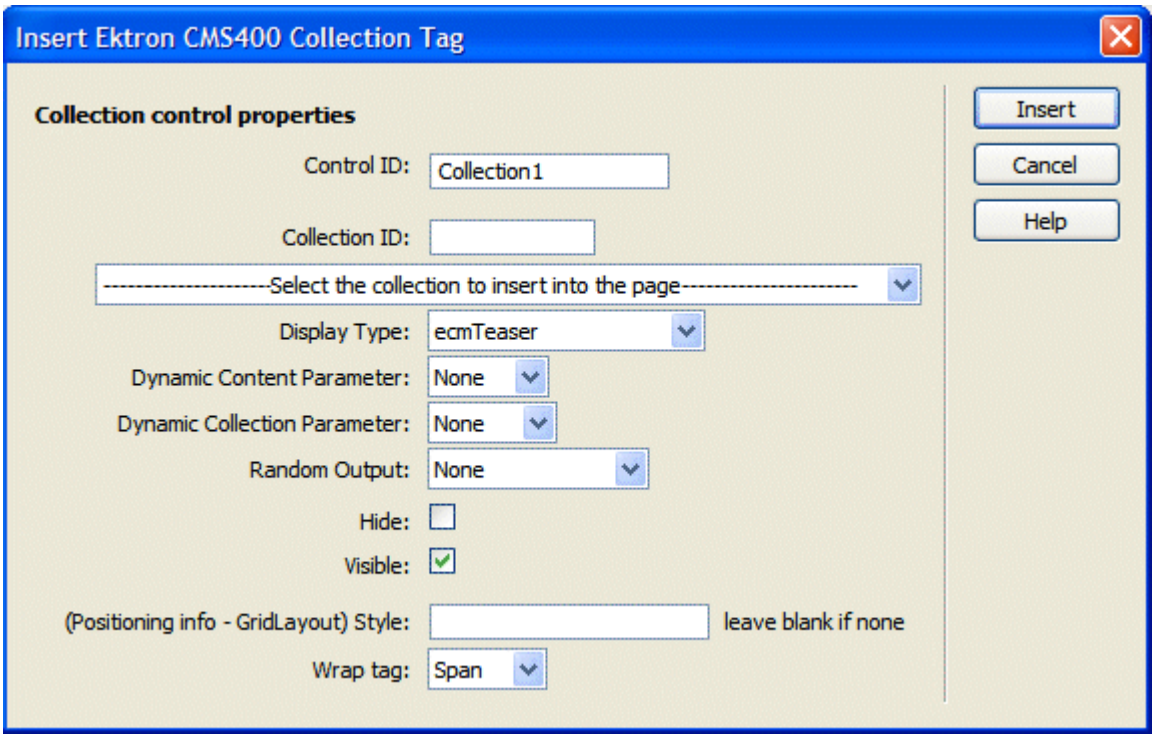
  or

- **Commands > Ektron CMS400.NET (Server Controls) > CMS Function...**

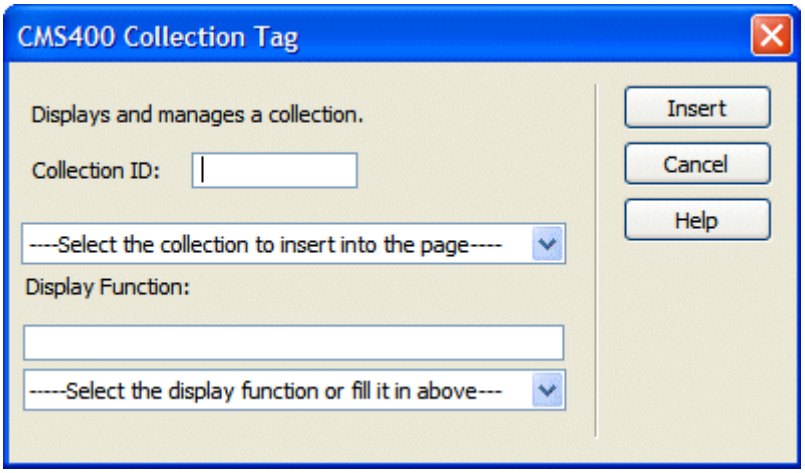
To insert a collection:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Functions, click **Collection**.

3. The Insert Ektron Collection box is displayed.



IMPORTANT! When inserting a collection using ASP, CF, JSP or PHP, a different dialog box appears.



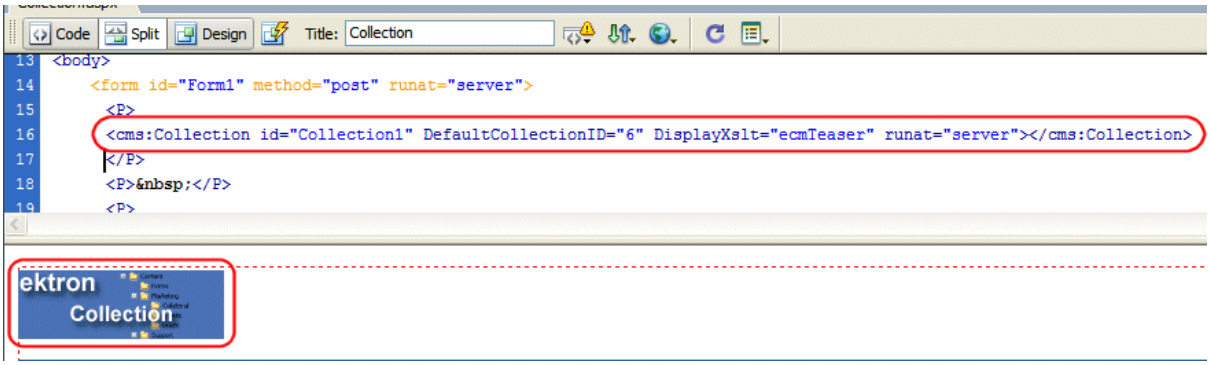
4. Use the following table to assist you with inserting a Collection.

Field	Description
Control ID Used in ASP.NET only.	A unique name for the function you are inserting.
Collection ID	If you know the collection ID, enter it into the textbox. If you do not know the Collection ID, you can navigate to it using the Select Collection drop down box.
Select Collection	The drop-down list is filled with a list of all the Collection titles that you have privileges to. When a title is selected, the Collection ID textbox is filled with that Collection's id number. If you insert the Collection ID, you do not need to select a category.
Display Type	Determines how the information on the page is displayed. **None** - databind only **ecmNavigation** - lists the title of every content block in the collection *See Also:* "Example of ecmNavigation Display" on page 105 **ecmTeaser** - lists the title of every content block in the collection plus the content summary *See Also:* "Example of ecmTeaser Display" on page 106
Dynamic Content Parameter Used in ASP.NET only.	Checks the QueryString for this value and replaces the collection with a content block when specified. Leave blank to always display the collection.
Dynamic Collection Parameter Used in ASP.NET only.	To make this collection dynamic, select **coll_id**. When you do, the function uses the collection passed as a URL parameter.
Random Output Used in ASP.NET only.	Displays random content within a collection. **None** - Displays all content in collection. **Random Teaser** - Displays a random content block's teaser. **Random Body** - Displays a random content block's body.

Field	Description
Hide Used in ASP.NET only.	Used to hide a collection in design time and run time. ☑ **Checked** = Hide collection ☐ **Unchecked** = Show collection Note: If Hide and Visible are both checked, the property is hidden.
Visible Used in ASP.NET only.	Used to show or hide the function at run time. ☑ **Checked** = Hide function ☐ **Unchecked** = Show function Note: If Hide and Visible are both checked, the property is hidden.
Style (Positioning Info - GridLayout) Used in ASP.NET only.	Set position information of a control as absolute, when using a GridLayout Web page. For example: style="Z-INDEX: 101; LEFT: 328px; POSITION: absolute; TOP: 240px"
Wrap Tag Used in ASP.NET only.	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code.

5. When completed, click **insert**.

   A graphic defines where the collection appears on your template.



Shown below is an example of the code that is inserted.

Platform	Code View
.NET	`<cms:Collection id=”Collection1” DefaultCollectionID=”0” DisplayXslt=”ecmTeaser” runat=”server”></cms:Collection>`
ASP	`<% ecmCollection 1,"ecmNavigation" %>`
ColdFusion	`<cfmodule id="1" displayFunction="ecmNavigation" template="#request.ecm.AppPath#ecmCollection.cfm">`
JSP	`<%= ecmCollection(6,"ecmTeaser") %>`
PHP	`<?php ecmCollection(1,"ecmNavigation"); ?>`

Inserting a List Summary

For a description of the List Summary function, see ”ListSummary Server Control” on page 139.

This tag appears on the following menu(s):

- **Commands > Ektron CMS400 - (ASP CF JSP PHP) > CMS Function...**

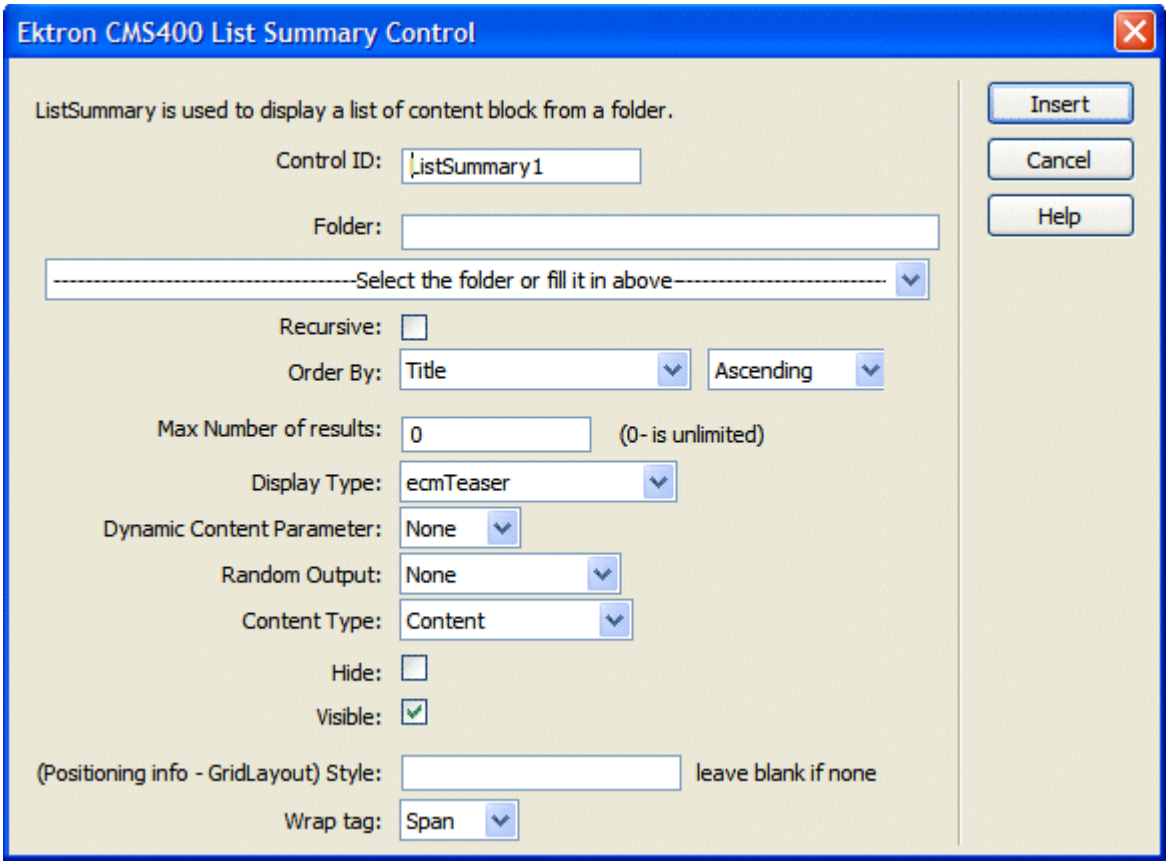
  or

- **Commands > Ektron CMS400.NET (Server Controls) > CMS Function...**

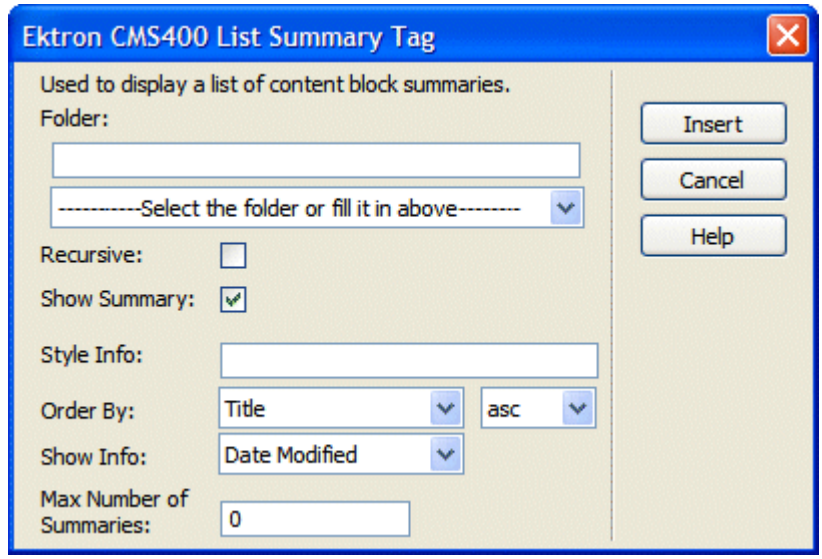
To insert a list summary:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Functions, click **List Summary**.

3. The Insert Ektron List Summary box is displayed.



IMPORTANT! When inserting a list summary using ASP, CF, JSP or PHP, a different dialog box appears.



4. Use the following table to assist you with inserting a metadata control.

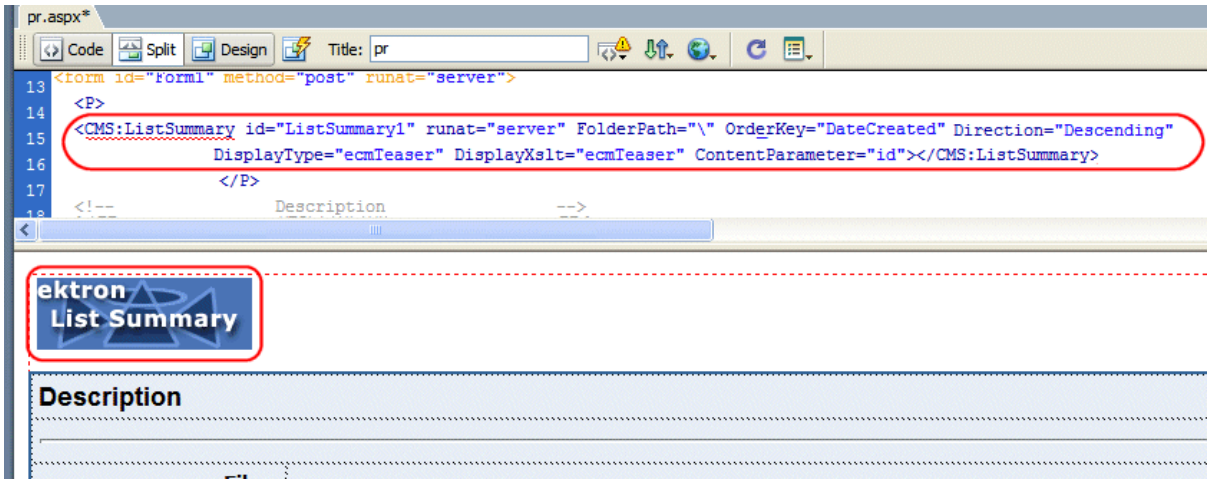
Field	Description
Control ID Used in ASP.NET only.	A unique name for the function you are inserting.
Folder	Enter the folder name manually. The default is the root or "\". If you do not know the folder path, you can navigate to it using the Select folder drop down box.
Select Folder	Select the folder the summary information comes from. The folder drop-down contains folders from your site. If you insert the folder path, you do not need to select a folder from the dropdown box.
Recursive	If checked, summary information is gathered from the folder selected and all of its child folders.

Field	Description
Order By	Specify how the summaries are ordered. Sort by the following criteria. • **Title** - The title of the content block • **Date Created** - The date the content block was created • **Date Modified** - The date the content block was last modified • **Last Author Last Name** - The last author's last name • **Last Author First Name** - The last author's first name • **Ascending** - Alphabetic or numerical order • **Descending** - Reverse alphabetic or numerical order
Max Number of results	Enter a number to specify how many results are displayed on the page (0=unlimited).
Display Type Used in ASP.NET only.	Determines how the information on the page is displayed. **None**-databind only **ecmNavigation** - lists the title of every content block in the folder *See Also:* "Example of ecmNavigation Display" on page 105 **ecmTeaser** - lists the title of every content block in the folder plus the content summary *See Also:* "Example of ecmTeaser Display" on page 106
Dynamic Content Parameter Used in ASP.NET only.	Checks the QueryString for this value and replaces the list summary with a content block when specified. Choose **None** to always display List Summary.
Random Output Used in ASP.NET only.	Displays random content within a list summary. **None** - Displays all content in list summary. **Random Teaser** - Displays a random content block's teaser. **Random Body** - Displays a random content block's body.
Content Type Used in ASP.NET only.	Defines what type of list summary appears. **Content** - Displays only content blocks in the folder. **All Types** - Displays all content in the folder. **Forms** - Displays only forms in the folder. **Archive_Content** - Displays archived content from the folder. **Archive_Forms** - Displays archived forms from the folder.

Field	Description
Hide Used in ASP.NET only.	Used to hide a list summary in design time and run time. ☑ **Checked** = Hide list summary ☐ **Unchecked** = Show list summary Note: If Hide and Visible are both checked, the property is hidden.
Visible Used in ASP.NET only.	Used to show or hide the function at run time. ☑ **Checked** = Hide function ☐ **Unchecked** = Show function Note: If Hide and Visible are both checked, the property is hidden.
Style (Positioning Info - GridLayout) Used in ASP.NET only.	Set position information of a control as absolute, when using a GridLayout Web page. For example: style="Z-INDEX: 101; LEFT: 328px; POSITION: absolute; TOP: 240px"
Wrap Tag Used in ASP.NET only.	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code.

5. When completed, click **insert**.

   A graphic defines where the list summary appears on your template.



Shown below is an example of the code that is inserted:

Platform	Code View
.NET	`<CMS:ListSummary id="ListSummary1" runat="server" FolderPath="\" OrderKey="DateCreated" Direction="Descending" DisplayType="ecmTeaser" DisplayXslt="ecmTeaser" ContentParameter="id"></CMS:ListSummary>`
ASP	`<% ecmListSummary "\Human Resources",1,1,"","DateModified,desc", "DateCreated",0,"","" %>`
ColdFusion	`<cfmodule template="#request.ecm.AppPath#ecmListSummary.cfm" Folder="\Human Resources" MaxNumber="0" Recursive="1" ShowSummary="1" StyleInfo="" OrderBy="DateModified,desc" ShowInfo="DateCreated">`
JSP	`<%= ecmListSummary("\Human Resources",0,1,"", "Title,asc","DateModified",0,"","") %>`
PHP	`<?php ecmListSummary("\Human Resources",1,1,"", "DateModified,desc","DateCreated",0,"",""); ?>`

Inserting a Metadata Function for ASP.NET pages

For a description of the Metadata function, see "MetaData Server Control" on page 178.

This tag appears on the following menu(s):

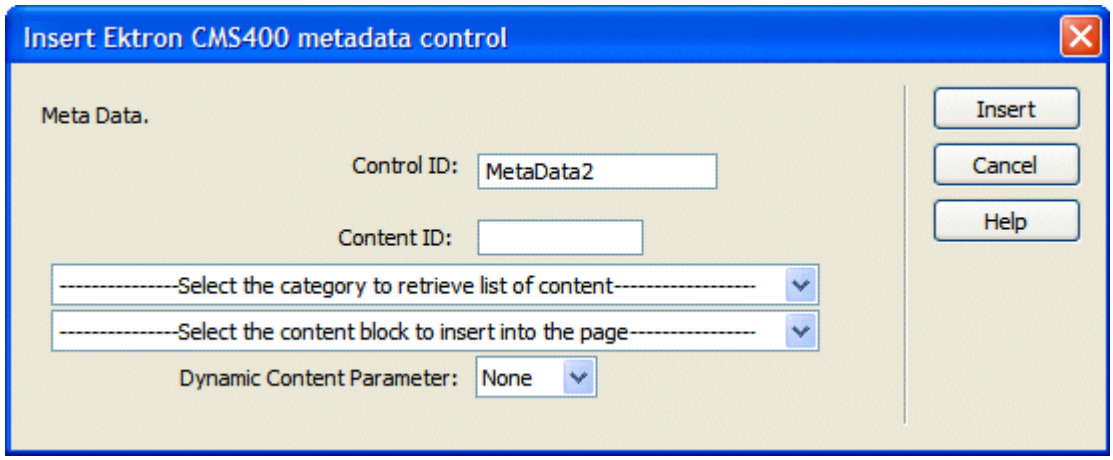
- **Commands > Ektron CMS400.NET (Server Controls) > CMS Function...**

To insert a metadata function:

1. Move the blinking cursor to the area that you want to place it.

NOTE The Ektron CMS400.NET Metadata function should be inserted within the <head> tags of your template to work properly.

2. From the list of Ektron CMS400.NET Functions, click on **Metadata**.
3. The Insert Ektron Metadata box is displayed.



4. Use the following table to assist you with inserting metadata.

Field	Description
Control ID	A unique name for the function you are inserting.
Content ID	Enter a valid content ID in the Content ID textbox. If you do not know the Content ID, you can navigate to it using the Select Category and Select Content Block drop down boxes.
Select Category	The drop-down list contains a list of content folders. If you insert the content ID, you do not need to select a category.
Select Content Block	The drop-down box is populated with a list of all the content block titles within that folder. When a title is selected, the content ID textbox is filled with that content block's id number. If you insert the content ID, you do not need to select a content block.

Field	Description
Dynamic Content Parameter	Gets or sets the QueryString parameter to read a content ID dynamically.

5. When completed, click **insert**.
6. The following line appears in the <body> of the HTML code.

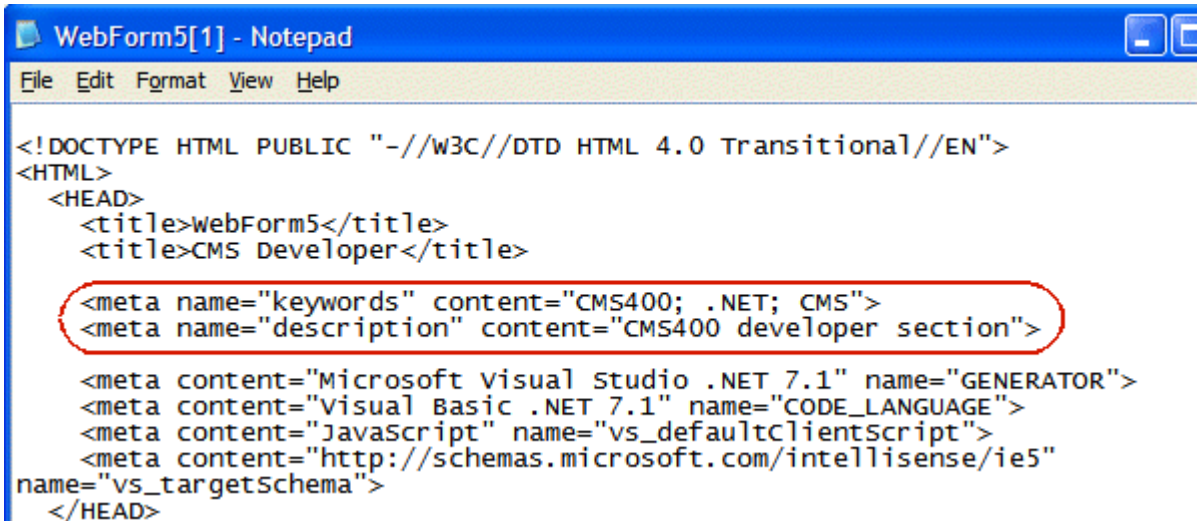
```
<cms:MetaData id="MetaData2" runat="server"
DefaultContentID="12"></cms:MetaData>
```

7. Move that line from the <body> tag into the <head> tag.

```
<HEAD>
    <title>WebForm5</title>
    <cms:MetaData id="Metadata2" runat="server" DefaultContentID="12"></cms:MetaData>
    <meta name="GENERATOR" content="Microsoft Visual Studio .NET 7.1">
    <meta name="CODE_LANGUAGE" content="Visual Basic .NET 7.1">
    <meta name="vs_defaultClientScript" content="JavaScript">
    <meta name="vs_targetSchema" content="http://schemas.microsoft.com/intellisense/i
</HEAD>
<body>
    <form id="Form1" method="post" runat="server">
    </form>
</body>
```

8. Save the Web form.
9. Open the Web page in the browser.
10. Right click on the Web page and click **View Source**. Look in the head tag. The meta tags from the content block are added to the page.

The metadata information added is shown below.



Shown below is an example of the code that is inserted:

Platform	Code View
.NET	`<cms:MetaData id="MetaData2" runat="server" DefaultContentID="12"></cms:MetaData>`
ASP	For ASP pages, see "Inserting a Metadata Function for ASP, CF, JSP or PHP pages" on page 293
ColdFusion	For ColdFusion pages, see "Inserting a Metadata Function for ASP, CF, JSP or PHP pages" on page 293
JSP	For JSP pages, see "Inserting a Metadata Function for ASP, CF, JSP or PHP pages" on page 293
PHP	For PHP pages, see "Inserting a Metadata Function for ASP, CF, JSP or PHP pages" on page 293

Inserting a Metadata Function for ASP, CF, JSP or PHP pages

For a description of the Metadata function, see "MetaData Server Control" on page 178.

This tag appears on the following menu(s):

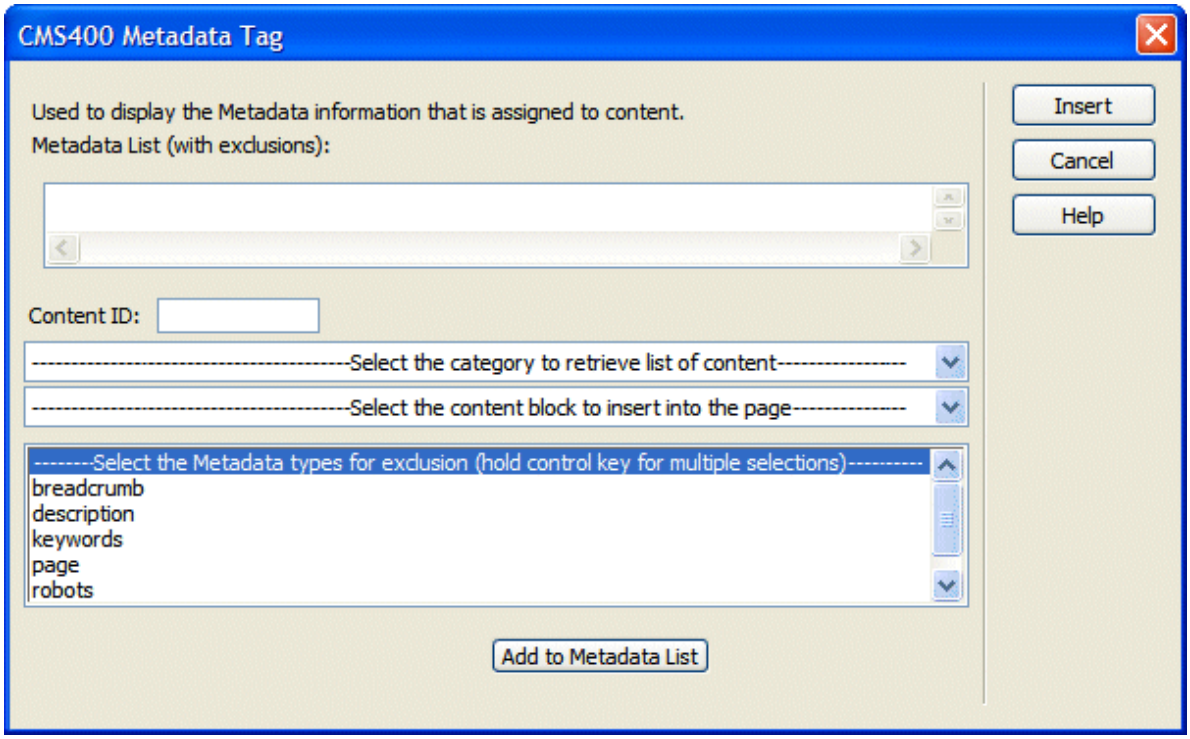
- **Commands > Ektron CMS400 - (ASP CF JSP PHP) > CMS Function...**

To insert a metadata function:

1. Move the blinking cursor to the area that you want to place it.

NOTE The Ektron CMS400.NET Metadata function should be inserted within the <head> tags of your template to work properly.

2. From the list of Ektron CMS400.NET Functions, click on **Metadata**.
3. The Insert Ektron Metadata box is displayed.



4. Use the following table to assist you with inserting metadata.

Field	Description
Metadata List	When you select a content ID and click the Add to Metadata List button, The content ID is added to this text box. This allows you to create a list of content blocks for which the metadata of each content block is added to the page.

Field	Description
Content ID	Enter a valid content ID in the Content ID textbox. If you do not know the Content ID, you can navigate to it using the Select Category and Select Content Block drop down boxes.
Select Category	The drop-down list contains a list of content folders. If you insert the content ID, you do not need to select a category.
Select Content Block	The drop-down box is populated with a list of all the content block titles within that folder. When a title is selected, the content ID textbox is filled with that content block's id number. If you insert the content ID, you do not need to select a content block.
Metadata types for exclusion	Select the metadata types that you do not want added to the metadata function you are inserting. Hold the control key down for multiple selections.

5. When completed, click **insert**.
6. The following line appears in the <body> of the HTML code.

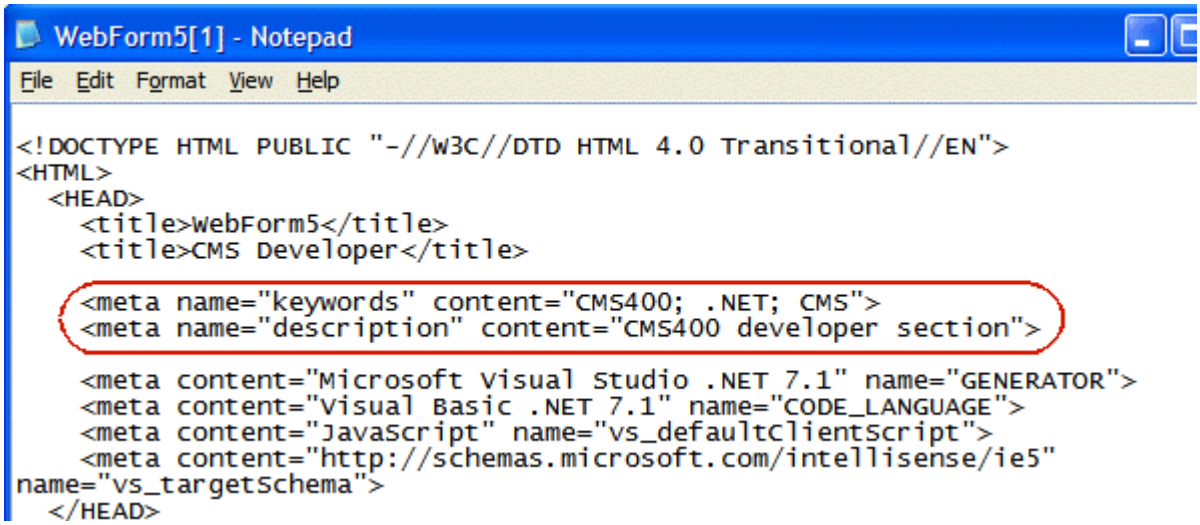
```
<cms:MetaData id="MetaData2" runat="server"
DefaultContentID="12"></cms:MetaData>
```

7. Move that line from the <body> tag into the <head> tag.

```
<HEAD>
    <title>WebForm5</title>
    <cms:MetaData id="Metadata2" runat="server" DefaultContentID="12"></cms:MetaData>
    <meta name="GENERATOR" content="Microsoft Visual Studio .NET 7.1">
    <meta name="CODE_LANGUAGE" content="Visual Basic .NET 7.1">
    <meta name="vs_defaultClientScript" content="JavaScript">
    <meta name="vs_targetSchema" content="http://schemas.microsoft.com/intellisense/i
</HEAD>
<body>
    <form id="Form1" method="post" runat="server">
    </form>
</body>
```

8. Save the Web form.
9. Open the Web page in the browser.
10. Right click on the Web page and click **View Source**. Look in the head tag. The meta tags from the content block are added to the page.

The metadata information added is shown below.



Shown below is an example of the code that is inserted:

Platform	Code View
.NET	For ASP.NET pages, see "Inserting a Metadata Function for ASP.NET pages" on page 290
ASP	`<% ecmMetadata "[1;Title][9][3;Keywords,Title]","" %>`
ColdFusion	`<cfmodule template="#request.ecm.AppPath#ecmMetadata.cfm" ItemList="[1;Title][9][3;Keywords,Title]">`
JSP	`<%= ecmMetadata("[12;keywords,Title]","") %>`
PHP	`<?php ecmMetadata("[1;Title][9][3;Keywords,Title]",""); ?>`

Inserting a Search Function

For a description of the Search functions, see "Search Server Control" on page 200

This tag appears on the following menu(s):

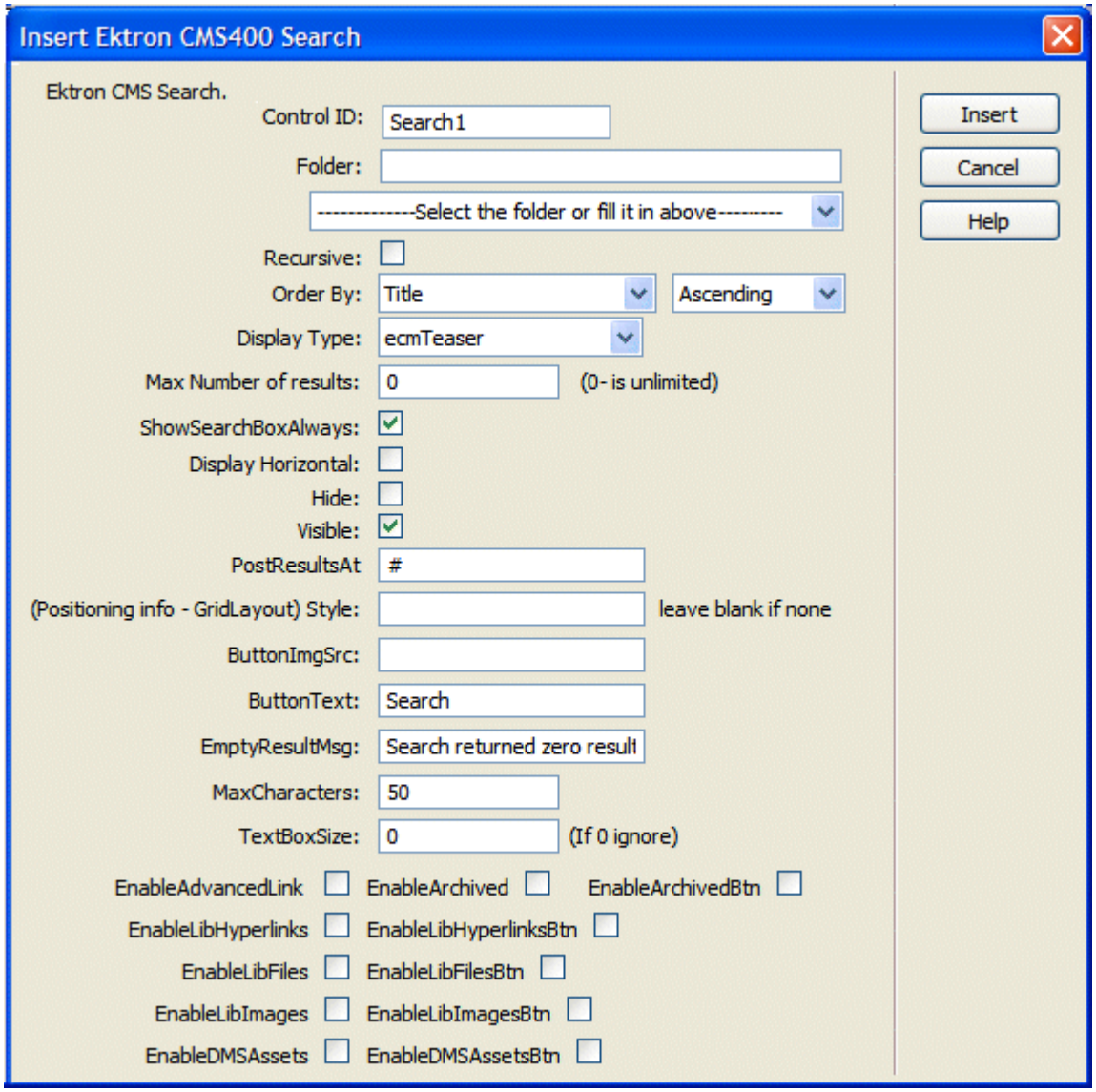
- **Commands > Ektron CMS400 - (ASP CF JSP PHP) > CMS Function...**

  or

- **Commands > Ektron CMS400.NET (Server Controls) > CMS Function...**

To insert a Search function:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Content functions, click on **Search**
3. The Insert Ektron Search box is displayed.



IMPORTANT! When inserting a list summary using ASP, CF, JSP or PHP, a slightly different dialog box appears.

4. Use the following table to assist you with inserting a search.

Field	Description
Control ID Used in ASP.NET only.	A unique name for the function you are inserting.
Folder	Type in the folder path to be searched. If you do not know the folder path, you can navigate to it using the Select folder drop down box.
Select Folder	Select a folder path to be searched. If you insert the folder path, you do not need to select a folder.
Recursive	If checked, search information is gathered from the folder selected and all of its child folders.
Order By Used in ASP.NET only.	Specify how you want to order the search results that are generated when a search is performed on you Web site. Also specify if you want the results in ascending or descending order. • **Title** - The title of the content block • **Date Created** - The date the content block was created • **Date Modified** - The date the content block was last modified • **Last Author Last Name** - The last author's last name • **Last Author First Name** - The last author's first name • **Ascending** - Alphabetic or numerical order • **Descending** - Reverse alphabetic or numerical order
Display Type Used in ASP.NET only.	Determines how the information on the page is displayed **None**-databind only **ecmNavigation** - lists the title of every content block in the folder **ecmTeaser** - lists the title of every content block in the folder plus the content summary
Max Number of results Used in ASP.NET only.	Enter a number to specify how many results are displayed on the page (0=unlimited).

Field	Description
ShowSearchBoxAlways Used in ASP.NET only.	If **unchecked**, the search box does not appear on PostBack.
Display Horizontally	The search can be displayed either vertically or horizontally on your Web page. Check off this box if you want it to appear horizontally.
Hide Used in ASP.NET only.	Used to hide a search in design time and run time. ☑ **Checked** = Hide search ☐ **Unchecked** = Show search Note: If Hide and Visible are both checked, the property is hidden.
Visible Used in ASP.NET only.	Used to show or hide the function at run time. ☑ **Checked** = Hide function ☐ **Unchecked** = Show function Note: If Hide and Visible are both checked, the property is hidden.
PostResultsAt (ASP.NET) Target Page (ASP, CF, JSP or PHP)	The path of the Web page where the search results will be posted. The # symbol appears in the text box by default. The # symbol tells the control to post back to the Web page that created the search. **Important:** A search control must exist on the posting page.
Style (Positioning Info - GridLayout) Used in ASP.NET only.	Set position information of a control as absolute, when using a GridLayout Web page. For example: style="Z-INDEX: 101; LEFT: 328px; POSITION: absolute; TOP: 240px"
ButtonImgSrc	If you wish to use an image for the search button, insert the location of the image in this field. If no image is specified, a standard submit button is used.
ButtonText	The button text is the text that is displayed on the standard submit button, or the alt text of a search button image.

Field	Description
EmptyResultMsg Used in ASP.NET only.	The return message displayed, if the search returns no hits. The default is: Search returned zero results
MaxCharacters	Specify the maximum amount of characters that is accepted in the text box.
TextBoxSize	Specify the size of the text box that is displayed on the Web page for users to enter the search keywords. The width is based on the number of characters.
Font Face Used in ASP, CF, JSP and PHP only.	The type of font to use for search function.
Font Color Used in ASP, CF, JSP and PHP only.	The color of the font used in the search function.
Font Size Used in ASP, CF, JSP and PHP only.	The size of the font used in the search function.
Wrap Tag Used in ASP.NET only.	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code.
EmptyResultMsg	The return message displayed if the search returns no hits. The default is: Search returned zero results

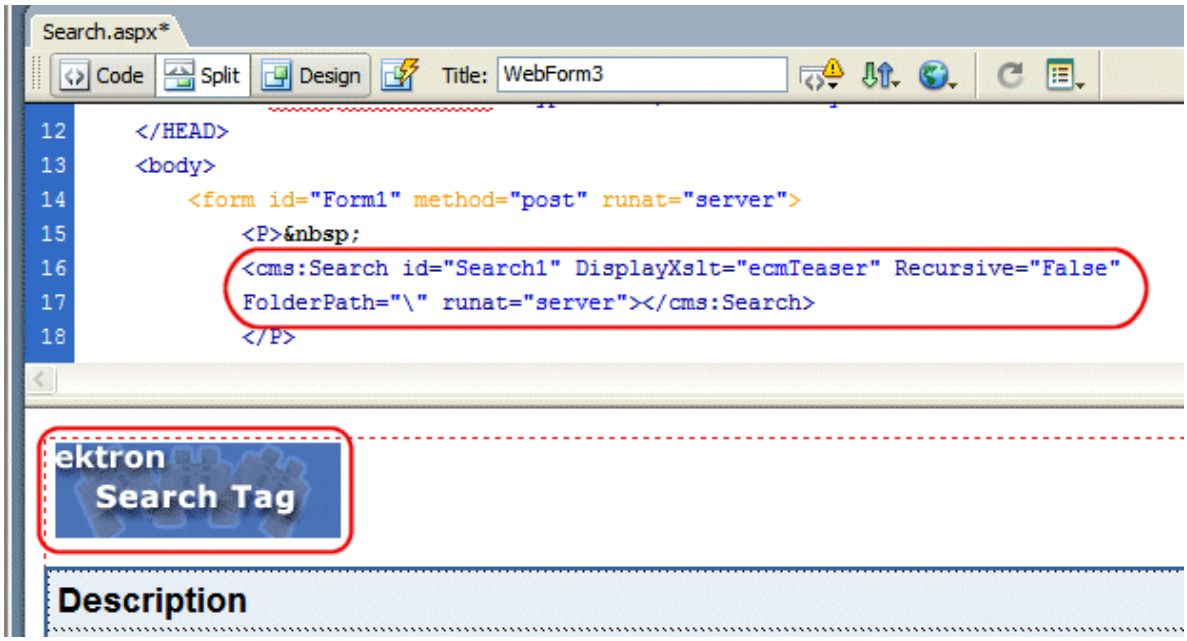
Field	Description
EnableAdvancedLink Used in ASP.NET only.	Set to true to display the link that makes the advanced properties available. ☑ **Checked** = Display Advanced link ☐ **Unchecked** = Hide Advanced link
EnableArchived Used in ASP.NET only.	Enables users to search archived content. The archived content must be archived as Archived Remain On Site.The default is False. ☑ **Checked** = Searching archived content is enabled. ☐ **Unchecked** = Searching archived content is disabled, unless Include Archived is checked in the Advanced search.
EnableArchivedBtn Used in ASP.NET only.	Provides a check box so users can decide if they want to perform an archive search. The default is False. ☑ **Checked** = The Include Archive check box is enabled in the advanced search properties. If used with EnableArchived, the check box is checked by default. ☐ **Unchecked** = The Include Archive check box is disabled in the advanced search properties.
EnableContent Used in ASP.NET only.	Enables users to search content. The default is Unchecked. ☑ **Checked** = Searching content is enabled. ☐ **Unchecked** = Searching content is disabled, unless Content is checked in the Advanced search.
EnableContentBtn Used in ASP.NET only.	Provides a check box so users can decide if they want to perform a content search. The default is Unchecked. ☑ **Checked** = The Content check box is enabled in the advanced search properties. If used with EnableContent, the check box is checked by default. ☐ **Unchecked** =The Content check box is disabled in the advanced search properties.
EnableDMSAssets Used in ASP.NET only.	Enables users to search DMS Assets. The default is Unchecked. ☑ **Checked** = Searching DMS Assets is enabled. ☐ **Unchecked** = Searching DMS Assets is disabled, unless DMS Assets is checked in the Advanced search.

Field	Description
EnableDMSAssetsBtn Used in ASP.NET only.	Provides a check box so users can decide if they want to perform a DMS Asset search. The default is Unchecked. ☑ **Checked** = The DMS Asset check box is enabled in the advanced search properties. If used with EnableContent, the check box is checked by default. ☐ **Unchecked** = The DMS Asset check box is disabled in the advanced search properties.
EnableForms Used in ASP.NET only.	Enables users to search Forms. The default is Unchecked. ☑ **Checked** = Searching Forms is enabled. ☐ **Unchecked** = Searching Forms is disabled, unless Forms is checked in the Advanced search.
EnableFormsBtn Used in ASP.NET only.	Provides a check box so users can decide if they want to perform a Forms search. The default is Unchecked. ☑ **Checked** = The Forms check box is enabled in the advanced search properties. If used with EnableContent, the check box is checked by default. ☐ **Unchecked** = The Forms check box is disabled in the advanced search properties.
EnableLibFiles Used in ASP.NET only.	Enables users to search Library Files. The default is Unchecked. ☑ **Checked** = Searching Library Files is enabled. ☐ **Unchecked** = Searching Library Files is disabled, unless Files is checked in the Advanced search.
EnableLibFilesBtn Used in ASP.NET only.	Provides a check box so users can decide if they want to perform a Library Files search. The default is Unchecked. ☑ **Checked** = The Files check box is enabled in the advanced search properties. If used with EnableLibFiles, the check box is checked by default. ☐ **Unchecked** = The Files check box is disabled in the advanced search properties.
EnableLibHyperlinks Used in ASP.NET only.	Enables users to search Library Hyperlinks. The default is Unchecked. ☑ **Checked** = Searching Library Hyperlinks is enabled. ☐ **Unchecked** = Searching Library Hyperlinks is disabled, unless Hyperlinks is checked in the Advanced search.

Field	Description
EnableLibHyperlinksBtn Used in ASP.NET only.	Provides a check box so users can decide if they want to perform a Library Hyperlinks search. The default is Unchecked. ☑ **Checked** = The Hyperlinks check box is enabled in the advanced search properties. If used with EnableLibHyperlinks, the check box is checked by default. ☐ **Unchecked** = The Hyperlinks check box is disabled in the advanced search properties.
EnableLibImages Used in ASP.NET only.	Enables users to search Library Images. The default is Unchecked. ☑ **Checked** = Searching Library Images is enabled. ☐ **Unchecked** = Searching Library Images is disabled, unless Images is checked in the Advanced search.
EnableLibImageBtn Used in ASP.NET only.	Provides a check box so users can decide if they want to perform a Library Images search. The default is Unchecked. ☑ **Checked** = The Images check box is enabled in the advanced search properties. If used with EnableLibImages, the check box is checked by default. ☐ **Unchecked** = The Images check box is disabled in the advanced search properties.

5. When completed, click **insert**.

   A graphic defines where the search appears on your template.



Shown below is an example of the code that is inserted:

Platform	Code View
.NET	`<cms:Search id="Search1" DisplayXslt="ecmTeaser" Recursive="False" FolderPath="\" runat="server"></cms:Search>`
ASP	`<% ecmSearch "\Products",1,"SearchDisplay.asp",10,25,"","Search", "Verdana","Blue","2",0,"" %>`
ColdFusion	`<CFMODULE template="#request.ecm.AppPath#ecmSearch.cfm" StartingFolder="\Products" Recursive="1" TargetPage="SearchDisplay.cfm" TextBoxSize="10" MaxCharacters="25" ButtonImageSrc="" ButtonText="Search" FontFace="Verdana" FontColor="Blue" FontSize="2" Horizontal="0">`
JSP	`<%= ecmSearch("\Products",1,"SearchDisplay.jsp",10,25,"","Search", "Verdana","Blue","2",0,"") %>`
PHP	`<?php ecmSearch("\Products",1,"SearchDisplay.php",10,25,"","Search", "Verdana","Blue","",0,""); ?>`

Inserting a Search Display Function

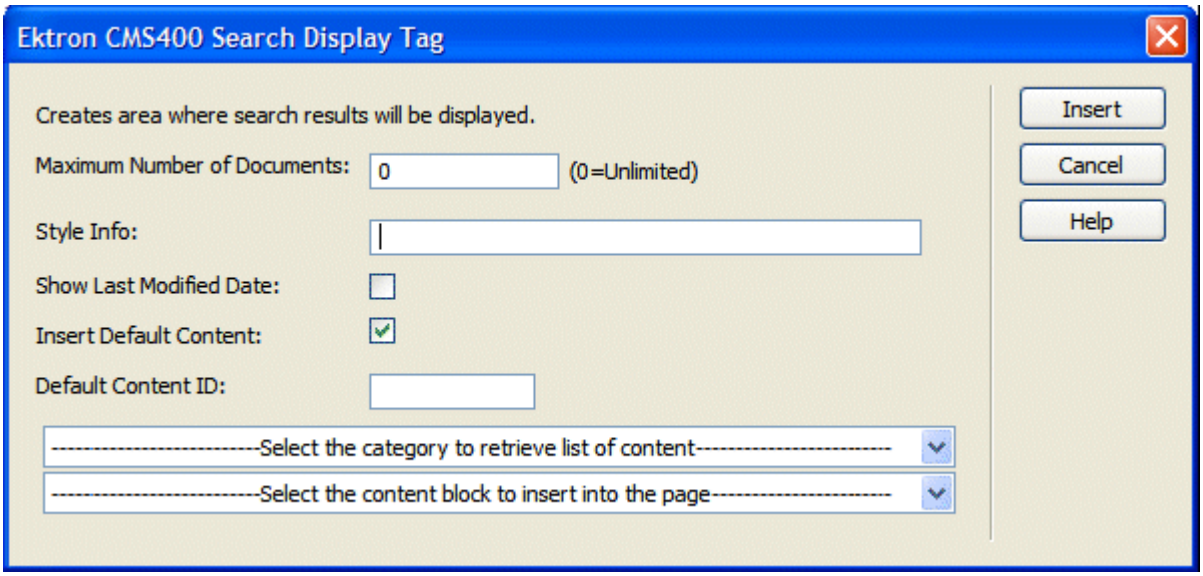
The search display custom tag creates the area where search results appear after a search is executed.

This tag appears on the following menu(s):

- **Commands > Ektron CMS400 - (ASP CF JSP PHP) > CMS Function...**

To insert a search display function:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Content functions, click on **Insert Search Display Tag...**
3. The Insert Ektron Search Display Tag box is displayed.



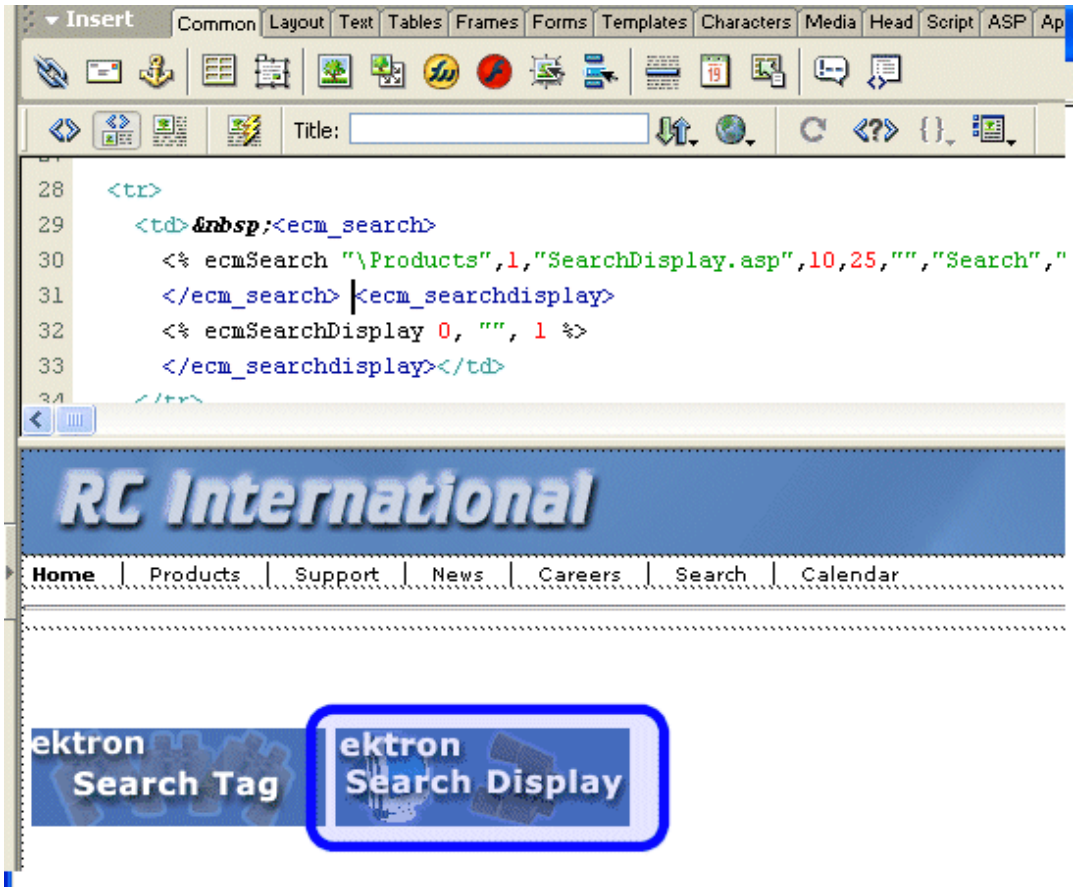
4. Use the following table to assist you with inserting a search display function.

Field	Description
Maximum Number of Documents	Specify the maximum amount of results that will be returned to the user when a search is performed.
Style Info	Insert an inline style sheet that will be used to format all the hyperlinks that are returned.

Field	Description
Show Last Modified Date	Check off this box to show the last modified date with the search result. This allows users to distinguish between updated and old content.
Insert Default Content	Specify the ID number of the content block that you wish to use as a default content block on the Web page when no search is executed.
Default Content ID	If you are working online, you may select the content block to use as a default content block from the list.

5. When completed, click **insert**.

   A graphic to define where the search display tag has been placed on your template is inserted.



Shown below is the code that will be inserted according to the platform that has been specified in the setup:

Platform	Code View (with no default content block set)
.NET	In ASP.NET, you do not need to add a separate search display.
ASP	`<% ecmSearchDisplay 0, "", 1 %>`
ColdFusion	`<cfmodule template="#request.ecm.AppPath#ecmSearchDisplay.cfm" MaxNumber="0" StyleInfo="" ShowDate="1">`
JSP	`<%= ecmSearchDisplay(0, "", 1)%>`
PHP	`<?php ecmSearchDisplay(0, "", 1)?>`

Inserting a Single Summary Function

For a description of the single summary function, see "Single Summary Function" on page 420

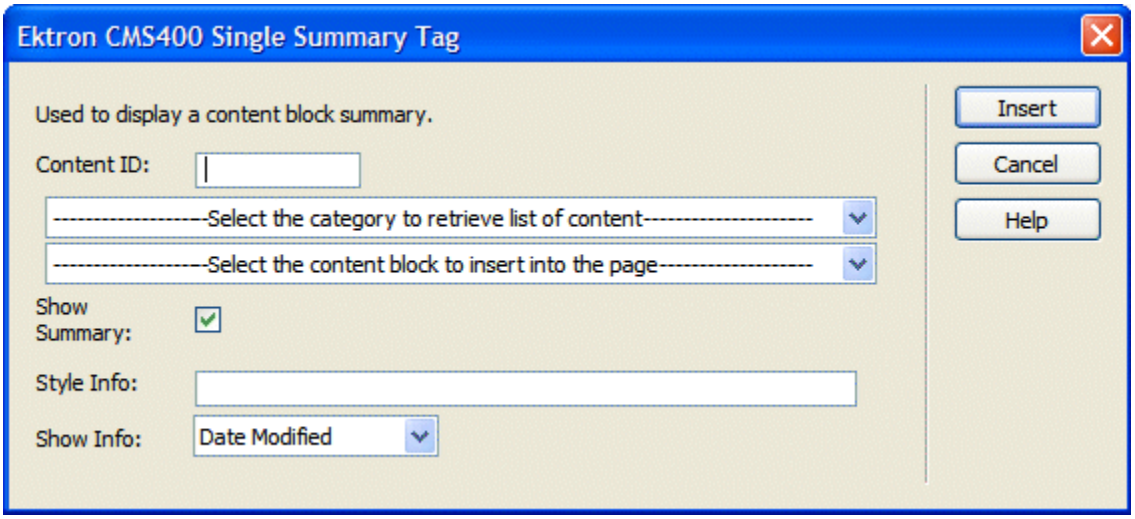
This tag appears on the following menu(s):

- **Commands > Ektron CMS400 - (ASP CF JSP PHP) > CMS Function...**

To insert a single summary:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Content functions, click on **Insert Single Summary Tag...**

3. The Insert Ektron Single Summary Tag box is displayed.

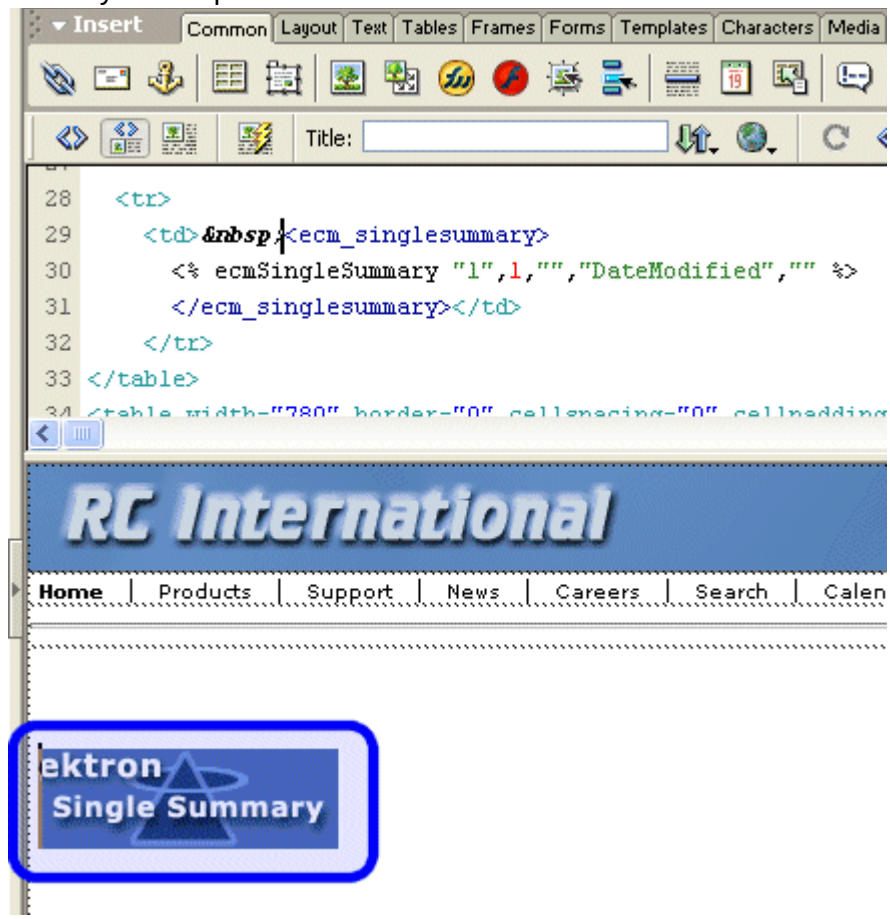


4. Use the following table to assist you with inserting a single summary tag.

Field	Description
Content ID	• Working Offline - Enter a valid content ID into the Content ID textbox. • Working Online - The top drop-down list will be filled with a list of content folders that you have privileges to. After choosing a content folder, the bottom drop-down box will be filled with a list of all the content block titles within that category. When a title is selected, the content ID textbox will be filled with that content block's id number.
Show Summary	If checked, content summaries will be displayed.
Style Info	Insert any inline style information, which will be applied to the generated text hyperlink, here.
Show Info	Select the type of information that will be displayed next to the generated hyperlinks.

5. When completed, click **insert**.

   A graphic to define where the single summary tag will appear on your template is inserted.



Shown below is the code that will be inserted according to the platform that has been specified in the setup:

Platform	Code View
ASP	`<% ecmSingleSummary "1",1,"","DateModified","" %>`
ColdFusion	`<cfmodule template="#request.ecm.AppPath#ecmSingleSummary.cfm" ContentID="1" ShowSummary="1" StyleInfo="" ShowInfo="DateModified">`
JSP	`<%= ecmSingleSummary("1",1,"","DateModified","") %>`
PHP	`<?php ecmSingleSummary("1",1,"","DateModified","") ?>`

Inserting a Random Content Function

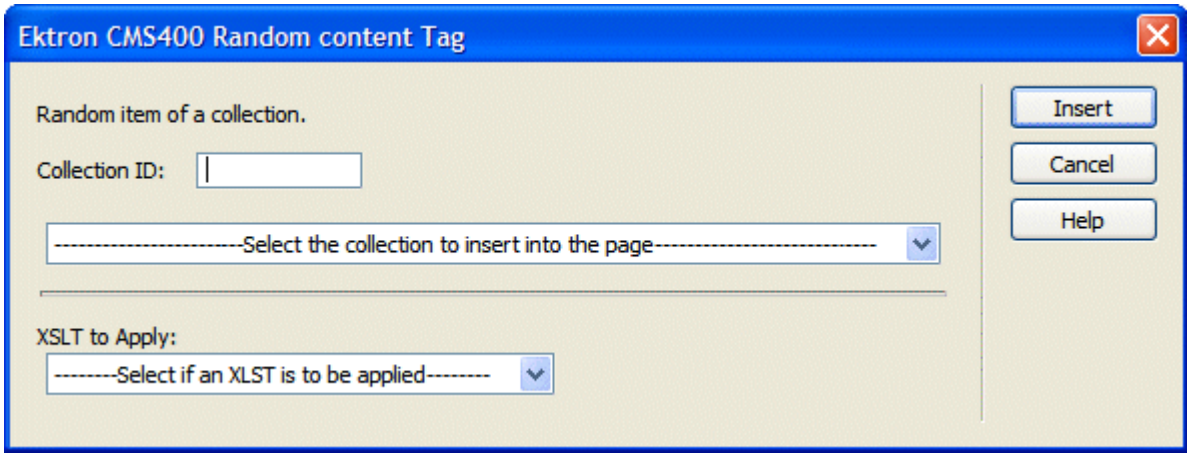
For a description of the random content function, see "Random Content" on page 443

This tag appears on the following menu(s):

- **Commands > Ektron CMS400 - (ASP CF JSP PHP) > CMS Function...**

To insert a random content function:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Content functions, click on **Insert Random Content Tag...**
3. The Insert Random Content Tag box is displayed.



4. Use the following table to assist you with inserting a random content tag.

Field	Description
Collection ID	Type in the collection ID, or select it from the drop down list, of the collection where the random content function will retrieve content from.
XSLT to apply *XML Content only*	If desired, you may specify an XSLT to apply to the content that gets displayed.

5. When completed, click **insert**.

   A graphic to define where the random content tag will appear on your template is inserted.

Shown below is the code that will be inserted according to the platform that has been specified in the setup:

Platform	Code View
ASP	`<% ShowRandomContent 3, 0 %>`
ColdFusion	`<cfmodule id="3" displayFunction="0" template="#request.ecm.AppPath#ecmShowRandomContent.cfm">`
JSP	`<%= ShowRandomContent(3, 0) %>`
PHP	`<?php ShowRandomContent(3, 0); ?>`

Inserting a Random Summary Function

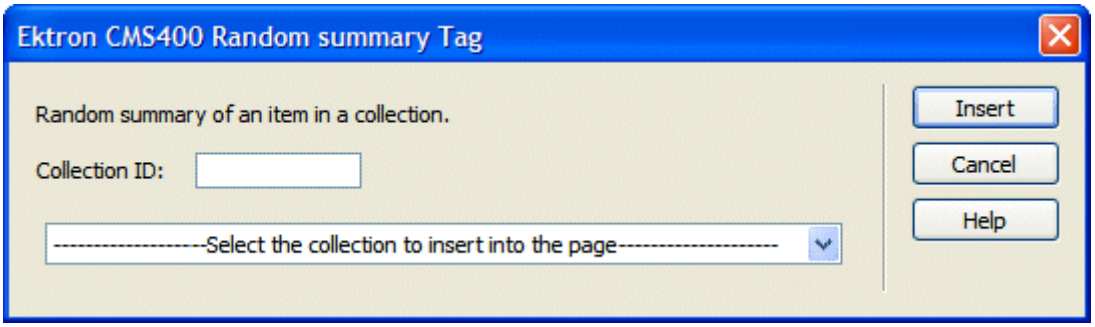
For a description of the random summary function, see "Random Summary" on page 444

This tag appears on the following menu(s):

- **Commands > Ektron CMS400 - (ASP CF JSP PHP) > CMS Function...**

To insert a random summary function:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Content functions, click on **Insert Random Summary Tag...**
3. The Insert Ektron CMS400.NET Random Summary Tag box is displayed.



4. Use the following table to assist you with inserting a random content tag.

Field	Description
Collection ID	Type in the collection ID, or select it from the drop down list, of the collection where the random summary function will retrieve summaries from.

5. When completed, click **insert**.

   A graphic to define where the random summary tag will appear on your template is inserted.

Shown below is the code that will be inserted according to the platform that has been specified in the setup:

Platform	Code View
ASP	`<% ShowRandomSummary (2) %>`
ColdFusion	`<cfmodule id="2" template="#request.ecm.AppPath#ecmShowRandomSummary.cfm">`
JSP	`<%= ShowRandomSummary(2) %>`
PHP	`<?php ShowRandomSummary (2); ?>`

Inserting an Event Calendar Function

For a description of the Event Calendar function, see "Calendar Server Control" on page 68.

This tag appears on the following menu(s):

- **Commands > Ektron CMS400 - (ASP CF JSP PHP) > CMS Function...**

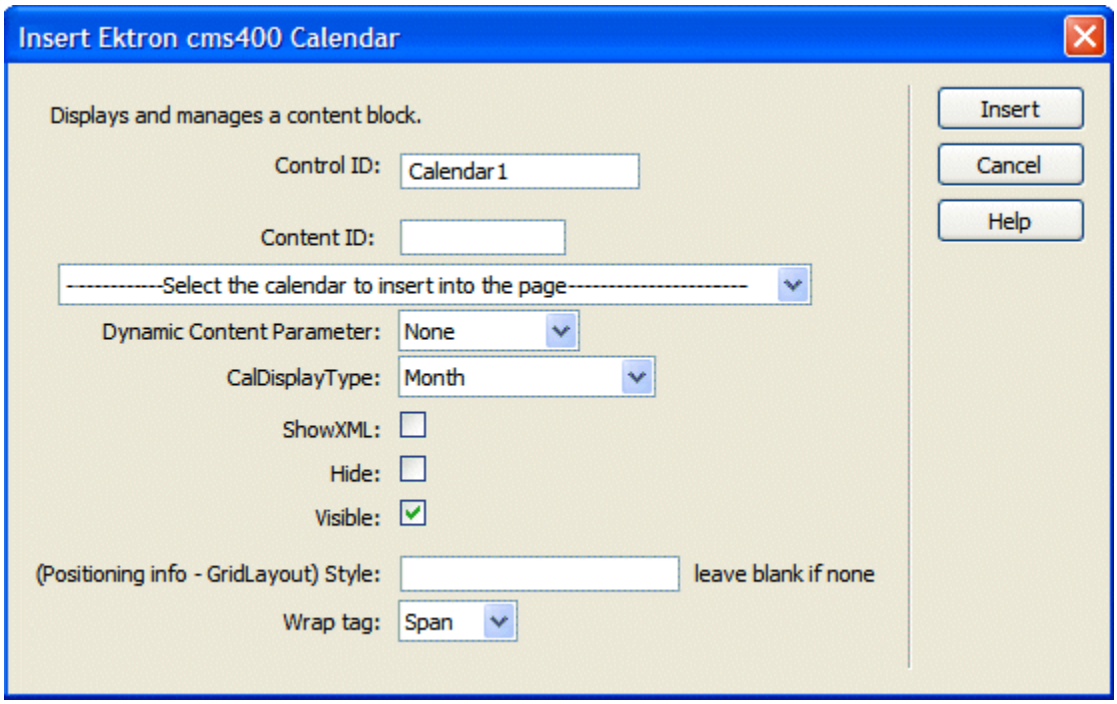
  or

- **Commands > Ektron CMS400.NET (Server Controls) > CMS Function...**

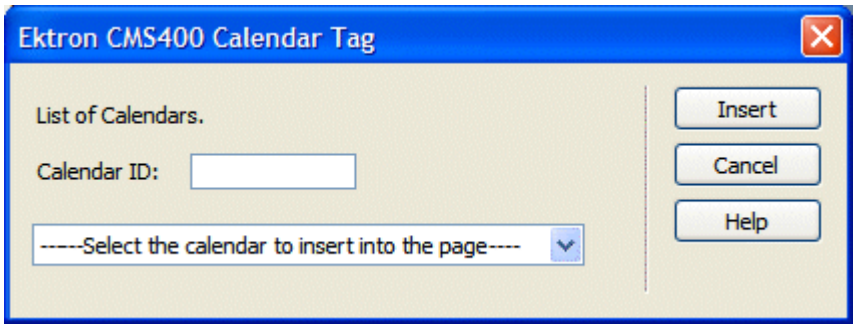
To insert an event calendar function:

NOTE Inserting a calendar function does not add a reference to a calendar style sheet. This step must be performed manually.

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Functions, click on **Calendar**
3. The Insert Calendar box is displayed.



IMPORTANT! When inserting an event calendar using ASP, CF, JSP or PHP, the dialog box that appears only allows you to set the Calendar ID or choose one from the list.



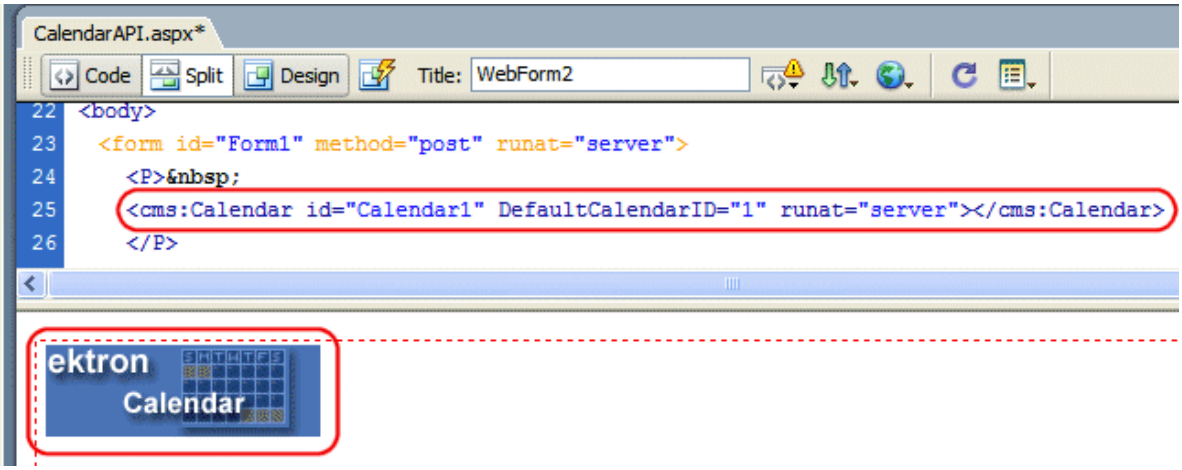
4. Use the following table to assist you with inserting a calendar.

Field	Description
Control ID Used in ASP.NET only.	A unique name for the function you are inserting.
Content ID	Type in the Content ID of the calendar that you want to insert. If you do not know the Content ID, you can navigate to it using the Select Calendar drop down box.
Select Calendar	The drop-down box is populated with a list of all the calendar titles. When a title is selected, the content ID textbox is filled with that content block's id number. If you insert the calendar ID, you do not need to select a calendar.
Dynamic Content Parameter Used in ASP.NET only.	To make this calendar dynamic, select **calendar_id**. When you do, this server control uses the calendar passed as a URL parameter.
CalDisplayType Used in ASP.NET only.	Specify the Calendar view type. The default is **Month**. • **Month** - A 30 day calendar month view. • **Day** - Single day view. • **Inline** - Display events that occur from the start date of the calendar through the end date. • **MonthUpWithEvent** - Displays a month calendar. Events appear to the right of the calendar when you click on a day.

Field	Description
ShowXML Used in ASP.NET only.	At runtime, output the raw XML data instead of the calendar. With this XML data, you can apply your own XSL to create your own custom display formats. Default is **Unchecked.** ☑ **Checked** = Show raw XML data ☐ **Unchecked** = Show Calendar
Hide Used in ASP.NET only.	Used to hide a calendar in design time and run time. ☑ **Checked** = Hide calendar ☐ **Unchecked** = Show calendar Note: If Hide and Visible are both checked, the property is hidden.
Visible Used in ASP.NET only.	Used to show or hide the function at run time. ☑ **Checked** = Hide function ☐ **Unchecked** = Show function Note: If Hide and Visible are both checked, the property is hidden.
Style (Positioning Info - GridLayout) Used in ASP.NET only.	Set position information of a control as absolute, when using a GridLayout Web page. For example: style="Z-INDEX: 101; LEFT: 328px; POSITION: absolute; TOP: 240px"
Wrap Tag Used in ASP.NET only.	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code.

5. When completed, click **insert**.

   A graphic defines where the calendar appears on your template.



Shown below is an example of the code that is inserted.

Platform	Code View
.NET	`<cms:Calendar id="Calendar1" DefaultCalendarID="1" runat="server"></cms:Calendar>`
ASP	`<% ecmEvtCalendar (1) %>`
ColdFusion	`<cfmodule Calendar_id="1" template="#request.ecm.AppPath#ecmEvtCalendar.cfm">`
JSP	`<%= ecmEvtCalendar(1) %>`
PHP	`<?php ecmEvtCalendar (1); ?>`

Inserting a Form

For a description of the Form function, see "FormBlock Server Control" on page 129

This tag appears on the following menu(s):

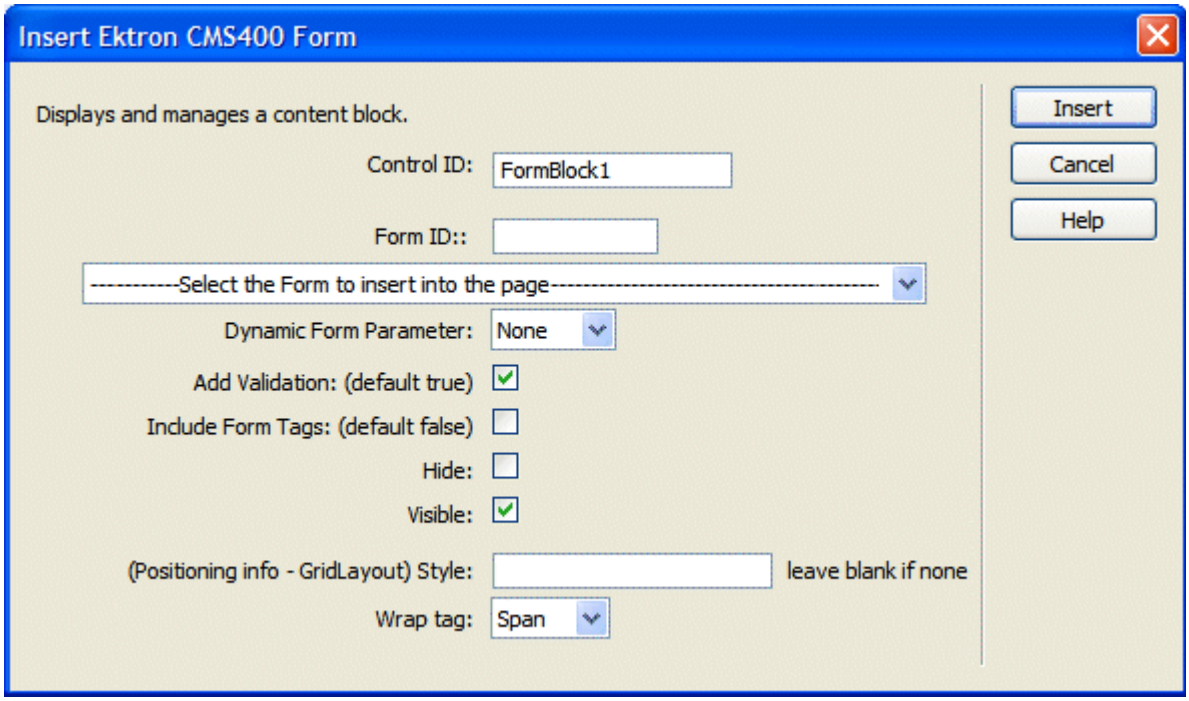
- **Commands > Ektron CMS400 - (ASP CF JSP PHP) > CMS Function...**

  or

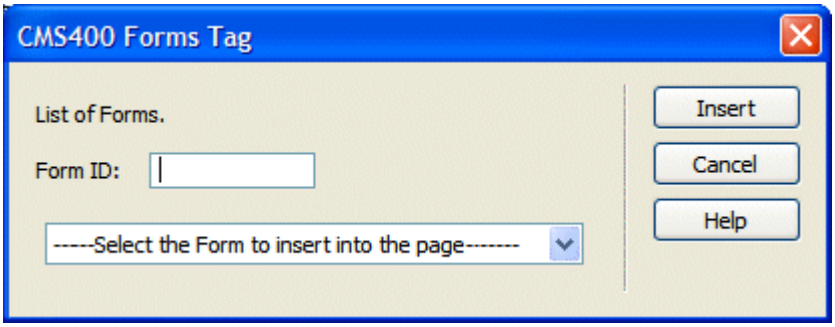
- **Commands > Ektron CMS400.NET (Server Controls) > CMS Function...**

To insert a form:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Content functions, click **Form**
3. The Insert Form box is displayed.



IMPORTANT! When inserting a form tag using ASP, CF, JSP or PHP, a different dialog box appears.



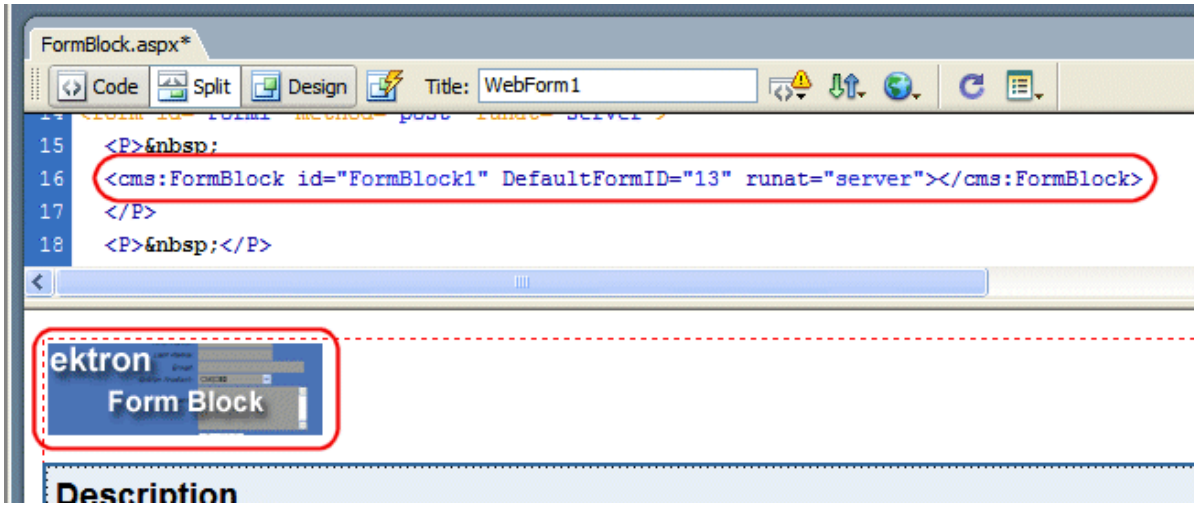
4. Use the following table to assist you with inserting a form.

Field	Description
Control ID Used in ASP.NET only.	A unique name for the function you are inserting.
Form ID	Type in the form ID that you want to insert. If you do not know the Form ID, you can navigate to it using the Select Form drop down box.
Select Form	The drop-down box is populated with a list of all the form titles. When a title is selected, the form ID textbox is filled with that form's id number. If you insert the Form ID, you do not need to select a form.
Dynamic Form Parameter Used in ASP.NET only.	To make this form block dynamic, select **ekfrm**. When you do, this control uses the form block passed as a URL parameter.
Add Validation Used in ASP.NET only.	When **checked**, adds Validation to your main runat=server form.
Include Form Tags Used in ASP.NET only.	Determines if tags are generated automatically or manually. *See Also:* "Automatic versus Manual Generation of Form Tags" on page 132
Hide Used in ASP.NET only.	Used to hide a form in design time and run time. ☑ **Checked** = Hide form ☐ **Unchecked** = Show form Note: If Hide and Visible are both checked, the property is hidden.
Visible Used in ASP.NET only.	Used to show or hide the function at run time. ☑ **Checked** = Hide function ☐ **Unchecked** = Show function Note: If Hide and Visible are both checked, the property is hidden.

Field	Description
Style (Positioning Info - GridLayout) Used in ASP.NET only.	Set position information of a control as absolute, when using a GridLayout Web page. For example: style="Z-INDEX: 101; LEFT: 328px; POSITION: absolute; TOP: 240px"
Wrap Tag Used in ASP.NET only.	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code.

5. When completed, click **insert**.

   A graphic defines where the form appears on your template.



Shown below is an example of the code that is inserted.

Platform	Code View
.NET	`<cms:FormBlock id="FormBlock1" DefaultFormID="13" runat="server"></cms:FormBlock>`
ASP	`<% ecmFormBlock (1) %>`
ColdFusion	`<cfmodule id="1" template="#request.ecm.AppPath#ecmFormBlock.cfm">`

Platform	Code View
JSP	`<%= ecmFormBlock(1) %>`
PHP	`<?php ecmFormBlock (1); ?>`

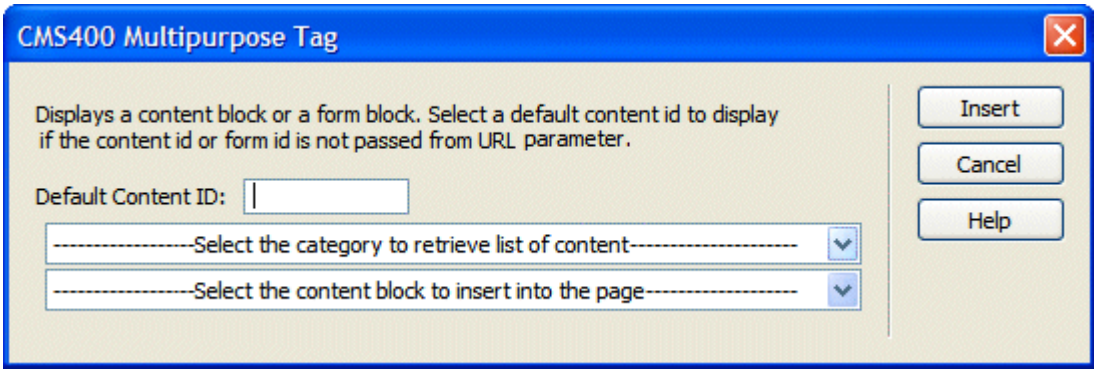
Inserting a Multipurpose content block

This tag appears on the following menu(s):

- **Commands > Ektron CMS400 - (ASP CF JSP PHP) > CMS Function...**

To insert a Multipurpose content block function:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Content functions, click on **Multipurpose Tag...**
3. The Insert Ektron CMS400.NET Multipurpose Tag box is displayed.

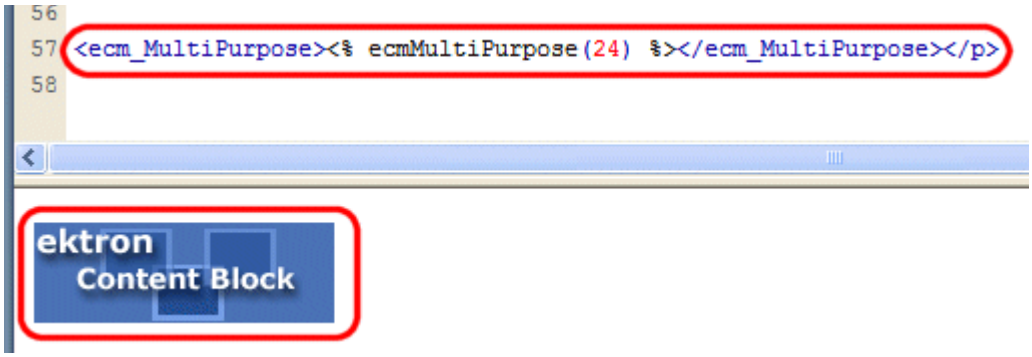


4. Use the following table to assist you with inserting a Multipurpose tag.

Field	Description
Default Content ID	Type in the default content ID, or select a category from the drop down list, then select a content block from the second dropdown list.

5. When completed, click **insert**.

   A graphic to define where the Multipurpose tag will appear on your template is inserted.



Shown below is the code that will be inserted according to the platform that has been specified in the setup:

Platform	Code View
.NET	In ASP.NET, the Content Block Tag functions as a Multipurpose Content block.
ASP	`<% ecmMultiPurpose(24) %>`
ColdFusion	`<cfmodule DefaultID="24" template="#request.ecm.AppPath#ecmMultipurpose.cfm">`
JSP	This function is not available as an Ektron Dreamweaver Extension for JSP.
PHP	`<?php ecmMultiPurpose(12); ?>`

Inserting an Advanced Calendar Tag

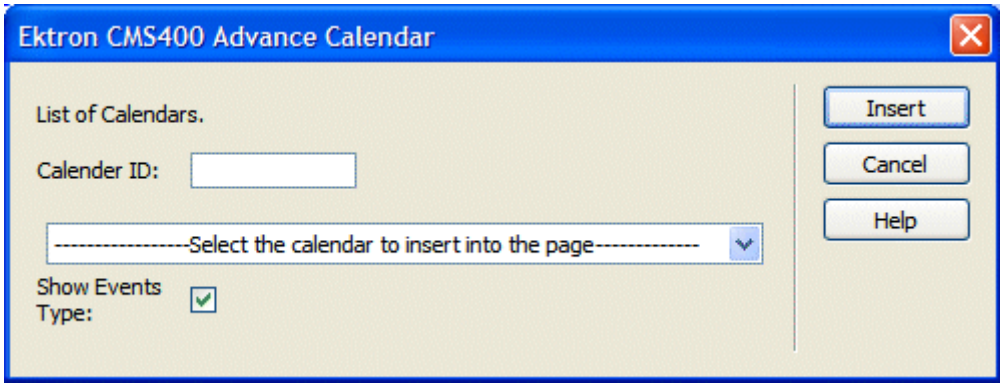
This tag appears on the following menu(s):

- **Commands > Ektron CMS400 - (ASP CF JSP PHP) > CMS Function...**

To insert an Advanced Calendar function:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Content functions, click on **Advanced Calendar...**

3. The Insert Ektron CMS400.NET Advanced Calendar Tag box is displayed.



4. Use the following table to assist you with inserting a Advanced Calendar tag.

Field	Description
Calendar ID	Type in the calendar ID, or select a calendar from the drop down list
Show Event Types	Shows a list of event types that can be used to sort events on the calendar. The default is checked. ☑ **Checked** - the list of event types appears with the calendar. ☐ **Unchecked** - the list of event types does not appear with the calendar.

5. When completed, click **insert**.

Shown below is the code that will be inserted according to the platform that has been specified in the setup:

Platform	Code View
ASP	`<% ecmAdvancedEvtCalendar 1, 1 %>`
ColdFusion	`<cfmodule ShowEvents ="1" Calendar_Id="3" template="#request.ecm.AppPath#ecmAdvanceEvtCalendar.cfm">`
JSP	`<%= ecmAdvancedEvtCalendar (1, 1) %>`
PHP	`<?php ecmAdvancedEvtCalendar(1, 1); ?>`

Inserting an XML Index Search

For a description of the XML Index Search Function, see ”IndexSearch Server Control” on page 234.

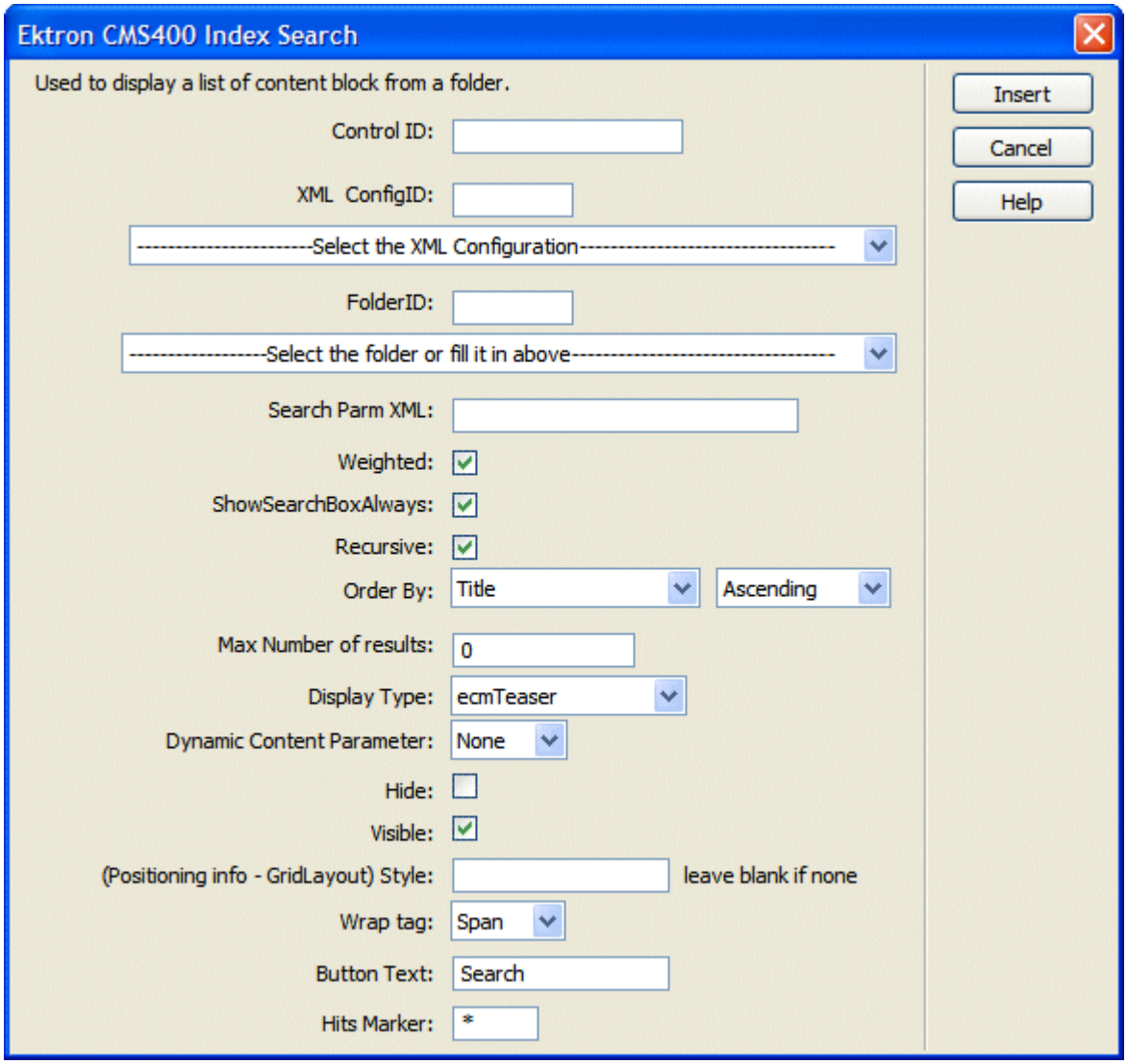
This tag appears on the following menu(s):

- **Commands > Ektron CMS400.NET (Server Controls) > CMS Function...**

To insert an XML Index Search:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET functions, click **XML Index Search**.

3. The Insert XML Index Search box is displayed.

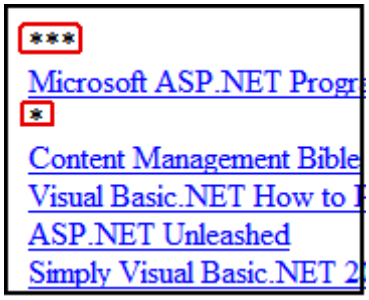


4. Use the following table to assist you with inserting an XML Index Search.

Field	Description
Control ID	A unique name for the function you are inserting.
XML ConfigID	ID of the XML configuration whose indexed fields appear on the search screen. If you do not know the XML ConfigID, you can navigate to it using the Select XML Configuration drop down box.

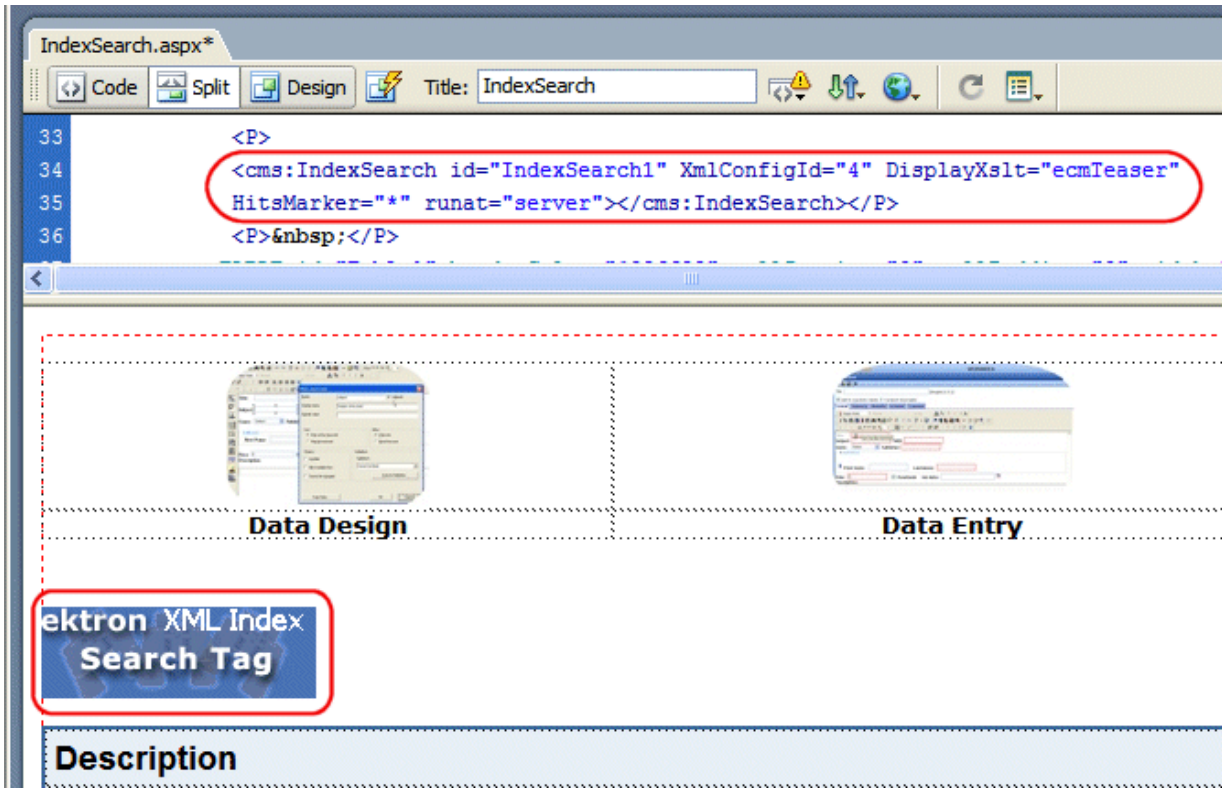
Field	Description
Select XML Configuration	The drop-down box is populated with a list of all the XML Configuration titles. When a title is selected, the XML ConfigID textbox is filled with that XML Configuration's id number. If you insert the XML ConfigID, you do not need to select a XML Configuration.
FolderID	The numeric id of the folder that you want to search. If you do not know the folder path, you can navigate to it using the Select folder drop down box.
Select Folder	Select a folder path to be searched. When a folder is selected, the folderID textbox is filled with that folder's id number. If you insert the Folder ID, you do not need to select a folder.
Search Parm XML	This property preserves settings that a developer would create in Visual Studio.
Weighted	If this value is **checked**, content blocks are ranked by the number of criteria matches. For example, if a content block matches all search criteria, it appears at the top of the results page. A hit marker appears for each match. If a second content block only matches one criterion, it also appears on the results page underneath the first content block. One asterisk appears above the page's title to indicate the single match. ☑ **Checked**- display every content block that matches *at least one* search criteria ☐ **Unchecked** - show only content blocks that match *all* search criteria
ShowSearchBoxAlways	If set to **unchecked**, the search box does not appear on when the results are posted to the Web page.
Recursive	Determines whether the XML Index Search Control uses this folder's child folders in the search.

Field	Description
Order By	Specify how you want to order the search results that are generated when a search is performed on you Web site. Also specify if you want the results in ascending or descending order. • **Title** - The title of the content block • **Date Created** - The date the content block was created • **Date Modified** - The date the content block was last modified • **Last Author Last Name** - The last author's last name • **Last Author First Name** - The last author's first name • **Ascending** - Alphabetic or numerical order • **Descending** - Reverse alphabetic or numerical order When you choose weighted search results, Ascending and Descending use how many criteria match your search. For example, if you have weighted results, and you choose Ascending, the results that match the most criteria appear first.
Max Number of results	The Maximum number of content blocks returned (0=unlimited).
Display Type	Determines how the information on the page is displayed **None**-databind only **ecmNavigation** - lists the title of every content block in the folder *See Also:* "Example of ecmNavigation Display" on page 105 **ecmTeaser** - lists the title of every content block in the folder plus the content summary *See Also:* "Example of ecmTeaser Display" on page 106
Dynamic Content Parameter	Checks the QueryString for the content block ID value and replaces the search with content when ID is specified. Leave blank to always display the search. By making content dynamic, when the content displayed on a Web page contains a link to another piece of content, you can display the second piece of content in the same Web page space. **None** - Use Default **ID** - The control uses the content block passed as a URL parameter.
Hide	Used to hide a form in design time and run time. ☑ **Checked** = Hide form ☐ **Unchecked** = Show form Note: If Hide and Visible are both checked, the property is hidden.

Field	Description
Visible	Used to show or hide the function at run time. ☑ **Checked** = Hide function ☐ **Unchecked** = Show function Note: If Hide and Visible are both checked, the property is hidden.
Style (Positioning Info - GridLayout)	Set position information of a control as absolute, when using a GridLayout Web page. For example: style="Z-INDEX: 101; LEFT: 328px; POSITION: absolute; TOP: 240px"
Wrap Tag	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code.
Button Text	The text that appears on the button.
Hits Marker	This symbol shows how many search criteria each hit matched. The following example shows the first hit met three of the criteria. The second group met one criterion. 

5. When completed, click **insert**.

   A graphic defines where the XML Index Search appears on your template.



Shown below is an example of the code that is inserted.

Platform	Code View
.NET	`<cms:IndexSearch id="IndexSearch1" XmlConfigId="4" DisplayXslt="ecmTeaser" HitsMarker="*" runat="server"></cms:IndexSearch>`
ASP	This function is not available as an Ektron Dreamweaver Extension for ASP.
ColdFusion	This function is not available as an Ektron Dreamweaver Extension for ColdFusion.
JSP	This function is not available as an Ektron Dreamweaver Extension for JSP.
PHP	This function is not available as an Ektron Dreamweaver Extension for PHP.

Inserting Display By Meta Value

For a description of the Display By Meta Value function, see ”MetaDataList Server Control” on page 183.

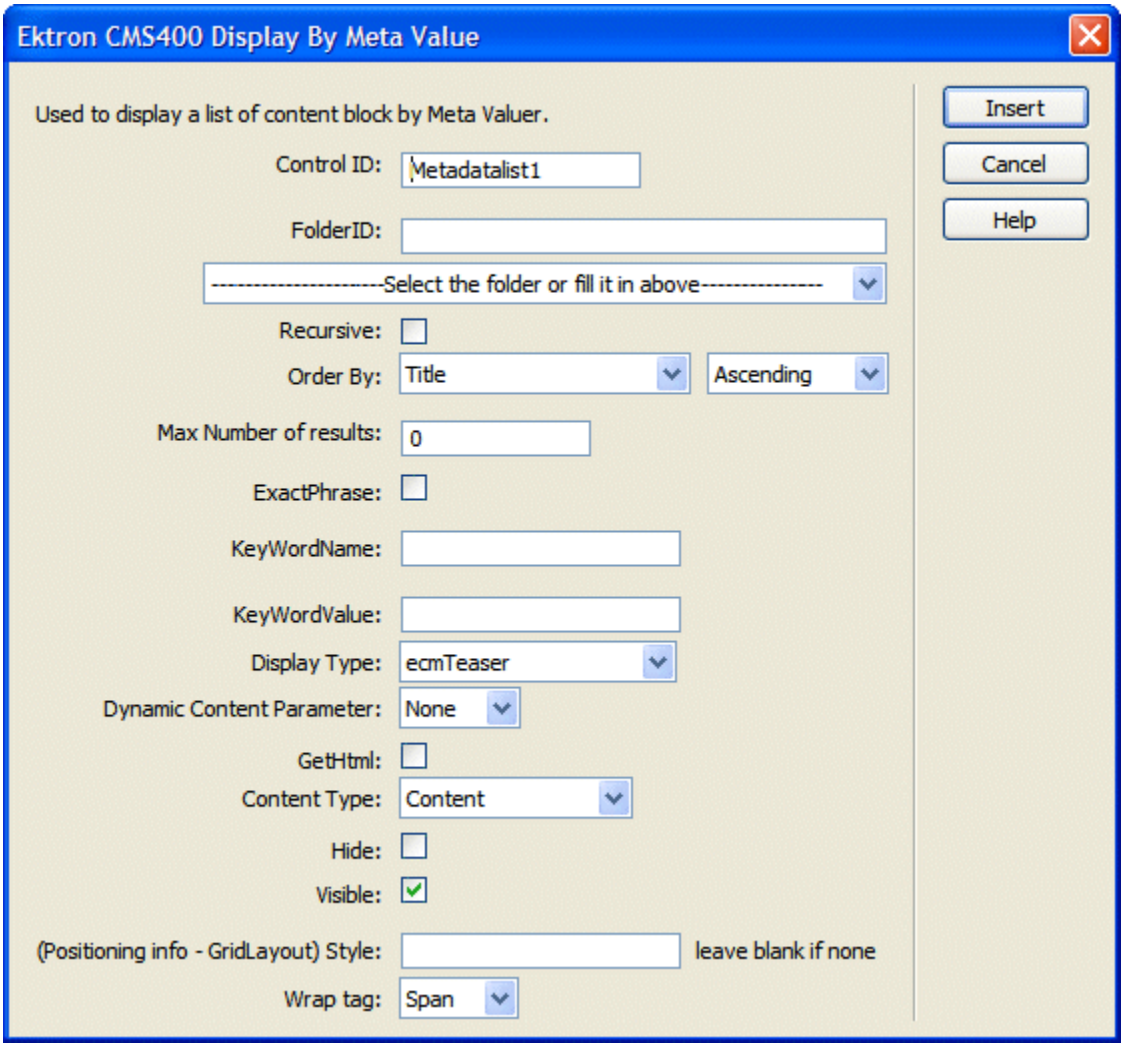
This tag appears on the following menu(s):

- **Commands > Ektron CMS400.NET (Server Controls) > CMS Function...**

To insert Display By Meta Value:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Functions, click **Display By Meta Value**

   The Display By Meta Value box appears.



3. Use the following table to assist you with inserting Display By Meta Value.

Field	Description
Control ID	A unique name for the function you are inserting.

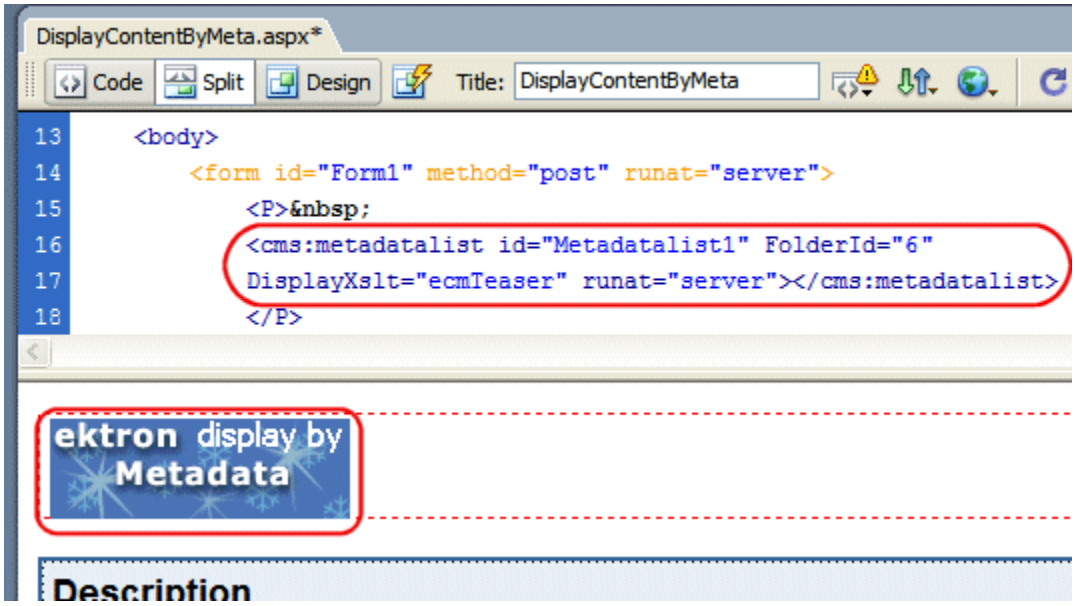
Field	Description
FolderID	The numeric id of the folder that you want to use. If you do not know the folder ID, you can navigate to it using the Select folder drop down box.
Select Folder	Select a folder path to be used. When a folder is selected, the folderID textbox is filled with that folder's id number. If you insert the Folder ID, you do not need to select a folder.
Recursive	Determines whether the Display By Meta Value control uses this folder's child folders.
Order By	• Specify how you want to order the results that are generated. Also specify if you want the results in ascending or descending order. • **Title** - The title of the content block • **Date Created** - The date the content block was created • **Date Modified** - The date the content block was last modified • **Last Author Last Name** - The last author's last name • **Last Author First Name** - The last author's first name • **ID** - The content blocks ID • **Ascending** - Alphabetic or numerical order • **Descending** - Reverse alphabetic or numerical order
Max Number of results	The Maximum number of content blocks returned (0=unlimited).
ExactPhrase	Determines whether the KeyWordValue needs to match the phrase exactly. For example, if "site" is the KeyWordValue and the title of a content block is "Welcome to the site" and ExactPhrase is set to true, you would not see the content block in the metadata list. This is because "site" does not equal "Welcome to the site". ☑ **Checked** = Match the exact phrase ☐ **Unchecked** = Doesn't need to match exact phrase
KeyWordName	The KeyWordName is the container where the KeyWordValues are located. Some examples of a KeyWordName might be "Keywords or Title." For information on creating KeyWordNames for Metadata, see the **Ektron CMS400.NET** Administrators Manual > "Managing Content" > "Metadata" > "Managing Metadata Definitions" > "Adding a Metadata Definitions"

Field	Description
KeyWordValue	The value that is associated with the KeyWordName. Some examples of a KeyWordValue might be “Home; page; or company”
Display Type	Determines how the information on the page is displayed. **None**-databind only **ecmNavigation** - lists the title of every content block in the folder *See Also:* ”Example of ecmNavigation Display” on page 105 **ecmTeaser** - lists the title of every content block in the folder plus the content summary *See Also:* ”Example of ecmTeaser Display” on page 106
Dynamic Content Parameter	Checks the QueryString for the content block ID value and replaces the Display By Meta Value Control with content when an ID is specified. Leave blank to always display the Display By Meta Value Control. **None** - Use Default **ID** - The control uses the content block passed as a URL parameter.
GetHtml	Set to **True** if you want to retrieve and display content (html body) for all content blocks in the collection. For example, display content inside a web server control such as a DataGrid.
Content Type	Defines what type of list summary appears. **Content** - Displays only content blocks in the folder. **All Types** - Displays all content in the folder. **Forms** - Displays only forms in the folder. **Archive_Content** - Displays archived content from the folder. **Archive_Forms** - Displays archived forms from the folder.
Hide	Used to hide the function in design time and run time. ☑ **Checked** = Hide function ☐ **Unchecked** = Show function Note: If Hide and Visible are both checked, the property is hidden.

Field	Description
Visible	Used to show or hide the function at run time. ☑ **Checked** = Hide function ☐ **Unchecked** = Show function Note: If Hide and Visible are both checked, the property is hidden.
Style (Positioning Info - GridLayout)	Set position information of a control as absolute, when using a GridLayout Web page. For example: style="Z-INDEX: 101; LEFT: 328px; POSITION: absolute; TOP: 240px"
Wrap Tag	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code.

4. When completed, click **insert**.

   A graphic defines where the Display By Meta Value appears on your template.



Shown below is an example of the code that is inserted:

Platform	Code View
.NET	`<cms:metadatalist id="Metadatalist1" FolderId="69" DisplayXslt="ecmTeaser" runat="server"></cms:metadatalist>`
ASP	This function is not available as an Ektron Dreamweaver Extension for ASP.
ColdFusion	This function is not available as an Ektron Dreamweaver Extension for ColdFusion.
JSP	This function is not available as an Ektron Dreamweaver Extension for JSP.
PHP	This function is not available as an Ektron Dreamweaver Extension for PHP.

Inserting a Standard Menu

For a description of the Standard Menu function, see "Menu Server Control" on page 153.

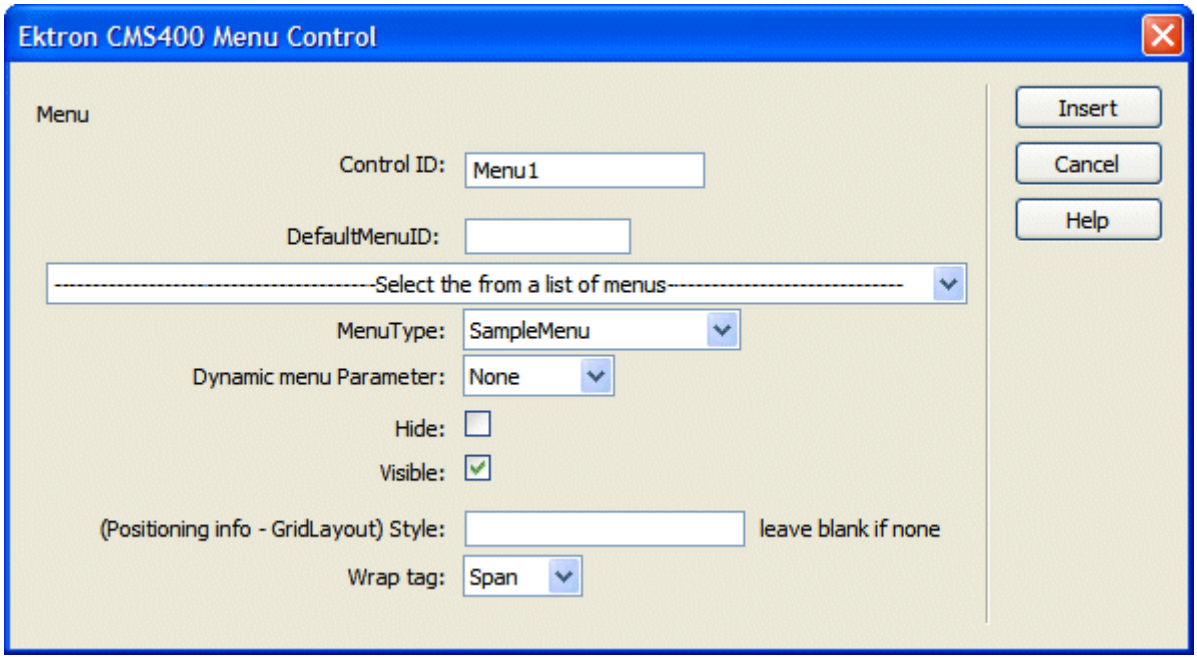
This tag appears on the following menu(s):

- **Commands > Ektron CMS400.NET (Server Controls) > CMS Function...**

To insert a Standard Menu:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Functions, click **Standard Menu**

The Standard Menu Control box appears.



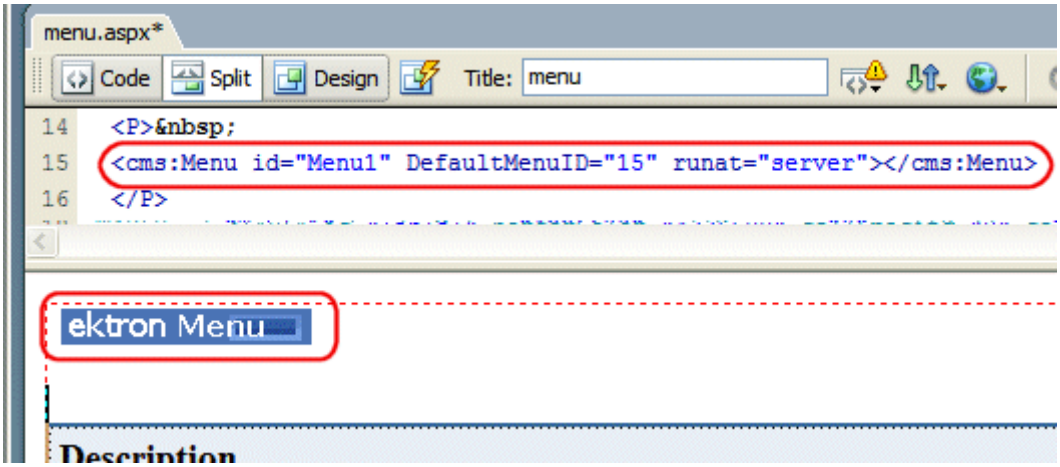
3. Use the following table to assist you with inserting a Standard Menu.

Field	Description
Control ID	A unique name for the function you are inserting.
DefaultMenuID	The numeric id of the menu that you want to insert. If you do not know the DefaultMenuID, you can navigate to it using the Select Menu drop down box.
Select Menu	Select a menu to be used. When a menu is selected, the DefaultMenuID textbox is filled with that menu's id number. If you insert the DefaultMenuID, you do not need to select a menu.
MenuType	The type of menu to render. **None** - databind only **SampleMenu** - A sample display, formatted as a bulleted menu list **TreeMenu** - A sample display, formatted as a folder tree. You can expand the tree by clicking on the folder icon. For more information on using the SampleMenu and TreeMenu Display, see "Using DisplayXslt Samples" on page 156.

Field	Description
Dynamic Menu Parameter	Checks the QueryString for the menu value and replaces the menu with content when ID is specified. Leave blank to always display the menu. **None** - Use Default **menu_ID** - The control uses the menu passed as a URL parameter.
Hide	Used to hide a menu in design time and run time. ☑ **Checked** = Hide function ☐ **Unchecked** = Show function Note: If Hide and Visible are both checked, the property is hidden.
Visible	Used to show or hide the function at run time. ☑ **Checked** = Hide function ☐ **Unchecked** = Show function Note: If Hide and Visible are both checked, the property is hidden.
Style (Positioning Info - GridLayout)	Set position information of a control as absolute, when using a GridLayout Web page. For example: style="Z-INDEX: 101; LEFT: 328px; POSITION: absolute; TOP: 240px"
Wrap Tag	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code.

4. When completed, click **insert**.

   A graphic defines where the Standard Menu appears on your template.



Shown below is an example of the code that is inserted:

Platform	Code View
.NET	`<cms:Menu id="Menu1" DefaultMenuID="15" runat="server"></cms:Menu>`
ASP	This function is not available as an Ektron Dreamweaver Extension for ASP.
ColdFusion	This function is not available as an Ektron Dreamweaver Extension for ColdFusion.
JSP	This function is not available as an Ektron Dreamweaver Extension for JSP.
PHP	This function is not available as an Ektron Dreamweaver Extension for PHP.

Inserting a DHTML Menu

For a description of the DHTML Menu function, see "DhtmlMenu Server Control" on page 170.

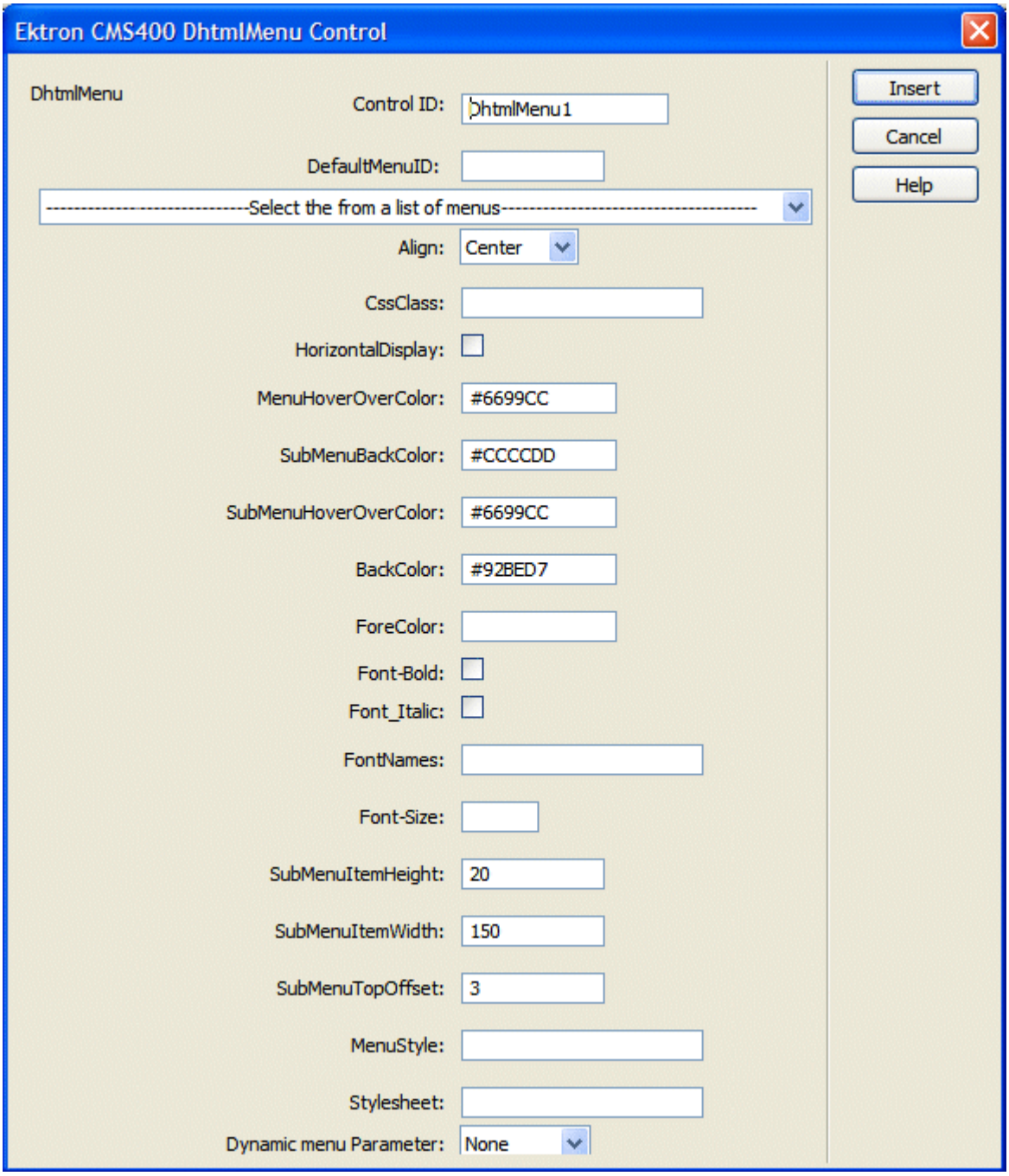
This tag appears on the following menu(s):

- **Commands > Ektron CMS400.NET (Server Controls) > CMS Function...**

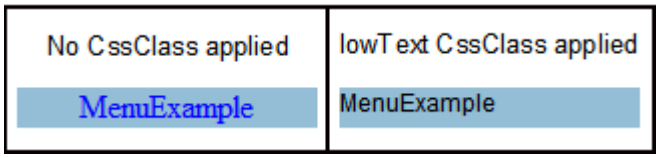
To insert a DHTML Menu:

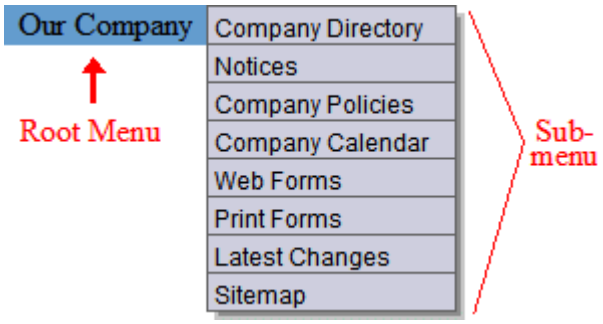
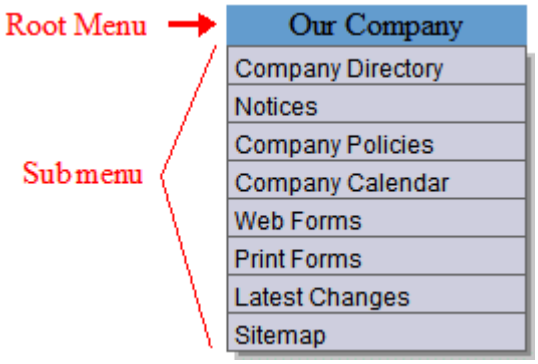
1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Functions, click **DHTML Menu**

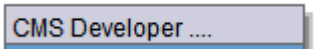
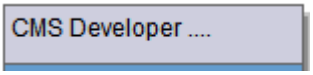
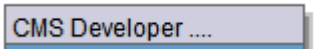
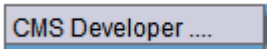
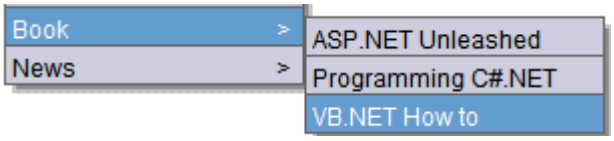
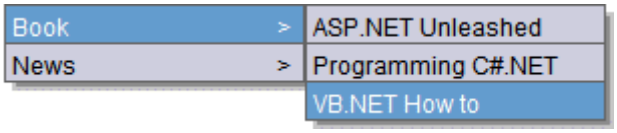
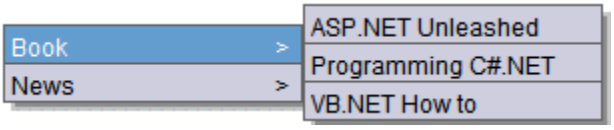
   The DHTML Menu Control box appears.



3. Use the following table to assist you with inserting a DHTML Menu.

Field	Description
Control ID	A unique name for the function you are inserting.
DefaultMenuID	The numeric id of the menu that you want to insert. If you do not know the DefaultMenuID path, you can navigate to it using the Select Menu drop down box.
Select Menu	Select a menu to be used. When a menu is selected, the DefaultMenuID textbox is filled with that menu's id number. If you insert the DefaultMenuID, you do not need to select a menu.
Align	Specifies the horizontal alignment of the root menu text. The default is **Center.** **NotSet** - Use the default setting. **Left** - Aligns text to the left. **Center** - Aligns text to the center. **Right** - Aligns text to the right. **Justify** - Aligns text justified to both margins.
CssClass	Apply a .css class to the root menu. Enter the class name from the .css defined in the Style sheet property. Since pop_style.css is the default style sheet, you can use its classes without defining it in the Stylesheet property. For example, if you enter lowtext in the property, the root menu changes as shown below. 

Field	Description
HorizontalDisplay	Displays submenu to the right of the root menu. This only affects the first submenu of the root menu. All subsequent submenus appear to the right of each submenu. The default is **False** **True** = Submenu appears to the right of the root menu  **False** = Submenu appears below the root menu 
MenuHoverOverColor	Sets the background color for the root menu when hovering over it.
SubMenuBackColor	Sets the background color for the submenu.
SubMenuHoverOverColor	Sets the background color for the submenu when hovering over it.
BackColor	Sets the background color for the root menu. This property appears in the **Appearance** category
ForeColor	Sets the color of the font for the root menu. This property appears in the **Appearance** category.

Field	Description
Font-Bold	Checking this box allows the font to appear **bold**.
Font_Italic	Checking this box allows the font to appear *italics.*
FontNames	Enter the name of a font to use.
Font-Size	The size of the font.
SubMenuItemHeight	Sets the height of each submenu item in pixels. The Default is 20 Example set to 20:  Example set to 30: 
SubMenuItemWidth	Sets the width of each submenu item in pixels. The Default is 150 Example set to 150:  Example set to 130: 
SubMenuTopOffset	Sets the location of the top of the submenu. This number is in pixels. The default is 5 Example set to 5:  Example set to 0 (zero):  Example set to -10: 

Field	Description
MenuStyle	A specified inline style that defines the root menu. For example, if you enter the following in the MenuStyle property box: `font: bold 18px Arial;color:royalblue; background-color:slateblue`
StyleSheet	Specify the location and style sheet for the submenu. For example: `\Workarea\csslib\mytest.css.` Leave blank to use the default. `pop_style.css` is the default style sheet. It is located in `\webroot\CMS400Example\Workarea\csslib`. For more information on style sheets, see "Using a Style Sheet" on page 175 The Stylesheet property can be used in conjunction with the CssClass property to define the root menu.
Dynamic Menu Parameter	Checks the QueryString for the menu value and replaces the menu with content when ID is specified. Leave blank to always display the menu. **None** - Use Default **menu_ID** - The control uses the menu passed as a URL parameter.

4. When completed, click **insert**.

   A graphic defines where the DHTML Menu appears on your template.

   Shown below is an example of the code that is inserted:

Platform	Code View
.NET	`<cms:DhtmlMenu id="DhtmlMenu1" DefaultMenuID="16" ForeColor="" runat="server"></cms:DhtmlMenu>`
ASP	This function is not available as an Ektron Dreamweaver Extension for ASP.
ColdFusion	This function is not available as an Ektron Dreamweaver Extension for ColdFusion.
JSP	This function is not available as an Ektron Dreamweaver Extension for JSP.
PHP	This function is not available as an Ektron Dreamweaver Extension for PHP.

Inserting a Language Select Box

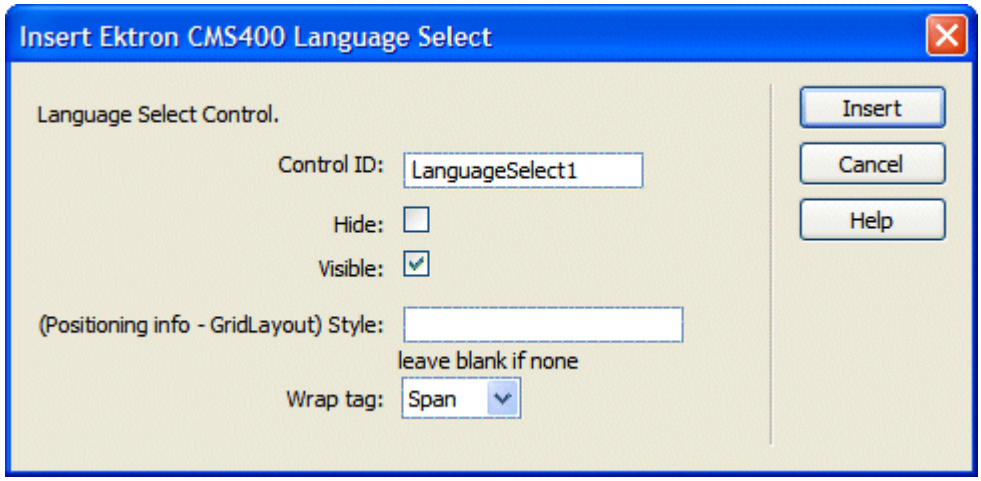
For a description of the Language Select Box, see "LanguageSelect Server Control" on page 137.

This tag appears on the following menu(s):

- **Commands > Ektron CMS400.NET (Server Controls) > CMS Function...**

To insert a Language Select Box:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Functions, click **Language Select**
3. The Language Select box appears.



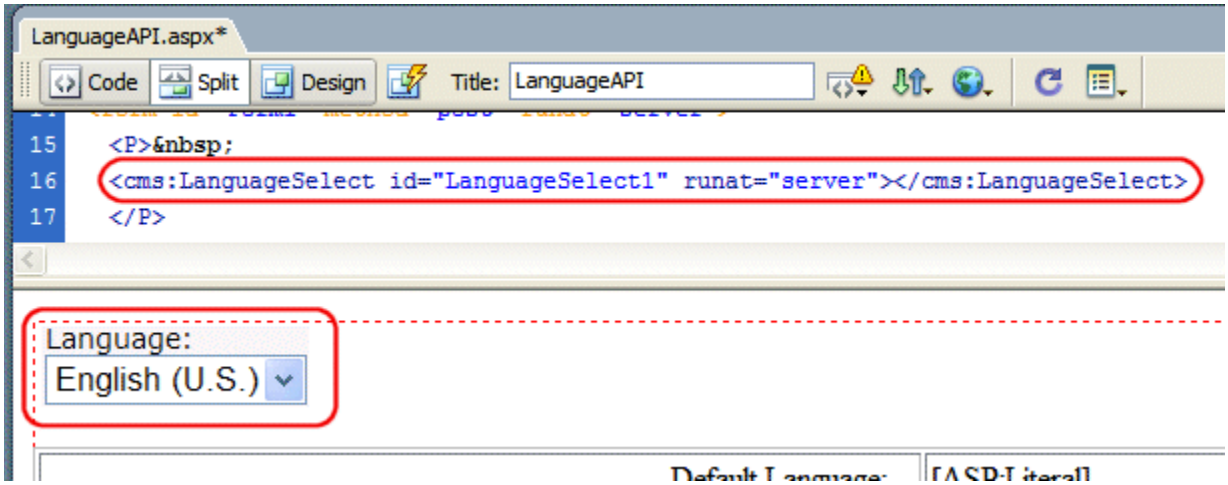
4. Use the following table to assist you with inserting a Language Select box.

Field	Description
Control ID	A unique name for the function you are inserting.

Field	Description
Hide	Used to hide a function in design time and run time. ☑ **Checked** = Hide function ☐ **Unchecked** = Show function Note: If Hide and Visible are both checked, the property is hidden.
Visible	Used to show or hide the function at run time. ☑ **Checked** = Hide function ☐ **Unchecked** = Show function Note: If Hide and Visible are both checked, the property is hidden.
Style (Positioning Info - GridLayout)	Set position information of a control as absolute, when using a GridLayout Web page. For example: style="Z-INDEX: 101; LEFT: 328px; POSITION: absolute; TOP: 240px"
Wrap Tag	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code.

5. When completed, click **insert**.

   A graphic defines where the Language Select appears on your template.



Shown below is an example of the code that is inserted.

Platform	Code View
.NET	`<cms:LanguageSelect id="LanguageSelect1" runat="server"></cms:LanguageSelect>`
ASP	This function is not available as an Ektron Dreamweaver Extension for ASP.
ColdFusion	This function is not available as an Ektron Dreamweaver Extension for ColdFusion.
JSP	This function is not available as an Ektron Dreamweaver Extension for JSP.
PHP	This function is not available as an Ektron Dreamweaver Extension for PHP.

Inserting a Language API

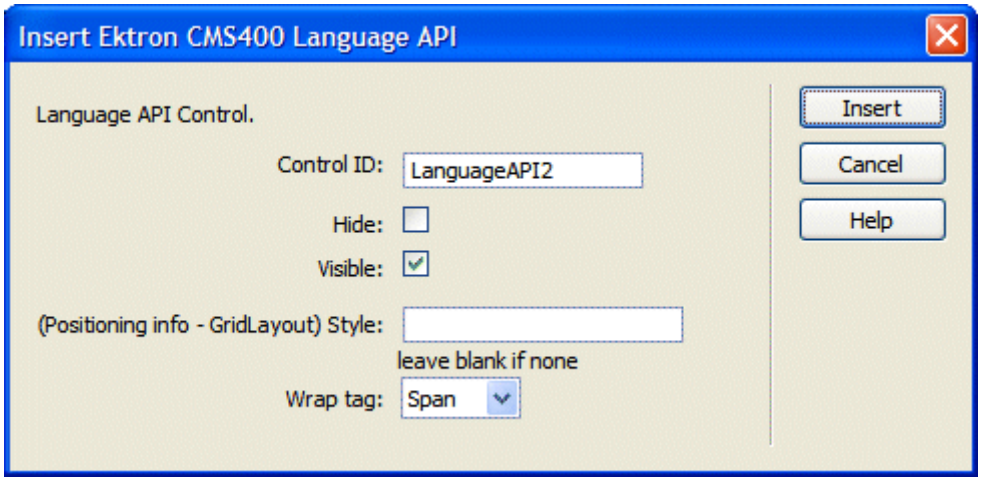
For a description of the Language API function, see "LanguageAPI Server Control" on page 134.

This tag appears on the following menu(s):

- **Commands > Ektron CMS400.NET (Server Controls) > CMS Function...**

To insert a Language API:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Functions, click **Language API**
3. The Language API box appears.



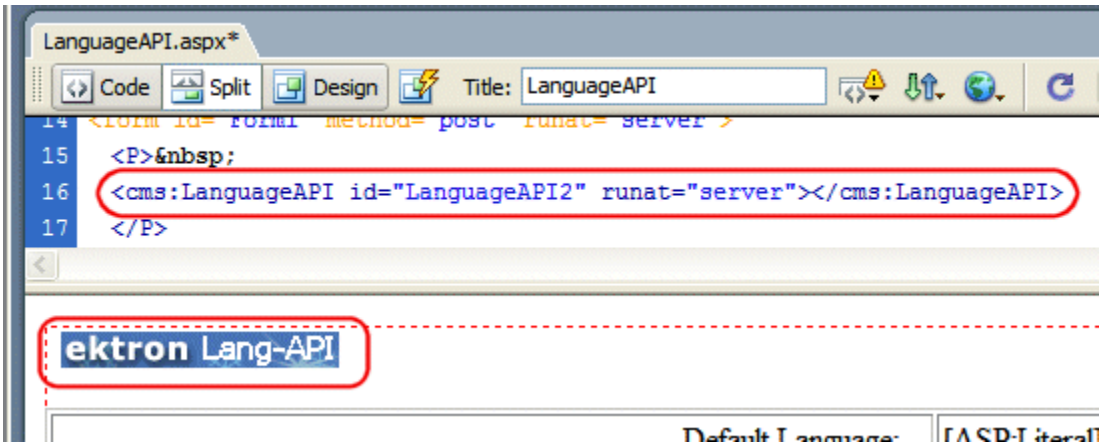
4. Use the following table to assist you with inserting a Language API box.

Field	Description
Control ID	A unique name for the function you are inserting.
Hide	Used to hide a Language API in design time and run time. ☑ **Checked** = Hide Language API ☐ **Unchecked** = Show Language API Note: If Hide and Visible are both checked, the property is hidden.
Visible	Used to show or hide the function at run time. ☑ **Checked** = Hide function ☐ **Unchecked** = Show function Note: If Hide and Visible are both checked, the property is hidden.

Field	Description
Style (Positioning Info - GridLayout)	Set position information of a control as absolute, when using a GridLayout Web page. For example: style="Z-INDEX: 101; LEFT: 328px; POSITION: absolute; TOP: 240px"
Wrap Tag	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code.

5. When completed, click **insert**.

   A graphic defines where the Language API appears on your template.



Shown below is an example of the code that is inserted:

Platform	Code View
.NET	`<cms:LanguageAPI id="LanguageAPI2" runat="server"></cms:LanguageAPI>`
ASP	This function is not available as an Ektron Dreamweaver Extension for ASP.
ColdFusion	This function is not available as an Ektron Dreamweaver Extension for ColdFusion.
JSP	This function is not available as an Ektron Dreamweaver Extension for JSP.

Platform	Code View
PHP	This function is not available as an Ektron Dreamweaver Extension for PHP.

Inserting an RSS Aggregator

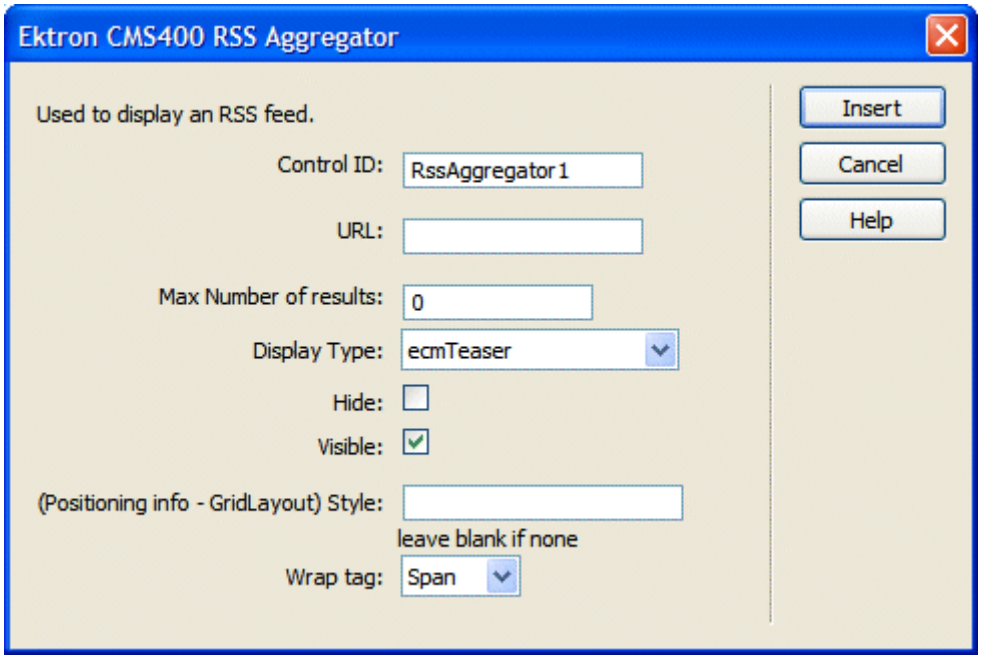
For a description of the RSS Aggregator function, see "RssAggregator Server Control" on page 196.

This tag appears on the following menu(s):

- **Commands > Ektron CMS400.NET (Server Controls) > CMS Function...**

To insert an RSS Aggregator:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Functions, click **RSS Aggregator**.
3. The RSS Aggregator dialog box appears.



4. Use the following table to assist you with inserting an RSS Aggregator.

Field	Description
Control ID	A unique name for the function you are inserting.
URL	The RSS feed path for the server control. For example: `http://msdn.microsoft.com/rss.xml`
Max Number of Results	The Maximum number of items from an RSS feed that are returned (0=unlimited).
Display Type	Determines how the information on the page is displayed **None**-databind only **ecmNavigation** - lists the title of every RSS feed item *See Also:* "Example of ecmNavigation Display" on page 105 **ecmTeaser** - lists a title and a description of every RSS feed item *See Also:* "Example of ecmTeaser Display" on page 106 **Path to Custom Xslt** - Enter the path to an Xslt that determines the display of the page
Hide	Used to hide a Language API in design time and run time. ☑ **Checked** = Hide ☐ **Unchecked** = Show Language API Note: If Hide and Visible are both checked, the property is hidden.
Visible	Used to show or hide the function at run time. ☑ **Checked** = Hide function ☐ **Unchecked** = Show function Note: If Hide and Visible are both checked, the property is hidden.
Style (Positioning Info - GridLayout)	Set position information of a control as absolute, when using a GridLayout Web page. For example: style="Z-INDEX: 101; LEFT: 328px; POSITION: absolute; TOP: 240px"

Field	Description
Wrap Tag	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code.

5. When completed, click **insert**.

   A graphic defines where the RSS Aggregator appears on your template.

Shown below is an example of the code that is inserted:

Platform	Code View
.NET	`<CMS:RssAggregator id="RssAggregator1" URL="http://msdn.microsoft.com/rss.xml" DisplayXslt="ecmTeaser" runat="server"></CMS:RssAggregator>`
ASP	This function is not available as an Ektron Dreamweaver Extension for ASP.
ColdFusion	This function is not available as an Ektron Dreamweaver Extension for ColdFusion.
JSP	This function is not available as an Ektron Dreamweaver Extension for JSP.
PHP	This function is not available as an Ektron Dreamweaver Extension for PHP.

Inserting a BreadCrumb Trail

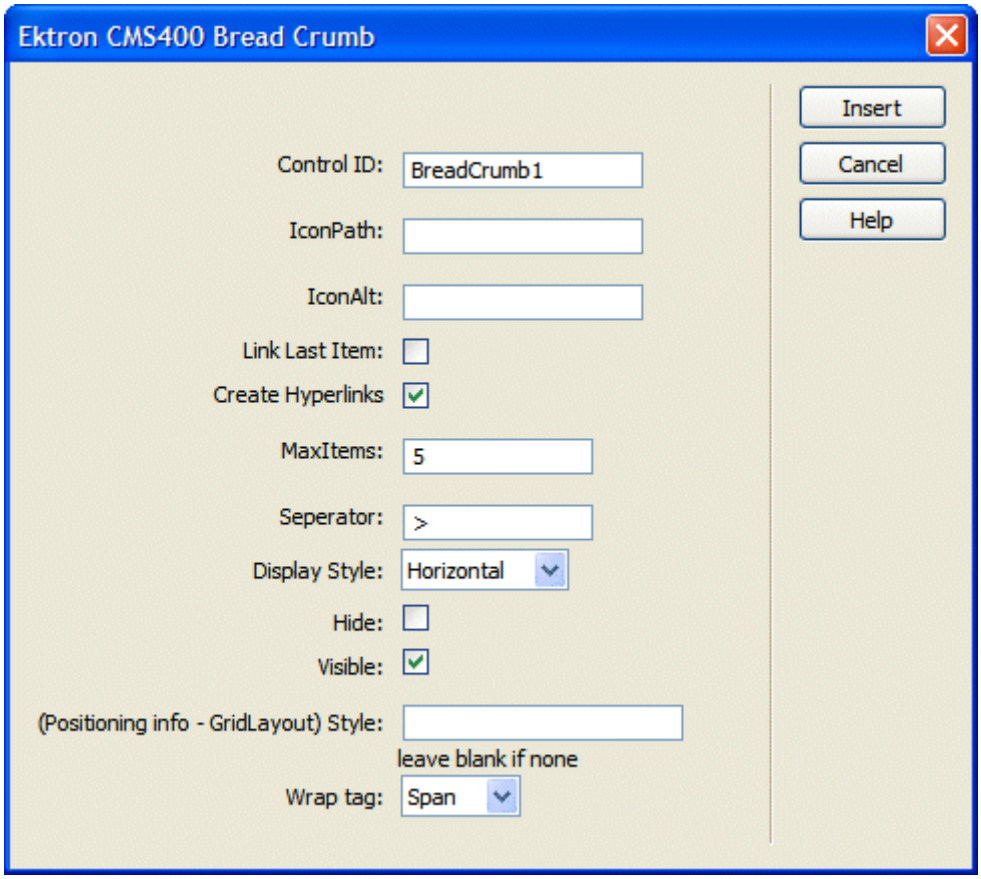
For a description of the BreadCrumb function, see "BreadCrumb Server Control" on page 57.

This tag appears on the following menu(s):

- **Commands > Ektron CMS400.NET (Server Controls) > CMS Function...**

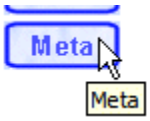
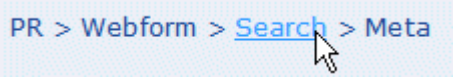
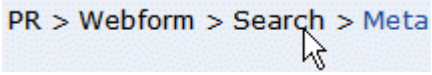
To insert a BreadCrumb Trail:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Functions, click **BreadCrumb**.
3. The BreadCrumb dialog box appears.



4. Use the following table to assist you with inserting a BreadCrumb Trail.

Field	Description
Control ID	A unique name for the function you are inserting.

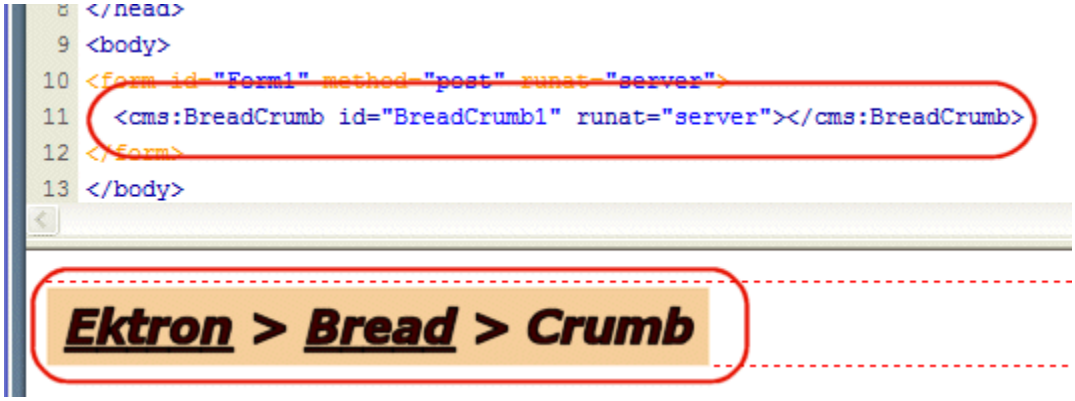
Field	Description
IconPath	If you want the breadcrumb trail to display an image to identify this Web form, enter the path to the image. **Important:** The image location must be relative to the Web root. For example: `\CMS400Example\Workarea\Images\bc_meta_icon.gif`
IconAlt	If you define an image in the **IconPath** property, enter any "Alt" text that should appear when a site visitor hovers the cursor over that image. Here is an example. 
Link Last Item	Use this property to determine whether the last breadcrumb item appears as a hyperlink on this Web form. If this property is set to **true**, and a user clicks the item, the current page reappears. **True** = last item is a hyperlink **False** = last item is an image and/or text only; the user cannot click on it
Create Hyperlinks	Unchecking this box makes the breadcrumb trail appear as non-hyperlinked plain text. The default is **Checked**. ☑ **Checked** = breadcrumb trail is hyperlinked  ☐ **Unchecked** = breadcrumb trail is plain text 
MaxItems	Enter the maximum number of items in the breadcrumb trail on this Web form. The default is **5**. If you set a value of 1 or greater and the user navigates beyond that number of pages, only the most recent pages appear. The older pages disappear from the trail. To allows an unlimited number of breadcrumb trail items on this Web form, enter zero (**0**).

Field	Description
Separator	Enter one or more characters to separate the items in a breadcrumb trail on this Web form. The default character is the greater than sign (**>**). Dev > WebForm
DisplayStyle	Indicate how to display the breadcrumb trail: horizontally or vertically. The default is Horizontal. This is an example of **Horizontal.** Dev > WebForm > pr > FAQ > Meta* This is an example of **Vertical**. > Dev > WebForm > pr > FAQ > Meta*
Hide	Used to hide a Language API in design time and run time. ☑ **Checked** = Hide ☐ **Unchecked** = Show Language API Note: If Hide and Visible are both checked, the property is hidden.
Visible	Used to show or hide the function at run time. ☑ **Checked** = Hide function ☐ **Unchecked** = Show function Note: If Hide and Visible are both checked, the property is hidden.
Style (Positioning Info - GridLayout)	Set position information of a control as absolute, when using a GridLayout Web page. For example: style="Z-INDEX: 101; LEFT: 328px; POSITION: absolute; TOP: 240px"

Field	Description
Wrap Tag	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code.

5. When completed, click **insert**.

   A graphic defines where the BreadCrumb Trail appears on your template.



Shown below is an example of the code that is inserted:

Platform	Code View
.NET	`<cms:BreadCrumb id="BreadCrumb1" runat="server"></cms:BreadCrumb>`
ASP	This function is not available as an Ektron Dreamweaver Extension for ASP.
ColdFusion	This function is not available as an Ektron Dreamweaver Extension for ColdFusion.
JSP	This function is not available as an Ektron Dreamweaver Extension for JSP.
PHP	This function is not available as an Ektron Dreamweaver Extension for PHP.

Inserting a Membership Control

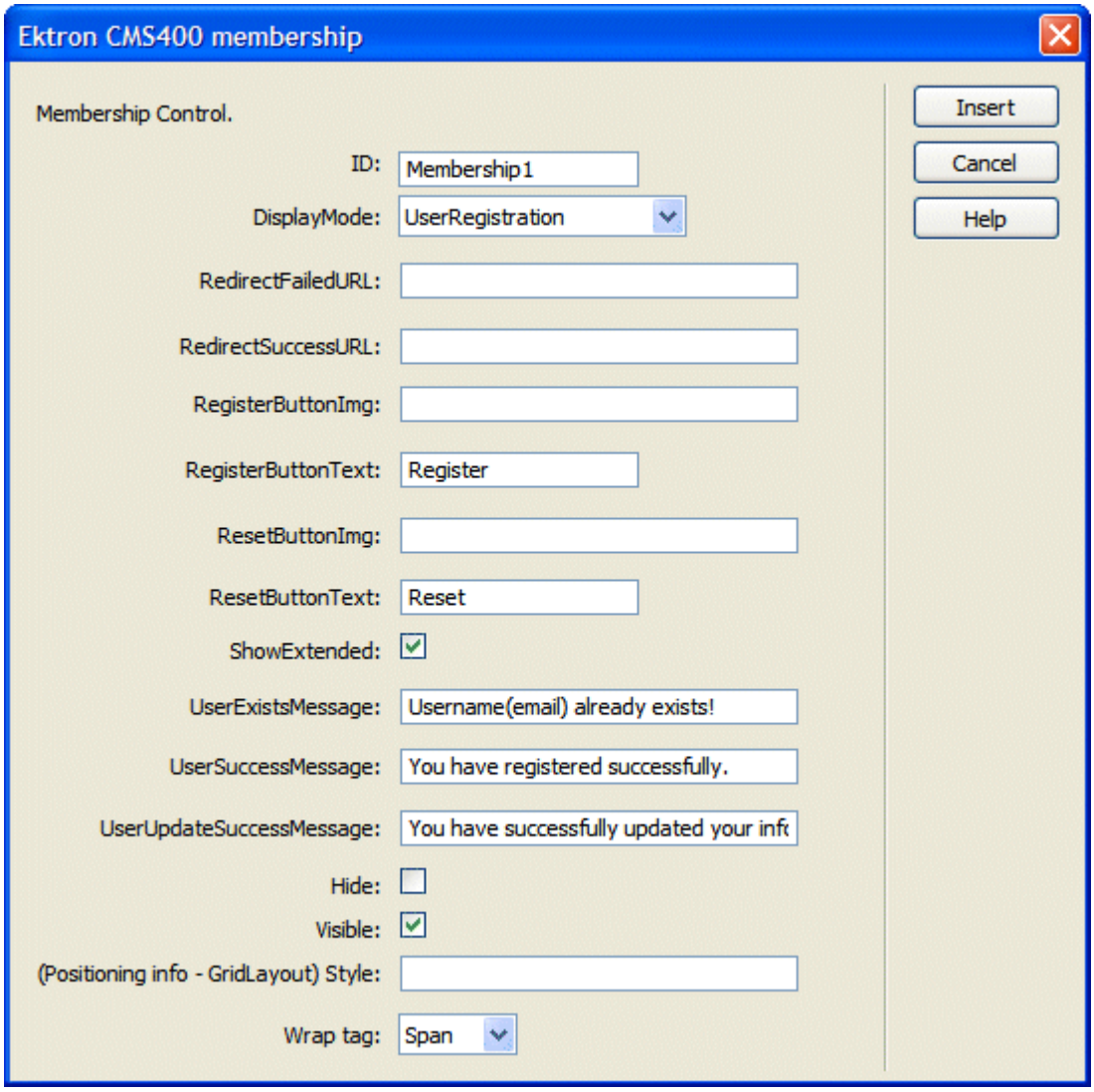
For a description of the Membership function, see "Membership Server Control" on page 146.

This tag appears on the following menu(s):

- **Commands > Ektron CMS400.NET (Server Controls) > CMS Function...**

To insert a Membership Control:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Functions, click **Membership**.
3. The Membership dialog box appears.



4. Use the following table to assist you with inserting a Membership Control.

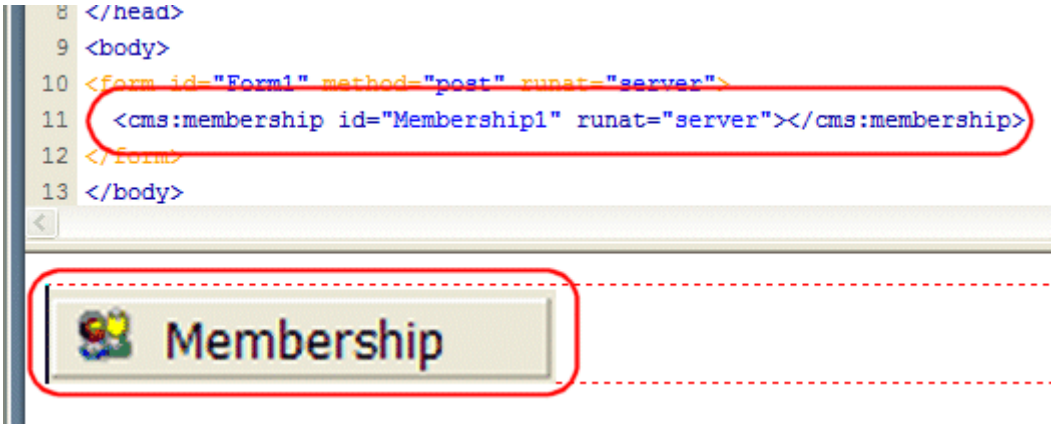
Field	Description
ID	A unique name for the function you are inserting.
DisplayMode	The type of membership form displayed. Choose from the following: • **UserRegistration** - Lets a site visitor to register as a membership user. • **ResetPassword** - Lets a membership user to reset his password. • **UnsubscribeSecured** - Lets a membership user unsubscribe. The membership user needs to enter his username and password to unsubscribe. • **UnsubscribeUnsecured** - Lets a membership user unsubscribe by entering his membership username only. • **AccountActivate** - Let a site visitor activate his membership account by entering the ID number sent him in the account verification email. For more information on the DisplayMode property, see "DisplayMode" on page 151.
RedirectedFailedURL	The URL where a membership user is sent if the registration fails. If the page resides in the same folder as the registration page, you only need to enter the name of the page. For example, RegFailed.aspx. If the redirect page is located in a subfolder, you need to add the folder path. For example, members\RegFailed.aspx.
RedirectedSuccessURL	The URL where a membership user is sent if the registration fails. If the page resides in the same folder as the registration page, you only need to enter the name of the page. For example, RegFailed.aspx. If the redirect page is located in a subfolder, you need to add the folder path. For example, members\RegFailed.aspx.
RegisterButtonImg	Lets you add an image in place of the text on the register button.
RegisterButtonText	Text that appears on the Register button. The default is Register. If you use a register button image, you do not see this text.
ResetButtonImg	Lets you add an image in place of the text on the reset button.
ResetButtonText	Text that appears on the reset button. The default is Reset. If you use a reset button image, you do not see this text.

Field	Description
ShowExtended	Decide if the Custom User Properties are available when using this control. The default setting is True. **True** = Show Custom User Properties **False** = Hide Custom User Properties **Note:** For more information on Custom User Properties, see the Administration Manual section “Custom User Properties”.
UserExistsMessage	The message that appears when a membership user already exists. The default message is: Username(email) already exists!
UserSuccessMessage	The message that appears when a membership user successfully registers. The default message is: You have registered successfully.
UserUpdateSuccessMessage	The message that appears when a membership user successfully updates their information. The default message is: You have successfully updated your information.
Hide	Used to hide a Language API in design time and run time. ☑ **Checked** = Hide ☐ **Unchecked** = Show Language API Note: If Hide and Visible are both checked, the property is hidden.
Visible	Used to show or hide the function at run time. ☑ **Checked** = Hide function ☐ **Unchecked** = Show function Note: If Hide and Visible are both checked, the property is hidden.
Style (Positioning Info - GridLayout)	Set position information of a control as absolute, when using a GridLayout Web page. For example: style="Z-INDEX: 101; LEFT: 328px; POSITION: absolute; TOP: 240px"

Field	Description
Wrap Tag	Allows a developer to specify a server control's tag. The default is **Span**. **Span** - The tag is used to designate an in-line portion of an HTML document as a span element. **Div** - The tag is used when you want to apply attributes to a block of code.

5. When completed, click **insert**.

   A graphic defines where the Membership Control appears on your template.



Shown below is an example of the code that is inserted:

Platform	Code View
.NET	`<cms:membership id="Membership1" runat="server"></cms:membership>`
ASP	This function is not available as an Ektron Dreamweaver Extension for ASP.
ColdFusion	This function is not available as an Ektron Dreamweaver Extension for ColdFusion.
JSP	This function is not available as an Ektron Dreamweaver Extension for JSP.
PHP	This function is not available as an Ektron Dreamweaver Extension for PHP.

XML Functions...

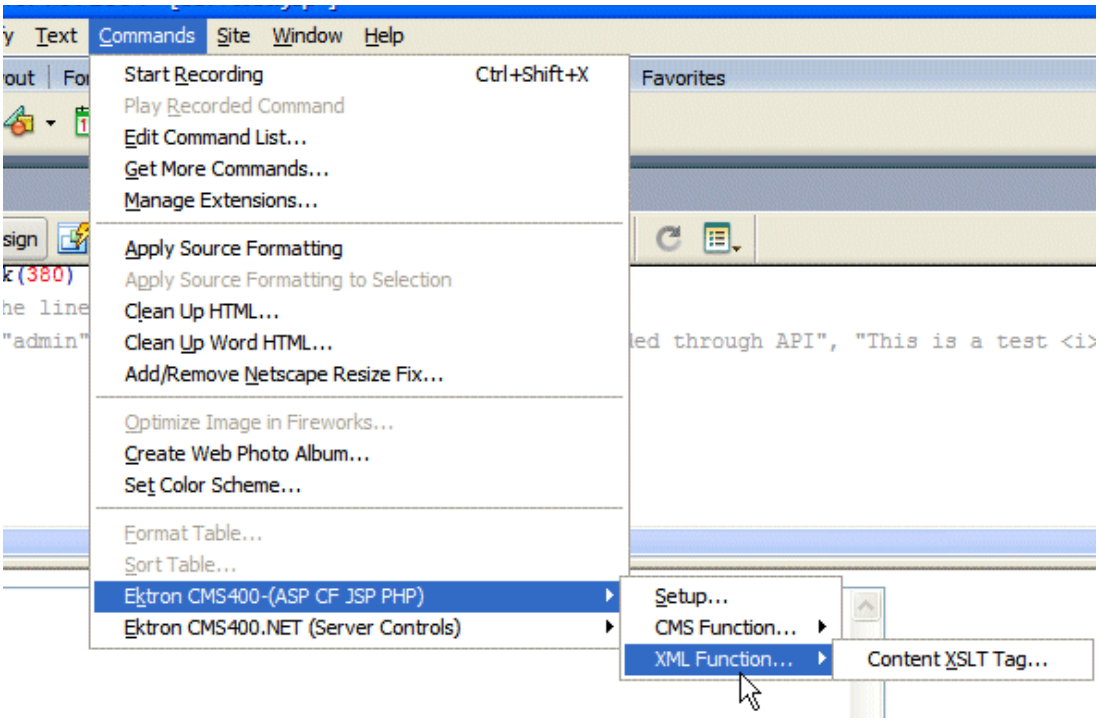
In addition to the many CMS functions, an additional XML related function may be inserted into your Web site templates. The following section explains this function.

Accessing XML Function Commands

To access the CMS XML Functions command menu, follow this path:

Commands > Ektron CMS400 - (ASP CF JSP PHP) > XML Functions...

A list of all CMS XML Functions that can be inserted is displayed.



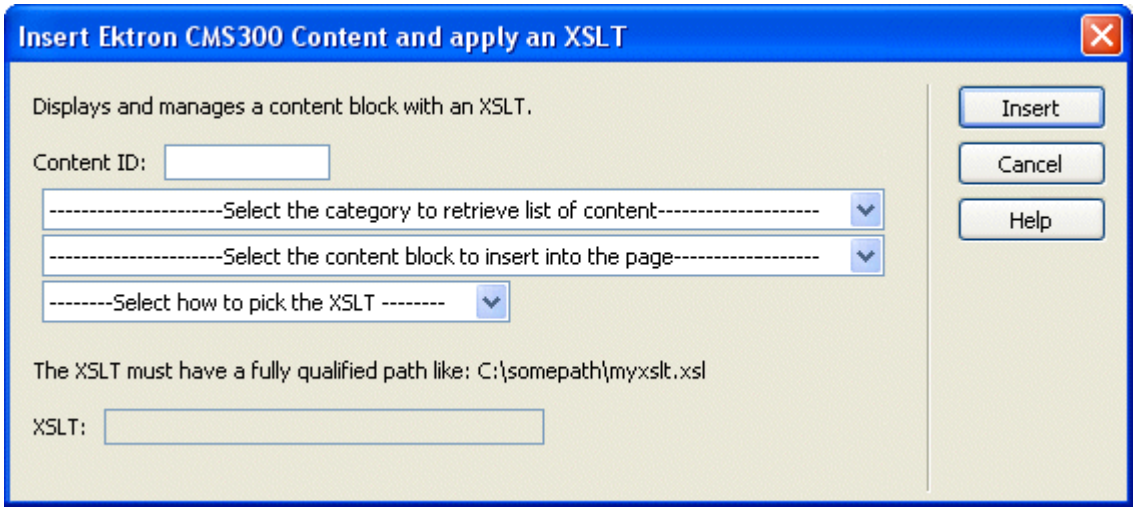
The following section explains how to insert the XML functions.

Insert Content XSLT Tag...

The insert content XSLT command allows you to insert an XML content block, and specify an XSLT (other than the default) to be applied to it.

To insert a content XSLT tag, perform the following steps:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET XML functions, click on **Insert Content XSLT Tag...**
3. The Insert Ektron CMS400.NET Content and Apply an XSLT Tag box is displayed.



4. Use the following table to assist you with this form.

Field	Description
Content ID	Either type in, or select from the drop down lists, the ID number of the XML content to insert.
Select XSLT	From the drop down list, select the XSLT you would like to apply to the XML content block.
Enter XSLT	If “Enter XSLT” is selected, you can insert the path and file name of the XSLT to apply to the content.

5. Click **insert**.

   A graphic showing you where the XML content block will be placed on the template is inserted.

Shown below is the code that will be inserted according to the platform that has been specified in the setup:

Platform	Code View
ASP	`<% ecmContentBlockEx 5, "", 1 %>`
ColdFusion	`<cfmodule id="5"  xsltpath="" xslttype="1" template="#request.ecm.AppPath#ecmContentBlockEx.cfm">`
JSP	`<%= ecmContentBlockEx(24, "", 0 ) %>`
PHP	`<?php ecmContentBlockEx (5, “”, 1); ?>`

Standard ASP.NET Methods

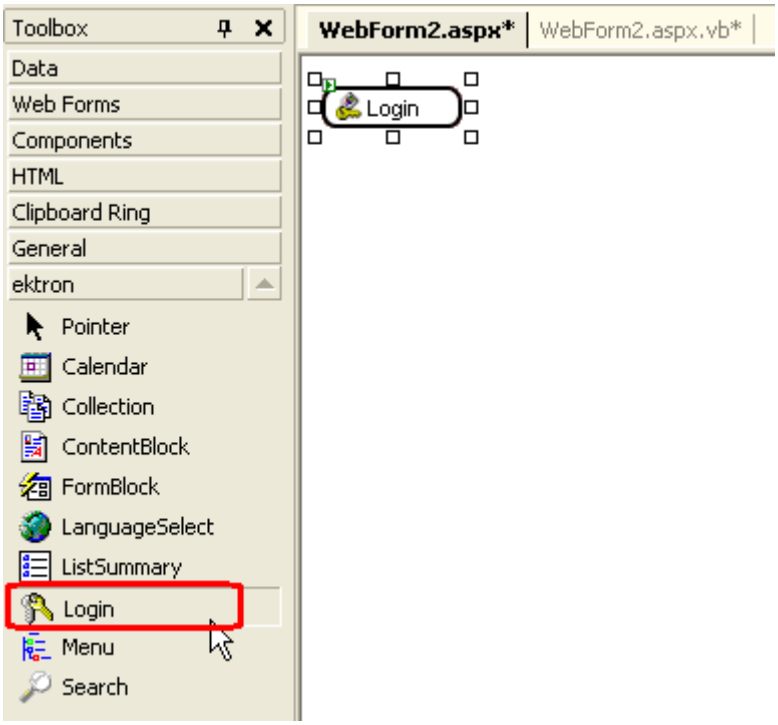
Use these methods to customize **Ektron CMS400.NET**. To use them, insert them onto your template pages to invoke content blocks, list summaries, and other objects stored in your database. They are the primary building blocks of your **Ektron CMS400.NET** Content Management System.

Replacement of Standard Methods

Beginning with Release 4.7, **Ektron CMS400.NET** utilizes ASP.NET server controls, which replace most Standard ASP.NET methods. With **Ektron CMS400.NET**'s new toolbox of pre-built, reusable, data-bindable ASP.NET controls, Ektron empowers developers working in Visual Studio .NET to rapidly integrate Ektron CMS components into their templates and sites. For more Information on Ektron CMS400.NET controls, see "Introduction to Ektron CMS400.NET Server Controls" on page 2.

The following examples shows the Login Server Control added by drag and drop vs. adding a Login method by the ASP.NET method.

Drag and Drop Login Server Control



Adding a Login function via an ASP.NET method.

```
<tr>

    <td>

        <%=AppUI.ecmLogin()%>

    </td>

</tr>
```

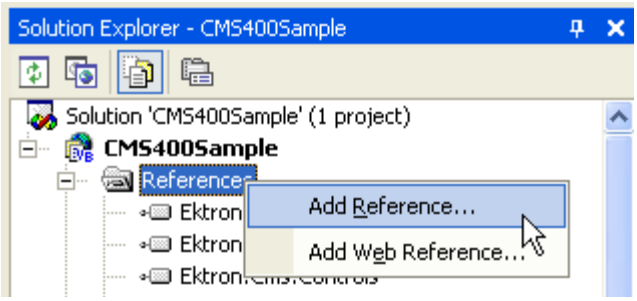
> **Best Practice**
>
> Ektron recommends using CMS400.NET server controls when available over Ektron standard ASP.NET methods. The methods are being replaced by the server controls.

NOTE You can still use the standard ASP.NET methods in **Ektron CMS400.NET**.

Referencing Methods in Visual Basic

Before you can use methods with Visual Basic, you must add a reference to them from Visual Basic. Follow these steps to do so.

1. Open Visual Basic.
2. Choose or add a project.
3. In Solution Explorer, right click **References** and choose **Add Reference**.

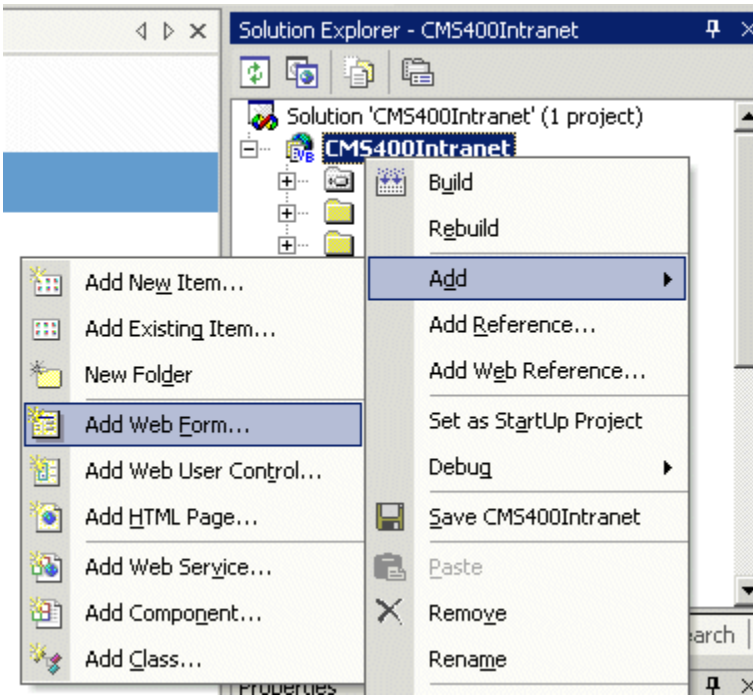


4. In the Add Reference dialog box, click **Browse**.
5. Go to `Inetpub/wwwroot/CMS400Example/Bin`.
6. To load the standard methods click `Ektron.Cms.UI.CommonUI.dll`. To load the XML Indexing methods, click `Ektron.Cms.Common.dll`.
7. Click **Open**, then **OK**.

Adding a New Web Form

To add a new Web form, follow these steps.

1. Open your project in Visual Studio .NET.
2. Right click the project.
3. Click **Add > Add Web Form**.



4. Click **Web Form**.
5. In the code behind, add an `Imports Ektron.Cms.UI.CommonUI` statement.

Requirements for Standard Methods

Ektron CMS400.NET maintains and displays content through the use of Ektron ASPX methods and ASP.NET assembly objects. To use the methods, include the following code in your template:

```
<%@ Page Language="vb" AutoEventWireup="false" %>

<%@ Import Namespace="Ektron.Cms.UI.CommonUI" %>

<% Dim AppUI As New ApplicationAPI %>
```

Each line is explained below.

Line	Description
`<%@ Page Language="vb" AutoEventWireup="false" %>`	A generic .net line of code that should always be placed in the template
`<%@ Import Namespace="Ektron.Cms.UI.CommonUI" %>`	An import statement that calls the namespace `Ektron.CMS.UI.CommonUI` to display information from the .net assembly file
`<% Dim AppUI As New ApplicationAPI %>`	Creates an instance of the ApplicationUI class that is used by each method

These lines call the Ektron .net assembly object to allow the page to function properly. They must appear at the top of each template file.

Standard Methods

To customize your Ektron CMS400.NET application, you need to be familiar with these methods.

Method	Description	More Information
Add Content	Programmatically adds a content block(s) to Ektron CMS400.NET	”Addcontent” on page 368
Collection	Displays links generated by collections.	”Collection Server Control” on page 103
Content Block	Displays and manages a content block.	”ContentBlock Server Control” on page 113
Event Calendar	Displays an event calendar	”Calendar Server Control” on page 68
Form	Displays a content block that is associated with a form.	”FormBlock Server Control” on page 129
IndexSearch	Displays on a Web page a set of search fields. A site visitor uses the page to search for information on your Web site.	”IndexSearch Server Control” on page 234

Method	Description	More Information
List Summary	Displays a list of content block summaries.	”ListSummary Server Control” on page 139
Login	Displays the login or logout button, which lets the user log in and use Ektron CMS400.NET.	”Login Server Control” on page 188
Metadata	Displays the metadata in the source of the Web page displaying the content.	”MetaData Server Control” on page 178
Menu	Customizes the DHTML menus feature	”DhtmlMenu Server Control” on page 170
Search	Inserts a text box with a search button in the template. When the user completes the search screen, it looks through XHTML content on the Web site.	”Search Server Control” on page 200
ShowSiteLanguage	Lets a site visitor select a language in which to view site.	”LanguageSelect Server Control” on page 137

Addcontent

The addcontent method lets you programmatically add a new content block to Ektron CMS400.NET without logging in. This method can be used for several applications, such as

- A Web form allowing you to collect data from users about the content block
- A loop function that moves information from a database to an Ektron CMS400.NET database

Here is the format for the addcontent method.

<% AddContent

(UserName,

Password,

Domain,

content_title,

content_comment,

ContentHtml,

SummaryHTML,

MetaInfo,

ContentLanguage,

FolderID,

GoLive,

EndDate,

ErrString

)%>

IMPORTANT! Every string that you pass to the function must be HTML encoded. Here is the VB.Net syntax for encoding a string.

```
myTitle = “John’s Book”
myTitle = Server.HtmlEncode(myTitle)
```

This function is basically the same as the AddContent function in ASP. See ”Add Content” on page 385 for details. The only difference between the ASPX function and the ASP one is the syntax for implementing it.

Add Content Example

The following is an example of the addcontent method hard coded in a template.

```
<tr>

  <td>

    <% AddContent ("admin", "admin", "", "Content Title", "Comment",
    "This is the <b>body</b> of the content.", "This is the
    <i>summary</i> for the content.", 1, 31, "06-14-2004 11:55 AM",
    "", "<metadata><meta id=""3"">ContentTitle</meta></metadata>",

    "") %>

  </td>

</tr>
```

Custom ASP Functions

Ektron CMS400.NET displays content on your Web site through the use of custom Ektron ASP functions and ComObjects. Your template you must include the following file to be able to use the custom functions:

```
<!-- #include file="applicationAPI.asp" -->
```

This file (**`application.asp`**) calls the file with the custom Ektron functions to allow the page to function properly. The `include` statement must appear at the top of every template file that uses an Ektron CMS400.NET custom ASP function.

IMPORTANT! `ContentWS.asmx` is the web services (WSDL) file that the custom ASP functions use to talk to **Ektron CMS400.NET**. The WSDLpath located in the `applicationAPI.asp` file must be set to location of the WSDL file. The default WSDLpath is `http://localhost/CMS400Example/WorkArea/WebServices/ContentWS.asmx.` If you have changed the location of `ContentWS.asmx` you need to update the WSDLpath. For example, if your site is `www.myektronsite.com`, the WSDLpath would be `http://www.myektronsite.com/WorkArea/WebServices/ContentWS.asmx.`

To customize Ektron CMS400.NET, you need to be familiar with these tags.

Function	Description	More Information
Calendar Functions		
Calendar	Displays a calendar with different display types. For example, a daily view calendar.	"Calendar Function" on page 448
Event Calendar	Displays a month view event calendar	"Event Calendar Function" on page 445
Collection Functions		
Collection	Displays links generated by collections	"Collection Function" on page 430

Function	Description	More Information
CollectionXML	Creates XML for a collection by wrapping tags around all of the collection items and their properties. This allows you to use an XSLT to define how the XML collection is displayed.	”XML Collection Function” on page 438
Random Content	Randomly displays the content of a collection item	”Random Content” on page 443
Random Summary	Randomly displays the summary of a collection item	”Random Summary” on page 444
RSS Collection	Creates an RSS feed out of the items in a collection.	”RSS Collection Function” on page 436
Content Functions		
Add Content	Allows a user to add content to **Ektron CMS400.NET** progammatically.	”Add Content” on page 385
Content Block	Displays and manages a content block	”Content Block Function” on page 375
Form	Displays a content block associated with a form	”Form Function” on page 452
Multipurpose	Displays either a content block or a form	”MultiPurpose Function” on page 455
List Summary Functions		
Archive ListSummary	Displays summaries of archived content blocks	”Archive List Summary Function” on page 419
List Summary	Displays a list of content block summaries	”List Summary Function” on page 406
List Summary XML	Creates XML for a list summary by wrapping tags around all of the collection items and their properties. This allows you to use an XSLT to define how the XML collection is displayed.	”XML List Summary Function” on page 414
RSS List Summary	Creates an RSS feed out of the items in a list summary.	”RSS Summary Function” on page 424

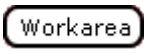
Function	Description	More Information
Single Summary	Displays the summary for a content block	”Single Summary Function” on page 420
Login Functions		
Login	Adds the login or logout button	”Login Function” on page 373
Menu Functions		
DropHTMLMenu	Used to display a DHTML menu	”DHTML Menus” on page 465
GenerateMenu	Works with DropHTML menu to display DHTML menus	”DHTML Menus” on page 465
XML Menu	Creates XML for a menu by wrapping tags around all of the menu items properties.	”XML Menu” on page 470
Metadata Functions		
Metadata	Displays the metadata for the current Web page	”Metadata Function” on page 456
Search Functions		
Search	Inserts a text box with a search button	”Search Functions” on page 389
Search Display	Creates an area to display search results	”Search Display Function” on page 400
Site Language Functions		
ShowSiteLanguage	Lets a site visitor select a language in which to view site.	”Show Site Language Function” on page 464

Login Function

The login function adds a login button on the template when displayed in a browser. Here it the format of the login function.

```
<% ecmLogin %>
```

This tag is responsible for displaying the following buttons on the Web page.

Button	Name	Description
	Login	When user is not logged into Ektron CMS400.NET, this button appears. Click this button to let the user log in to the application.
	Logout	Once logged in, this button replaces the login button to allow the user to log out from the Ektron CMS400.NET Web site.
	Workarea	When logged in, this button appears under the logout button allowing the user to access their workarea.
	Preview On	Lets the user to preview the entire Web site as if all checked-in content was published.
	Preview Off	Turns off preview mode.

Login Tag Example

The following example places a login button in a table cell inside an asp template.

```
<tr>
    <td>
        <% ecmLogin %>
    </td>
</tr>
```

When inserted properly, a login button appears on the Web page.

Clicking the login button opens a login window in which a user enters the username and password. Upon successful completion, the user is logged in to the Ektron CMS400.NET Web site.

Placing Your Login Button

You can add as many login buttons to a template as you like. You might want to include a login button on each template, or have a special Web page called login.asp, from which users can log in without the public being able to access the login page.

Content Block Function

The content block tag displays a content block on an Ektron CMS400.NET Web page. There are two functions to choose from:

- ecmContentBlock
- ecmContentBlockEX

ecmContentBlock allows you display a content block without having to define an XSLT.

With ecmContentBlockEX, you can display XML content by defining an XSLT or applying one of the predefined XSLTs located in **Ektron CMS400.NET**. For more information on predefined XSLTs, see the **Administrator's Manual** > "Managing XML" > "Adding an XML Configuration" > "Adding an XML Configuration Using External XML Files".

Here is the format for the ecmContentBlock function.

```
<%ecmContentBlockEX     ID          %>
```

Here is the format for the ecmContentBlockEX function.

```
<%ecmContentBlockEX     ID,
                        "XSLT",
                        Override    %>
```

NOTE As an alternative, you can use the multipurpose function, which can display either a content block or a content block associated with a form. For more information, see "MultiPurpose Function" on page 455.

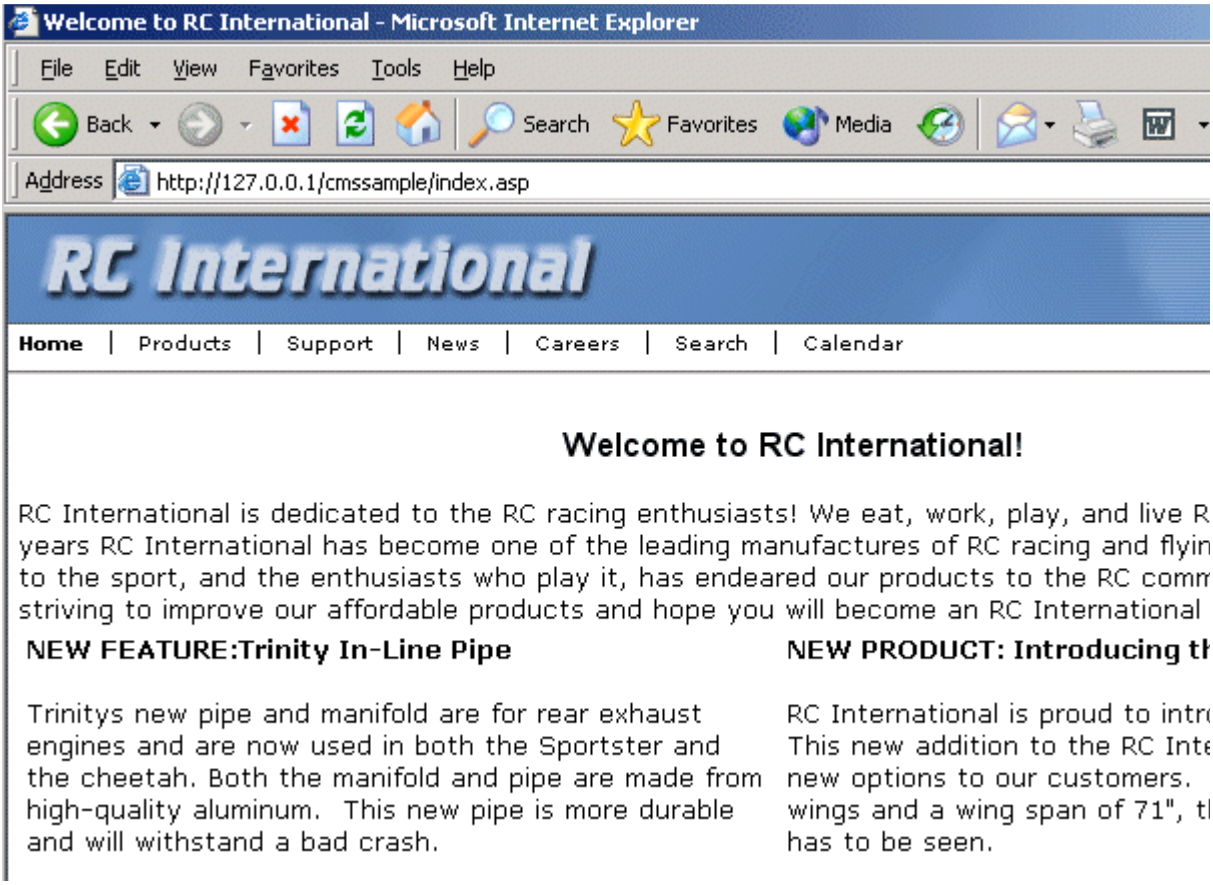
The table below explains the attributes of the ecmContentBlock function.

Attribute	Description	Required
ID	ID number assigned of the content block that will appear	Yes

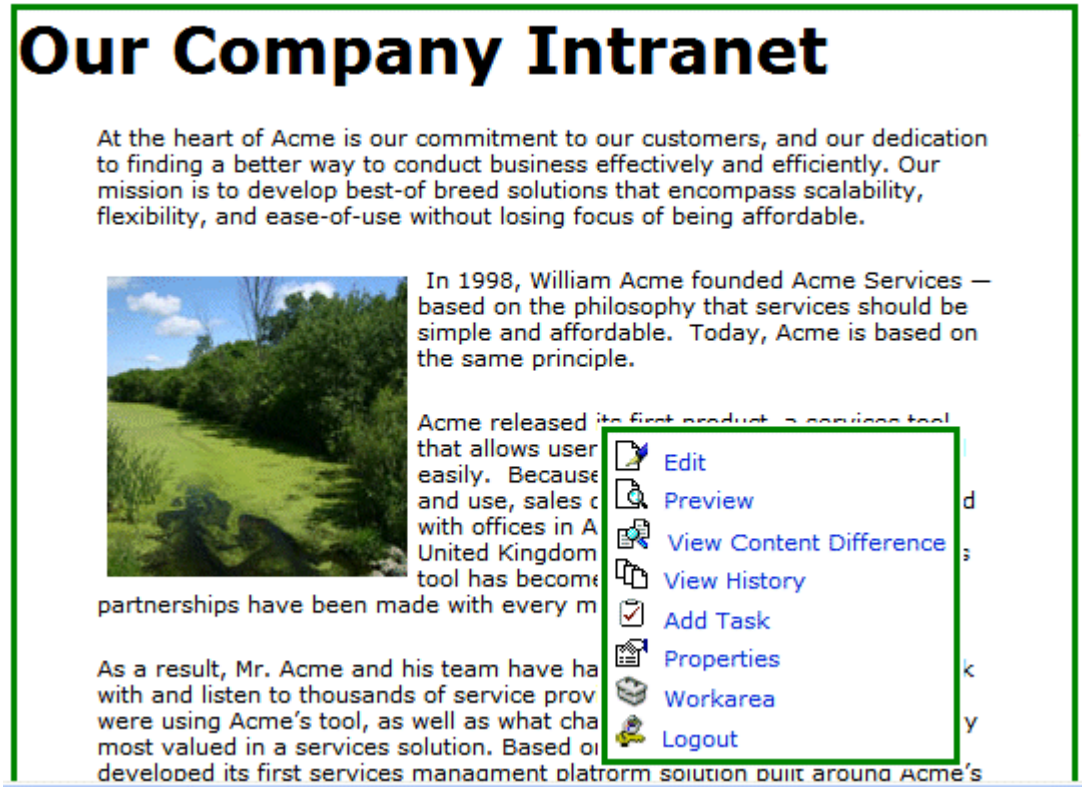
The table below explains the attributes of the ecmContentBlockEX function.

Attribute	Description	Required
ID	ID number assigned of the content block that will appear	Yes
XSLT	File name and path to a valid XSLT to transform the XML content. *See Also:* "The Function's Arguments" on page 382	No
Override	If no XSLT is specified, then enter 1 to use XSLT1, 2 for XSLT2, or 3 for XSLT3. *See Also:* "The Function's Arguments" on page 382	No

In normal view, these tags retrieve the content block from the database and displays it in the template.



If a user is logged into Ektron CMS400.NET, the content block tag displays a border around the content block and when they right click on the content, a menu is provided for managing it.



Content Block Examples

Ektron CMS400.NET has two types of content blocks.

Type	Displays	More information
Static	One content block on the Web page	"Static Content Block" on page 379
Dynamic	A content block whose ID is passed through the URL	"Dynamic Content Block" on page 380

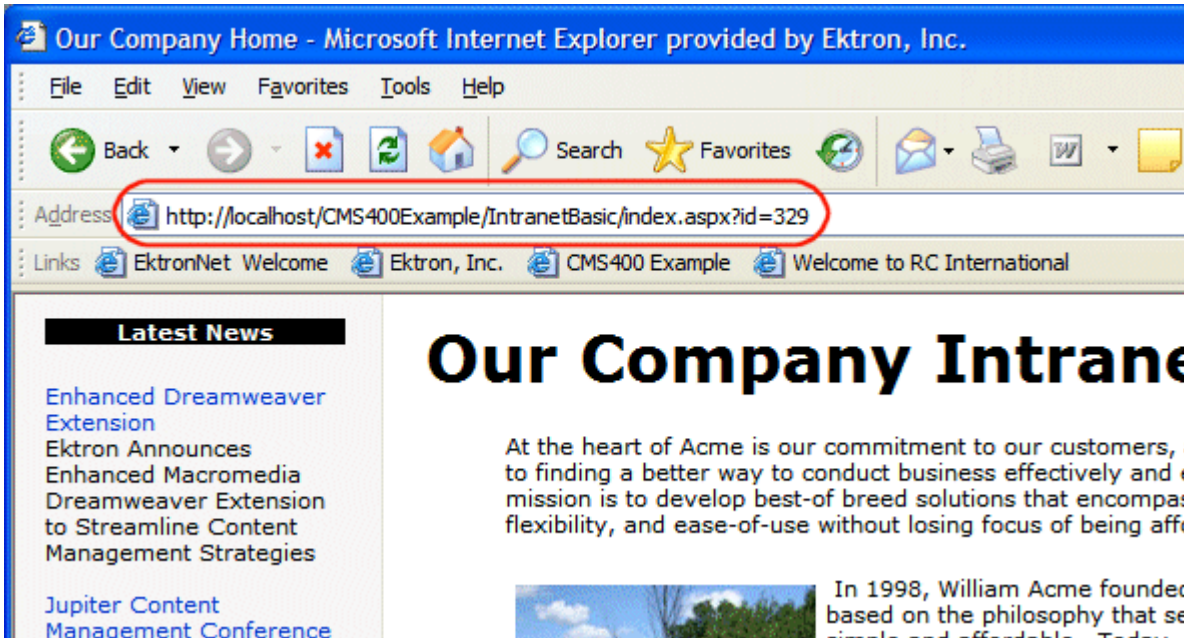
In addition, a content block can contain XHTML or XML content. For more information, see "XML Content Block" on page 381.

Static Content Block

The following illustrates how to insert a static content block on a template.

```
<tr>
    <td>
        <% ecmContentBlock(329) %>
    </td>
</tr>
```

In the example above, the function retrieves the content block with an id of 329 from the database to the browser.

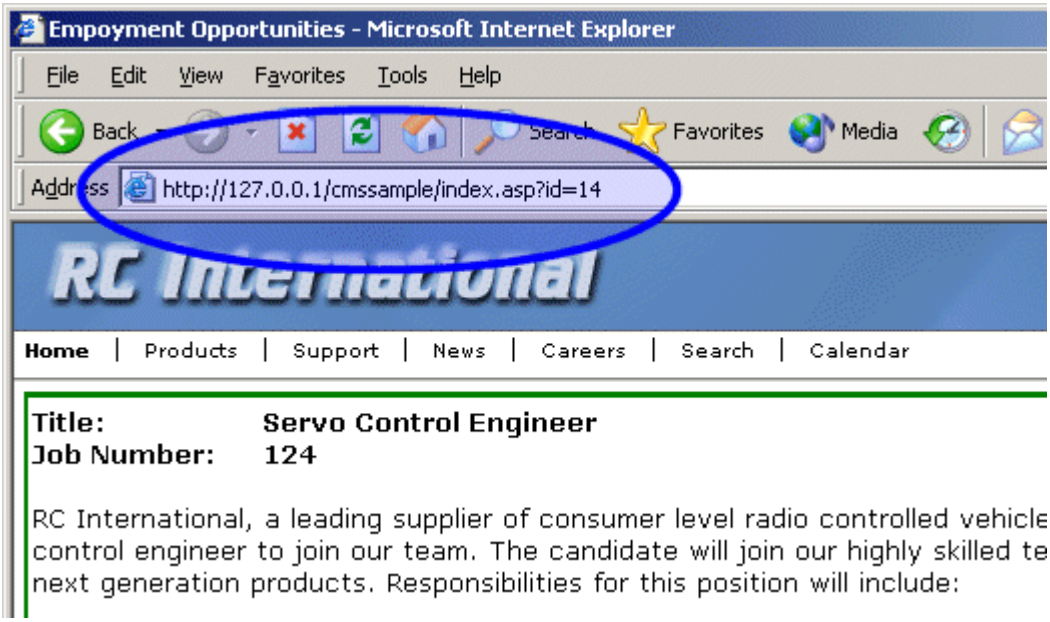


Dynamic Content Block

The following example shows the content block tag where the id is being passed as a URL parameter. Use this format with dynamic template.

```
<tr>
    <td>
        <% ecmContentBlock(request.QueryString("id")) %>
    </td>
</tr>
```

When a user accesses this page, it adds the content block ID number to the tag's template, and the content block with the corresponding ID appears on the Web page in the specified template.



The `id=` convention is used by **Ektron CMS400.NET** to generate quicklinks when content blocks are created. (For more information, see the **Ektron CMS400.NET** Administrator manual section

“Managing Content” > “Forming URLs For Ektron CMS400.NET Documents.”)

The dynamic `ecmContentBlock` function is almost the same as the static one (see ”Static Content Block” on page 379). The only difference is the parameter, which makes the function dynamic. The parameter uses a query string call (`Request.QueryString`) to read the URL from the browser’s address bar. It treats everything after the question mark as a list of key/value pairs separated by ampersands. So, `Request.QueryString` passes the key as a parameter and returns its associated value. For example, in URL `<http://localhost/CMS400Example/index.asp?id=1&LangType=1036>`, the following QueryString calls return these values.

QueryString Call	Return value
Request.QueryString(“id”)	1
Request.QueryString(“LangType”)	1036

XML Content Block

Here is an example of displaying an XML content block. Note that you specify a content block and an XSLT, which determines how the content block is displayed.

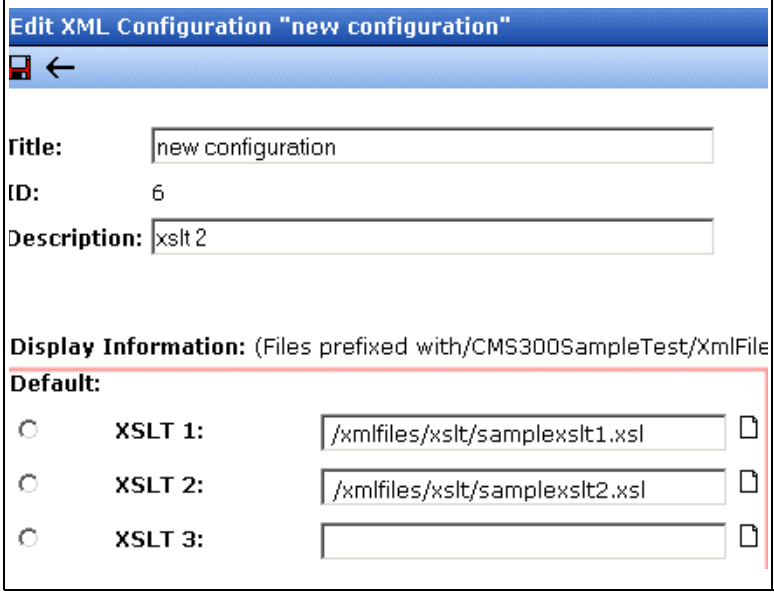
```
<tr>
    <td>
        <% ecmContentBlockEx 13, “”, 1 %>
    </td>
</tr>
```

This example displays the content block ID=13, using XSLT1 as the display XSLT.

The Function's Arguments

The ecmContentBlockEx takes three arguments. The first argument identifies the content block to display. The second one is optional and, if used, specifies an external XSLT file.

If the second argument does not exist, the third argument specifies an XSLT identified in the Edit XML Configuration screen. The following table provides additional detail about the second and third arguments.

To use this display XSLT	Enter this for argument 2	Enter this for argument 3
The default XSLT specified in the Edit XML configuration screen (illustrated below). Note that, in this example, the **XSLT Packaged** option is the default XSLT since it is selected.  	""	none

To use this display XSLT	Enter this for argument 2	Enter this for argument 3
XSLT Packaged: from the Edit XML configuration screen (that is, the XSLT created in the Data Designer) Note: If an XSLT package has not been created, and zero (0) is the third argument, the default XSLT specified in the Edit XML configuration screen is used.	“”	0
XSLT 1 from the Edit XML configuration screen	“”	1
XSLT 2 from the Edit XML configuration screen	“”	2
XSLT 3 from the Edit XML configuration screen	“”	3
An absolute or relative path to an XSLT	An external XSLT file not specified in the Edit XML Configuration screen. For example `<% ecmContentBlockEx 13, "samplexslt.xsl"%>`	does not matter - if a value exists in argument 2, argument 3 is ignored

See Also: The “Managing XML” chapter of the **Ektron CMS400.NET** Administrator manual

Dynamic and Static Combination

Here is an example of how you can use the two different content tags in the same table cell by using an IF statement

```
<tr>

    <td>

        <% if request.QueryString("id")<> "" then

            ecmContentBlock(request.QueryString("id"))else

            ecmContentBlock(14) end if %>

    </td>

<tr>
```

In this sample code, when a person accesses this page without passing through the content's ID number, the content with an ID=14 will be displayed. If you pass a content id through the URL like **index.asp?id=4,** the content block ID=4 will be displayed.

Add Content

The add content function allows you to programmatically add a new content block to Ektron CMS400.NET without the need of logging in.

This function can be for several applications including, but not limited to:

- Applied to a Web form, allowing you to collect data from users about the content block
- Creating a loop function that moves information from a database to an Ektron CMS400.NET database

Here is the format for the add content function:

```
<% AddContent    Username,
                 Password,
                 Domain,
                 content_title,
                 content_comment,
                 ContentHtml,
                 SummaryHTML,
                 ContentLanguage,
                 FolderID,
                 GoLive,
                 EndDate,
                 MetaInfo          %>
```

IMPORTANT! Every string that you pass to the function must be HTML encoded. Here is the VB.Net syntax for encoding a string.

```
myTitle = “John’s Book”
myTitle = Server.HtmlEncode(myTitle)
```

Explained in the table below are the attributes for the add content block function.

Attribute	Description
Username	The username of the user to be authenticated for permissions to see whether the user is able to add a content block.
Password	The password for the username being authenticated.
Domain	If using Active Directory, specify the domain needed to authenticate the user.
Content_title	Specify a title for the content block being added.
Content_comment	Specify a historical comment for the content being added.
ContentHTML	Actual content for the content block.
SummaryHTML	Summary information about the content block.
ContentLanguage	Language of content.
FolderID	ID number of the content folder where the content block will be added to.
GoLive	Date and time the content will go live. Must match date/time format set in setup.asp.
EndDate	Date and time the content will expire. Must match date/time format set in setup.asp.
MetaInfo	Specify the information for the metadata which will be used as a reference point for finding other words or information.

Add Content Example

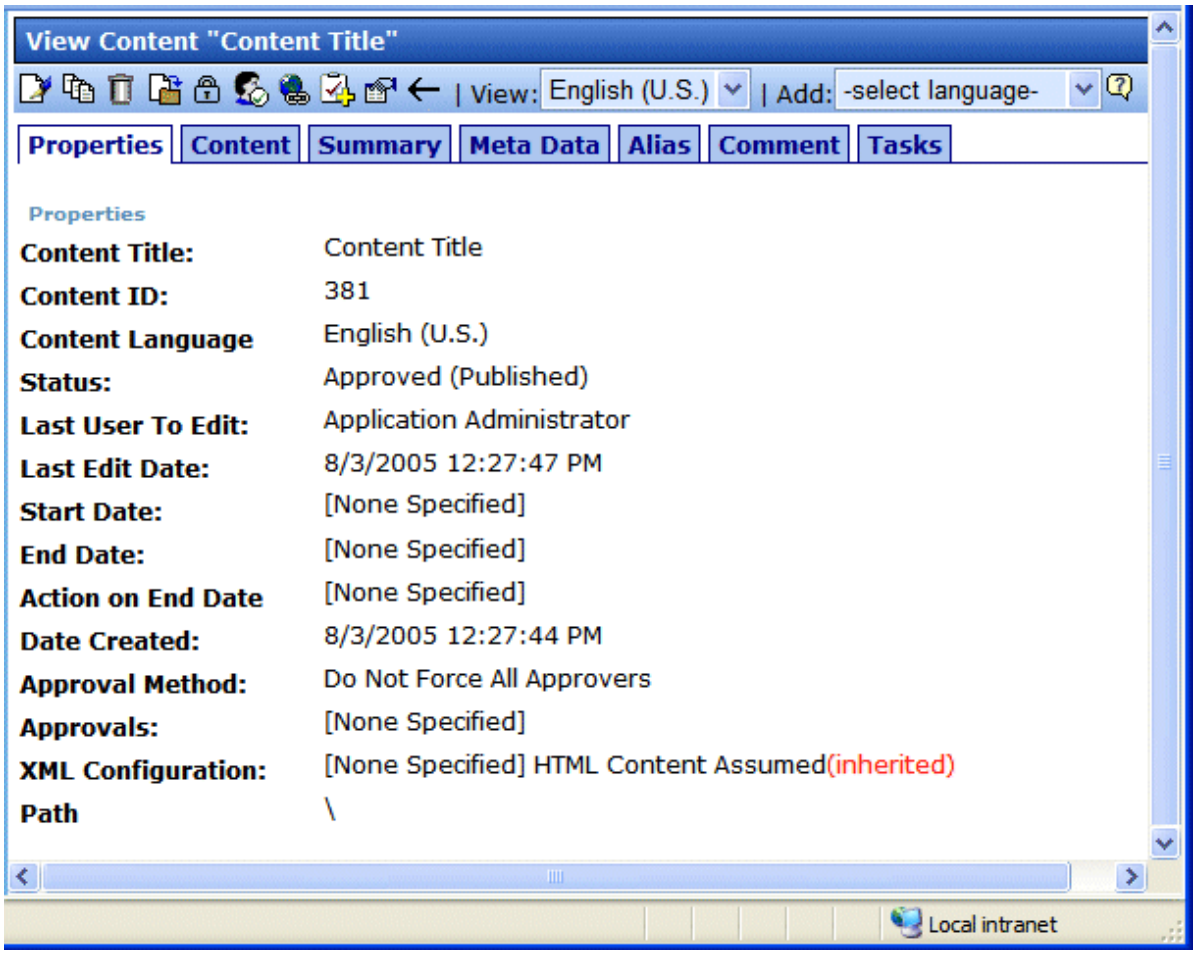
The following is an example of the add content function in a template:.

```
<tr>

    <td>

        <% AddContent "admin", "admin", "", "Content Title", "Comment",

            "This is the <b>body</b> of the content.",

            "This is the <i>summary</i> for the content.", 1033, 0, "", "", "" %>

    </td>

</tr>
```

The previous example has the following properties:

- The user “admin” is used to be authenticated.
- The title of the new content block is “Control Title”.
- The comment for the content is “Comment”.
- The body of the content is “This is the **body** of the content”.
- The summary of the content block is “This is the *summary* for the content.
- The language ID = 1033, or English.
- The folder ID = 0 is where the content is added.
- No start date is specified.
- No end date is specified.
- No metadata is specified.

When this function is executed, the content is added to Ektron CMS400.NET. Displayed here is the record of the content in the workarea.



Search Functions

The search tag is used to display a text box, with a search button on the template. When seen on the template of a Web page, users can enter text and click on the search button to search through all the HTML content on the Web site. To create a search, Ektron has created two search functions:

- ecmSearch
- ecmSearchNoFormTag

Both functions have the same parameters and perform the same search. The only difference is ecmSearch automatically inserts form tags, ecmSearchNoFormTag does not.

The following is an example of form tags.

```
<form name="ecmsearch" method="post" action='search-
display.asp'>

</form>
```

When using ecmSearch, form tags are added automatically. When using ecmSearchNoFormTag you need to add the tags manually

When using the search function, ecmSearch and ecmSearchNoFormTag can be interchanged. Shown here is the format for the ecmSearch and ecmSearchNoFormTag:

```
<% ecmSearch
   or
   ecmSearchNoFormTag        "Starting Folder",
                             Recursive,
                             "Target Page",
                             Text Box Size,
                             Maximum Characters,
                             "Button Image Source",
                             "Button Text",
                             "Font Face",
                             "Font Color",
                             "Font Size",
                             Horizontal,
                             "Spare"                %>
```

The following table provides information about all the attributes for the search tags.

Attribute	Description	Required ?	For more information, see
Starting Folder	The folder you wish to search.	Yes	”Starting Folder” on page 393
Recursive	Whether to search subfolders. 0=No, 1 = Yes	Yes	”Recursive” on page 393
Target Page	The Web page to output the search results.	Yes	”Target Page” on page 394
Text Box	The size of the text box for user input.	Yes	”Text Box” on page 394

Attribute	Description	Required ?	For more information, see
Maximum Characters	The maximum number of characters that the input text box can accept.	No	”Maximum Characters” on page 395
Button Image Source	The image location for the search button. Blank (“”) = standard submit button.	No	”Button Image Source” on page 395
Button Text	The text used for the button if the standard submit button is chosen, or alternative text for the button if an image is chosen.	No	”Button Text” on page 396
Font Face	The font of the search tag text.	No	”Font Face” on page 397
Font Color	The font color of the search tag text. Blank (“”) = Black.	No	
Font Size	The font size of the search tag text.	No	”Font Size” on page 398
Horizontal	Determines if the search options are in a row or a column. 1 = horizontal, 0 = vertical	Yes	”Horizontal” on page 399
Spare1	Currently not used.	Must be “”	”Spare 1” on page 399

Search Tag Example

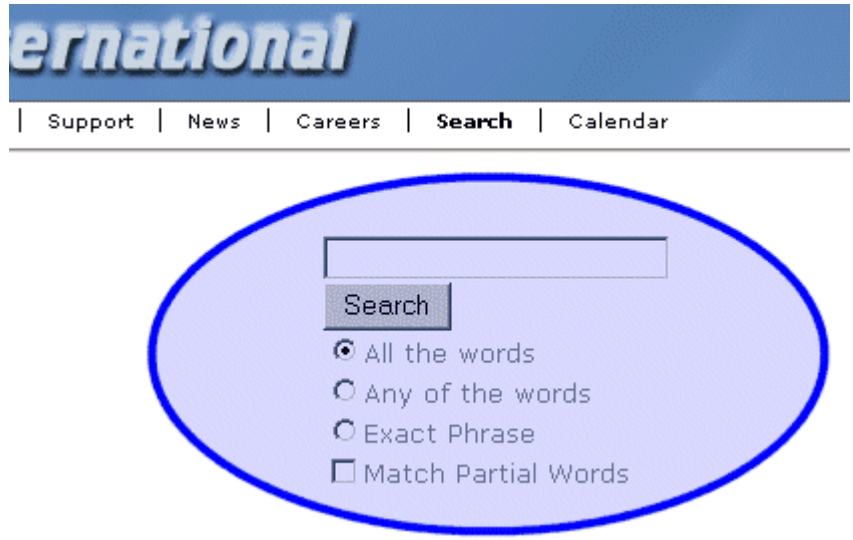
The following example places a text field and a search button in a table cell inside an ASP template.

```
<tr>

    <td>

        <% ecmSearch "\",1,"search-

        display.asp",25,200,"","Search","Verdana",

        "#808080","2",0,"" %>

    </td>

</tr>
```

The previous example has the following properties:

- Searches the root folder, in this case the `\content` folder.
- Search is recursive.
- The target page is “search-display.asp”.
- The size of the text box is 25.
- The search text box allows up to 200 characters.
- No image is used for a button. Instead, a standard submit button is used.
- The text on the submit button is “Search”.
- The search options appear in Verdana with a color of #808080 and a size of 2pt.
- The search options are arranged vertically.

When added to a template, and visited, the search tag will appear similar to the following:



Search Tag Attributes

The search tag's attributes are described below.

Starting Folder

In this attribute, specify the content folder where the search begins. There are two options for this attribute.

Value	Search begins in
"\"	root folder
"\folder path"	specified folder

Recursive

When you setup the search to be recursive, you are allowing the search to search all the sub-folders associated with the starting folder you had specified. You have two options for this attribute:

Value	Search begins in
1	Recursive search (will search sub-folders of the starting folder).
0	Non-Recursive search (will only search starting folder).

Target Page

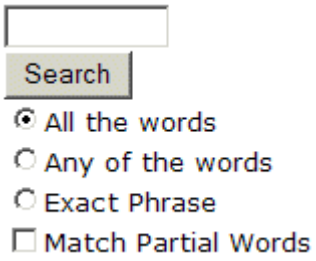
The page that displays the search results. You may enter any form of a URL for this page.

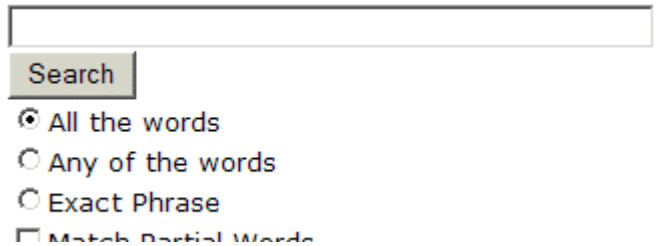
NOTE The target page *must* include the searchdisplay function to display search results.

To use the same page, enter a pound sign (#) into the field.

Text Box

This attribute sets the size of the search text box. It can be any number, depending on the size of the search text box that you want.

Text Box Size	Web Page View
10	

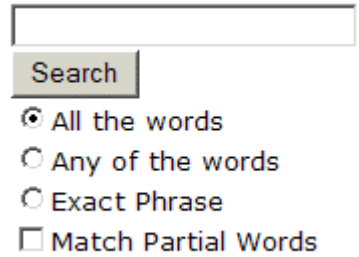
Text Box Size	Web Page View
50	

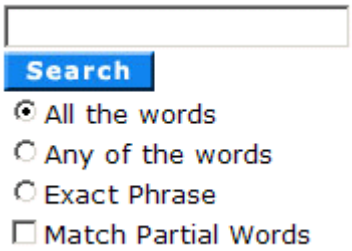
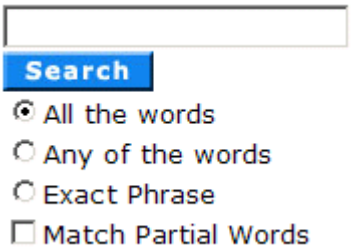
Maximum Characters

This attribute determines the maximum number of characters that a user can enter for search text.

Button Image Source

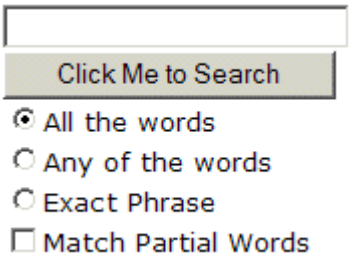
To use an image for the search button, enter the image's filename and path.

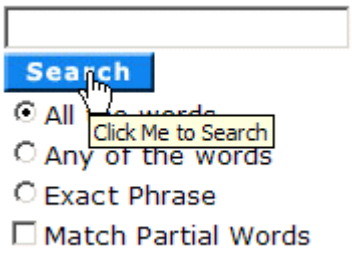
Button Image Source	Web Page View
“”	

Button Image Source	Web Page View
“search.jpg”	
“http://www.ektron.com/images/search.jpg”	

Button Text

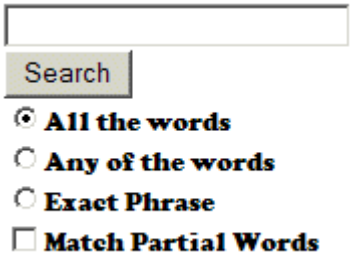
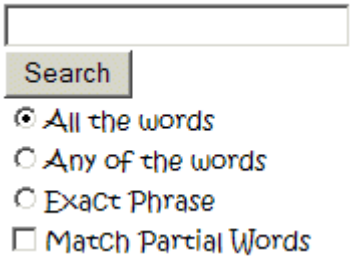
If no search button image is specified, the text in this field appears on the button. If an image is being used for the search button, this text is the image’s alt text.

Button Text	Web Page View
“Click Me to Search”	

Button Text	Web Page View
"Click Me to Search"	

Font Face

This font is used for the search options displayed on the Web page.

Font Face	Web Page View
Cooper Black	
Kristen ITC	

Font Color

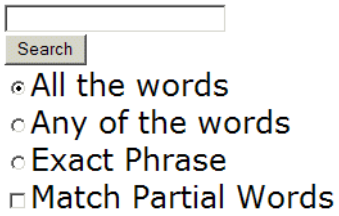
The color of the text in the search tag. You may enter the color name or a hexidecimal value of the color.

Font Color	Web Page View
Red	Search All the words Any of the words Exact Phrase Match Partial Words
#14DA14	Search All the words Any of the words Exact Phrase Match Partial Words

Font Size

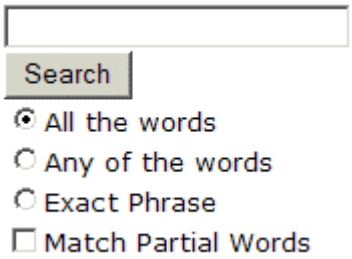
This is the size of the text on the search tag.

Font Size	Web Page View
1	Search All the words Any of the words Exact Phrase Match Partial Words

Font Size	Web Page View
5	

Horizontal

This attribute determines whether the search options are displayed vertically or horizontally.

Attribute	Web Page View
0 (vertical)	
1 (horizontal)	

Spare 1

At the end of the Search tag is a attribute called Spare 1. This attribute has not yet been assigned an function. As of now, when you enter a search tag, you must pass this attribute “”.

Search Display Function

The search display custom tag creates the area where search results appear after a search is executed.

Here is the format of the ecmSearchDisplay tag.

```
<% ecmSearchDisplay      "MaxNumberReturn",
                         "StyleInfo",
                         ShowDate                %>
```

Below is a description of the tag's attributes.

Attribute	Description	Required	For more information, see
Max Number of content blocks Returned	Maximum number of content blocks returned. (0=Unlimited)	Yes	"Max Number of Content Blocks Returned" on page 403
Style Info	An HTML style string used within the HTML "hyperlink" generated by the search output. Can control font, color, size, etc.	No	"Style Info" on page 404
ShowDate	Displays the last modified date of the content block. 0=No 1=Yes	No	"Show Date" on page 404

Search Display Example

Insert the following on the page that displays search results. You may place it on the same page as the search tag or a separate page.

```
<tr>
    <td>
        <% ecmSearchDisplay 0,"","1" %>
    </td>
</tr>
```

This search display example has the following properties:

- Returns unlimited results when the search is performed.
- Contains no style information.
- Shows the last modified date of the content.

When the tag is added to a template, and a search is performed, the results appear similar to the following.

Search Display with Default Content

You can also set up the search display tag to display a default content block if a search is not performed. Then, if a search is

performed, the results replace the content block. The sample code below does that:

```
<tr>

    <% if (Request.Form("ecmsearchtype") <> "") then %>

        <td>

            <% ecmSearchDisplay 0,"" %>

        </td>

    <% else %>

        <td>

            <% ecmContentBlock(1) %>

        </td>

    <% end if %>

</tr>
```

Search Display Attributes

Listed below are the attributes for the SearchDisplay tag.

Max Number of Content Blocks Returned

This attribute determines the number of search results displayed on a page. You may enter any integer. To display unlimited results, enter 0 (zero).

Max Returns	Web Page View
0	**Search Results** RC Sportster RC Redstar RC Lilly RC Cheetah Products Page RC International Ships RC Sportster RC International Hosts RC Conference RC International Announces the RC Cheetah
3	**Search Results** RC Sportster RC Redstar RC Lilly

Style Info

An HTML style string used within the HTML “hyperlink” generated by the search output. Can control font, color, size, etc.

Style Info	Web Page View
“”	**Search Results** RC Sportster RC Redstar RC Lilly RC Cheetah Products Page RC International Ships RC Sportster RC International Hosts RC Conference RC International Announces the RC Cheetah
"font-family:arial; font-weight:bold; background-color:#cccccc; border:solid blue 1pt; margin:2px; width:50%;"	**Search Results** RC Sportster RC Redstar RC Lilly RC Cheetah Products Page RC International Ships RC Sportster RC International Hosts RC Conference RC International Announces the RC Cheetah

Show Date

This attribute determines if the content block’s last modified date appears next to the title. This is useful when determining which content block is the most recent. You have two choices.

Attribute	Web Page View
0	**Search Results** RC Sportster RC Redstar RC Lilly RC Cheetah Products Page RC International Ships RC Sportster RC International Hosts RC Conference RC International Announces the RC Cheetah
1	**Search Results** RC Sportster *(8/27/2002 7:52:40 PM)* RC Redstar *(8/27/2002 7:52:01 PM)* RC Lilly *(8/27/2002 7:51:26 PM)* RC Cheetah *(8/27/2002 7:50:53 PM)* Products Page *(8/27/2002 7:50:09 PM)* RC International Ships RC Sportster *(8/27/2002 2:46:12 PM)* RC International Hosts RC Conference *(8/27/2002 2:44:44 PM)* RC International Announces the RC Cheetah *(8/27/2002 2:43:01 PM)*

List Summary Function

The list summary function is used to display the list of content summaries on a Web page.

Shown here is the format of the list summary function:

```
<% ecmListSummary    "Folder",
                     Recursive,
                     ShowSummary,
                     "StyleInfo",
                     "OrderBy, Asc/Desc",
                     "ShowInfo",
                     MaxNumber,
                     "ObjType",
                     "OptionList"                %>
```

A list and description of the attributes are shown in this table:

Attribute	Description	Options
Folder	This is the folder the summary information will come from.	Any folder
Recursive	Allow the tag to get summary info from the child folders.	0 = Not Recursive 1 = Recursive
ShowSummary	Do you want the content summary to be displayed?	0 = No 1 = Yes
StyleInfo	Inline style information for the generated text hyperlink.	Any string

Attribute	Description	Options
OrderBy, ASC/DESC	Order the hyperlinks by:	“Title” “DateModified” “DateCreated” “Startdate” “AuthorFname” “AuthorLname”
	Order the hyperlinks in ascending or descending order.	“Title,asc” “Title,desc”
ShowInfo	Information that will appear to the right of the hyperlink.	“DateModified” “DateCreated” “AuthorFname” “AuthorLname”
MaxNumber	Maximum number of summaries returned.	0 = Unlimited
ObjType	Determines whether the list summary is for the folder or a single piece of content. For example: If the folder property = \books and “Folder” is chosen as the ObjType, the list summary for the folder with books appears. If the folder property = 1 and you choose “Content” as the ObjType, a single summary for content with an id = 1 appears. *See Also:* ”Single Summary Function” on page 420 The default ObjType is “Folder” when left blank.	“““ “Folder“ “Content”
OptionList	Not yet implemented	MUST BE ““

List Summary Example

Shown below is an example of the ListSummary function in an ASP Web page.

```
<TR>
    <TD>
        <% ecmListSummary "\Marketing\News",0,1,"",
            "Title,asc","DateModified",0,"","" %>
    </TD>
</TR>
```

The previous example has the following properties:

- Displays summaries from the \Marketing\News folder.
- Not recursive.
- The summaries will be shown.
- No style was applied to the hyperlinks.
- The results will be displayed in ascending order by title.
- The date modified of each content block will be displayed.
- Unlimited results will be shown.

Here is how the summaries will appear on the web page.

Latest News

RC International Announces the RC Cheetah *(8/27/2002 2:43:01 PM)*
RC International, a leading RC racing company, has announced the availability of its new flagship product, the RC Cheetah.

RC International Hosts RC Conference *(8/27/2002 2:44:44 PM)*
RC International, the fastest growing company in the RC industry, announces plans to host the 3rd annual RC conference in Atlanta, Georgia.

List Summary Attributes

Listed below are all the different attributes and descriptions about them.

Starting Folder

In this attribute, you will need to specify which content folder you will want to get all the summaries from. There are two options for this attribute:

Value	Description
“”	By leaving the value blank, all the summaries from the root content folder will be displayed.
“\Folder Path	You can specify a specific folder to display the summaries from by inserting the folder path

Recursive

When you setup the search to be recursive, you are allowing the search to search all the subfolders associated with the starting folder you had specified. You have two options for this attribute:

Value	Description
1	Recursive - summaries in the starting folder’s subfolders will also be displayed.
0	Non-recursive - only summaries in the starting folder will be displayed.

Show Summary

The show summary attribute displays or suppresses the summary in the summary list. You have two options for this attribute.

Attribute	Web Page View
0	RC International Ships RC Sportster RC International Hosts RC Conference RC International Announces the RC Cheet
1	RC International Announces the RC Cheetah *(8/27/2002 2:43:01 PM)* RC International, a leading RC racing company, has announced the availability of its new flagship product, the RC Cheetah. RC International Hosts RC Conference *(8/27/2002 2:44:44 PM)* RC International, the fastest growing company in the RC industry, announces plans to host the 3rd annual RC conference in Atlanta, Georgia.

Style Info

In this field, you may enter inline style information for the generated text hyperlinks displayed on the screen.

Style Info	Web Page View
“”	RC International Announces the RC Cheetah *(8/27/2002 2:43:01* RC International, a leading RC racing company, has announced th availability of its new flagship product, the RC Cheetah. RC International Hosts RC Conference *(8/27/2002 2:44:44 PM)* RC International, the fastest growing company in the RC industry, announces plans to host the 3rd annual RC conference Atlanta, Georgia.

Style Info	Web Page View
"font-family:arial; font-weight:bold; background-color:#cccccc; border:solid blue 1pt; margin:2px; width:100%;"	**RC International Announces the RC Cheetah** RC International, a leading RC racing company, has announced tl availability of its new flagship product, the RC Cheetah. **RC International Hosts RC Conference** RC International, the fastest growing company in the RC industry, announces plans to host the 3rd annual RC conference Atlanta, Georgia.

Order By

This attribute will sort the summary results by different options. There are five different ways to sort the summaries:

Value	Summaries sorted by
Title	title
DateModified	date modified
Date Created	date created
Startdate	The Start Date assigned to the content block when created.
AuthorFname	author's first names
AuthorLname	author's last name

Ascending/Descending

This attribute is an addition to the previous OrderBy attribute. By default, the list of the summaries are descending, meaning they will follow the path of A-B-C, or 1-2-3. When set to ascending, they will order by Z-Y-X or 3-2-1.You will have the option to re arrange these to display either in ascending or descending order.

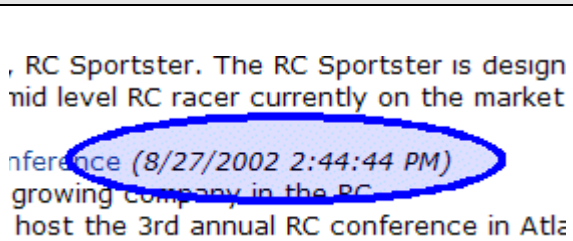
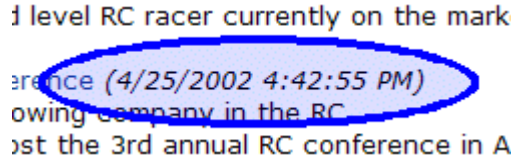
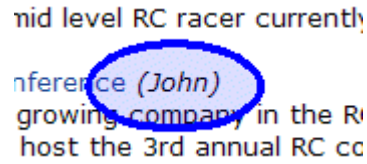
To do this, add the following to the OrderBy attribute:

Value	Description
asc, 1	This will order the Summaries in an ascending order
desc, 0	This will order the Summaries in a descending order. This is the default setting.

By using the ascending/descending option, you will have the option to set the order of content blocks from most recent to oldest, or vice versa. As well as ordering the list by the user's first names or last names in ascending or descending order.

Show Info

The show info field determines what sort of information is displayed to the right of the hyperlinked title. There are four different options that you may choose from:

Value	Description	Web Page View
DateModified	Displays the date the content was last modified.	
***DateCreated	Displays the date when the content was first created.	
AuthorFname	Displays the first name of the user who last edited the content block.	

Value	Description	Web Page View
AuthorLname	Displays the last name of the user who last edited the content block.	ct, RC Sportster. The RC Sp d mid level RC racer currentl Conference *(Doe)* st growing company in the R to host the 3rd annual RC c

Max Number

This attribute determines how many summaries appear on a page. Enter the number of summaries to be displayed. To display all summaries, enter 0 (zero) and you will display unlimited results.

XML List Summary Function

The XML List Summary function wraps XML around the properties of a list of content that's created as a list summary in the Ektron CMS400.NET Workarea. Once created, you can use the XML to display the list summary information. The ecmListSummaryXML function allows you to easily define how the list summary data appears on the Web page.

Shown here is the format of the ecmListSummaryXML function.

```
<% ecmListSummaryXML  "Folder",
                      "Recursive",
                      "OrderBy",
                      "MaxNumber",
                      "ObjType",
                      "SummaryType",      %>
```

The ecmListSummaryXml function attributes are explained below.

Attribute	Description	Options
Folder	This is the folder the summary information will come from. *See Also:* "Starting Folder" on page 409	Any folder
Recursive	Allow the tag to get summary info from the child folders. *See Also:* "Recursive" on page 409	0 = Not Recursive 1 = Recursive
OrderBy, ASC/ DESC	Order the hyperlinks by: *See Also:* "Order By" on page 411	"Title" "DateModified" "DateCreated" "Startdate" "AuthorFname" "AuthorLname"
	Order the hyperlinks in ascending or descending order.	"Title,asc" "Title,desc"

Attribute	Description	Options
MaxNumber	Maximum number of summaries returned. *See Also:* "Max Number" on page 413	**0** (Zero) = Unlimited
ObjType	Determines whether the list summary is for the folder or a single piece of content. For example: If the folder property = \books and "Folder" is chosen as the ObjType, the list summary for the folder with books appears. If the folder property = 1 and you choose "Content" as the ObjType, a single summary for content with an id = 1 appears. *See Also:* "Single Summary Function" on page 420 The default ObjType is "Folder" when left blank.	"" "Folder" "Content"

When using the XML list summary function, the following tags are created around the properties of each item in the list summary.

XML Tags	Description
<Content></Content>	Declares a new section of content in the XML collection.
<ID></ID>	The content block ID. For example: `<ID>27</ID>`
<Title></Title>	The title of the content block. For example: `<Title>HTML for the World Wide Web with XHTML and CSS</Title>`
<QuickLink></QuickLink>	The quicklink associated with the content block. For example: `<QuickLink>/CMS400Example/collection.aspx?id=27</QuickLink>`
<Teaser></Teaser>	The summary of the content block. For example: `<Teaser>The easiest HTML book weve seen that still manages to be comprehensive.</Teaser>`

XML Tags	Description
<StartDate></StartDate>	The content block's start date formatted as a .NET date type For example: `<StartDate>1/1/0001 12:00:00 AM</StartDate>`
<DateModified></DateModified>	The last date the content block was modified. For example: `<DateModified>1/19/2005 4:18:06 PM</DateModified>`
<EndDate></EndDate>	The date the content block stops running on the Web site For example: `<EndDate>12/31/9999 12:00:00 AM</EndDate>`
<LastEditorFname></LastEditorFname>	The first name of the last person who edited the content block. For example: `<LastEditorFname>John</LastEditorFname>`
<LastEditorLname></LastEditorLname>	The last name of the last person who edited the content block. For example: `<LastEditorLname>Johnson</LastEditorLname>`
<DisplayStartDate></DisplayStartDate>	The content block's start date. It is formatted as a string that represents Ektron CMS400's display of the date.
<FolderID></FolderID>	The ID of the folder in which the content block is contained. For example: `<FolderID>0</FolderID>`
<ContentStatus></ContentStatus>	The status of the content block.
<Language></Language>	The language of the content block. For example: <Language>1033</Language>
<DisplayDateModified></DisplayDateModified>	edited. It is formatted as a string that represents Ektron CMS400's display of the date. For example: `<DisplayDateModified>1/19/2005 4:18:06 PM</DisplayDateModified>`

XML Tags	Description
<DisplayEndDate></DisplayEndDate>	The content block's end date. It is formatted as a string that represents Ektron CMS400's display of the date. For example: `<DisplayEndDate>12/31/9999 12:00:00 AM</DisplayEndDate>`

Here is how the raw XML list summary output appears before an XSLT is applied.

```
<Content>
    <ID>24</ID>
    <Title>ASP.NET Unleashed</Title>
    <QuickLink>/CMS400Example/collection.aspx?id=24</QuickLink>
    <Teaser><p><em>ASP.NET Unleashed, Second Edition</em> is really big, really thorough, and
really <i>good</i>.</p></Teaser>
    <StartDate>1/1/0001 12:00:00 AM</StartDate>
    <DateModified>1/12/2005 7:53:31 PM</DateModified>
    <EndDate>12/31/9999 12:00:00 AM</EndDate>
    <LastEditorFname>Application</LastEditorFname>
    <LastEditorLname>Administrator</LastEditorLname>
    <DisplayStartDate></DisplayStartDate>
    <FolderID>0</FolderID>
    <ContentStatus></ContentStatus>
    <Language>0</Language>
    <DisplayDateModified>1/12/2005 7:53:31 PM</DisplayDateModified>
    <DisplayEndDate></DisplayEndDate>
</Content>
<Content>
    <ID>25</ID>
    <Title>Microsoft ASP.NET Programming with Microsoft Visual C#.NET</Title>
    <QuickLink>/CMS400Example/collection.aspx?id=25</QuickLink>
    <Teaser><p>Teach yourself how to write high-performance Web applications with ASP.NET and
Visual C# .NET--one step at a time.</p></Teaser>
    <StartDate>1/1/0001 12:00:00 AM</StartDate>
    <DateModified>1/4/2005 7:16:21 PM</DateModified>
    <EndDate>12/31/9999 12:00:00 AM</EndDate>
    <LastEditorFname>Application</LastEditorFname>
    <LastEditorLname>Administrator</LastEditorLname>
    <DisplayStartDate></DisplayStartDate>
    <FolderID>0</FolderID>
    <ContentStatus></ContentStatus>
    <Language>0</Language>
    <DisplayDateModified>1/4/2005 7:16:21 PM</DisplayDateModified>
    <DisplayEndDate></DisplayEndDate>
</Content>
```

Archive List Summary Function

The Archive list summary function displays on a Web page a list of content summaries for archived content blocks. For example, you might have a current news page and an archived news page. Use the List Summary Function to display current news, and this function to display older news stories.

On the Content Block **schedule** screen, the user must set an **End Date** then choose the Archive and Display option to have the content blocks appear via this function.

Shown here is the format of the Archive List Summary function:

```
<% ecmListArchiveSummary    "StartingFolder",
                            Recursive,
                            ShowSummary,
                            "StyleInfo",
                            "OrderBy, Asc/Desc",
                            "ShowInfo",
                            MaxNumber,
                            "Spare1",
                            "Spare2"                %>
```

This function has the same parameters as the List Summary Function. See ”List Summary Function” on page 406 for details.

Single Summary Function

The single summary tag is used to display the summary of one content block in the Ektron CMS400.NET Web site, instead of a list of all the summaries in a content folder. Seen below is the single summary tag:

```
<% ecmSingleSummary   "ContentID",
                      ShowSummary,
                      "StyleInfo",
                      "ShowInfo",
                      "Spare1"          %>
```

You can look at the single summary tag as just a simpler form of the List summary tag. The format for each of them is almost identical, but the single summary tag does not require as many attributes to be passed.

A list and description of the attributes are shown in this table:

Attribute	Description	Required?
Content ID	This is the ID number of the content block the summary information will come from.	Yes
ShowSummary	Do you want the content summary to be displayed?	Yes
StyleInfo	Inline style information for the generated text hyperlink.	No
ShowInfo	Information that will appear to the right of the hyperlink.	No
Spare1	Not yet implemented	MUST BE “”

Single Summary Example

Shown below is an example of the single summary function in an ASP Web page.

```
<TR>
    <TD>
        <%  ecmSingleSummary "14",1,"","DateModified","" %>
    </TD>
</TR>
```

The previous example had the following properties:

- Displays the summary for the content block with an ID=14
- The summary will be shown
- No style was applied to the hyperlinks
- The last date the content was modified will be displayed

Here is how the summary will appear on the web page.

Servo Control Engineer #124 *(10/29/2002 6:51:02 PM)*
RC International is looked for a highly skilled servo control engineer to join our team.

Single Summary Attributes

Listed below are all the different attributes and descriptions about them.

Content ID

In this attribute, you will need to specify which content block you will want the summary to be displayed from.

Show Summary

The show summary attribute will allow or disallow the summary to be displayed in the summary list. You have two options for this attribute:

Value	Web Page View
0	Servo Control Engineer #124
1	Servo Control Engineer #124 RC International is looked for a highly skilled servo control engineer to join our team.

Style Info

In this field, you may enter inline style information for the generated text hyperlinks displayed on the screen.

Style Info	Web Page View
“““	Servo Control Engineer #124 RC International is looked for a highly skilled servo control engineer to join our team.
font-family:arial; font-weight:bold; background-color:#cccccc; border:solid blue 1pt; margin:2px; width:100%;	Servo Control Engineer #124 RC International is looked for a highly skilled servo control engineer to join our team.

Show Info

The show info field determines what sort of information is displayed to the right of the hyperlinked title. There are five different options that you may choose from:

Value	Displays the...	Web Page View
ID	ID number for the content block.	#124 (14)
DateModified	date the content was last modified.	124 (8/27/2002 2:40:26 PM)
DateCreated	date when the content was first created.	124 (4/25/2002 4:54:42 PM)
AuthorFname	first name of the user who last edited the content block.	124 (Mike)
AuthorLname	last name of the user who last edited the content block	124 (Johnson)

Spare 1

There is one attribute fields at the end that are currently not used. This is here for future scalability. When you insert a single summary tag into your template, you must leave this attributes as “”

RSS Summary Function

The RSS summary function is used to create an XML list of content summaries formatted for RSS Syndication.

Shown here is the format of the RSS summary function:

```
<% ecmRssSummary    "Folder",
                    Recursive,
                    "OrderBy, Asc/Desc",
                    MaxNumber,                %>
```

A list and description of the attributes are shown in this table:

Attribute	Description	Options
Folder	This is the folder the summary information will come from.	Any folder
Recursive	Allow the tag to get summary info from the child folders.	0 = Not Recursive 1 = Recursive
OrderBy, ASC/DESC	Order the hyperlinks by:	“Title” “DateModified” “DateCreated” “Startdate” “AuthorFname” “AuthorLname”
	Order the hyperlinks in ascending or descending order.	“Title,asc” “Title,desc”
MaxNumber	Maximum number of summaries returned.	0 = Unlimited

RSS Summary Example

Shown below is an example of the ecmRssSummary function in an ASP Web page.

```
<TR>
    <TD>
        <% ecmRssSummary "\Books",0 "Title",0 %>
    </TD>
</TR>
```

The previous example has the following properties:

- Displays summaries from the \Books folder.
- Not recursive.
- The results will be displayed in ascending order by title.
- Unlimited results will be shown.

Here is how an RSS summary appears when viewed without a RSS reader on a web page.

```
<?xml version="1.0" ?>
<rss xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" version="2.0">
    <channel>
        <title>Books</title>
        <link>http://localhost/CMS400Example/WorkArea/WebServices/ContentWS.asmx</link>
        <description></description>
        <item>
            <title>Visual Basic.NET How to Program</title>
            <link>http://localhost/CMS400Example/collection.aspx?id=23</link>
            <description>Visual Basic .NET How to Program, Second Edition is an awesome resource for the beginning VB.NET developer who is serious about the craft of programming. In 1,500 pages, this book covers an incredible amount of territory, from the evolution of programming languages and web development all the way to advanced data structures and web services.</description>
            <author>Application Administrator</author>
            <pubDate>3/3/2005 5:15:30 PM</pubDate>
        </item>
        <item>
            <title>Simply Visual Basic.NET 2003</title>
            <link>http://localhost/CMS400Example/collection.aspx?id=26</link>
            <description>An Application-Driven Tutorial Approachp</description>
            <author>Application Administrator</author>
            <pubDate>1/8/2005 10:50:30 AM</pubDate>
        </item>
        <item>
            <title>Microsoft ASP.NET Programming with Microsoft Visual C#.NET</title>
            <link>http://localhost/CMS400Example/collection.aspx?id=25</link>
            <description>Teach yourself how to write high-performance Web applications with ASP.NET and Visual C# .NET--one step at a time.</description>
            <author>Application Administrator</author>
            <pubDate>1/4/2005 7:16:21 PM</pubDate>
        </item>
        <item>
            <title>HTML for the World Wide Web with XHTML and CSS</title>
            <link>http://localhost/CMS400Example/collection.aspx?id=27</link>
            <description>The easiest HTML book weve seen that still manages to be comprehensive, and to stay firmly grounded in the reality of todays demanding users and complex browsers.</description>
            <author>Application Administrator</author>
            <pubDate>1/19/2005 4:18:06 PM</pubDate>
        </item>
                </channel>
</rss>
```

RSS Summary Attributes

Listed below are all the different attributes and descriptions about them.

Starting Folder

In this attribute, you will need to specify which content folder you will want to get all the summaries from. There are two options for this attribute:

Value	Description
“”	By leaving the value blank, all the summaries from the root content folder will be displayed.
“\Folder Path	You can specify a specific folder to display the summaries from by inserting the folder path

Recursive

When you setup the search to be recursive, you are allowing the search to search all the subfolders associated with the starting folder you had specified. You have two options for this attribute:

Value	Description
1	Recursive - summaries in the starting folder’s subfolders will also be displayed.
0	Non-recursive - only summaries in the starting folder will be displayed.

Order By

This attribute will sort the summary results by different options. There are five different ways to sort the summaries:

Value	Summaries sorted by
Title	title

Value	Summaries sorted by
DateModified	date modified
Date Created	date created
Startdate	The Start Date assigned to the content block when created.
AuthorFname	author's first names
AuthorLname	author's last name

Ascending/Descending

This attribute is an addition to the previous OrderBy attribute. By default, the list of the summaries are descending, meaning they will follow the path of A-B-C, or 1-2-3. When set to ascending, they will order by Z-Y-X or 3-2-1.You will have the option to re arrange these to display either in ascending or descending order.

To do this, add the following to the OrderBy attribute:

Value	Description
asc, 1	This will order the Summaries in an ascending order
desc, 0	This will order the Summaries in a descending order. This is the default setting.

By using the ascending/descending option, you will have the option to set the order of content blocks from most recent to oldest, or vice versa. As well as ordering the list by the user's first names or last names in ascending or descending order.

Max Number

This attribute determines how many summaries appear on a page. Enter the number of summaries to be displayed. To display all summaries, enter 0 (zero) and you will display unlimited results.

Collection Function

The collection function is used to display the list of links that were created as a collection in the Ektron CMS400.NET Workarea. The ecmCollection function is highly customizable, allowing you to easily define how the collection data appears on the Web page.

Shown here is the format of the ecmCollection function.

```
<% ecmCollection      "id",
                      "Display Function"     %>
```

The ecmCollection function attributes are explained below.

Attribute	Description
id	The id of the collection that you would like to be displayed is defined here.
displayFunction	Specify a function that is defined in the API and determines how the appearance of the collection data on a Web page. The display function must also be defined in the ecmCollection function.

See Also: The **Ektron CMS400.NET** User Manual Section "Working with Collections" > "Comparison of Collections, Menus, and the List Summary Features"

Collection Examples

The following are examples of the collection function in the Ektron CMS400.NET sample Web site. One example uses the `ecmNavigation` display function, and the other uses the `ecmTeaser` display function.

The `ecmNavigation` and `ecmTeaser` display functions are sample display functions included with Ektron CMS400.NET.

The ecmNavigation Display Function

The following sample collection function is being used as a navigation menu in the Ektron CMS400.NET sample Web site.

```
<% ecmCollection 1, "ecmNavigation" %>
```

In this example, the collection with an ID=1 is displayed, and the `ecmNavigation` function defines the appearance of the collection data on a Web page.

Here is how a collection might appear on a Web page.

Below is the source code for the `ecmNavigation` display function.

```
Function ecmNavigation(cInfo)

    dim html, info

    html = "<table border=""0"" cellspacing=""0"" cellpadding=""0""

                width=""100%""">"

    for each info in cInfo

        html = html & "<tr><td>  <a href="""

        html = html & info("ContentLinks")

        html = html & """>"

        html = html & Cstr(info("ContentTitle"))

        html = html & "</a></td></tr><tr><td> </td></tr>"

    next

    html = html & "</table>"

    ecmNavigation = html

end Function
```

As you can see, the `ecmNavigation` function is a simple ASP function that creates a table of the links to the specified collection. It displays the title of the content blocks by using the `info("ContentTitle"))` building block.

The ecmTeaser Display Function

The following is an example of a collection function being used as a navigation menu in the Ektron CMS400.NET sample Web site.

```
<% ecmCollection 2, "ecmTeaser" %>
```

In this example, the collection with an ID=2 is displayed, and the `ecmTeaser` function defines the appearance of the collection data on the Web page. `ecmTeaser` displays the title and summary of each content block in the collection.

Here is how the collection might appear on a Web page.

The format of the collection on the Web page depends on the displayFunction that is used. Below is the source code for the display function `ecmTeaser`.

```
Function ecmTeaser(cInfo)

    dim html, info

    html = "<table border=""0"" cellspacing=""0"" cellpadding=""0""

                width=""100%"">"

    for each info in cInfo

        html = html & "<tr><td><a href="""

        html = html & CStr(info("ContentLinks"))

        html = html & """ >"

        html = html & Cstr(info("ContentTitle"))

        html = html & "</a> ("

        html = html & info("DateModified")

        html = html & ")</td></tr><tr><td>"

        html = html & Cstr(info("ContentTeaser"))

        html = html & "</td></tr><tr><td> </td></tr>"

    next

    html = html & "</table>"

    ecmTeaser = html

end Function
```

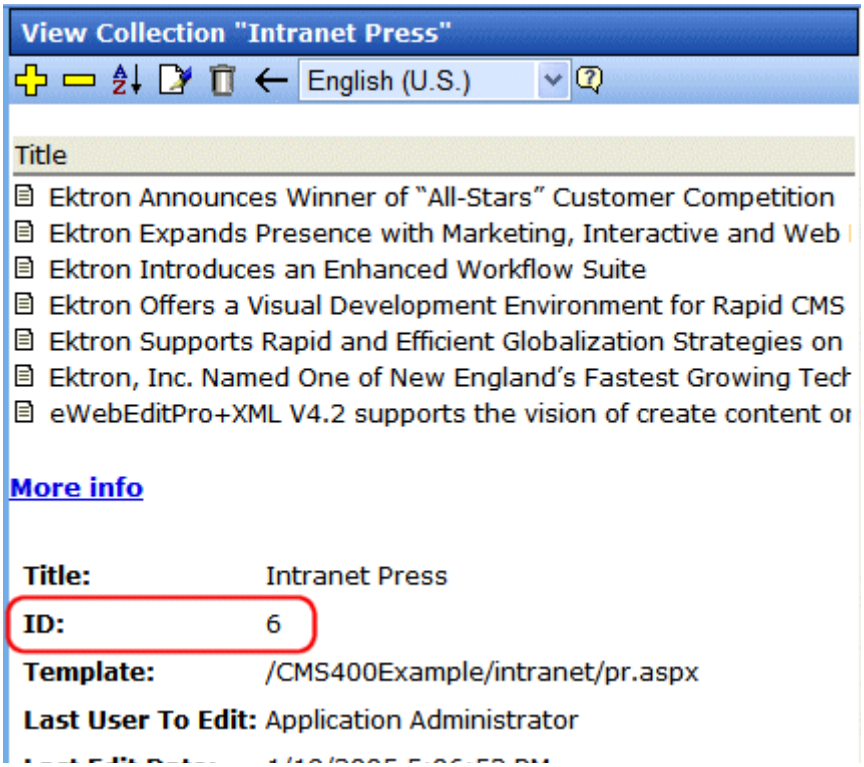
As you can see, the `ecmTeaser` function is an ASP function that creates a table of links to the specified collection. The `ecmTeaser` function displays the:

- Content title
- Date modified
- Content block summary

Collection Attributes

ID

The collection ID number is assigned to the collection when it is created.



DisplayFunction

A display function determines which elements of the content blocks in a collection appear on a Web page. Via ASP scripting, you can create display functions or use any of the three included functions. You must define the display function in two places.

- In the ecmCollection function as a case
- In the API file as a function

RSS Collection Function

The RSS collection function is used to create an XML list of links that were created as a collection in the Ektron CMS400.NET Workarea. The ecmRssCollection function is highly customizable, allowing you to easily define how the collection data appears on the Web page.

Shown here is the format of the ecmRssCollection function.

```
<% ecmRssCollection     "id"                    %>
```

The ecmRssCollection function attributes are explained below.

Attribute	Description
id	The id of the collection that you would like to be displayed is defined here.

See Also: The **Ektron CMS400.NET** User Manual Section "Working with Collections" > "Comparison of Collections, Menus, and the List Summary Features"

Here is how an RSS Collection appears when viewed without a RSS reader on a web page.

```
<?xml version="1.0" ?>
<rss xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" version="2.0">
    <channel>
        <title>Books</title>
        <link>http://localhost/CMS400Example/WorkArea/WebServices/ContentWS.asmx</link>
        <description></description>
        <item>
            <title>Visual Basic.NET How to Program</title>
            <link>http://localhost/CMS400Example/collection.aspx?id=23</link>
            <description>Visual Basic .NET How to Program, Second Edition is an awesome resource for the beginning VB.NET developer who is serious about the craft of programming. In 1,500 pages, this book covers an incredible amount of territory, from the evolution of programming languages and web development all the way to advanced data structures and web services.</description>
            <author>Application Administrator</author>
            <pubDate>3/3/2005 5:15:30 PM</pubDate>
        </item>
        <item>
            <title>Simply Visual Basic.NET 2003</title>
            <link>http://localhost/CMS400Example/collection.aspx?id=26</link>
            <description>An Application-Driven Tutorial Approachp</description>
            <author>Application Administrator</author>
            <pubDate>1/8/2005 10:50:30 AM</pubDate>
        </item>
        <item>
            <title>Microsoft ASP.NET Programming with Microsoft Visual C#.NET</title>
            <link>http://localhost/CMS400Example/collection.aspx?id=25</link>
            <description>Teach yourself how to write high-performance Web applications with ASP.NET and Visual C# .NET--one step at a time.</description>
            <author>Application Administrator</author>
            <pubDate>1/4/2005 7:16:21 PM</pubDate>
        </item>
        <item>
            <title>HTML for the World Wide Web with XHTML and CSS</title>
            <link>http://localhost/CMS400Example/collection.aspx?id=27</link>
            <description>The easiest HTML book weve seen that still manages to be comprehensive, and to stay firmly grounded in the reality of todays demanding users and complex browsers.</description>
            <author>Application Administrator</author>
            <pubDate>1/19/2005 4:18:06 PM</pubDate>
        </item>
                </channel>
</rss>
```

XML Collection Function

The XML collection function wraps XML around the parameters of a list of content that's created as a collection in the Ektron CMS400.NET Workarea. Once created you can use the XML to display the collection information. The ecmCollectionXML function allows you to easily define how the collection data appears on the Web page.

Shown here is the format of the ecmCollectionXML function.

```
<% ecmCollectionXML    "id", "GetHtml"        %>
```

The ecmCollectionXML function attributes are explained below.

Attribute	Description
id	The id of the collection that you would like to be displayed is defined here.
GetHtml	Determines whether the HTML contained within the content is included in the XML output. 0 = Do not get HTML 1 = Get HTML Note: Ektron recommends using 0 (zero). HTML content can be large and this could cause display problems when your XML collection is displayed on your Web page.

See Also: The **Ektron CMS400.NET** User Manual Section "Working with Collections" > "Comparison of Collections, Menus, and the List Summary Features"

When using the XML Collection function, the following tags are created around the properties of each item in the collection.

XML Tags	Description
<Content></Content>	Declares a new section of content in the XML collection.
<ID></ID>	The content block ID. For example: `<ID>27</ID>`
<Title></Title>	The title of the content block. For example: `<Title>HTML for the World Wide Web with XHTML and CSS</Title>`
<QuickLink></QuickLink>	The quicklink associated with the content block. For example: `<QuickLink>/CMS400Example/collection.aspx?id=27</QuickLink>`
<Teaser></Teaser>	The summary of the content block. For example: `<Teaser>The easiest HTML book weve seen that still manages to be comprehensive.</Teaser>`
<Html></Html>	All of the HTML content contained in a content block. *See Also:* "GetHtml" on page 438 For example: `<Html>ASP.NET Unleashed, Second Edition is really big, really thorough, and really <i>good</i>. The explanations are simple and clear as anything weve seen in a .NET book. The sample code addresses virtually every facet of ASP.NET development, from little gems (code for validating credit cards) to full-fledged web stores. Best of all, Stephen Walther identifies the mistakes ASP.NET developers are most likely to make -- and presents detailed solutions. </Html>`
<StartDate></StartDate>	The content block's start date formatted as a .NET date type For example: `<StartDate>1/1/0001 12:00:00 AM</StartDate>`

XML Tags	Description
<DateModified></DateModified>	The last date the content block was modified. For example: `<DateModified>1/19/2005 4:18:06 PM</DateModified>`
<EndDate></EndDate>	The date the content block stops running on the Web site For example: `<EndDate>12/31/9999 12:00:00 AM</EndDate>`
<LastEditorFname></LastEditorFname>	The first name of the last person who edited the content block. For example: `<LastEditorFname>John</LastEditorFname>`
<LastEditorLname></LastEditorLname>	The last name of the last person who edited the content block. For example: `<LastEditorLname>Johnson</LastEditorLname>`
<Hyperlink></Hyperlink>	The hyperlink shown in the collection for linking to the content block. For example: `<Hyperlink><a href="/CMS400Example/collection.aspx?id=27">HTML for the World Wide Web with XHTML and CSS</a></Hyperlink>`
<DisplayStartDate></DisplayStartDate>	The content block's start date. It is formatted as a string that represents Ektron CMS400's display of the date.
<FolderID></FolderID>	The ID of the folder in which the content block is contained. For example: `<FolderID>0</FolderID>`
<ContentStatus></ContentStatus>	The status of the content block.
<Language></Language>	The language of the content block.
<DisplayDateModified></DisplayDateModified>	edited. It is formatted as a string that represents Ektron CMS400's display of the date. For example: `<DisplayDateModified>1/19/2005 4:18:06 PM</DisplayDateModified>`

XML Tags	Description
<DisplayEndDate></DisplayEndDate>	The content block's end date. It is formatted as a string that represents Ektron CMS400's display of the date. For example: `<DisplayEndDate>12/31/9999 12:00:00 AM</DisplayEndDate>`
<EndDateAction></EndDateAction>	What should happen to the content block on its end date. • archive display • archive expire • refresh report For example: `<EndDateAction>Archive_Expire</EndDateAction>`
<Comment></Comment>	Comments contained in the content block. For example: `<Comment>move picture below text</Comment>`

Here is how the raw XML Collection output appears before an XSLT is applied.

```
<Content>
    <ID>24</ID>
    <Title>ASP.NET Unleashed</Title>
    <QuickLink>/CMS400Example/collection.aspx?id=24</QuickLink>
    <Teaser><p><em>ASP.NET Unleashed, Second Edition</em> is really big, really thorough, and
really <i>good</i>.</p></Teaser>
    <Html></Html>
    <StartDate>1/1/0001 12:00:00 AM</StartDate>
    <DateModified>1/12/2005 7:53:31 PM</DateModified>
    <EndDate>12/31/9999 12:00:00 AM</EndDate>
    <LastEditorFname>Application</LastEditorFname>
    <LastEditorLname>Administrator</LastEditorLname>
    <Hyperlink><a href="/CMS400Example/collection.aspx?id=24">ASP.NET Unleashed</a></
Hyperlink>
    <DisplayStartDate></DisplayStartDate>
    <FolderID>0</FolderID>
    <ContentStatus></ContentStatus>
    <Language>0</Language>
    <DisplayDateModified>1/12/2005 7:53:31 PM</DisplayDateModified>
    <DisplayEndDate></DisplayEndDate>
    <EndDateAction>Archive_Expire</EndDateAction>
    <Comment></Comment>
</Content>
<Content>
    <ID>25</ID>
    <Title>Microsoft ASP.NET Programming with Microsoft Visual C#.NET</Title>
    <QuickLink>/CMS400Example/collection.aspx?id=25</QuickLink>
    <Teaser><p>Teach yourself how to write high-performance Web applications with ASP.NET and
Visual C# .NET--one step at a time.</p></Teaser>
    <Html></Html>
    <StartDate>1/1/0001 12:00:00 AM</StartDate>
    <DateModified>1/4/2005 7:16:21 PM</DateModified>
    <EndDate>12/31/9999 12:00:00 AM</EndDate>
    <LastEditorFname>Application</LastEditorFname>
    <LastEditorLname>Administrator</LastEditorLname>
    <Hyperlink><a href="/CMS400Example/collection.aspx?id=25">Microsoft ASP.NET Programming
with Microsoft Visual C#.NET</a></Hyperlink>
    <DisplayStartDate></DisplayStartDate>
    <FolderID>0</FolderID>
    <ContentStatus></ContentStatus>
    <Language>0</Language>
    <DisplayDateModified>1/4/2005 7:16:21 PM</DisplayDateModified>
    <DisplayEndDate></DisplayEndDate>
    <EndDateAction>Archive_Expire</EndDateAction>
    <Comment></Comment>
</Content>
```

Random Content

The random content function is used to randomly display the content of a content block that belongs in a specified collection.

Displayed here is the format for the random content function

```
<% ShowRandomContent "id", XSLTId %>
```

Where the "**id**" represents the ID number of the collection the content blocks, that will be displayed, belong to.

And "**XSLTId**" is the ID number of the display XSLT that will be applied to the content blocks in the collections.

Random Summary

Similar to the random content function, the random summary function randomly displays a content block summary that belongs in a specified collection.

Here is the format for the random content function

```
<% ShowRandomSummary “id” %>
```

The “id” represents ID number of the collection to which the content blocks belong.

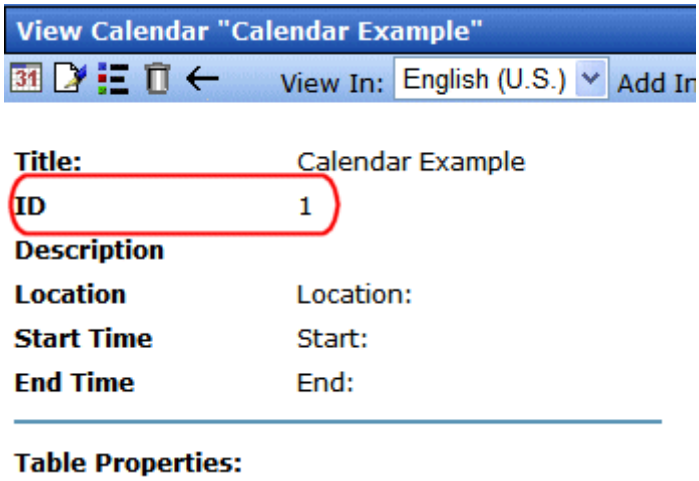
Event Calendar Function

The ecmEvtCalendar function displays a month view event calendar that has been created in Ektron CMS400.NET. Here is the format for the event calendar custom function.

```
<% ecmEvtCalendar "id" %>
```

NOTE You can use ecmCalendar to call a calendar and display a different view, such as daily. For more Information, see "Calendar Function" on page 448

The **"id"** argument represents the ID number of the event calendar in the Workarea.



Event Calendar Style Sheet

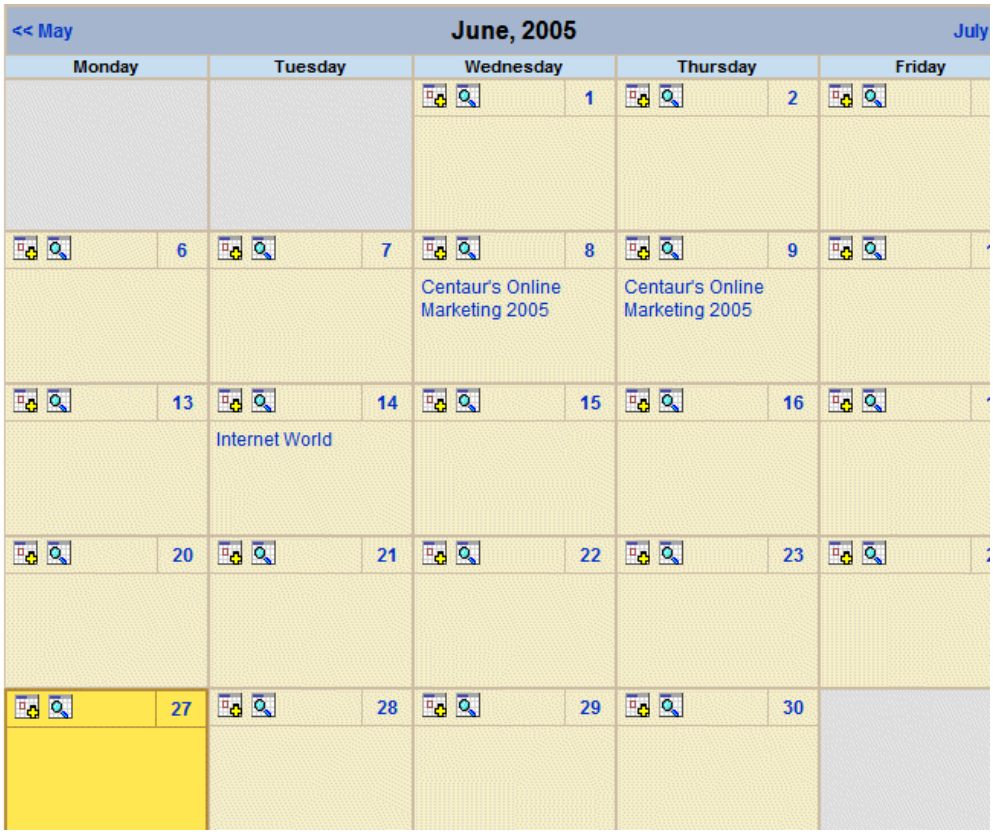
Before you insert an event Calendar into your Ektron CMS400.NET template, you should make a reference to the style sheet, `\CMS400example\calendarStyles.css`, in your template. This style sheet contains styles that define the Calendar.

Event Calendar Sample

Here is an example of an event Calendar in an Ektron CMS400.NET template.

```
<tr>
    <td>
        <% ecmEvtCalendar "1" %>
    </td>
</tr>
```

When this code is displayed in a browser, the Calendar with an ID-1 is displayed. Here is the calendar displayed on a Web Page

If...	Calendar Display
Logged in	

If...	Calendar Display
Logged Out	

As you can see, when you are logged into the Web site, you can add and view Calendar events. When logged out, you can only view calendar events.

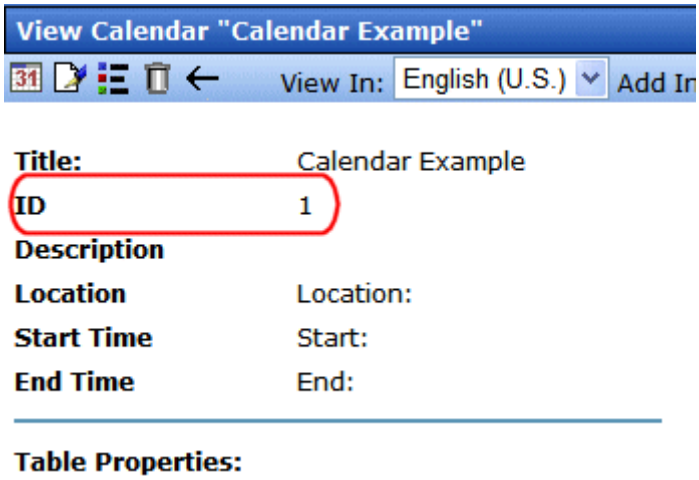
Calendar Function

The ecmCalendar function displays a calendar that has been created in Ektron CMS400.NET. Here is the format for the event calendar custom function.

```
<% ecmCalendar “id”, “DisplayType” %>
```

NOTE While you can display a month view calendar with ecmCalendar, you can also use ecmEvtCalendar to display a month view calendar. For more Information, see ”Event Calendar Function” on page 445

The ecmCalendar function attributes are explained below.

Attribute	Description
id	The **“id”** argument represents the ID number of the event calendar in the Workarea. 

Attribute	Description
DisplayType	Specify the Calendar display type. • **Month** - A 30 day calendar month view. • **Day** - Single day view. • **Inline** - Display events that occur from the start date of the calendar through the end date. • **MonthUpWithEvent** - Displays a month calendar. Events appear to the right of the calendar when you click on a day.

Calendar Style Sheet

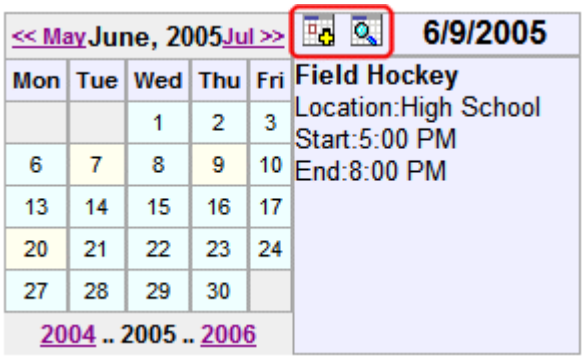
Before you insert an event Calendar into your Ektron CMS400.NET template, you should make a reference to the style sheet, `\CMS400example\calendarStyles.css`, in your template. This style sheet contains styles that define the Calendar.

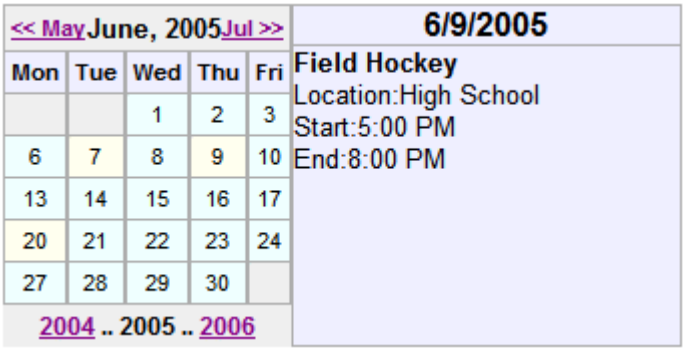
Event Calendar Sample

Here is an example of an event Calendar in an Ektron CMS400.NET template.

```
<tr>
    <td>
        <% ecmCalendar "1", “MonthUpWithEvent” %>
    </td>
</tr>
```

When this code is displayed in a browser, the Calendar with an ID of 1 and DisplayType of MonthUpWithEvent is displayed. Here is the calendar displayed on a Web Page

If...	Calendar Display
Logged in	Notice the add event and view event buttons that appear when logged in. 

If...	Calendar Display
Logged Out	

As you can see, when you are logged into the Web site, you can add and view Calendar events. When logged out, you can only view calendar events.

Form Function

The ecmFormBlock and ecmFormBlockNoFormTag functions display an HTML form content block. Both functions have the same parameters and display the form the same way. The only difference is ecmFormBlock automatically inserts form tags, ecmFormBlockNoFormTag does not.

The following is an example of form tags.

```
<form name="myform" action="#" onsubmit="return
EkFmValidate(this);" method="post">

</form>
```

When using ecmFormBlock, form tags are added automatically. When using ecmFormBlockNoFormTag, form tags need to be added manually.

For more information about HTML forms, see the **Ektron CMS400.NET** Administrator manual chapter “Working with HTML Forms.”

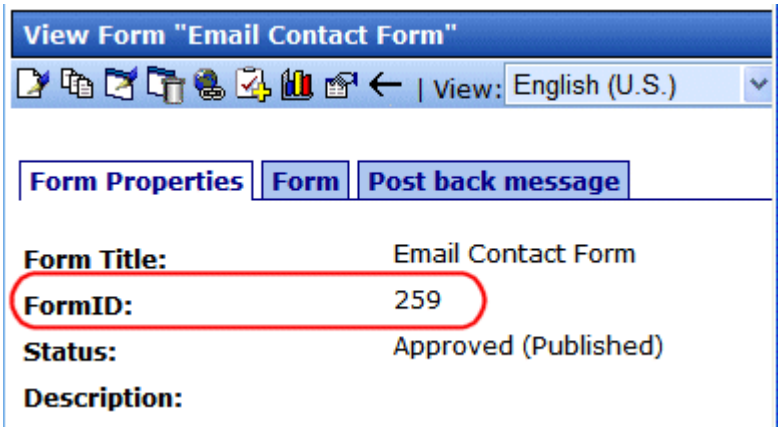
IMPORTANT! If you create a template for an existing form content block, you must manually change its quicklink to point to the new template. This change does not occur automatically. This procedure is described in the Ektron CMS400.NET Administrator manual chapter “Managing Library Assets”, section “Updating Default Template for Multiple Quicklinks.”

Here are the two formats for the form custom function.

<% ecmFormBlock “id” %>
or
<% ecmFormBlockNoFormTag “id” %>

NOTE As an alternative, you can use the multipurpose function, which can display either a content block or a content block associated with a form. For more information, see ”MultiPurpose Function” on page 455.

The “id” argument represents the form’s ID number, visible when viewing the form from the View Form screen.



ecmFormBlock and ecmFormBlockNoFormTag are interchangeable when using this function. Here is an example of a form content block in an Ektron CMS400.NET template.

```
<tr>

    <td>

        <% ecmFormBlock "367" %>

    </td>

</tr>
```

When this code is read by a browser, the following is displayed.

When a visitor to your site submits a form, the form uses the form information to process the data.

MultiPurpose Function

The ecmMultiPurpose function displays either a content block or an content block associated with a form. As a result, you can use one template to display both types of content block.

Here is the format for the Multipurpose function:

```
<% ecmMultiPurpose “id” %>
```

To pass in a standard content block, pass the id value in the url parameter, as illustrated below.

```
http://localhost/cms400sample/index.asp?id=2
```

To pass in a Form content block, pass the ekfrm value in the url parameter, as illustrated below.

```
http://localhost/cms400sample/index.asp?ekfrm=2
```

NOTE For more information about passing the id value as a url parameter, see ”Dynamic Content Block” on page 380.

If the id value or ekfrm value does not exist, a blank template appears.

If the url parameter is not passed in or recognized, the function calls ecmContentBlock and passes the DefaultID value as the id argument. If the DefaultID value is zero (0), ecmContentBlock is not called.

Metadata Function

The metadata tag is used to display the metadata for one or more content blocks in the Ektron CMS400.NET Web site. This custom tag lets you generate the list of metadata edited by the authors and insert it in the source of the Web page. Below is an example of the metadata custom tag.

```
<%  ecmMetaData "ItemList","Spare1" %>
```

The attributes are described below.

Attribute	Description	Required?
ItemList	List of the content block ID numbers, and exclusions *See Also:* "Creating an Item List" on page 456 and "Item List" on page 461	No
Spare1	Not yet implemented *See Also:* "Spare 1" on page 463	MUST BE ""

Creating an Item List

The Item List for the ecmMetadata tag has the following format.

```
"[content block id; Exclusion list]...[content block id; Exclusion list]"
```

Attribute	Description	Required?
[(left bracket)	Delineates the beginning of a content block id/exclusion list pair	Yes
Content Block ID	The ID of the content block whose metadata you want to retrieve	Yes

Attribute	Description	Required?
; (semicolon)	Separator: Separates content block id from the exclusion list	No (**Yes** if including an **exclusion** list)
Exclusion List	A comma delimited list of metadata you do not want to be retrieved for this content block.	No
] (right bracket)	Delineates the end of a content block id/exclusion list pair	Yes

ecmMetadata Example

For this example, our administrator has defined the following metadata types for Ektron CMS400.NET.

Name	Style	Type	Editable	Required	Remove Dups	Case Sensitive
Title	n/a	HTML	Yes	Yes	No	n/a
Keywords	Name	Meta	Yes	Yes	Yes	No
Content-Type	http-equiv	Meta	No	Yes	Yes	n/a

Our example Web page has two content blocks displayed.

- content block 1, the company's home page introduction text
- content block 2, a sidebar containing the lead story for the week

The metadata entered by your content editors for these two content blocks are:

Content Block 1:
Title – "Welcome to Our Home Page"
Keywords – "Home; page; company; Welcome"
Content-Type -- "text/html; charset=iso-8859-1"

Content Block 2:
Title – “Announcing our New Product!”
Keywords – “Announce; company; product; widget”
Content-Type -- “text/html; charset=iso-8859-1“

To include both content blocks’ metadata in a Web page, the ecmMetadata call looks like this.

```
<html>
    <head>
        <% ecmMetadata “[1][2;Title,Content-Type]”, “” %>
        (rest of head)
    </head>
    <body>
        (rest of web page document)
    </body>
</html>
```

In the above example, the ecmMetadata call is contained in the `<head>` of the HTML document (since all HTML metadata is required to exist there). When the Web page is viewed, the Web page source contains the following information:

```
<html>
    <head>
        <Title>Welcome to Our Home Page</Title>
        <meta name=”Keywords” content= Home; page; company; Welcome; Announce;
        product; widget”>
        <meta http-equiv=”Content-Type” content=”text/html; charset=iso-8859-1”>
        (rest of head)
    </head>
    <body>
        (rest of web page document)
```

```
    </body>

</html>
```

The ecmMetadata constructed three HTML tags based on the administrator defined parameters and the ecmMetadata call.

The metadata tag was constructed in the page because:

- **[1]** - Displays all metadata content for the content block ID=1. In this example, the title, keywords, and content-type were defined, so they were displayed as:

```
<Title>Welcome to Our Home Page</Title>

    <meta name="Keywords" content= Home; page; company; Welcome;

        Announce; product; widget">

    <meta http-equiv="Content-Type" content="text/html;

        charset=iso-8859-1">
```

- **[2;title,content-type]** - Displays all metadata content for content block ID=2, *except* for its title and content-type definitions.

```
<Title>Welcome to Our Home Page</Title>

    <meta name="Keywords" content= Home; page; company; Welcome;

        Announce;  product; widget">

    <meta http-equiv="Content-Type" content="text/html;

        charset=iso-8859-1">
```

The next section provides more examples.

More ecmMetadata Examples

Shown below is an example of the metadata function in an ASP Web page.

```
<TR>

    <TD>

        <%  ecmMetaData "[1][2;title][14;keywords]","" %>
```

```
    </TD>
</TR>
```

The previous example has the following properties:

- All metadata for content block ID=1 is inserted into the page
- All metadata for the content block ID=2 except for its title is inserted into the page
- All metadata for the content block ID=14 except for its keywords is inserted into the page

Using a Dynamic Metadata Tag

You can use a metadata tag in a template to define the metadata for a specific content block. However, you can also use a dynamic metadata tag combined with a content block tag to show the metadata for all content blocks on a Web page.

This allows you to create a template and not have to keep updating the tag by adding the new ID and exclusion list. Shown below is the sample code for this.

```
<!-- #include virtual="/CMS400.NETsample/CMS400.NETscripts/ektronAPI.asp" -->
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
    <head>
        <% Dim id
            if (request.QueryString("id")<> "") then
                id = request.QueryString("id")
            else
                id = 1
            end if
                ecmMetadata "[" & id & "]", ""
        %>
    </head>
    <body>
```

```
        <% ecmContentBlock(request.QueryString("id")) %>

    </body>

</html>
```

In this example, if you specify a content block in the URL and the page is displayed, the metadata takes that ID and displays the proper metadata in the source for the content block.

ecmMetadata Attributes

Below are the attributes and their descriptions.

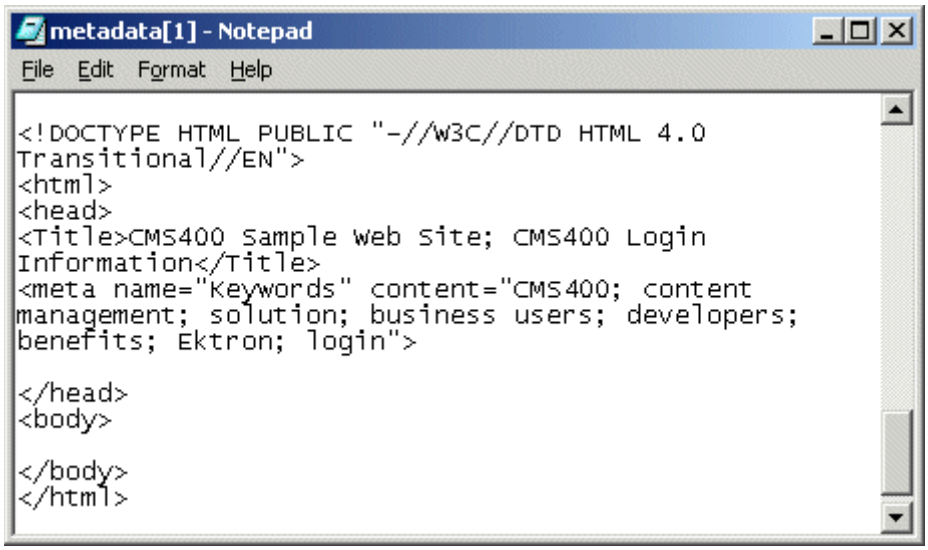
Item List

In this attribute of the ecmMetadata tag, you need to specify two values.

Value	Specifies
ID	The content block ID whose metadata will appear
Exclusion List	The metadata to be excluded. *See Also:* "Creating an Item List" on page 456

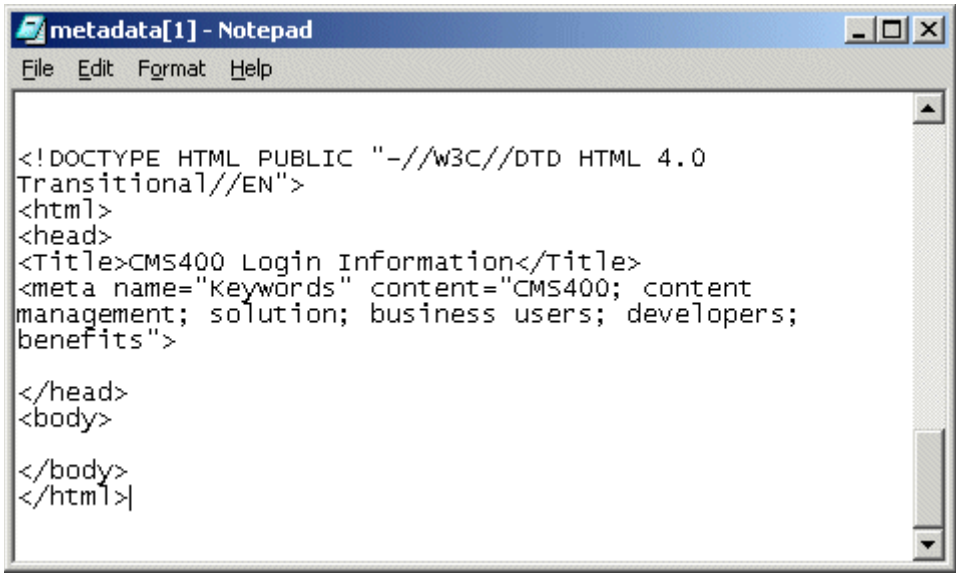
Some examples might be:

```
<%  ecmMetaData "[1][2]","" %>
```



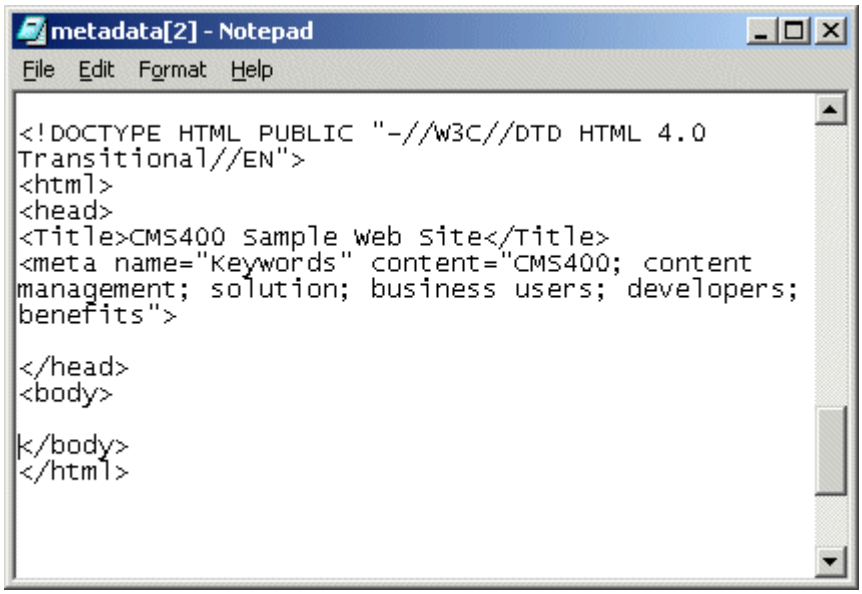
In this example, all metadata from content block ID=1 and 2 is displayed.

```
<%  ecmMetaData "[1;title][2;keywords]","" %>
```



In this example, all the metadata from content block ID=1 will be displayed *except* for the title. Also, all the metadata for content block ID=2 is displayed *except* its keywords.

```
<%  ecmMetaData "[1][2;title,keywords]","" %>
```



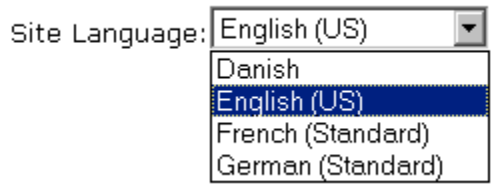
In this example, all metadata from content block ID=1 is displayed, and all metadata from content block ID=2 is displayed *except* its title and keywords.

Spare 1

One attribute field at the end is currently not used. It is for future scalability. When you insert a single metadata tag into a template, you must leave this attributes as “”.

Show Site Language Function

This function lets a visitor to your Web site select a language in which to view the site. Here is what the function looks like when published on a Web page.



It lists all languages selected on the **Settings > Configuration > Language settings** screen. (For more information, see the “Multi-Language Support “chapter of the **Ektron CMS400.NET** Administrator manual.)

Below is an example of the Show Site Language custom tag.

```
<% ecmShowSiteLanguage(FieldName)%>
```

You can place this function in any location of any page on your site. By default, it appears in the lower right corner of the home page.

NOTE This function only works with Asp and ASP.Net.

The attribute is described in this table.

Attribute	Description

FieldName	An optional argument. You can leave it blank and allow the drop-down list to be populated by the available languages for your Web site. For example:
	`<%=ecmShowSiteLanguage(‘’)%>`
	Or you can pass in an argument such as:
	`<%=ecmShowSiteLanguage(‘mysitelanguage’)%>`

DHTML Menus

Ektron CMS400.NET's Menu feature lets users create and maintain a dropdown menu system for your Web site. The Menu feature lets you display menu options that link to content blocks, library assets, external hyperlinks and submenus.

To learn more about menus, please refer to the "Menu Feature" chapter of the **Ektron CMS400.NET** User Manual.

Additional Documentation in Ektron Knowledge Base

This chapter explains how to implement standard menus that ship with **Ektron CMS400.NET**. Ektron's Web site also provides Knowledge Base articles that explain how to extend the standard menu features.

Displaying a Menu on a Web Page

GenerateMenu and DropMenu

IMPORTANT! The menu functions described below are only one example of how to implement menus. The scripts reside in the Application API file. You can implement your own menu system using JavaScript.

Use the `GenerateNextGenMenu` and `DropNextGenMenu` custom functions to display a DHTML menu on a Web page. Place `GenerateNextGenMenu` once on the page to provide an infrastructure for `DropNextGenMenu`. Then, place `DropNextGenMenu` as needed to implement individual menus.

Follow these steps to display a menu on a Web template.

1. Place the following code directly after the page's `<body>` tag.

```
<%=(ecmDHTML_GenerateNextGenMenu("menu id number"))%>
```

For example:

```
<%=(ecmDHTML_GenerateNextGenMenu("1"))%>
```

To display more than one menu, separate each id number with a comma. For example:

```
<%=(ecmDHTML_GenerateNextGenMenu("1,2,3"))%>
```

2. If the user clicks *text* to display the menu, place the following code within the template file where the text would appear.

```
<%= ecmDHTML_DropNextGenMenu(menu id number,"text user clicks to display menu")%>
```

For example:

```
<%= ecmDHTML_DropNextGenMenu(1,"Products")%>
```

If the user clicks *a graphic* to invoke the menu, use this syntax:

```
<%= ecmDHTML_DropNextGenMenu(menu id number,'<img src=graphic file name>')%>
```

For example:

```
<%= ecmDHTML_DropNextGenMenu(1,'<img src=mymenugraphic.jpg>')%>
```

Displaying Menus in a Multi-language System

If your Web site supports multiple languages, and the user selects a language before viewing the menu, if an edition of the menu is available in that language, it appears. If not, nothing appears.

For more information, see the **Ektron CMS400.NET** Administrator Manual section "Multi-Language Support "> "Working with Multi-Language Content" > "Working with Menus in a Multi-Language System."

Customizing the Menu's Appearance

You can customize a menu's appearance, such as the font style, color, and size. So, you have the flexibility to coordinate a menu with the design your Web site.

To customize a menu in **Ektron CMS400.NET**, you can change the ek_menu.js file. *See Also:* "ek_Menu.js" on page 467

ek_Menu.js

You can change the look and feel of a menu by changing the ek_menu.js file. If you used the default settings when installing **Ektron CMS400.NET**, this file is in `webroot/CMS400Example/Workarea/java/`.

Use the following table to help you change menu attributes.

Attribute name	Specifies the	Default value
fontSize	Size of menu text	14
fontWeight	Thickness of menu text You can set FontWeight by name (`Bold` or `Plain`) or value. Values range from 0 -1000.	Plain
fontFamily	Font style of menu text *See Also:* "FontFamily" on page 468	arial,helvetica,espy, sans-serif
fontColor	Color of menu text Note: When changing the color, you can specify the color name or code.	#000000
fontColorHiLite	Color of text when menu item is highlighted	#ffffff
bgColor	Color of border between menu items	#555555
menuBorder	Width (in pixels) of border around main menu	1
menuItemBorder	Width (in pixels) of border around each menu item	1
menuItemBgColor	Background color of menu items	#6699ff
menuLiteBgColor	Color of main menu border highlight *See Also:* "menuLiteBgColor" on page 468	#ffffff
menuBorderBgColor	Color of the main menu border	#6699ff
menuHiLiteBgColor	Background color of highlighted menu item	#000084

Attribute name	Specifies the	Default value
menuContainerBgColor	Background color of the *container* Note: The container is like a tag that displays the menu. When you set the container color, it appears at the top and left side of the menu frame only.	#6699ff
childMenuIcon	Path to the submenu icon (the image used to indicate that a submenu is available from this menu item). To display a different image, either provide an external path to it or upload the image to **Ektron CMS400.NET** then provide a local path.	`./Workarea/images/application/arrows.gif`
childMenuIconHiLite	The path to the icon that appears when you hover the mouse over a menu item that leads to a submenu. To display a different image, either provide an external path to it or upload the image to **Ektron CMS400.NET** then provide a local path.	`./Workarea/images/application/arrows2.gif`

FontFamily

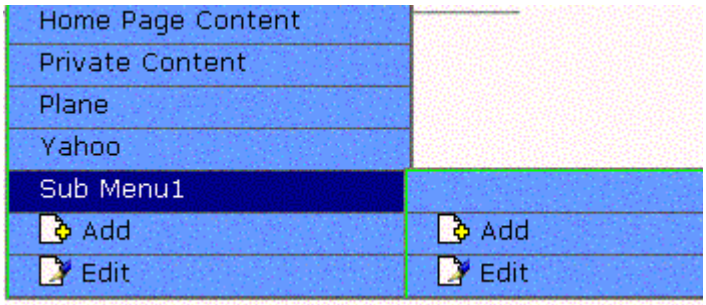
Use the fontFamily attribute to specify the menu font. It must be supported by the browser.

You can assign to this property several, comma-separated values.

NOTE To display a font family other than the browser default, specify only the font name and no other comma-separated values. If you specify more than one value for the font family, Ektron CMS400.NET uses the default IE settings.

menuLiteBgColor

Example


Home Page Content
Private Content
Plane
Yahoo
Sub Menu1
Add
Edit
Add
Edit

XML Menu

The ecmGetMenuXML function wraps XML around the properties of a menu created in the Ektron CMS400.NET Workarea. Once created you can use the XML to display the menu. The ecmGetMenuXML function allows you to easily define how the menu data appears on the Web page.

Shown here is the format of the ecmGetMenuXML function.

```
<% ecmGetMenuXML (id) %>
```

The ecmGetMenuXml function attributes are explained below.

Attribute	Description
Id	The id that associated with the menu.

Here is how the raw XML Menu output appears before an XSLT is applied.

```
<Item>
    <ItemID>1</ItemID>
    <ItemType>Submenu</ItemType>
    <ItemTitle>MenuExample</ItemTitle>
    <ItemImage></ItemImage>
    <ItemImageOverride>False</ItemImageOverride>
    <Menu>
        <ID>1</ID>
        <Title>MenuExample</Title>
        <Template></Template>
        <Type>content</Type>
        <Link></Link>
        <ParentID>0</ParentID>
        <AncestorID>1</AncestorID>
        <FolderID>0</FolderID>
        <Description></Description>
        <ImageOverride>False</ImageOverride>
        </Item>
        <Item>
            <ItemID>2</ItemID>
            <ItemType>Submenu</ItemType>
            <ItemTitle>Book</ItemTitle>
            <ItemImage></ItemImage>
            <ItemImageOverride>False</ItemImageOverride>
            <Menu>
                <ID>2</ID>
                <Title>Book</Title>
                <Template></Template>
                <Type>content</Type>
                <ParentID>1</ParentID>
                <ImageOverride>False</ImageOverride>
                <Item>
                    <ItemID>23</ItemID>
                    <ItemType>content</ItemType>
                    <ItemTitle>VB.NET How to</ItemTitle>
                    <ItemImage></ItemImage>
                    <ItemLink>/CMS400Example/collection.aspx?id=23</ItemLink>
                    <ItemTarget>2</ItemTarget>
                </Item>
            </Menu>
        </Item>
    </Menu>
</Item>
```

Custom ColdFusion Functions

Ektron CMS400.NET maintains and display content through the use of custom Ektron ColdFusion functions and ComObjects.

When using custom ColdFusion functions, the following paths must be updated in the application.cfm.

```
<cfset application.WSDLpath = "http://localhost/CMS400Example/WorkArea/WebServices/
ContentWS.asmx?WSDL" />
```

and

```
<cfset application.AppPath = "/CMS400Example/WorkArea/platforms/ColdFusion/" />
```

The `application.WSDLpath` needs to updated with the location of the WSDL file. For example, For example, if your site is `www.myektronsite.com`, the WSDLpath would be `http://www.myektronsite.com/WorkArea/WebServices/ContentWS.asmx`.

The application.AppPath needs to updated with the path to the ColdFusion custom Functions.

To customize you Ektron CMS400.NET application, you will need to be familiar with these tags

Function	Description	More Information
Calendar Functions		
Event Calendar	Displays a month view event calendar	”Event Calendar Function” on page 490
Collection Functions		
Collection	Displays links generated by collections	”Collection Function” on page 484

Function	Description	More Information
CollectionXML	Creates XML for a collection by wrapping tags around all of the collection items and their properties. This allows you to use an XSLT to define how the XML collection is displayed.	”XML Collection Function” on page 489
Random Content	Randomly displays the content of a collection item	”Random Content Function” on page 478
Random Summary	Randomly displays the summary of a collection item	”Random Summary Function” on page 478
RSS Collection	Creates an RSS feed out of the items in a collection.	”RSS Collection Function” on page 490
Content Functions		
Add Content	Allows a user to add content to **Ektron CMS400.NET** progammatically.	”Add Content” on page 476
Content Block	Displays and manages a content block	”Content Block Function” on page 475
Form	Displays a content block associated with a form	”Form Function” on page 493
Multipurpose	Displays either a content block or a form	”MultiPurpose Function” on page 494
Language Functions		
Language Select	Lets a site visitor select a language in which to view site.	”Language Select Function” on page 499
List Summary Functions		
Archive ListSummary	Displays summaries of archived content blocks	”Archive List Summary Function” on page 483
List Summary	Displays a list of content block summaries	”List Summary Function” on page 480

Function	Description	More Information
List Summary XML	Creates XML for a list summary by wrapping tags around all of the collection items and their properties. This allows you to use an XSLT to define how the XML collection is displayed.	”XML List Summary Function” on page 481
RSS List Summary	Creates an RSS feed out of the items in a list summary.	”RSS List Summary Function” on page 482
Single Summary	Displays the summary for a content block.	”Single Summary Function” on page 483
Login Functions		
Login	Adds the login or logout button	”Login Function” on page 475
Menu Functions		
DropHTMLMenu	Used to display a DHTML menu	”DHTML Menus” on page 497
GenerateMenu	Works with DropHTML menu to display DHTML menus	”DHTML Menus” on page 497
XML Menu	Creates XML for a menu by wrapping tags around all of the menu items properties.	”XML Menu Function” on page 498
Metadata Functions		
Metadata	Displays the metadata for the current Web page	”Metadata Function” on page 494
Search Functions		
Search	Inserts a text box with a search button	”Search Function” on page 478
Search Display	Creates an area to display search results	”Search Display Function” on page 480

Login Function

This tag is responsible for displaying the following buttons on the Web page.

This function is basically the same as the Login function in ASP. See "Login Function" on page 373 for details. The only difference is the syntax used to implement the function. This is listed below.

Login Tag Example

```
<tr>

    <td>

        <cfmodule template="#request.ecm.AppPath#ecmLogin.cfm">

    </td>

</tr>
```

Content Block Function

The content block and content blockEX tags are used to display a content block on an Ektron CMS400.NET Web page.

Shown here is the format for the content block function:

```
<CFMODULE template="#request.ecm.AppPath#ecmContentBlock.cfm"

           ID=""

>
```

Shown here is the format for the content blockEX function:

```
<CFMODULE template="#request.ecm.AppPath#ecmContentBlockEx.cfm"

           ID=""

           XSLT=""

           Override=""

>
```

This functions are basically the same as the ContentBlock and ContenBlockEX function in ASP. See "Content Block Function" on page 375 for details.

The only difference between the Coldfusion function and the ASP one is the syntax for implementing it. The ColdFusion examples are below.

Static Content Block

```
<tr>

   <td>

      <CFMODULE id="1" template="#request.ecm.AppPath#ecmContentBlock.cfm">

    </td>

</tr>
```

Dynamic Content Block

```
<tr>

  <td>

      <CFMODULE id="#id#" template="#request.ecm.AppPath#ecmContentBlock.cfm">

  </td>

</tr>
```

XML Content Block

```
  <tr>

    <td>

      <CFMODULE id="13" XSLT= “” override= “1”

      template="#request.ecm.AppPath#ecmContentBlockEx.cfm">

    </td>

  </tr>
```

Add Content

The add content function allows you to programmatically add a new content block to Ektron CMS400.NET without the need of logging in.

Here is the format for the add content function:

```
<CFModule template="#request.ecm.AppPath#ecmAddContent.cfm"   Username="",
                                                              Password="",
                                                              Domain="",
                                                              content_title="",
                                                              content_comment="",
                                                              ContentHtml="",
                                                              SummaryHTML="",
                                                              ContentLanguage="",
                                                              FolderID="",
                                                              GoLive="",
                                                              EndDate="",
                                                              MetaInfo=""          >
```

This function is basically the same as the Add Content function in ASP. See "Add Content" on page 385 for details.

The only difference between the Coldfusion function and the ASP one is the syntax for implementing it. The ColdFusion example is below.

Add Content Example

The following is an example of the add content function in a template:.

```
<tr>

    <td>

        <CFModule template="#request.ecm.AppPath#ecmAddContent.cfm "AddContent
UserName="admin", Password="admin", Domain="", content_title="Content Title",
content_comment="Comment", ContentHTML="This is the <b>body</b> of the content.",
SummaryHTML="This is the <i>summary</i> for the content.", ContentLanguage="1033",
FolderID="0", GoLive="", EndDate="", MetaInfo="">

    </td>

</tr>
```

Random Content Function

The random content function is used to randomly display the content of a content block that belongs to a specified collection. Below is the format for the random content function:

```
<CFMODULE template="#request.ecm.AppPath#ecmShowRandomContent.cfm"

                    ID=""

                    XSLT=""

                    Override=""

        >
```

This function is basically the same as the Random Content function in ASP. See ”Random Content” on page 443 for details.

Random Summary Function

SImilar to the random content function, the random summary function can randomly display a content block summary that belongs to a specified collection. Here is the format for the random content function

```
<CFMODULE template="#request.ecm.AppPath#ecmShowRandomSummary.cfm"

                    ID=""

>
```

This function is basically the same as the Random Summary function in ASP. See ”Random Summary” on page 444 for details.

Search Function

The search tag is used to display a text box, with a search button on the template. When seen on the template on the Web page, the users will be allowed to enter text and click on the search button to search through all the content on the Web site.

Shown here is the format for the ecmSearch tag.

```
        <CFMODULE template="#request.ecm.AppPath#ecmSearch.cfm"
```

```
                StartingFolder=""

                Recursive=""

                TargetPage=""

                TextBoxSize=""

                MaxCharacters=""

                ButtonImageSrc=""

                ButtonText=""

                FontFace=""

                FontColor=""

                FontSize=""

                Horizontal=""

        >
```

This function is basically the same as the Search function in ASP. See ”Search Functions” on page 389 for details.

The only difference between the Coldfusion function and the ASP one is the syntax for implementing it. The ColdFusion example is below.

Search Tag Example

```
<tr>

    <td>

        <CFMODULE template="#request.ecm.AppPath#ecmSearch.cfm"

            StartingFolder="\marketing" Recursive="1" TargetPage="search-

            display.cfm" TextBoxSize="25" MaxCharacters="200"

            ButtonImageSrc="" ButtonText="Search" FontFace="Verdana"

            FontColor="##808080" FontSize="2" Horizontal="0">

    </td>

</tr>
```

Search Display Function

The search display custom tag created the area where all the search results show after the search has been executed.

Shown here is the format of the ecmSearchDisplay tag:

```
<CFMODULE template="#request.ecm.AppPath#ecmSearchDisplay.cfm"

            MaxNumber=""

            StyleInfo=""

            ShowDate=""

>
```

This function is basically the same as the Search Display function in ASP. See ”Search Display Function” on page 400 for details.

The only difference between the Coldfusion function and the ASP one is the syntax for implementing it. The ColdFusion example is below.

Search Display Example

```
<tr>

    <td>

        <CFMODULE template="#request.ecm.AppPath#ecmSearchDisplay.cfm"

        MaxNumber="0" StyleInfo="" ShowDate="1">

    </td>

</tr>
```

List Summary Function

The list summary function is used to display the list of content summaries on a Web page. Shown here is the format of the list summary function:

```
<CFMODULE template="#request.ecm.AppPath#ecmListSummary.cfm"

            Folder=""

            MaxNumber=""
```

```
            Recursive=""

            ShowSummary=""

            StyleInfo=""

            OrderBy=""

            ShowInfo=""

>
```

This function is basically the same as the List Summary function in ASP. See ”List Summary Function” on page 406 for details.

The only difference between the Coldfusion function and the ASP one is the syntax for implementing it. The ColdFusion example is below.

List Summary Example

```
<TR>

    <TD>

        <CFMODULE template="#request.ecm.AppPath#ecmListSummary.cfm"

        Folder="\Marketing\News" MaxNumber="0" Recursive="0"

        ShowSummary="1" StyleInfo="" OrderBy="Title,asc" ShowInfo="" >

    </TD>

</TR>
```

XML List Summary Function

The XML List Summary function wraps XML around the properties of a list of content, created as a list summary in the Ektron CMS400.NET Workarea. Once created you can use the XML to display the list summary information. The ecmListSummaryXML function allows you to easily define how the list summary data appears on the Web page.

```
<TR>

    <TD>

        <CFMODULE template="#request.ecm.AppPath#ecmListSummaryXML.cfm"

        Folder="" Recursive="" OrderBy=""
```

```
        MaxNumber="" ObjType="" SummaryType="" >

    </TD>

</TR>
```

This function is basically the same as the List Summary XML function in ASP. See "XML List Summary Function" on page 414 for details.

The only difference between the Coldfusion function and the ASP one is the syntax for implementing it. The ColdFusion example is below.

XML List Summary Example

```
<TR>

    <TD>

        <CFMODULE template="#request.ecm.AppPath#ecmListSummaryXML.cfm"

        Folder="\books\" Recursive="0" OrderBy="Title,asc"

         MaxNumber="0" ObjType="folder" SummaryType="alltypes">

    </TD>

</TR>
```

RSS List Summary Function

The RSS summary function is used to create an XML list of content summaries for RSS Syndication. Shown here is the format of the RSS list summary function:

```
<CFMODULE template="#request.ecm.AppPath#ecmListSummary.cfm"

           Folder=""

           MaxNumber=""

           Recursive=""

           OrderBy=""

>
```

This function is basically the same as the RSS Summary function in ASP. See "RSS Summary Function" on page 424 for details.

The only difference between the Coldfusion function and the ASP one is the syntax for implementing it. The ColdFusion example is below.

List Summary Example

```
<TR>

    <TD>

        <CFMODULE template="#request.ecm.AppPath#ecmListSummary.cfm"

        Folder="\books" MaxNumber="0" Recursive="0"

        OrderBy="Title,asc" >

    </TD>

</TR>
```

Archive List Summary Function

The Archive list summary function displays on a Web page a list of content summaries for archived content blocks. For example, you might have a current news page and an archived news page. Use the List Summary Function to display current news, and this function to display older news stories.

On the Content Block **schedule** screen, the user must set an **End Date** then choose the Archive and Display option to have the content blocks appear via this function.

This function has the same parameters as the List Summary Function. See ”List Summary Function” on page 480 for details.

Single Summary Function

The single summary tag is used to display the summary of one content block in the Ektron CMS400.NET Web site, instead of a list of all the summaries in a content folder. Seen below is the single summary tag:

```
<cfmodule template="#request.ecm.AppPath#ecmSingleSummary.cfm"

            ContentID=""
```

```
            ShowSummary="1"

            StyleInfo=""

            ShowInfo=""

>
```

This function is basically the same as the Single Summary function in ASP. See "Single Summary Function" on page 420 for details.

The only difference between the Coldfusion function and the ASP one is the syntax for implementing it. The ColdFusion example is below.

Single Summary Example

```
<TR>

    <TD>

        <cfmodule template="#request.ecm.AppPath#ecmSingleSummary.cfm"

        ContentID="14" ShowSummary="1" StyleInfo=""

        ShowInfo="DateModified">

    </TD>

</TR>
```

Collection Function

The collection function is used to display the list of links that were created as a collection in the Ektron CMS400.NET Workarea. The ecmCollection function is highly customizable, allowing you to easily define how the collection data will be displayed on the Web page.

Shown here is the format of the ecmCollection function.

```
    <CFMODULE template="#request.ecm.AppPath#ecmCollection.cfm"

                id="2"

                displayFunction="ecmTeaser"

    >
```

This function is basically the same as the Collection function in ASP. See ”Collection Function” on page 430 for details.

The only difference between the Coldfusion function and the ASP one is the syntax for implementing it. The ColdFusion example is below.

Example Using the ecmNavigation Display Function

The following is an example of a collection function being used as a navigation menu in the Ektron CMS400.NET sample Web site.

```
<CFMODULE id="1" displayFunction="ecmNavigation"

template="#request.ecm.AppPath#ecmCollection.cfm">
```

As stated earlier, the format of the collection on the Web page depends on the displayFunction that is used. Below is the source code for the display function “ecmNavigation”.

```
<cfparam name="Attributes.nId" default="">

<cfparam name="Attributes.href" default="">

<cfparam name="Attributes.target" default="">

<cfparam name="Attributes.startStyle" default="">

<cfparam name="Attributes.endStyle" default="">

<cfset LocalError = "">

<cfobject class="CMS400.NETContentBrv2.CMS400.NETContent" Action="Create" name="cmsObj">

<cfset cNavs =
cmsObj.GetEcmCollectionItems(request.ecm.AppConfStr,Attributes.nId,0,request.ecm.cmsPreview,r
equest.ecm.user_id,request.ecm.site_id,LocalError)>

<cfif cNavs.Count()>

    <cfset nString = "">

    <cfloop collection=#cNavs# Item=nav>

        <cfset nString = nString & "<a href="""">

        <cfif len(Attributes.href)>

            <cfif FindNoCase("?", Attributes.hrefs)>

                <cfset nString = nString & Attributes.href & "&id=" &

                nav.Item("ContentID")>
```

```
            <cfelse>

                <cfset nString = nString & Attributes.href & "?id=" &

                nav.Item("ContentID")>

            </cfif>

        <cfelse>

            <cfset nString = nString & nav.Item("ContentLinks")>

        </cfif>

        <cfset nString = nString & """ ">

        <cfif len(Attributes.target)>

            <cfset nString = nString & "target=""" & Attributes.target & """ ">

        </cfif>

        <cfset nString = nString & ">">

        <cfif len(Attributes.startStyle)>

            <cfset nString = nString & Attributes.startStyle>

        </cfif>

        <cfset nString = nString & nav.Item("ContentTitle")>

        <cfif len(Attributes.endStyle)>

            <cfset nString = nString & Attributes.endStyle>

        </cfif>

        <cfset nString = nString & "</a>">

    </CFLOOP>

        <cfmodule str="#nString#" nId="#Attributes.nId#"
template="#request.ecm.AppPath#ecmshowCollectionsContent.cfm">

</cfif>

<cfset nNavs = "Nothing">

<cfset cmsObj = "Nothing">
```

As you can see, the ecmNavigation function is a simple ASP function that creates a table of the links that belong to the collection specified. The function also displays the title of the content blocks by using the “info(“ContentTitle”)) building block.

Example Using the ecmTeaser Display Function

The following is an example of a collection function being used as a navigation menu in the Ektron CMS400.NET sample Web site.

```
<CFMODULE id="2" displayFunction="ecmTeaser"

template="#request.ecm.AppPath#ecmCollection.cfm">
```

In this example, the collection with an ID=2 is displayed, and the function “ecmTeaser” defines how to display the collection data on a Web page. Here is how the collection would appear on a Web page.

As stated earlier, the format of the collection on the Web page depends on the displayFunction that is used. Below is the source code for the display function “ecmTeaser”.

```
<cfparam name="Attributes.nID" default="">

<cfparam name="Attributes.href" default="">

<cfparam name="Attributes.ShowSummary" default="">

<cfparam name="Attributes.StyleInfo" default="">

<cfparam name="Attributes.target" default="">

<cfset ErrorString = "">

<cfobject class="CMS400.NETContentBrv2.CMS400.NETContent" Action="Create" name="tObj">
```

```
<cfset cTeasers = tObj.GetEcmCollectionItems(request.ecm.AppConfStr, Attributes.nId, 0,
request.ecm.cmsPreview, request.ecm.user_id, request.ecm.site_id, ErrorString)>

<cfset tString = "<table border=""0"" cellspacing=""0"" cellpadding=""0"" width=""100%"">">

<cfif cTeasers.Count()>

    <cfloop collection=#cTeasers# Item=cT>

        <cfset tString = tString & "<tr><td>">

        <cfset tString = tString & "<a href=""">

        <cfif len(Attributes.href)>

            <cfif FindNoCase("?", Attributes.hrefs)>

                <cfset tString = tString & Attributes.href & "&id=" &

                cT.Item("ContentID")>

            <cfelse>

                <cfset tStringi = tString & Attributes.href & "?id=" &

                cT.Item("ContentID")>

            </cfif>

        <cfelse>

            <cfset tString = tString & cT.Item("ContentLinks")>

        </cfif>

        <cfset tString = tString & """ ">

        <cfif len(Attributes.target)>

            <cfset tString = tString & "target=""" & Attributes.target & """ ">

        </cfif>

        <cfset tString = tString & ">">

        <cfset tString = tString & cT.Item("ContentTitle") & "</a>" & " (" &

        DateFormat(cT.Item("DateModified"), "mm/dd/yyyy") & " " &

        TimeFormat(cT.Item("DateModified"), "h:mm:ss tt") & ")</td></tr><tr><td>" &

        cT.Item("ContentTeaser")>

        <cfset tString = tString & "</td></tr><tr><td> </td></tr>">

    </cfloop>
```

```
    <cfset tString = tString & "</table>">

        <cfmodule str="#tString#" nId="#Attributes.nId#"
template="#request.ecm.AppPath#ecmshowCollectionsContent.cfm">

</cfif>
```

As you can see, the ecmTeaser function is a simple ASP function that creates a table of the links that belong to the collection specified.

The ecmTeaser function also displays the:

- Content Title
- Date Modified
- Content Teaser

Collection Variables

See ”Collection Attributes” on page 435

DisplayFunction

The display function defines how collection data appears on a Web page. Using ColdFusion scripting, you can create your own display functions, or use the three included functions.

XML Collection Function

The XML collection function wraps XML around the parameters of a list of content that’s created as a collection in the Ektron CMS400.NET Workarea. Once created you can use the XML to display the collection information. The ecmCollectionXML function allows you to easily define how the collection data appears on the Web page.

Shown here is the format of the ecmCollection function.

```
<CFMODULE template="#request.ecm.AppPath#ecmCollectionXML.cfm"

            id=""

            getHtml=""

>
```

This function is basically the same as the XML Collection function in ASP. See "XML Collection Function" on page 438 for details.

The only difference between the Coldfusion function and the ASP one is the syntax for implementing it. The ColdFusion example is below.

```
<CFMODULE template="#request.ecm.AppPath#ecmCollectionXML.cfm" id="3" getHtml="0">
```

RSS Collection Function

The RSS collection function is used to create an XML list of links that were created as a collection in the Ektron CMS400.NET Workarea and display them in RSS Syndication.

Shown here is the format of the ecmRssCollection function.

```
<CFMODULE template="#request.ecm.AppPath#ecmRssCollection.cfm"

           id="2"

>
```

This function is basically the same as the RSS Collection function in ASP. See "RSS Collection Function" on page 436 for details.

The only difference between the Coldfusion function and the ASP one is the syntax for implementing it.

Event Calendar Function

The ecmEvtCalendar function is used to display an event calendar that has been created in the Ektron CMS400.NET Workarea.

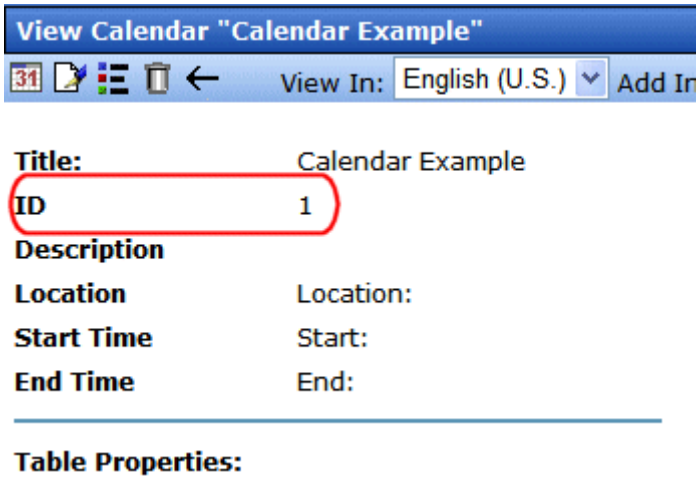
Displayed here is the format for the event calendar custom function:

```
<CFMODULE template="#request.ecm.AppPath#ecmEvtCalendar.cfm" Calendar_ID="" DisplayType="">
```

This function is basically the same as the Event Calendar function in ASP. See "Event Calendar Function" on page 445 for details.

The only differences between the Coldfusion function and the ASP one are the use of the DisplayType parameter and the syntax for

implementing it. The ecmEvtCalendar function parameters are defined below.

Attribute	Description
id	The **"id"** argument represents the ID number of the event calendar in the Workarea. 

Attribute	Description
DisplayType	Specify the Calendar display type. • **Month** - A 30 day calendar month view. • **Day** - Single day view. • **Inline** - Display events that occur from the start date of the calendar through the end date. • **MonthUpWithEvent** - Displays a month calendar. Events appear to the right of the calendar when you click on a day.

Here is a sample of the Event Calendar syntax.

Event Calendar Sample

```
    <tr>

         <td>

             <CFModule template="#request.ecm.AppPath#ecmEvtCalendar.cfm" calendar_id="24"
DisplayType="monthupwithevent">

         </td>

    </tr>
```

Form Function

The ecmFormBlock function displays an HTML form content block. For more information about HTML forms, see the **Ektron CMS400.NET** Administrator manual chapter “Working with HTML Forms.”

IMPORTANT! If you create a template for an existing form content block, you must manually change its quicklink to point to the new template. This change does not occur automatically. This procedure is described in the Ektron CMS400.NET Administrator manual chapter “Managing Library Assets”, section “Updating Default Template for Multiple Quicklinks.”

Here is the format for the form custom function

```
<CFMODULE template="#request.ecm.AppPath#ecmFormBlock.cfm" ID="" >
```

This function is basically the same as the Form function in ASP. See ”Form Function” on page 452 for details.

The only difference between the Coldfusion function and the ASP one is the syntax for implementing it. The ColdFusion example is below.

Here is an example of a form content block in an Ektron CMS400.NET template.

```
    <tr>

         <td>

                 <CFModule template="#request.ecm.AppPath#ecmFormBlock" form_id="1">
```

```
        </td>

    </tr>
```

MultiPurpose Function

The ecmMultiPurpose function displays either a content block or an content block associated with a form. As a result, you can use one template to display both types of content block.

Here is the format for the Multipurpose function

```
<CFMODULE template="#request.ecm.AppPath#ecmMultiPurpose.cfm"  DefaultID="" >
```

To pass in a standard content block, pass the id value in the url parameter, as illustrated below.

```
http://localhost/CMS400Example/index.asp?id=2
```

To pass in a Form content block, pass the ekfrm value in the url parameter, as illustrated below.

```
http://localhost/CMS400Example/index.asp?ekfrm=2
```

NOTE For more information about passing the id value as a url parameter, see ”Dynamic Content Block” on page 476.

If the id value or ekfrm value does not exist, a blank template appears.

If the url parameter is not passed in or recognized, the function calls ecmContentBlock and passes the DefaultID value as the id argument. If the DefaultID value is zero (0), ecmContentBlock is not called.

Metadata Function

The metadata tag is used to display the metadata for one or more content blocks in the Ektron CMS400.NET Web site. Inserting this custom tag will allow you to generate the list of metadata that is edited by the authors, and insert it in the source of the Web page.

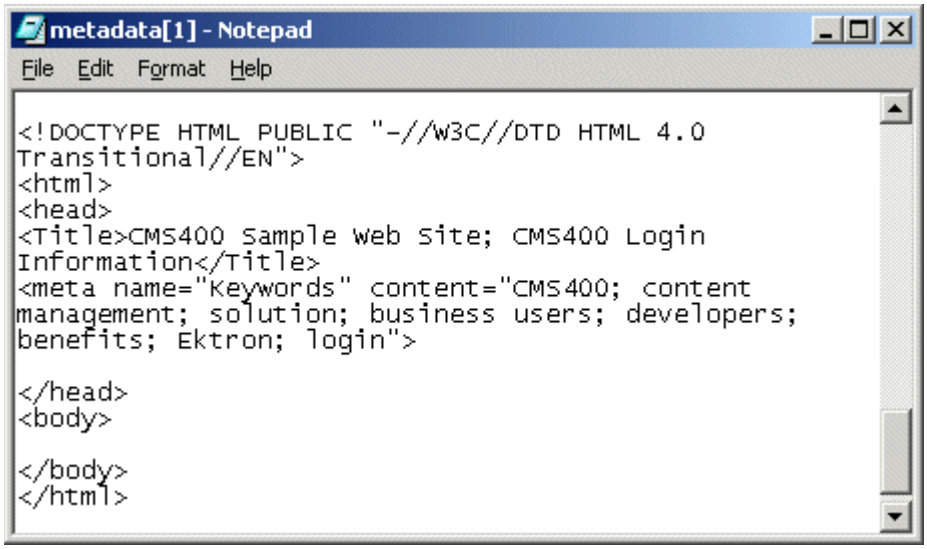
Shown below is an example of what the metadata custom tag looks like.

```
<cfmodule template="#request.ecm.AppPath#ecmMetadata.cfm" ItemList="" >
```

This function is basically the same as the Metadata function in ASP. See ”Metadata Function” on page 456 for details.

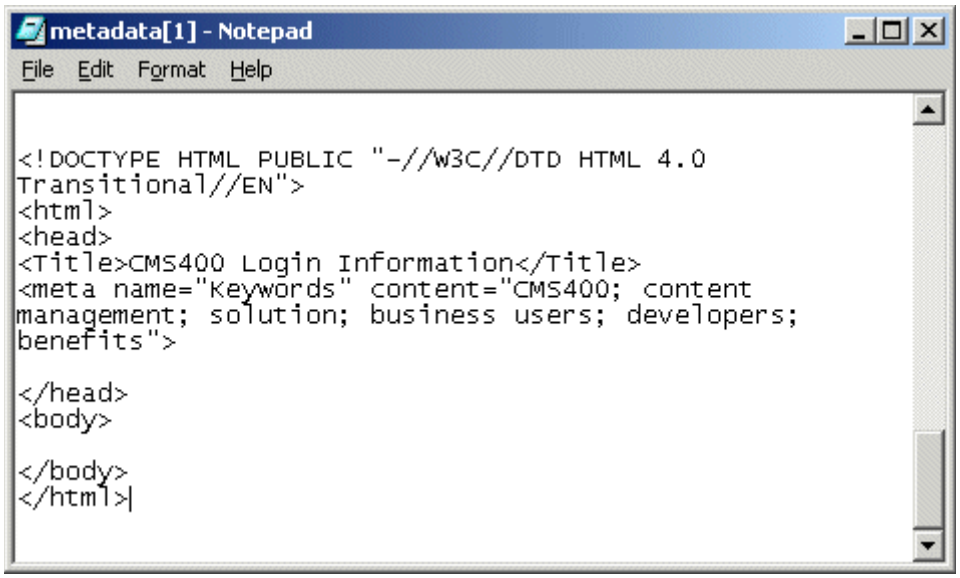
The only difference between the Coldfusion function and the ASP one is the syntax for implementing it. The ColdFusion examples are below.

```
<cfmodule template="#request.ecm.AppPath#ecmMetadata.cfm" ItemList="[1][2]">
```

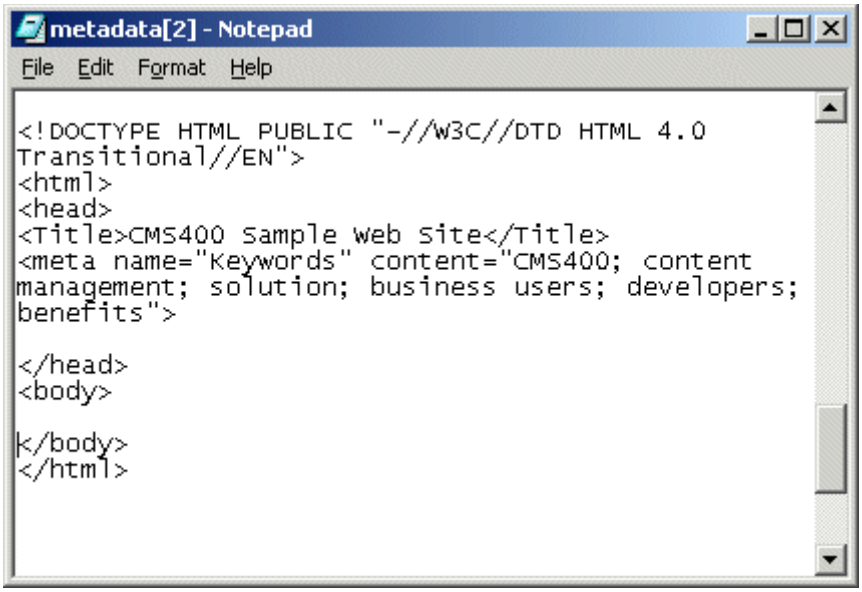



```
<cfmodule template="#request.ecm.AppPath#ecmMetadata.cfm"
```

```
ItemList="[1;title][2;keywords]>
```




```
<cfmodule template="#request.ecm.AppPath#ecmMetadata.cfm"

ItemList="[1][2;title,keywords]>
```



DHTML Menus

Ektron CMS400.NET's Menu feature lets users create and maintain a dropdown menu system for your Web site. The Menu feature lets you display menu options that link to content blocks, library assets, external hyperlinks and submenus.

To learn more about menus, please refer to the "Menu Feature" chapter of the **Ektron CMS400.NET** User Manual.

Additional Documentation in Ektron Knowledge Base

This chapter explains how to implement standard menus that ship with **Ektron CMS400.NET**. Ektron's Web site also provides Knowledge Base articles that explain how to extend the standard menu features.

Displaying a Menu on a Web Page

GenerateMenu and DropMenu

IMPORTANT! The menu functions described below are only one example of how to implement menus. You can implement your own menu system using JavaScript.

Use the `GenerateNextGenMenu` and `DropNextGenMenu` custom functions to display a DHTML menu on a Web page. Place `GenerateNextGenMenu` once on the page to provide an infrastructure for `DropNextGenMenu`. Then, place `DropNextGenMenu` as needed to implement individual menus.

Follow these steps to display a menu on a Web template.

1. Place the following code directly after the page's `<body>` tag.

```
<CFMODULE Ids="menu id number"
template="#request.ecm.AppPath#ecmDHTML_GenerateNextGenMenu.cfm">
```

For example:

```
<CFMODULE Ids="1" template="#request.ecm.AppPath#ecmDHTML_GenerateNextGenMenu.cfm">
```

To display more than one menu, separate each id number with a comma. For example:

```
<CFMODULE Ids="1,2,3" template="#request.ecm.AppPath#ecmDHTML_GenerateNextGenMenu.cfm">
```

2. If the user clicks *text* to display the menu, place the following code within the template file where the text would appear.

```
<CFMODULE MenuId="menu id number" Title="text user clicks to display menu"
template="#request.ecm.AppPath#ecmDHTML_DropNextGenMenu.cfm">
```

For example:

```
<CFMODULE MenuId="1" Title="Products"
template="#request.ecm.AppPath#ecmDHTML_DropNextGenMenu.cfm">
```

If the user clicks *a graphic* to invoke the menu, use this syntax:

```
<CFMODULE MenuId="menu id number" Title= '<img src=graphic file name>'
template="#request.ecm.AppPath#ecmDHTML_DropNextGenMenu.cfm">
```

For example:

```
<CFMODULE MenuId="1" Title= '<img src=mymenugraphic.jpg>'
template="#request.ecm.AppPath#ecmDHTML_DropNextGenMenu.cfm">
```

Customizing the Menu's Appearance

You can customize a menu's appearance, such as the font style, color, and size. So, you have the flexibility to coordinate a menu with the design your Web site.

To customize a menu in **Ektron CMS400.NET**, you can change the ek_Menu.js file.

ek_Menu.js

You can change the look and feel of a menu by changing the ek_menu.js file. If you used the default settings when installing **Ektron CMS400.NET**, this file is in `webroot/CMS400Example/Workarea/java/`. For a description of the ek_menu.js file, see "ek_Menu.js" on page 467.

XML Menu Function

The ecmGetMenuXML function wraps XML around the properties of a menu created in the Ektron CMS400.NET Workarea. Once created you can use the XML to display the menu. The

ecmGetMenuXML function allows you to easily define how the menu data appears on the Web page.

```
<CFMODULE id="" template="#request.ecm.AppPath#ecmGetMenuXML.cfm">
```

This function is basically the same as the XML Menu function in ASP. See "XML Menu" on page 470 for details.

The only difference between the Coldfusion function and the ASP one is the syntax for implementing it. The Coldfusion example is below.

```
<CFMODULE id="1" template="#request.ecm.AppPath#ecmGetMenuXML.cfm">
```

Language Select Function

This function lets a visitor to your Web site select a language in which to view the site. Here is what the function looks like when published on a Web page.

Shown here is the format of the ecmLanguageSelect function.

```
<CFMODULE template="#request.ecm.AppPath#ecmLanguageSelect.cfm">
```

This function is basically the same as the Show Site Language function in ASP. See "Show Site Language Function" on page 464 for details.

The only difference between the Coldfusion function and the ASP one is the syntax for implementing it.

Custom PHP Functions

Ektron CMS400.NET maintains and displays content via custom Ektron PHP functions and ComObjects. For the custom PHP functions to work properly, do the following:

- Make sure the `php_soap.dll` file is in the `c:\php` folder on the server where you installed PHP.
- Add the following line to the php.ini file.

  `extension=php_soap.dll`

  The php.ini file is located in `C:\Windows\php.ini`.
- In the `ApplicationAPI.php` file, set the WSDLpath. The `applicationAPI.php` file is located by default in:

  `http://localhost/CMS400Example/WorkArea/platforms/PHP/applicationAPI.php`

IMPORTANT! `ContentWS.asmx` is the web services (WSDL) file that the custom PHP functions use to talk to **Ektron CMS400.NET**. The WSDLpath located in the `applicationAPI.php` file must be set to location of the WSDL file. The default WSDLpath is `http://localhost/CMS400Example/WorkArea/WebServices/ContentWS.asmx.` If you have changed the location of `ContentWS.asmx` you need to update the WSDLpath. For example, if your site is `www.myektronsite.com`, the WSDLpath would be `http://www.myektronsite.com/WorkArea/WebServices/ContentWS.asmx.`

- Add the following line to the top of each template.

  `<?php include 'applicationAPI.php'; ?>`

  This file calls the Ektron custom functions file to allow the page to function properly. It *must* appear at the top of the template file.
- If you have upgraded your Ektron CMS400.NET from an Ektron CMS300 version, add the following line between the head tags for each template.

IMPORTANT! This only needs to be done if you are upgrading from CMS300 to CMS400.NET

```
<meta http-equiv="Content-type" content="text/html; charset=utf-8" />
```

To customize you Ektron CMS400.NET application, you need to be familiar with these functions.

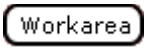
Function	Description	More Information
Calendar Functions		
Event Calendar	Displays a month view event calendar	”Event Calendar Function” on page 517
Collection Functions		
Collection	Displays links generated by collections	”Collection Function” on page 511
CollectionXML	Creates XML for a collection by wrapping tags around all of the collection items and their properties. This allows you to use an XSLT to define how the XML collection is displayed.	”XML Collection Function” on page 515
Random Content	Randomly displays the content of a collection item	”Random Content Function” on page 516
Random Summary	Randomly displays the summary of a collection item	”Random Summary Function” on page 517
RSS Collection	Creates an RSS feed out of the items in a collection.	”RSS Collection Function” on page 516
Content Functions		
Add Content	Allows a user to add content to **Ektron CMS400.NET** progammatically.	”Add Content” on page 505
Content Block	Displays and manages a content block	”Content Block Function” on page 504
Form	Displays a content block associated with a form	”Form Function” on page 520
Multipurpose	Displays either a content block or a form	”MultiPurpose Function” on page 521
List Summary Functions		

Function	Description	More Information
Archive ListSummary	Displays summaries of archived content blocks	”Archive List Summary Function” on page 509
List Summary	Displays a list of content block summaries	”List Summary Function” on page 508
List Summary XML	Creates XML for a list summary by wrapping tags around all of the collection items and their properties. This allows you to use an XSLT to define how the XML collection is displayed.	”XML List Summary Function” on page 510
RSS List Summary	Creates an RSS feed out of the items in a list summary.	”RSS List Summary Function” on page 511
Single Summary	Displays the summary for a content block	”Single Summary Function” on page 509
Login Functions		
Login	Adds the login or logout button	”Login Function” on page 503
Menu Functions		
DropHTMLMenu	Used to display a DHTML menu	”DHTML Menus” on page 524
GenerateMenu	Works with DropHTML menu to display DHTML menus	”DHTML Menus” on page 524
XML Menu	Creates XML for a menu by wrapping tags around all of the menu items properties.	”XML Menu Function” on page 526
Metadata Functions		
Metadata	Displays the metadata for the current Web page	”Metadata Function” on page 522
Search Functions		
Search	Inserts a text box with a search button	”Search Function” on page 506

Function	Description	More Information
Search Display	Creates an area to display search results	"Search Display Function" on page 507
Language Functions		
Language Select	Lets a site visitor select a language in which to view site.	"Language Select Function" on page 527

Login Function

This function is responsible for displaying the following buttons on the Web page.

Button	Name	Description
	Login	When user is not logged into Ektron CMS400.NET, this button appears. Click this button to let the user log in to the application.
	Logout	Once logged in, this button replaces the login button to allow the user to log out from the Ektron CMS400.NET Web site.
	Workarea	When logged in, this button appears under the logout button allowing the user to access their workarea.
	Preview On	Lets the user to preview the entire Web site as if all checked-in content was published.
	Preview Off	Turns off preview mode.

This function is basically the same as the Login function in ASP. See "Login Function" on page 373 for details. The only difference is the syntax used to implement the function. This is listed below.

Login Function Example

```
<tr>

    <td>

        <?php ecmlogin(); ?>

    </td>

</tr>
```

Content Block Function

The ecmContentBlock and ecmContentBlockEX functions are used to display a content block on an Ektron CMS400.NET Web page.

Shown here is the format for the ecmContentBlock function:

```
<?php ecmContentBlock (ID) ?>
```

Here is the format for the ecmContentBlockEX function:

```
<?php ecmContentBlockEx (ID, “XSLT”, Override) ?>
```

NOTE As an alternative, you can use the multipurpose function, which can display either a content block or a content block associated with a form. For more information, see ”MultiPurpose Function” on page 521.

This function is basically the same as the Content Block function in ASP. See ”Content Block Function” on page 375 for details.

The only differences between the PHP function and the ASP one are the examples of implementing it. The PHP examples are below.

Static Content Block

```
<tr>

   <td>

      <?php  ecmContentBlock(1) ?>

   </td>

</tr>
```

Dynamic Content Block

```
<tr>

   <td>

      <?php  ecmContentBlock($id) ?>

   </td>

</tr>
```

XML Content Block

```
<tr>

    <td>

       <?php ecmContentBlockEx (13, “”, 1) ?>

    </td>

</tr>
```

Add Content

The add content function allows you to programmatically add a new content block to Ektron CMS400.NET without the need of logging in.

Here is the format for the add content function:

```
<? AddContent        (“Username”
                     “Password”,
                     “Domain”,
                     ‘content_title”,
                     “content_comment”,
                     “ContentHtml”,
                     “SummaryHTML”,
                     “ContentLanguage”,
                     “FolderID”,
                     “GoLive”,
                     “EndDate”,
                     “MetaInfo”);            ?>
```

This function is basically the same as the Add Content function in ASP. See ”Add Content” on page 385 for details.

The only difference between the PHP function and the ASP one is the syntax for implementing it. The PHP example is below.

Add Content Example

The following is an example of the add content function in a template:.

```
<tr>

    <td>

        <? AddContent ("admin", "admin", "", "Content Title", "Comment",
"This is the <b>body</b> of the content.", "This is the <i>summary</i>
for the content.", "1033", "0", "", "", ""); ?>

    </td>

</tr>
```

Search Function

The search function is used to display a text box with a search button on the template. When seen on the template on the Web page, a user can enter text and click the search button to search through all the content on the Web site.

Ektron has created two search functions:

- ecmSearch
- ecmSearchNoFormTag

Both functions have the same parameters and perform the same search. The only difference is ecmSearch automatically inserts form tags, ecmSearchNoFormTag does not.

The following is an example of form tags.

```
<form name="ecmsearch" method="post" action='search-
display.asp'>

</form>
```

When using ecmSearch, form tags are added automatically. When using ecmSearchNoFormTag you need to add the tags manually

Here is the format for the search function. You can interchange ecmSearch and ecmSearchNoFormTag when using this function.

```
<?php ecmSearch( “StartingFolder”,

        Recursive,

        "targetpage",

        textbox,

        MaxCharacters,

        "ButtonImgSrc",

        "ButtonText",

        "FontFace",

        "FontColor",

        "FontSize",

        Horizontal,

        "Spare" );

?>
```

This function is basically the same as the Search function in ASP. See ”Search Functions” on page 389 for details.

The only difference between the PHP function and the ASP one is the syntax for implementing it. The PHP example is below.

Search Function Example

```
<TR>

 <TD>

  <?php ecmSearch("\\Marketing",1,"search-
  display.php",25,200,"","Search","Verdana","#808080","2",0,""); ?>

 </TD>

</TR>
```

Search Display Function

The search display custom function created the area where all the search results show after the search has been executed.

Here is the format of the ecmSearchDisplay function:

```
<?php ecmSearchDisplay( MaxNumbers,"StyleInfo", ShowDate ); ?>
```

This function is basically the same as the Search Display function in ASP. See "Search Display Function" on page 400 for details.

The only difference between the PHP function and the ASP one is the syntax for implementing it. The PHP example is below.

Search Display Function Example

```
<tr>
   <td>
      <?php ecmSearchDisplay( 0,"",1); ?>
   </td>
</tr>
```

List Summary Function

The list summary function is used to display the list of content summaries on a Web page. Here is the format of the list summary function.

```
<?php ecmListSummary ( "Folder",
        Recursive,
        ShowSummary,
        "StyleInfo",
        "OrderBy",
        "ShowInfo",
        MaxNumber,
        "Spare 1",
        "Spare2");
?>
```

This function is basically the same as the List Summary function in ASP. See "List Summary Function" on page 406 for details.

The only difference between the PHP function and the ASP one is the syntax for implementing it. The PHP example is below.

List Summary Example

```
<TR>

    <TD>

        <?PHP ecmListSummary ("\Marketing\News",0,1,"",

        "Title,asc","DateModified",0,"","",""); ?>

    </TD>

</TR>
```

Archive List Summary Function

The Archive list summary function displays on a Web page a list of content summaries for archived content blocks. For example, you might have a current news page and an archived news page. Use the List Summary Function to display current news, and this function to display older news stories.

On the Content Block **schedule** screen, the user must set an **End Date** then choose the Archive and Display option to have the content blocks appear via this function.

This function has the same parameters as the List Summary Function. See ”List Summary Function” on page 508 for details.

Single Summary Function

The single summary function is used to display the summary of one content block in the Ektron CMS400.NET Web site, instead of a list of all the summaries in a content folder. Seen below is the single summary function:

```
<?php ecmSingleSummary( “ContentID”,

        ShowSummary,

        “StyleInfo”,

        “ShowInfo”,
```

```
        "Spare1");

?>
```

This function is basically the same as the Single Summary function in ASP. See "Single Summary Function" on page 420 for details.

The only difference between the PHP function and the ASP one is the syntax for implementing it. The PHP example is below.

Single Summary Example

```
<TR>

   <TD>

       <?PHP ecmSingleSummary ("14",1,"","DateModified",""); ?>

   </TD>

</TR>
```

XML List Summary Function

The XML List Summary function wraps XML around the properties of a list of content, created as a list summary in the Ektron CMS400.NET Workarea. Once created, you can use the XML to display the list summary information. The ecmListSummaryXML function allows you to easily define how the list summary data appears on the Web page.

<?php ecmListSummaryXML ("Folder", Recursive, "OrderBy", MaxNumber, "ObjType", "SummaryType"); ?>

This function is basically the same as the List Summary XML function in ASP. See "XML List Summary Function" on page 414 for details.

The only difference between the PHP function and the ASP one is the syntax for implementing it. The PHP example is below.

```
<?php ecmListSummaryXML("\books\\",0,"Title,asc",0,"folder",""); ?>
```

RSS List Summary Function

The RSS summary function is used to create an XML list of content summaries for RSS Syndication. Shown here is the format of the RSS list summary function:

<?php ecmRssSummary (“Folder”, Recursive, “OrderBy”, MaxNumber); ?>

This function is basically the same as the RSS Summary function in ASP. See ”RSS Summary Function” on page 424 for details.

The only difference between the PHP function and the ASP one is the syntax for implementing it. The PHP example is below.

```
<?php  ecmRssSummary("\Books\\", 0, "Title,asc", 0); ?>
```

Collection Function

The collection function is used to display the list of links that were created as a collection in the Ektron CMS400.NET Workarea. The ecmCollection function is highly customizable, allowing you to easily define how the collection data will be displayed on the Web page.

Shown here is the format of the ecmCollection function.

```
<?PHP ecmCollection( “id”,

    “displayFunction”;

?>
```

This function is basically the same as the Collection function in ASP. See ”Collection Function” on page 430 for details.

The only difference between the PHP function and the ASP one is the syntax for implementing it. The PHP example is below.

Collection Examples

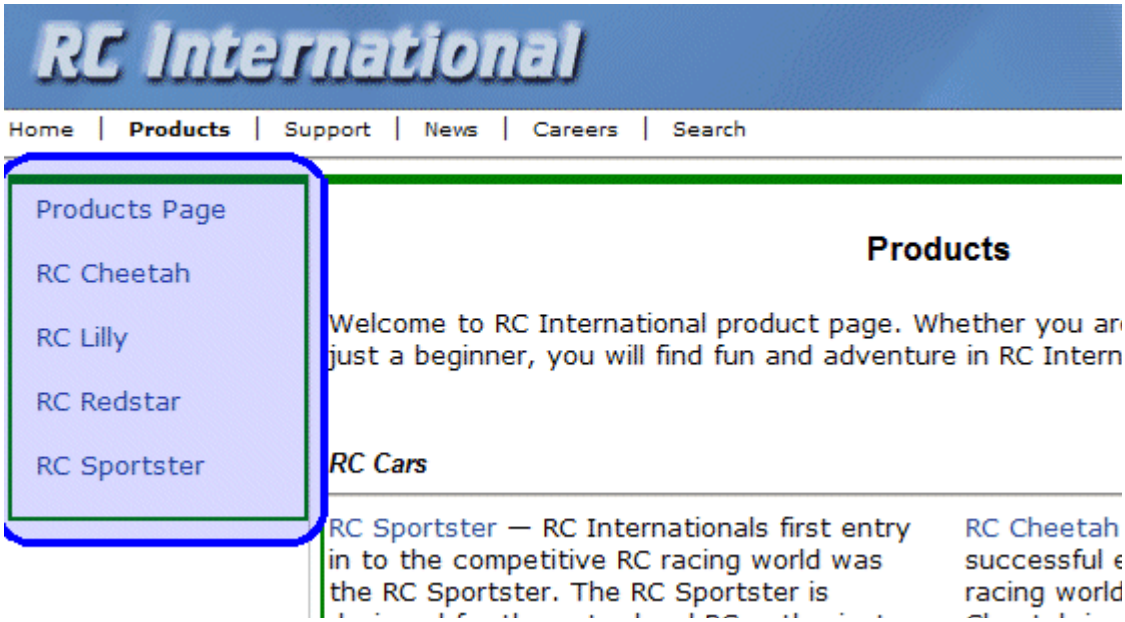
Example Using the ecmNavigation Display Function

The following is an example of a collection function being used as a navigation menu in the Ektron CMS400.NET sample Web site.

```
<?php ecmCollection(1, "ecmNavigation") ?>
```

In this example, the collection with an ID=1 is displayed, and the function “ecmNavigation” is used to define how the collection data is displayed on the Web page.

Here is how the collection would appear on the Web page.



The format of the collection on the Web page depends on the displayFunction that is used.

Below is the source code for the display function “ecmNavigation”.

```
Function ecmNavigation($cInfo){

    global $html, $info;

    $html = "<table border=\"0\" cellspacing=\"0\" cellpadding=\"0\"

        width=\"100%\">";

    $mycount = $cInfo->count();

    for ($iloop=0; $iloop < $mycount; ++$iloop) {

        $info = $cInfo->Item($iloop+1);

        $html = $html."<tr><td>  <a href=\"";

        $html = $html.$info->Item("ContentLinks");

        $html = $html."\">";

        $html = $html.$info->Item("ContentTitle");

        $html = $html."</a></td></tr><tr><td> </td></tr>";

    }

    $html = $html."</table>";

    return $html;

}
```

As you can see, the ecmNavigation function is a simple PHP function that creates a table of the links that belong to the collection specified.

The ecmNavigation function also displays the title of the content blocks by using the “info(“ContentTitle”)) building block.

Example Using the ecmTeaser Display Function

The following is an example of a collection function being used as a navigation menu in the Ektron CMS400.NET sample Web site.

```
<?php ecmCollection (2, "MyDisplayFunction"); ?>
```

In this example, the collection with an ID=2 is displayed, and the function “ecmTeaser” is used to define how the collection data is displayed on the Web page.

Shown here is how the collection would appear on the Web page.

As stated earlier, the format of the collection on the Web page depends on the displayFunction that is used.

Shown below is the source code for the display function "ecmTeaser"

```
Function ecmTeaser($cInfo){

    global $html, $info;

    $html = "<table border=\"0\" cellspacing=\"0\" cellpadding=\"0\" width=\"100%\">";

    $mycount = $cInfo->count();

    for ($iloop=0; $iloop < $mycount; ++$iloop) {

        $info = $cInfo->Item($iloop+1);

        $html = $html."<tr><td><a href=\"";

        $html = $html.$info->Item("ContentLinks");

        $html = $html."\" >";

        $html = $html.$info->Item("ContentTitle");

        $html = $html."</a> (";

        $html = $html.date("m/d/Y g:i:s A", $info->Item("DateModified"));

        $html = $html.")</td></tr><tr><td>";

        $html = $html.$info->Item("ContentTeaser");

        $html = $html."</td></tr><tr><td> </td></tr>";
```

```
    }

    $html = $html."</table>";

    return $html;

}
```

As you can see, the ecmTeaser function is a simple ASP function that creates a table of the links that belong to the collection specified.

The ecmTeaser function also displays the:

- Content Title
- Date Modified
- Content Teaser

Collection Variables

See ”Collection Attributes” on page 435

DisplayFunction

As explained earlier, the display function is a function that you create that will define how the collection data will be displayed on the Web page.

By using simple PHP scripting, you can create your own display functions, or use the three included functions.

XML Collection Function

The XML collection function wraps XML around the parameters of a list of content that’s created as a collection in the Ektron CMS400.NET Workarea. Once created you can use the XML to display the collection information. The ecmCollectionXML function allows you to easily define how the collection data appears on the Web page.

Shown here is the format of the ecmCollection function.

```
<?php ecmCollectionXML (id, GetHtml); ?>
```

This function is basically the same as the XML Collection function in ASP. See "XML Collection Function" on page 438 for details.

The only difference between the PHP function and the ASP one is the syntax for implementing it. The PHP example is below.

```
<?php ecmCollectionXML (3,0); ?>
```

RSS Collection Function

The RSS collection function is used to create an XML list of links that were created as a collection in the Ektron CMS400.NET Workarea and display them in RSS Syndication.

Shown here is the PHP format of the ecmRssCollection function.

```
<?php ecmRssCollection (id); ?>
```

This function is basically the same as the RSS Collection function in ASP. See "RSS Collection Function" on page 436 for details.

The only difference between the PHP function and the ASP one is the syntax for implementing it.

Random Content Function

The random content function is used to randomly display the content of a content block that belongs in a specified collection.

Displayed here is the format for the random content function

```
<?php ShowRandomContent (id); ?>
```

This function is basically the same as the Random Content function in ASP. See "Random Content" on page 443 for details.

Random Summary Function

SImilar to the random content function, the random summary function is used to randomly display the summary of a content block that belongs in a specified collection.

Displayed here is the format for the random content function

```
<?php ShowRandomSummary (id, XsltID); ?>
```

This function is basically the same as the Random Summary function in ASP. See ”Random Summary” on page 444 for details.

Event Calendar Function

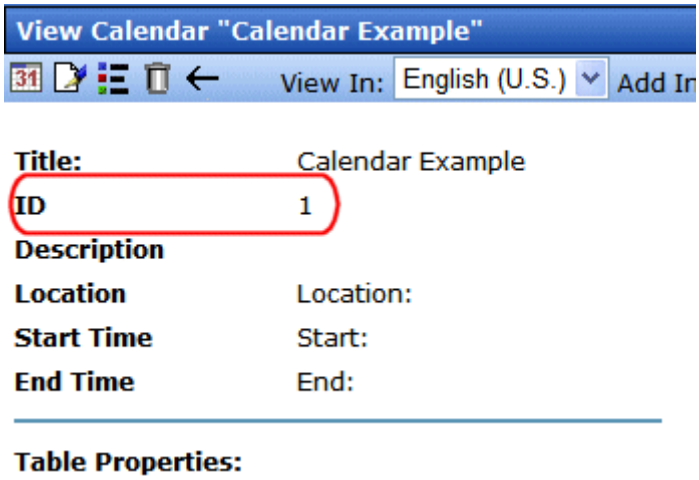
The ecmEvtCalendar function is used to display an event calendar that has been created in the Ektron CMS400.NET Workarea.

```
<?php ecmEvtCalendar(id, “Displaytype”); ?>
```

This function is basically the same as the Event Calendar function in ASP. See ”Event Calendar Function” on page 445 for details.

The only differences between the PHP function and the ASP one are the use of the DisplayType parameter and the syntax for

implementing it. The ecmEvtCalendar function parameters are defined below.

Attribute	Description
id	The **"id"** argument represents the ID number of the event calendar in the Workarea. 

Attribute	Description
DisplayType	Specify the Calendar display type. • **Month** - A 30 day calendar month view. • **Day** - Single day view. • **Inline** - Display events that occur from the start date of the calendar through the end date. • **MonthUpWithEvent** - Displays a month calendar. Events appear to the right of the calendar when you click on a day.

Event Calendar Sample

Here is an example of the event Calendar in an Ektron CMS400.NET template.

```
<tr>
    <td>
        <?php ecmEvtCalendar(24, “MonthUpWithEvent”); ?>
    </td>
</tr>
```

Form Function

The ecmFormBlock and ecmFormBlockNoFormTag functions display an HTML form content block. Both functions have the same parameters and display the form the same way. The only difference is ecmFormBlock automatically inserts form tags, ecmFormBlockNoFormTag does not.

The following is an example of form tags.

```
<form name="myform" action="#" onsubmit="return
EkFmValidate(this);" method="post">

</form>
```

When using ecmFormBlock, form tags are added automatically. When using ecmFormBlockNoFormTag, form tags need to be added manually.

For more information about HTML forms, see the **Ektron CMS400.NET** Administrator manual chapter “Working with HTML Forms.”

IMPORTANT! If you create a template for an existing form content block, you must manually change its quicklink to point to the new template. This change does not occur automatically. This procedure is described in the Ektron CMS400.NET Administrator manual chapter “Managing Library Assets”, section “Updating Default Template for Multiple Quicklinks.”

Here is the format for the form custom function

<?php ecmFormBlock(id); ?>

<?php ecmFormBlockNoForTag(id); ?>

This function is basically the same as the Form function in ASP. See ”Form Function” on page 452 for details.

The only difference between the PHP function and the ASP one is the syntax for implementing it. The PHP example is below.

```
<tr>
      <td>
            <?php ecmFormBlock(1); ?>
      </td>
</tr>
```

NOTE As an alternative, you can use the multipurpose function, which can display either a content block or a content block associated with a form. For more information, see ”MultiPurpose Function” on page 521.

MultiPurpose Function

The ecmMultiPurpose function displays either a content block or a content block associated with a form. As a result, you can use one template to display both types of content block.

Here is the format for the Multipurpose function

<?php ecmMultiPurpose(ID); ?>

To pass in a standard content block, pass the id value in the url parameter, as illustrated below.

```
http://localhost/CMS400Example/index.asp?id=2
```

To pass in a Form content block, pass the ekfrm value in the url parameter, as illustrated below.

```
http://localhost/CMS400Example/index.asp?ekfrm=2
```

NOTE For more information about passing the id value as a url parameter, see "Dynamic Content Block" on page 505.

If the id value or ekfrm value does not exist, a blank template appears.

If the url parameter is not passed in or recognized, the function calls ecmContentBlock and passes the DefaultID value as the id argument. If the DefaultID value is zero (0), ecmContentBlock is not called.

Metadata Function

The metadata function is used to display the metadata for one or more content blocks in the Ektron CMS400.NET Web site. Inserting this custom function allows you to generate the list of metadata that is edited by the authors, and insert it in the source of the Web page. Shown below is an example of what the metadata custom function looks like.:

```
<?php ecmMetaData ("ItemList","Spare1";) ?>
```

This function is basically the same as the Metadata function in ASP. See "Metadata Function" on page 456 for details.

The only difference between the PHP function and the ASP one is the syntax for implementing it. The PHP examples are below.

To include both content blocks' metadata in the web page, an ecmMetadata call would look like this.

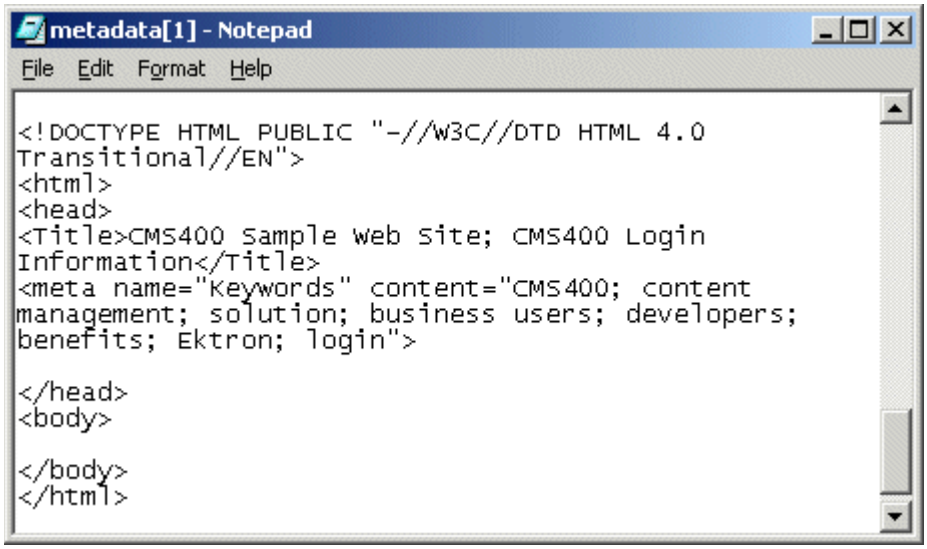
```
<html>

    <head>

        <?php ecmMetadata ("[1][2;Title,Content-Type]", ""); ?>

        (rest of head)

    </head>

    <body>

        (rest of web page document)
```

```
    </body>

</html>
```

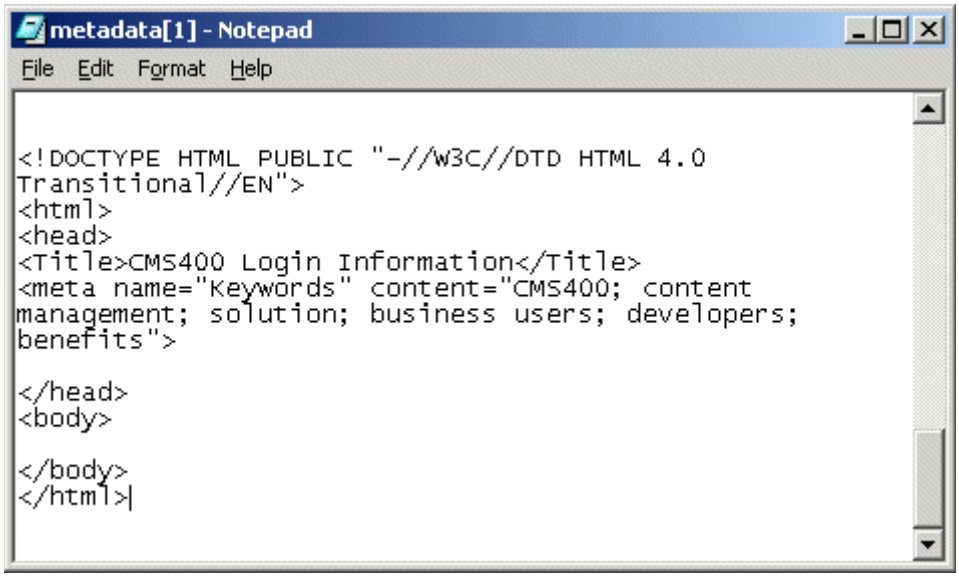
In the above example the ecmMetadata call is contained in the <head> of the HTML document (since all HTML metadata is required to exist there). Once the Web page is viewed the Web page source will contain the following information:

```
<html>

    <head>

        <Title>Welcome to Our Home Page</Title>

        <meta name=”Keywords” content= Home; page; company; Welcome; Announce;

        product; widget”>

        <meta http-equiv=”Content-Type” content=”text/html; charset=iso-8859-1”>

        (rest of head)

    </head>

    <body>

        (rest of web page document)

    </body>

</html>
```

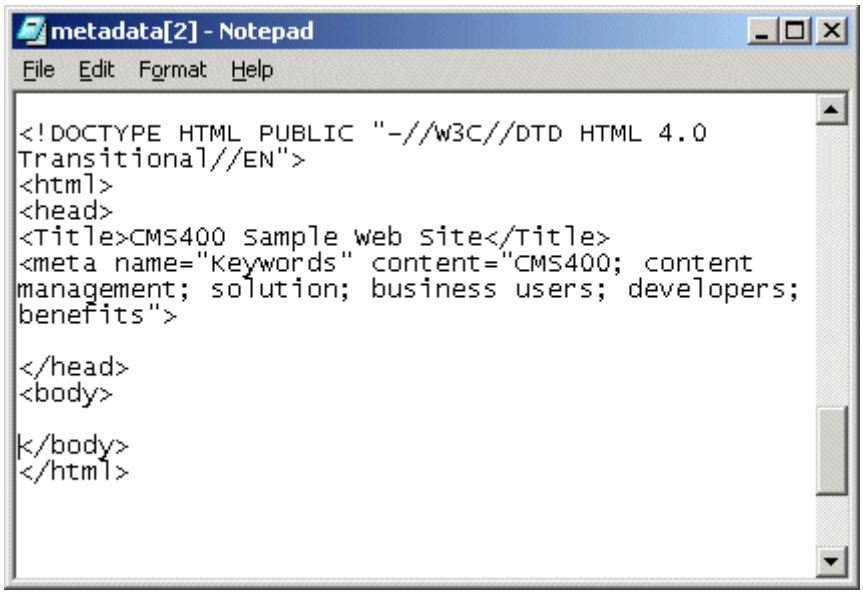
```
<?PHP ecmMetaData ("[1][2]","") ?>
```



```
<?PHP ecmMetaData ("[1;title][2;keywords]","") ?>
```




```
<?PHP ecmMetaData ("[1][2;title,keywords]","") ?>
```



DHTML Menus

Ektron CMS400.NET's Menu feature lets users create and maintain a dropdown menu system for your Web site. The Menu feature lets

you display menu options that link to content blocks, library assets, external hyperlinks and submenus.

To learn more about menus, please refer to the “Menu Feature” chapter of the **Ektron CMS400.NET** User Manual.

Additional Documentation in Ektron Knowledge Base

This chapter explains how to implement standard menus that ship with **Ektron CMS400.NET**. Ektron’s Web site also provides Knowledge Base articles that explain how to extend the standard menu features.

Displaying a Menu on a Web Page

GenerateMenu and DropMenu

IMPORTANT! The menu functions described below are only one example of how to implement menus. You can implement your own menu system using JavaScript.

Use the `GenerateNextGenMenu` and `DropNextGenMenu` custom functions to display a DHTML menu on a Web page. Place `GenerateNextGenMenu` once on the page to provide an infrastructure for `DropNextGenMenu`. Then, place `DropNextGenMenu` as needed to implement individual menus.

Follow these steps to display a menu on a Web template.

1. Place the following code directly after the page’s `<body>` tag.

```
<?php ecmDHTML_GenerateNextGenMenu("menu id number"); ?>
```

   For example:

```
<?php ecmDHTML_GenerateNextGenMenu("1"); ?>
```

   To display more than one menu, separate each id number with a comma. For example:

```
<?php ecmDHTML_GenerateNextGenMenu("1,2,3"); ?>
```

2. If the user clicks *text* to display the menu, place the following code within the template file where the text would appear.

```
<?php ecmDHTML_DropNextGenMenu menu ID number,"text user clicks to display menu"; ?>"
```

   For example:

```
<?php ecmDHTML_DropNextGenMenu 1,"MyMenu1"; ?>
```

If the user clicks *a graphic* to invoke the menu, use this syntax:

```
<?php ecmDHTML_DropNextGenMenu menu ID number, '<img src=graphic file name>'; ?>
```

For example:

```
<?php ecmDHTML_DropNextGenMenu 1,'<img src=mymenugraphic.jpg>'; ?>
```

Customizing the Menu's Appearance

You can customize a menu's appearance, such as the font style, color, and size. So, you have the flexibility to coordinate a menu with the design your Web site.

To customize a menu in **Ektron CMS400.NET**, you can change the ek_Menu.js file.

ek_Menu.js

You can change the look and feel of a menu by changing the ek_menu.js file. If you used the default settings when installing **Ektron CMS400.NET**, this file is in *webroot*/CMS400Example/Workarea/java/. For a description of the ek_menu.js file, see "ek_Menu.js" on page 467.

XML Menu Function

The ecmGetMenuXML function wraps XML around the properties of a menu created in the Ektron CMS400.NET Workarea. Once created you can use the XML to display the menu. The ecmGetMenuXML function allows you to easily define how the menu data appears on the Web page.

```
<?php ecmGetMenuXML($Id); ?>
```

This function is basically the same as the XML Menu function in ASP. See "XML Menu" on page 470 for details.

The only difference between the PHP function and the ASP one is the syntax for implementing it. The PHP example is below.

```
<?php ecmGetMenuXML(1); ?>
```

Language Select Function

This function lets a visitor to your Web site select a language in which to view the site. Here is what the function looks like when published on a Web page.

Shown here is the format of the ecmLanguageSelect function.

```
<?php ecmLanguageSelect(); ?>
```

This function is basically the same as the Show Site Language function in ASP. See "Show Site Language Function" on page 464 for details.

The only difference between the PHP function and the ASP one is the syntax for implementing it.

Custom API

As you have seen, the formatting for the custom functions used in Ektron CMS400.NET for PHP are defined in the file ApplicationAPI.php. As a developer, you may make changes to this file to further enhance how the custom functions appear on your Web site.

However, Ektron recommends creating a copy of the file, and renaming it to an arbitrary name, for example, MyCustomAPI.php, and including that file when creating Ektron CMS400.NET templates. This ensures that your custom API will not be overwritten if you upgrade to another version of Ektron CMS400.NET.

Custom JSP Functions

Ektron CMS400.NET maintains and displays content via custom Ektron JSP functions. For the custom JSP functions to work properly, do the following:

- In the `applicationapi.jsp` file, set the WSSoap_address. The `applicationapi.jsp` file is located by default in:

  `http://localhost/CMS400Example/WorkArea/platforms/JSP/applicationapi.jsp`

IMPORTANT! `ContentWS.asmx` is the web services file that the custom JSP functions use to talk to **Ektron CMS400.NET**. The WSSoap_address located in the `applicationapi.jsp` file must be set to location of the Web services file. The default WSSoap_address is `http://localhost/CMS400Example/WorkArea/WebServices/ContentWS.asmx.` If you have changed the location of `ContentWS.asmx` you need to update the WSSoap_address. For example, if your site is `www.myektronsite.com`, the WSSoap_address would be `http://www.myektronsite.com/WorkArea/WebServices/ContentWS.asmx.`

- Add the following line to the top of each template.

  ```
  <%@ include file="applicationapi.jsp" %>
  ```

  This file calls the Ektron custom functions file to allow the page to function properly. It *must* appear at the top of the template file.

- Make sure the `contentws.jar` file is included in your JSP environment

- Ektron recommends running Axis when using a JSP environment. We have tested on TomCat and JRUN with Axis v1.2.1. You can download Axis from http://ws.apache.org/axis/java/.

To customize you Ektron CMS400.NET application, you need to be familiar with these functions.

Function	Description	More Information
Calendar Functions		
Calendar	Displays a calendar with different display types. For example, a daily view calendar. For more information on the types of calendars that can be dsiplayed, see "DisplayType" on page 449	"Calendar Function" on page 532
Event Calendar	Displays a month view event calendar	"Event Calendar Function" on page 533
Collection Functions		
Collection	Displays links generated by collections	"Collection Function" on page 533
CollectionXML	Creates XML for a collection by wrapping tags around all of the collection items and their properties. This allows you to use an XSLT to define how the XML collection is displayed.	"XML Collection Function" on page 534
Random Content	Randomly displays the content of a collection item	"Random Content Function" on page 535
Random Summary	Randomly displays the summary of a collection item	"Random Summary Function" on page 535
RSS Collection	Creates an RSS feed out of the items in a collection.	"RSS Collection Function" on page 536
Content Functions		
Add Content	Allows a user to add content to **Ektron CMS400.NET** progammatically.	"Add Content" on page 536
Content Block	Displays and manages a content block	"Content Block Function" on page 537
Form	Displays a content block associated with a form	"Form Function" on page 538
Language Functions		

Function	Description	More Information
Language Select	Lets a site visitor select a language in which to view site.	”Language Select Function” on page 540
Login Functions		
Login	Adds the login or logout button	”Login Function” on page 540
Menu Functions		
DropHTMLMenu	Used to display a DHTML menu	”DHTML Menus” on page 541
GenerateMenu	Works with DropHTML menu to display DHTML menus.	”DHTML Menus” on page 541
XML Menu	Creates XML for a menu by wrapping tags around all of the menu items properties.	”XML Menu Function” on page 543
Metadata Functions		
Metadata	Displays the metadata for the current Web page.	”Metadata Function” on page 544
Search Functions		
Search	Inserts a text box with a search button.	”Search Function” on page 546
Search Display	Creates an area to display search results.	”Search Display Function” on page 548
Summary Functions		
List Summary	Displays a list of content block summaries.	”List Summary Function” on page 548
List Summary XML	Creates XML for a list summary by wrapping tags around all of the collection items and their properties. This allows you to use an XSLT to define how the XML collection is displayed.	”XML List Summary Function” on page 549
RSS List Summary	Creates an RSS feed out of the items in a list summary.	”RSS List Summary Function” on page 550

Function	Description	More Information
Single Summary	Displays the summary for a content block.	”Single Summary Function” on page 550

Calendar Function

The ecmCalendar function displays a calendar that has been created in Ektron CMS400.NET. Here is the format for the event calendar custom function.

```
<%= ecmCalendar (int calendarID, String “displayType”) %>
```

NOTE While you can display a month view calendar with ecmCalendar, you can also use ecmEvtCalendar to display a month view calendar. For more Information, see ”Event Calendar Function” on page 533

This function is basically the same as the Calendar function in ASP. See ”Calendar Function” on page 448 for details.

The only difference between the JSP function and the ASP one is the syntax for implementing it.

Calendar Sample

Here is an example of the Calendar function in an Ektron CMS400.NET template.

```
<tr>
    <td>
        <%= ecmEvtCalendar(24, “MonthUpWithEvent”) %>
    </td>
</tr>
```

Event Calendar Function

The ecmEvtCalendar function is used to display an event calendar that has been created in the Ektron CMS400.NET Workarea.

```
<%= ecmEvtCalendar(int calendar id) %>
```

This function is basically the same as the Event Calendar function in ASP. See "Event Calendar Function" on page 445 for details.

The only difference between the JSP function and the ASP one is the syntax for implementing it.

Event Calendar Sample

Here is an example of the event Calendar in an Ektron CMS400.NET template.

```
<tr>
    <td>
        <%= ecmEvtCalendar(24) %>
    </td>
</tr>
```

Collection Function

The collection function is used to display the list of links that were created as a collection in the Ektron CMS400.NET Workarea. The ecmCollection function is highly customizable, allowing you to easily define how the collection data will be displayed on the Web page.

Shown here is the format of the ecmCollection function.

```
<%= ecmCollection(int id, String "displayFunction") %>
```

This function is basically the same as the Collection function in ASP. See ”Collection Function” on page 430 for details.

The only difference between the JSP function and the ASP one is the syntax for implementing it.

Collection Sample

Here is an example of the Collection function in an Ektron CMS400.NET template.

```
<tr>
    <td>
        <%= ecmCollection(24, “ecmTeaser”) %>
    </td>
</tr>
```

XML Collection Function

The XML collection function wraps XML around the parameters of a list of content that’s created as a collection in the Ektron CMS400.NET Workarea. Once created you can use the XML to display the collection information. The ecmCollectionXML function allows you to easily define how the collection data appears on the Web page. Shown here is the format of the ecmCollection function.

```
<%= ecmCollectionXML (int id, boolean GetHtml) %>
```

This function is basically the same as the XML Collection function in ASP. See ”XML Collection Function” on page 438 for details.

The only difference between the JSP function and the ASP one is the syntax for implementing it. The JSP example is below.

```
<tr>
    <td>
        <%= ecmCollectionXML(24, True) %>
    </td>
</tr>
```

Random Content Function

The random content function is used to randomly display the content of a content block that belongs in a specified collection.

Displayed here is the format for the random content function

```
<%= ecmShowRandomContent (int collectionID, int xsltId) %>
```

This function is basically the same as the Random Content function in ASP. See ”Random Content” on page 443 for details.

Random Summary Function

SImilar to the random content function, the random summary function is used to randomly display the summary of a content block that belongs in a specified collection.

Displayed here is the format for the random content function

```
<%= ShowRandomSummary (int collectionID) %>
```

This function is basically the same as the Random Summary function in ASP. See ”Random Summary” on page 444 for details.

RSS Collection Function

The RSS collection function is used to create an XML list of links that were created as a collection in the Ektron CMS400.NET Workarea and display them in RSS Syndication.

Shown here is the JSP format of the ecmRssCollection function.

```
<%= ecmRssCollection (int id) %>
```

This function is basically the same as the RSS Collection function in ASP. See "RSS Collection Function" on page 436 for details.

Add Content

The add content function allows you to programmatically add a new content block to Ektron CMS400.NET without the need of logging in.

Here is the format for the add content function:

```
<%= ecmAddContent (String UserName, String Password, String Domain, String content_title,
String content_comment, String ContentHtml, String SummaryHTML, String ContentLanguage,
String FolderID, String GoLive, String EndDate, String MetaInfo) %>
```

This function is basically the same as the Add Content function in ASP. See "Add Content" on page 385 for details.

The only difference between the JSP function and the ASP one is the syntax for implementing it. The JSP example is below.

Add Content Example

The following is an example of the add content function in a template:.

```
<tr>

    <td>

        <%= ecmAddContent ("admin", "admin", "", "Content Title",
"Comment", "This is the <b>body</b> of the content.", "This is the
<i>summary</i> for the content.", "1033", "0", "", "", "") %>

    </td>

</tr>
```

Content Block Function

The ecmContentBlock and ecmContentBlockEX functions are used to display a content block on an Ektron CMS400.NET Web page.

Shown here is the format for the ecmContentBlock function:

```
<%= ecmContentBlock (int id) %>
```

Here is the format for the ecmContentBlockEX function:

```
<%= ecmContentBlockEX (int id, String xslt, int OverrideId) %>
```

These function is basically the same as the Content Block functions in ASP. See ”Content Block Function” on page 375 for details.

The only difference between the JSP function and the ASP one is the examples of implementing it. The JSP examples are below.

Static Content Block

```
<tr>

    <td>
```

```
        <%= ecmContentBlock(1) %>

    </td>

</tr>
```

Dynamic Content Block

```
<tr>

    <td>

        <%= ecmContentBlock($id) %>

    </td>

</tr>
```

XML Content Block

```
<tr>

     <td>

         <%= ecmContentBlockEx (13, “”, 1) %>

     </td>

</tr>
```

Form Function

The ecmFormBlock function displays an HTML form content block. This function has a parameter that lets you decide whether to have form tags added automatically or manually. The following is an example of form tags.

```
<form name="myform" action="#" onsubmit="return
EkFmValidate(this);" method="post">

</form>
```

For more information about HTML forms, see the **Ektron CMS400.NET** Administrator manual chapter “Working with HTML Forms.”

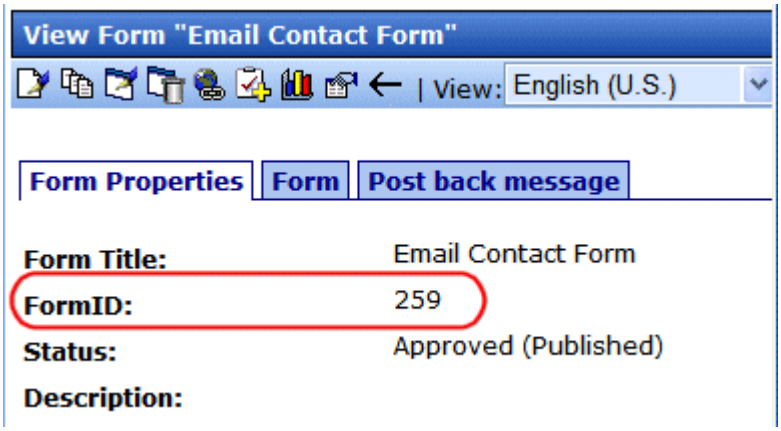
IMPORTANT! If you create a template for an existing form content block, you must manually change its quicklink to point to the new template. This change does not occur automatically. This procedure is described in the Ektron CMS400.NET

Administrator manual chapter “Managing Library Assets”, section “Updating Default Template for Multiple Quicklinks.”

Here is the format for the ecmFormBlock function:

```
<%= ecmFormBlock (int id, boolean formTagReqd) %>
```

The ecmFormBlock function attributes are explained below.

Attribute	Description
id	The “id” argument represents the form’s ID number, visible when viewing the form from the View Form screen. 

formTagReqd	Allows form tags to be added automatically.
	This is an example of form tags.
	`<form name="myform" action="#" onsubmit="return EkFmValidate(this);" method="post">`
	`</form>`
	true = Add form tags automatically
	false = Add form tags manually

This function is basically the same as the Form function in ASP. See ”Form Function” on page 452 for details.

The only differences between the JSP function and the ASP one are the formTagReqd parameter and the syntax for implementing it. The JSP example is below.

```
<tr>
```

```
        <td>
            <%= ecmFormBlock(1, true) %>
        </td>
    </tr>
```

Language Select Function

This function lets a visitor to your Web site select a language in which to view the site. Here is what the function looks like when published on a Web page.

Shown here is the format of the ecmLanguageSelect function.

```
<%= ecmLanguageSelect() %>
```

This function is basically the same as the Show Site Language function in ASP. See "Show Site Language Function" on page 464 for details.

The only difference between the JSP function and the ASP one is the syntax for implementing it.

Login Function

This function is responsible for displaying the following buttons on the Web page.

Button	Name	Description
Login	Login	When user is not logged into Ektron CMS400.NET, this button appears. Click this button to let the user log in to the application.

Button	Name	Description
Logout	Logout	Once logged in, this button replaces the login button to allow the user to log out from the Ektron CMS400.NET Web site.
Workarea	Workarea	When logged in, this button appears under the logout button allowing the user to access their workarea.
Preview	Preview On	Lets the user to preview the entire Web site as if all checked-in content was published.
Preview	Preview Off	Turns off preview mode.

Shown here is the format of the ecmLogin function.

<%= ecmlogin() %>

This function is basically the same as the Login function in ASP. See "Login Function" on page 373 for details. The only difference is the syntax used to implement the function. This is listed below.

Login Function Example

```
<tr>

    <td>

        <%= ecmlogin() %>

    </td>

</tr>
```

DHTML Menus

Ektron CMS400.NET's Menu feature lets users create and maintain a dropdown menu system for your Web site. The Menu feature lets

you display menu options that link to content blocks, library assets, external hyperlinks and submenus.

To learn more about menus, please refer to the “Menu Feature” chapter of the **Ektron CMS400.NET** User Manual.

Additional Documentation in Ektron Knowledge Base

This chapter explains how to implement standard menus that ship with **Ektron CMS400.NET**. Ektron’s Web site also provides Knowledge Base articles that explain how to extend the standard menu features.

Displaying a Menu on a Web Page

GenerateMenu and DropMenu

IMPORTANT! The menu functions described below are only one example of how to implement menus. You can implement your own menu system using JavaScript.

Use the `GenerateNextGenMenu` and `DropNextGenMenu` custom functions to display a DHTML menu on a Web page. Place `GenerateNextGenMenu` once on the page to provide an infrastructure for `DropNextGenMenu`. Then, place `DropNextGenMenu` as needed to implement individual menus.

Follow these steps to display a menu on a Web template.

1. Place the following code directly after the page’s `<body>` tag.

```
<%= ecmDHTML_GenerateNextGenMenu("menu id number") %=>
```

   For example:

```
<%= ecmDHTML_GenerateNextGenMenu("1") %>
```

   To display more than one menu, separate each id number with a comma. For example:

```
<%= ecmDHTML_GenerateNextGenMenu("1,2,3") %>
```

2. If the user clicks *text* to display the menu, place the following code within the template file where the text would appear.

```
<%= ecmDHTML_DropNextGenMenu menu ID number,"text user clicks to display menu" %>"
```

   For example:

```
<%= ecmDHTML_DropNextGenMenu 1,"MyMenu1" %>
```

If the user clicks *a graphic* to invoke the menu, use this syntax:

```
<%= ecmDHTML_DropNextGenMenu menu ID number, '<img src=graphic file name>' %>
```

For example:

```
<%= ecmDHTML_DropNextGenMenu 1,'<img src=mymenugraphic.jpg>' %>
```

Customizing the Menu's Appearance

You can customize a menu's appearance, such as the font style, color, and size. So, you have the flexibility to coordinate a menu with the design your Web site.

To customize a menu in **Ektron CMS400.NET**, you can change the ek_Menu.js file.

ek_Menu.js

You can change the look and feel of a menu by changing the ek_menu.js file. If you used the default settings when installing **Ektron CMS400.NET**, this file is in `webroot/CMS400Example/Workarea/java/`. For a description of the ek_menu.js file, see "ek_Menu.js" on page 467.

XML Menu Function

The ecmGetMenuXML function wraps XML around the properties of a menu created in the Ektron CMS400.NET Workarea. Once created you can use the XML to display the menu. The ecmGetMenuXML function allows you to easily define how the menu data appears on the Web page.

```
<%= ecmGetMenuXML(int id) %>
```

This function is basically the same as the XML Menu function in ASP. See "XML Menu" on page 470 for details.

The only difference between the JSP function and the ASP one is the syntax for implementing it. The JSP example is below.

```
<%= ecmGetMenuXML(1) %>
```

Metadata Function

The metadata function is used to display the metadata for one or more content blocks in the Ektron CMS400.NET Web site. Inserting this custom function allows you to generate the list of metadata that is edited by the authors, and insert it in the source of the Web page. Shown below is an example of what the metadata custom function looks like:

```
<%= ecmMetadata(String itemlist, java.lang.object spare1) %>
```

This function is basically the same as the Metadata function in ASP. See "Metadata Function" on page 456 for details.

The only difference between the JSP function and the ASP one is the syntax for implementing it. The JSP examples are below.

To include both content blocks' metadata in the web page, an ecmMetadata call would look like this.

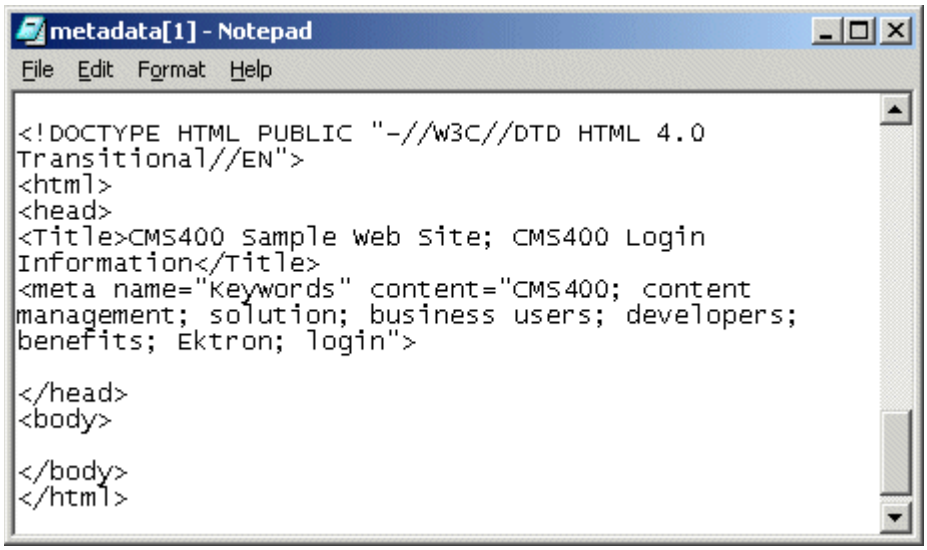
```
<html>

    <head>

        <%= ecmMetadata ("[1][2;Title,Content-Type]", "") %>

        (rest of head)

    </head>

    <body>

        (rest of web page document)

    </body>

</html>
```

In the above example the ecmMetadata call is contained in the <head> of the HTML document (since all HTML metadata is required to exist there). Once the Web page is viewed the Web page source will contain the following information:

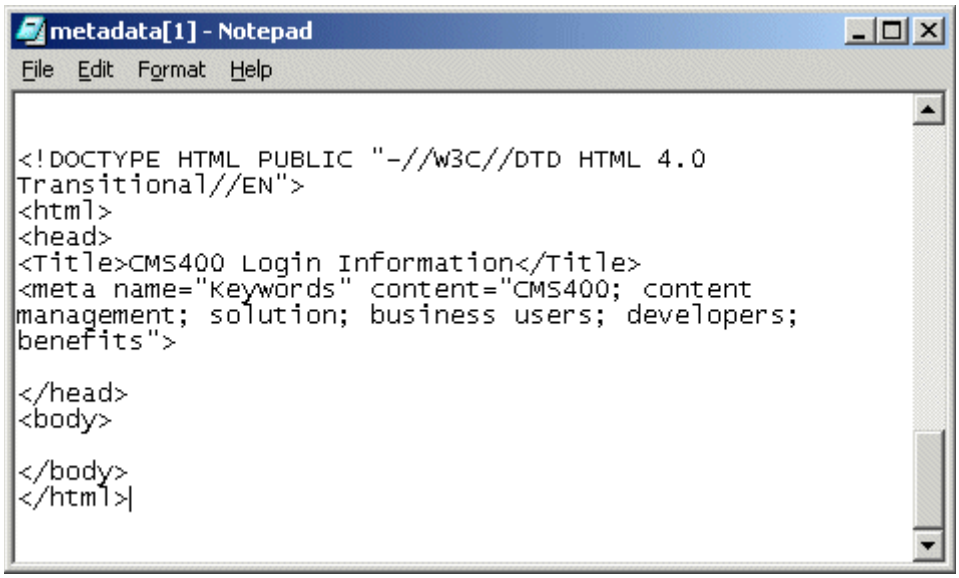
```
<html>

    <head>

        <Title>Welcome to Our Home Page</Title>
```

```
        <meta name="Keywords" content= Home; page; company; Welcome; Announce;
        product; widget">
        <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
        (rest of head)
    </head>
    <body>
        (rest of web page document)
    </body>
</html>
```

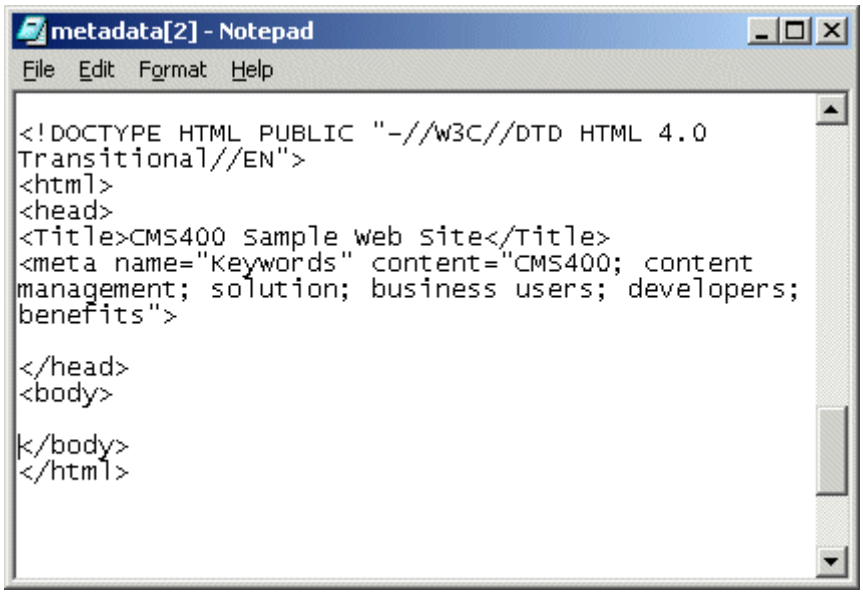
```
<%= ecmMetaData ("[1][2]","") %>
```



```
<%= ecmMetaData ("[1;title][2;keywords]","") %>
```




```
<%= ecmMetaData ("[1][2;title,keywords]","") %>
```



Search Function

The search function is used to display a text box with a search button on the template. When seen on the template on the Web

page, a user can enter text and click the search button to search through all the content on the Web site.

Ektron has created two search functions:

- ecmSearch
- ecmSearchNoFormTag

Both functions have the same parameters and perform the same search. The only difference is ecmSearch automatically inserts form tags, ecmSearchNoFormTag does not.

The following is an example of form tags.

```
<form name="ecmsearch" method="post" action='search-
display.asp'>

</form>
```

When using ecmSearch, form tags are added automatically. When using ecmSearchNoFormTag you need to add the tags manually

Here is the format for the search function. You can interchange ecmSearch and ecmSearchNoFormTag when using this function.

<%= ecmSearch(java.lang.String startingFolder, boolean recursive, java.lang.String target, int textBoxSize, int maxCharacters, java.lang.String buttonImageScr, java.lang.String buttonText, java.lang.String fontFace, java.lang.String fontColor, java.lang.String fontSize, boolean horizontal, java.lang.Object spare1) %>

This function is basically the same as the Search function in ASP. See ”Search Functions” on page 389 for details.

The only difference between the JSP function and the ASP one is the syntax for implementing it. The JSP example is below.

Search Function Example

```
<TR>

 <TD>

  <%= ecmSearch("\\",true,"search-
display.jsp",25,200,"","Search","Verdana","#808080","2",false,"") %>

 </TD>
```

```
</TR>
```

Search Display Function

The search display custom function creates the area where all the search results are shown after the search has been executed.

Here is the format of the ecmSearchDisplay function:

```
<%= ecmSearchDisplay(int maxNumber, String styleInfo, boolean ShowDate,
boolean showSummary) %>
```

This function is basically the same as the Search Display function in ASP. See "Search Display Function" on page 400 for details.

The only difference between the JSP function and the ASP one is the syntax for implementing it. The JSP example is below.

Search Display Function Example

```
<tr>

   <td>

      <%= ecmSearchDisplay(0,"",true,true) %>

   </td>

</tr>
```

List Summary Function

The list summary function is used to display a list of content summaries on a Web page. Here is the format of the list summary function.

```
<%= ecmListSummary (String folder, boolean recursive, boolean showSummary,
String styleInfo, String OrderBy, String showInfo, int maxNumber, String objType,
String optionList, String summaryType) %>
```

This function is basically the same as the List Summary function in ASP. See "List Summary Function" on page 406 for details.

The only difference between the JSP function and the ASP one is the syntax for implementing it. The JSP example is below.

List Summary Example

```
<TR>

    <TD>

        <%= ecmListSummary ("\Marketing\News",false,true,"",

        "Title,asc","DateModified",0,"","","") %>

    </TD>

</TR>
```

XML List Summary Function

The XML List Summary function wraps XML around the properties of a list of content, created as a list summary in the Ektron CMS400.NET Workarea. Once created, you can use the XML to display the list summary information. The ecmListSummaryXML function allows you to easily define how the list summary data appears on the Web page.

<%= ecmListSummaryXML (String Folder, int Recursive, String OrderBy, int MaxNumber, String ObjType, String SummaryType) %>

This function is basically the same as the List Summary XML function in ASP. See ”XML List Summary Function” on page 414 for details.

The only difference between the JSP function and the ASP one is the syntax for implementing it. The JSP example is below.

```
<%= ecmListSummaryXML("\\books\\",0,"Title,asc",0,"folder",""); ?>
```

RSS List Summary Function

The RSS summary function is used to create an XML list of content summaries for RSS Syndication. Shown here is the format of the RSS list summary function:

<%= ecmRssSummary (java.lang.String Folder, boolean Recursive, java.lang.String OrderBy, int MaxNumber) %>

This function is basically the same as the RSS Summary function in ASP. See "RSS Summary Function" on page 424 for details.

The only difference between the JSP function and the ASP one is the syntax for implementing it. The JSP example is below.

```
<%= ecmRssSummary("\\Books\\", false, "Title,asc", 0) %>
```

Single Summary Function

The single summary function is used to display the summary of one content block in the Ektron CMS400.NET Web site, instead of a list of all the summaries in a content folder. Seen below is the single summary function:

<%= ecmRssSummary (int contentID, boolean showSummary, java.lang.String showInfo, java.lang.String styleInfo, java.lang.String spare1) %>

This function is basically the same as the Single Summary function in ASP. See "Single Summary Function" on page 420 for details.

The only difference between the JSP function and the ASP one is the syntax for implementing it. The JSP example is below.

Single Summary Example

```
<TR>

   <TD>

       <%= ecmSingleSummary ("14",true,"","","") %>

   </TD>

</TR>
```

Web Services

Introduction to Web Services

Web Services (WS) are reusable software components that can be consumed by variety of clients, independent of operating system, programming language, or Web server, using standard Internet technologies. Because Web Services allow heterogeneous systems to communicate with each other in a standard way, they have received quick acceptance for data interchange over the Web.

It's easy for anyone to integrate your WS into their application. The client application can be a Web application, a Windows application, a Java application, a Visual Studio.net application, or a host of others -- it does not matter as long as it supports Web Services.

A Web Service consists of one or more methods that expose functionality. In this sense, a Web Service is an API that can be accessed using SOAP over HTTP. (SOAP is an XML-based, lightweight protocol that defines the message format of Web Services.)

For example, you might develop a purchasing application that automatically obtains price information from several vendors, lets the user select a vendor, submits the order and then tracks the shipment until it is received. The vendor application, in addition to exposing its services on the Web, might in turn use XML Web services to check a customer's credit, charge the customer's account and set up the shipment with a shipping company.

This manual does not explain how Web Services work. To learn more about them, try these sites.

- http://msdn.microsoft.com/webservices/understanding/readme/default.aspx

- http://aspalliance.com/jnuckolls/articles/introws/default.aspx
- http://aspnet.4guysfromrolla.com/articles/062602-1.aspx
- http://www.15seconds.com/Issue/010430.htm

CMS400.NET's Implementation of Web Services

CMS400.NET supports XML Web Services. This means that you can create pages that offer your content to any Web-enabled device, and you can create pages that consume existing Web Services. CMS400.NET's Web Services feature can return data separated from presentation. In combination with XML support, this feature lets consumers retrieve your XML data independent of presentation information and act on it however they see fit.

Benefits of Using Web Services

Web Services make it easy to connect with partners. They can also deliver more personal, integrated experiences to users via smart devices, including PCs. WSs can save time and money by reducing development time, and increase revenue by making your own XML Web services available to others.

WS Components and CMS400.NET Architecture

Web Services require the installation of the .NET Framework on the server.

Ektron's Web Service, ContentWS.asmx, runs within the .NET Framework. Through the Application API class in the Ektron.cms.commonUI.dll, .NET managed code can communicate with asp.net assembly code. ContentWS.asmx utilizes the ektron.com.commonui.dll file to provide access to the Content Business Objects (content blocks, menus, collections, list summaries, etc.).

Software and Hardware Requirements

- .NET framework, version 1.1 SP2, must be installed on the server (for more information, see http://msdn.microsoft.com/netframework/).

- Ektron CMS400.NET, version 4.5 or higher

Installation of Files

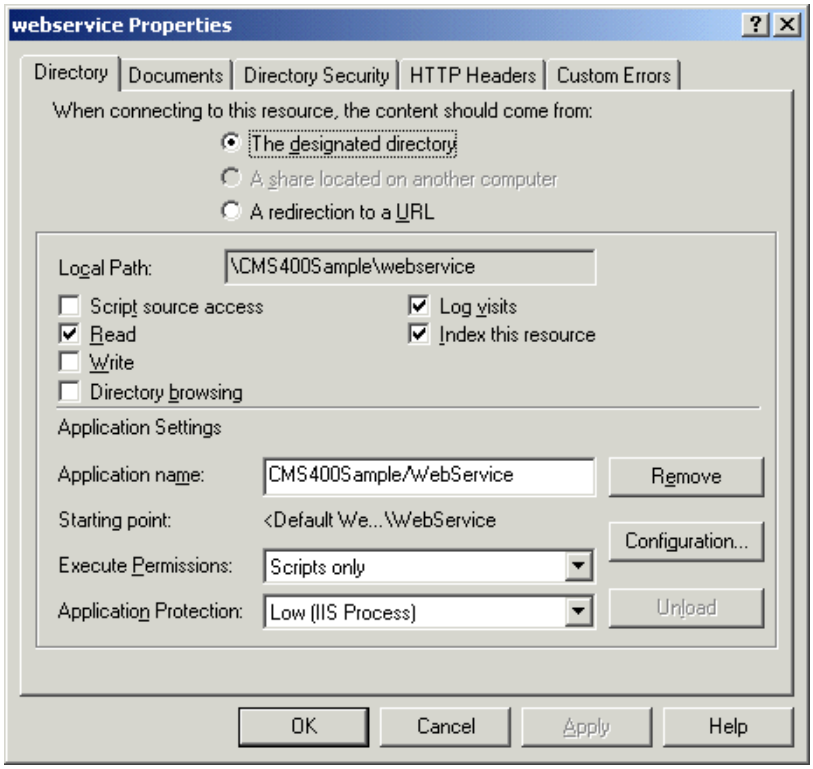
Below is a list of the files and their locations that make up the Web Services feature. They are installed when you install CMS400.NET.

File or folder location	Description
webroot/CMS400Example/Workarea/webservices/ContentWS.asmx	Entry point for Web Service. This Web page displays examples of all methods.
webroot/CMS400Example/bin	Directory for compiled Web Service code
webroot/CMS400Example/bin/WebService.dll	Web Service dll

Setup Instructions

After you install CMS400.NET, you need to open IIS, select `CMS400Example/workarea`, and make it an application. To make a folder an application, follow these steps.

1. Open IIS.
2. Right click one of the folders listed above.
3. Select **Properties**. The Properties dialog appears.



4. Make sure you are on the **Directory** tab, which is selected by default.
5. Press the **Create** button next to the Application Name field.
6. Assign the folder's name in the **Application name** field.
7. Press **OK**.

If you are creating a new directory for your Web site, you must copy the following files into it from `webroot/CMS400Example/bin`.

- WebService.dll
- Ektron.Cms.BE.Content.dll
- Ektron.Cms.BE.Font.dll
- Ektron.Cms.BE.Library.dll
- Ektron.Cms.BE.License.dll
- Ektron.Cms.BE.Message.dll
- Ektron.Cms.BE.Module.dll

- Ektron.Cms.BE.Site.dll
- Ektron.Cms.BE.ToolBar.dll
- Ektron.Cms.BE.User.dll
- Ektron.Cms.BE.UserMgmt.dll
- Ektron.Cms.Common.dll
- Ektron.Cms.DataAccess.dll
- Ektron.Cms.DataRW.dll
- Ektron.Cms.Global.dll
- Ektron.Cms.UI.CommonUI.dll
- Ektron.Cms.UI.dll
- Ektron.Cms.WebserviceUI.dll

Providing Web Service Information in Your CMS

CMS400.NET Methods

CMS400.NET implements a single Web Service, ContentWS.asmx, that exposes several methods for retrieving CMS400.NET content. These methods mirror those in the WebService.dll file.

The methods determine the *kind* of content that is made available (for example, a content block, a collection of content blocks, a summary list) along with the *presentation* of that information (for example, an HTML string or an XML data structure representing content block information).

Below is a list of methods that ContentWS exposes with a brief description of functionality and differences from the applicationAPI implementation.

All methods are demonstrated on the sample page `webroot/CMS400Example\Workarea\webservices\ContentWS.asmx`. For more

information, see "The CMS400.NET Content Sample Page" on page 559.

Method	Exposes this applicationApi method as a web service method	Returns	Changes to standard parameter values
ContentBlock	ecmContentBlock	The content block as an HTML string.	
ContentBlockEx	ecmContentBlockEx	The content block as an XML string. If necessary, performs XSLT transformation on content prior to its return.	
GetContentBlock	ecmContentBlock	An XML data structure representing information in the ContentBlock.	
Collection	ecmCollection	A Collection presented as an HTML string whose format is determined by the display function that you specify. You can create a custom display function in a separate .NET dll.	
GetCollection	ecmCollection	An XML data structure representing information about a Collection.	additional parameter: GetHTML. Possible values: **1** -returns html content for each collection item **0** - does not return it
ListSummary	ecmListSummary	List Summary information as an HTML string.	
GetListSummary	ecmListSummary	An XML data structure representing information about the List Summary.	parameters ShowSummary, StyleInfo, ShowInfo not available.

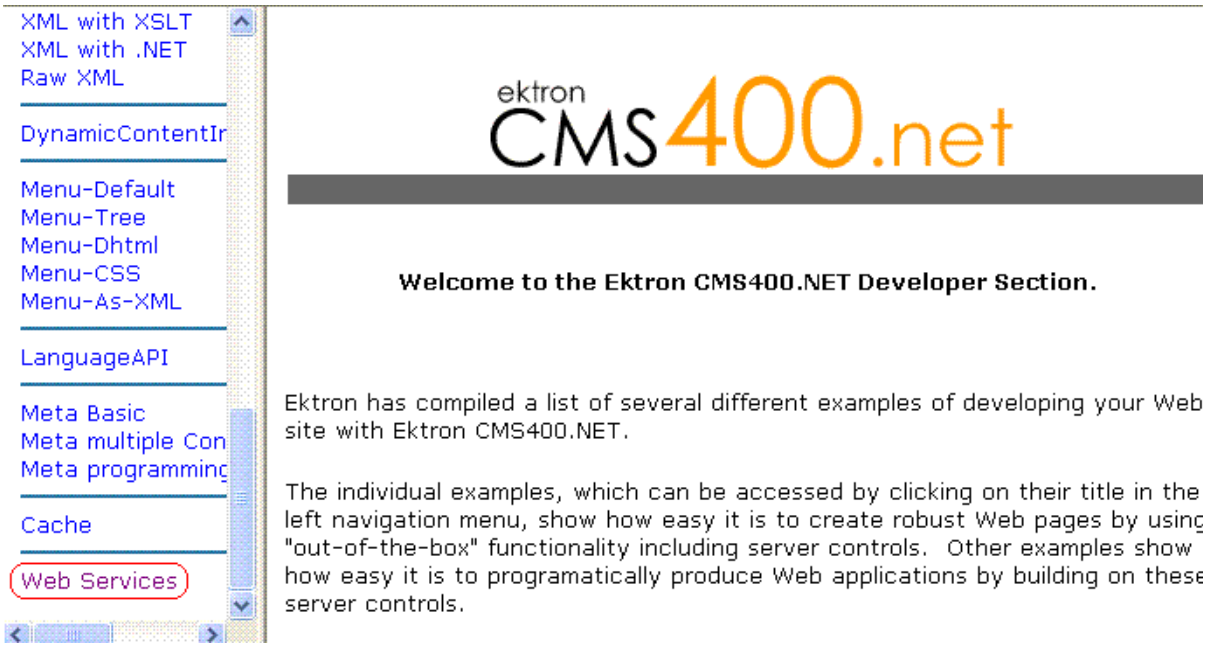
SingleSummary	ecmSingleSummary	Single Summary information as an HTML string.	
GetSingleSummary	ecmSingleSummary	An XML data structure representing the Single Summary information.	parameters StyleInfo, ShowSummary, ShowInfo not available.
MetaData	ecmMetaData	Meta Data information as an HTML string	does not have the unused spare parameter.
GetMetaData	ecmMetaData	An XML data structure representing Meta Data information	does not have the unused spare parameter.
SearchDisplay	ecmSearchDisplay *See Also:* "The Search Display Method Parameters" on page 562	Search Display information as an HTML string	
GetSearchDisplay	ecmSearchDisplay *See Also:* "The Search Display Method Parameters" on page 562	An XML data structure representing the Search Display information	StyleInfo and ShowDate parameters not available
GenerateHTMLMenu	ecmGenerateMenu	Menu details as an html string	Parameters as menu ids (comma separated value)
DropHTMLMenu	ecmDropMenu	Required menu as an html string	Parameters: • Root menu id (that is, the generateHTML-Menu value) • MenuId - required menuid • title - required title

GetFormBlock	ecmFormBlock	Form information as an HTML string	Parameters: • formId • formdata generated through webservicesUI • formtag required flag (set to **true** to embed `<form>` tag; otherwise, set to **false**)
GetEventsByCalendar	ecmEvtCalendar	Calendar information as an HTML string	• Calendar id • event id • month • year • url • querystring
Show RandomSummary	ecmShowRandomSummary	Randomly displays the summary content of a collection item.	• collection ID
ShowRandomContent	ecmShowRandomContent	Randomly displays the content of a content block in a specified collection	• collection ID • xsltID

AddContent	ecmAddContent	Lets you programmatically add a new content block to Ektron CMS400 without logging in.	• Username • password • Domain • content_title • content_comment • ContentHTML • SummaryHTML • ContentLan-guage • FolderID • GoLive • EndDate • MetaInfo • ErrString

The CMS400.NET Content Sample Page

The CMS400.NET content sample page (*webroot*`\CMS400Example\Workarea\webservices\ContentWS.asmx`) lists the methods described in the table above. You can access it by signing on to the developer sample site (`http://localhost/CMS400Example/dev.aspx`) and clicking Web Services from the bottom of the left frame (illustrated below).



Click each method on the page to see sample calls and output using various protocols.

When you open the detail page, you see **Test** followed by parameters.

ecmGetContentBlock

Test

To test the operation using the HTTP POST protocol, click the 'Invoke' button.

Parameter	Value
ContentId:	
UserID:	
SitePath:	
Preview:	
SiteLanguage:	

Invoke

SOAP

The following is a sample SOAP request and response. The **placeholders** shown need to be replaced with actual values.

```
POST /CMS400Example/Workarea/Webservices/ContentWS.asmx HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.ektron.com/cms/WebService/ContentWS/ecmGetContentBlock"
```

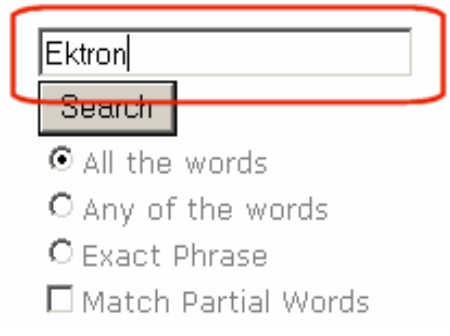
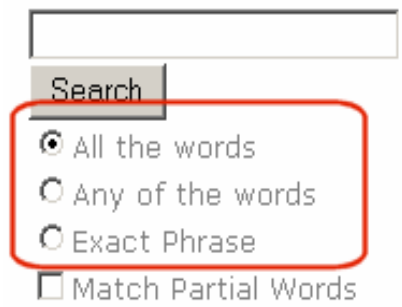
The top section of the page lets you test the method by using the HTTP Get protocol to retrieve data from the sample database. To do this, you must insert valid parameter values.

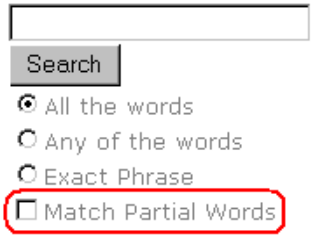
The next section of the CMS400.NET content sample page displays the following kinds of sample output.

- A sample SOAP request and response.
- A sample HTTP GET request and response.
- A sample HTTP POST request and response.

Review the test pages to learn the kind of content you can retrieve from this method.

The Search Display Method Parameters

Parameter	Where Defined in Application API.xxx	How Implemented by CMS
Search Text	n/a	Search text used in search method. 
Search Type	n/a	Type of search used in search method.  Options **and** - all the words **or** - any of the words **exact phrase** - exact phrase Note: This parameter is handled automatically by CMS when displayed on Web page. For Web Services, however, this needs to be defined.
Recursive	Search	Defines whether search is recursive .· **1** - recursive search **0** - non-recursive search

Parameter	Where Defined in Application API.xxx	How Implemented by CMS
Starting Folder	Search	Describes path to folder in which search begins. · Here are some examples. • \ - root content folder • **\Products\RC Cars** - begins search in the RC Cars folder.
Allow Fragments	n/a	Selected by user on page that performs search.  Options **0** (false) - do not allow fragments. **1** (true) - allow fragments. If fragments are allowed, and the user enters ”AT” as the search string, the search returns topics with these words: pat, ate, hatter. If fragments are *not* allowed, and the user enters ”AT” as the search string, the search only returns topics with the word: at. Note: This parameter is handled automatically by CMS when displayed on Web page. For Web Services, this needs to be defined.
Max Number	Search Display	Specify the maximum number of topics to display when a search is performed. For example: **0** - unlimited **1** - one result **7** - seven results

Parameter	Where Defined in Application API.xxx	How Implemented by CMS
Style Info	Search Display	Specify an inline style that is applied to the hyperlinked portion of topics that result from a search. For example: `"font-family:arial; font-weight:bold; backgroundcolor:#cccccc; border:solid blue 1pt; margin:2px; width:50%;"`
Show Date	Search Display	Specify whether the last modified date is displayed with search results.· **0** - date is not displayed. **1** - date is displayed.

Sample Output from One Method

The following is an example of the content that can be retrieved from one of the methods, GetContentBlock. As you can see, the HTML content is provided within the `<ContentHtml>` XML element.

GetContentBlock

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentBlockResult xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="urn:ektron:webservices:content:cms400:v45">
<ErrorString />
<Item>
   <ContentTitle>Plastic Molder #123</ContentTitle>
   <ContentHtml>
<?xml version="1.0"?> <jobposting> <job-title>Plastics Molder</job-title>
<job-number>123</job-number> <description><p>RC International, a leading
supplier of consumer level radio controlled vehicles, is looking for an
experienced plastics molder to join our team. The candidate will join our
highly skilled team of enthusiastic molders to bring to life our next
generation products. RC International is experiencing extremely rapid
growth and offers many opportunities for advancement.</p></description>
<qualifications> <ul> <li>3 years experience in commercial grade plastic
molding technologies.</li> <li>Excellent written and oral communication
skills.</li> <li>Ability to work with very little supervision.</li> <li>An
AS degree in chemistry.</li> </ul> </qualifications> <offering> <p>First
and second shift opportunities exist. RC International offers competitive
pay, an extensive benefits package including a 401k plan, profit sharing,
and health coverage.</p> </offering> <contact> <contact-name>Human
```

```
Resources Manager</contact-name> <contact-email>jobs@CMS400site.com</
contact-email> <contact-fax>1-555-555-5555</contact-fax> </contact> </
jobposting>
</ContentHtml>

<Xslt1>C:\Inetpub\wwwroot\CMS400Example\XMLFiles\xslt\rc_jobs_display.xsl</
Xslt1>
  <Xslt2 />
  <Xslt3 />
  <Xslt4 />
  <Xslt5 />
  <DefaultXslt>1</DefaultXslt>
 </Item>
</ContentBlockResult>
```

The News.aspx Sample (non-XML)

To help you understand how the samples work, this section explains the news sample .NET page, `news.aspx`, which is installed to *`webroot`*`/CMS400Example/WebServiceDemo/`. Specifically, four key sections of code are explained:

- Identifying the Code Behind File
- Setting Up an Instance of Web Services
- Retrieving the Web Service's URL
- Getting Content Blocks

Identifying the Code Behind File

Here is the first line in the news.aspx file:

```
<%@ Page CodeBehind="news.aspx.vb" Language="vb" AutoEventWireup="false"
Inherits="WebServicesDemo.news" %>
```

This line indicates that the page's functionality is set up in the Code Behind Visual Basic file, news.aspx.vb. The rest of the news.aspx file contains only page display information.

Setting Up an Instance of Web Services

The following code in news.aspx.vb sets up an instance of the Web Services. Every page that uses Web Services must contain such a declaration.

```
Dim wsCms400WebService As New ContentWS.ContentWS
```

Getting Content Blocks

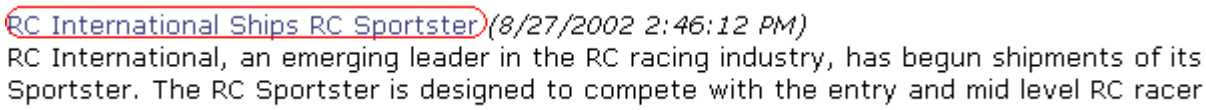
The following lines of news.aspx.vb code retrieve all content blocks in the sample database's news folder formatted as an HTML string.

```
If (iId > 0) Then
   strHTML = wsCms400WebService.ContentBlock(iId)
   strHTML = Replace(strHTML, ".asp", ".aspx")
   NewsInfo.InnerHtml = strHTML

Else
   strHTML = wsCms400WebService.ListSummary("\Marketing\News", 0, 1, "",
"Title,asc", "DateModified", 0, "", "")
   strHTML = Replace(strHTML, "news.asp", "WebServiceDemo/news.aspx")
   NewsInfo.InnerHtml = strHTML
End If
```

NOTE Since the database was created with .NET, you could use this code to get the HTML for the requested ID:
`NewsInfo.InnerHtml = wsCms400WebService.ContentBlock(iId)`

As you can see, we are assigning to the variable NewsInfo.InnerHtml the Web service (wsCms400WebService) followed by the ContentBlock method. (For a list of methods, see "CMS400.NET Methods" on page 555.)

If the user selects a content block on the List Summary page (that is, clicks the blue text of the content title), that action passes an id value. The blue text is circled in the illustration below.



A new page appears, using the ContentBlock method to display the selected block.

```
If (iId > 0) Then
strHTML = wsCms400WebService.ContentBlock(iId)
```

The ContentBlock method retrieves the content block as an HTML string, so the result looks like this.

Click here for a complete list of operations.

GetContentBlock

Test

To test the operation using the HTTP GET protocol, click the 'Invoke' button.

Parameter	Value
ContentId:	
	Invoke

SOAP

The following is a sample SOAP request and response. The **placeholders** shown need to be replaced with actual values.

```
POST /CMS400Example/Workarea/Webservices/ContentWS.asmx HTTP/1.1
Host: 192.168.0.219
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "urn:ektron:webservices:content:cms300:v21/GetContentBlock"
```

If the user does not select a content block (which is true when the page first appears), the page uses the ListSummary method to display all blocks in the `marketing/news` folder using the List Summary format.

```
Else
strHTML =
wsCms400WebService.ListSummary("\Marketing\News", 0, 1, "", "Title,asc",
"DateModified", 0, "", "")
```

The ListSummary method retrieves the content blocks as an HTML string, so the page looks like this.

Latest News

RC International Announces the RC Cheetah *(8/27/2002 2:43:01 PM)*
RC International, a leading RC racing company, has announced the availability of its new flagship product, the RC Cheetah.

RC International Hosts RC Conference *(8/27/2002 2:44:44 PM)*
RC International, the fastest growing company in the RC industry, announces plans to host the 3rd annual RC conference in Atlanta, Georgia.

RC International Ships RC Sportster *(8/27/2002 2:46:12 PM)*
RC International, an emerging leader in the RC racing industry, has begun shipments of its first product, RC Sportster. The RC Sportster is designed to compete with the entry and mid level RC racer currently on the market.

The News.aspx Sample (XML)

The news.aspx sample file in the `/cmd400Example/WebServiceDemo` folder includes all elements described in "The News.aspx Sample (non-XML)" on page 565 *plus* demonstrations of how to retrieve XML content.

Setting Up Instances of Web Services

In addition to creating an instance of the Web Service, the XML version of news.aspx.vb creates instances of the ContentBlock and ListSummary methods, which are used on the page.

```
Dim wsCms400WebService As New localhost.CMS400Content()
Dim gcbCms400ContentXml As localhost.ContentBlockResult
Dim glsCms400ListSummaryResult As localhost.ListSummaryResult
```

Getting Content Blocks

Look at this code from news.aspx.vb.

```
        glsCms400ListSummaryResult =
wsCms400WebService.GetListSummary("\Marketing\News", 0, 1, "", "Title,asc",
"DateModified", 0, "", "")
        If (Len(glsCms400ListSummaryResult.ErrorString) = 0) Then
strHTML = "<table><tr><td><b>Content Block Information Return from the
     GetListSummary Web service call</b></td></tr>"
strHTML = strHTML & "<tr><td> </td></tr>"
For iLoop = 0 To (glsCms400ListSummaryResult.Item.Length - 1)
strHTML = strHTML & "<tr><td><b>Content ID: </b>" &
     glsCms400ListSummaryResult.Item(iLoop).ContentID & "</td></tr>"
```

```
strHTML = strHTML & "<tr><td><b>Content Title: </b>" &
     glsCms400ListSummaryResult.Item(iLoop).ContentTitle & "</td></tr>"
strHTML = strHTML & "<tr><td><b>Date Created: </b>" &
     glsCms400ListSummaryResult.Item(iLoop).DisplayDateCreated & "</td></
tr>"
strHTML = strHTML & "<tr><td><b>Date Modified: </b>" &
      glsCms400ListSummaryResult.Item(iLoop).DisplayDateModified & "</td></
tr>"
strHTML = strHTML & "<tr><td><b>Modified By: </b>" &
     glsCms400ListSummaryResult.Item(iLoop).EditorFname & " " &
     glsCms400ListSummaryResult.Item(iLoop).EditorLname & "</td></tr>"
strHTML = strHTML & "<tr><td><b>Content Teaser: </b></td></tr>"
     strHTML = strHTML & "<tr><td>" &
glsCms400ListSummaryResult.Item(iLoop).ContentTeaser & "</td></tr>"
strHTML = strHTML & "<tr><td> </td></tr>"
Next
```

This code loops through all content blocks in the `Marketing\News` folder. For each block, the page uses the Web Service method `glsCms400ListSummaryResult.Item(iLoop)` to display the following XML data.

- Content ID
- Content Title
- Date Created
- DateModified
- Modified By
- Content Teaser

The result looks like this.

Latest News

Content Block Information Return from the GetListSummary Web service call

Content ID: 11
Content Title: RC International Announces the RC Cheetah
Date Created: 25-Apr-2002 04:43:42 PM
Date Modified: 27-Aug-2002 02:43:01 PM
Modified By: Application Administrator
Content Teaser:
RC International, a leading RC racing company, has announced the availability of its new flagship product, the RC Cheetah.

Content ID: 10
Content Title: RC International Hosts RC Conference
Date Created: 25-Apr-2002 04:42:55 PM
Date Modified: 27-Aug-2002 02:44:44 PM
Modified By: Application Administrator
Content Teaser:
RC International, the fastest growing company in the RC industry, announces plans to host the 3rd annual RC conference in Atlanta, Georgia.

Content ID: 9
Content Title: RC International Ships RC Sportster
Date Created: 25-Apr-2002 04:42:10 PM
Date Modified: 27-Aug-2002 02:46:12 PM

The lower portion of the page also displays the content block in a ListSummary format, as described in"Getting Content Blocks" on page 566.

Latest News

RC International Announces the RC Cheetah *(8/27/2002 2:43:01 PM)*
RC International, a leading RC racing company, has announced the availability of its new flagship product, the RC Cheetah.

RC International Hosts RC Conference *(8/27/2002 2:44:44 PM)*
RC International, the fastest growing company in the RC industry, announces plans to host the 3rd annual RC conference in Atlanta, Georgia.

RC International Ships RC Sportster *(8/27/2002 2:46:12 PM)*
RC International, an emerging leader in the RC racing industry, has begun shipments of its first product, RC Sportster. The RC Sportster is designed to compete with the entry and mid level RC racer currently on the market.

If the user selects a content block and clicks the blue text of any title, a new page appears with

- information about the block (title and any XSLTs used for rendering).
- a demonstration of the GetContentBlock method to display the content block in “raw” HTML or XML format.
- a demonstration of the ContentBlock method to display the content block in processed HTML or XML format.

Here is the code that displays the top section of the page.

```
strHTML = "<table><tr><td><b>Content Block Information from GetContentBlock
Web Service: </b></td></tr>"
strHTML = strHTML & "<tr><td><b>Content Block Title: </b>" &
gcbCms400ContentXml.Item.ContentTitle & "</td></td>"
strHTML = strHTML & "<tr><td><b>Xslt Used for Rendering: </b>"
Select Case (gcbCms400ContentXml.Item.DefaultXslt)
   Case 1
      strTmp = gcbCms400ContentXml.Item.Xslt1
   Case 2
      strTmp = gcbCms400ContentXml.Item.Xslt2
   Case 3
      strTmp = gcbCms400ContentXml.Item.Xslt3
   Case 4
      strTmp = gcbCms400ContentXml.Item.Xslt4
   Case 5
      strTmp = gcbCms400ContentXml.Item.Xslt5
End Select

If (Len(strTmp) = 0) Then
   strTmp = "None Used (HTML Content assumed)"
```

This code uses the `gcbCms400ContentXml` Web Service to get the content title: `gcbCms400ContentXml.Item.ContentTitle.`

The code also uses the gcbCms3400ContentXml Web Service to retrieve any XSLTs assigned to the content. If no XSLTs are assigned (as is the case with this sample page), the message "None Used (HTML Content assumed)" appears.

This code results in the following section of the page.

Latest News

Content Block Information from GetContentBlock Web Service:
Content Block Title: RC International Announces the RC Cheetah
Xslt Used for Rendering: None Used (HTML COntent assumed)

Next, the page uses the GetContentBlock method to display the content block in “raw” HTML or XML format. Here is the code that displays that section of the page.

```
strHTML = strHTML & "<tr><td><b>Raw HTML (or XML) from GetContentBlock Web
Service: </b></td></tr>"
strHTML = strHTML & "<tr><td> </td></tr>"
strHTML = strHTML & "<tr><td>" & strRawHtml & "</td></tr>"
strHTML = strHTML & "<tr><td> </td></tr>"
```

Finally, the page uses the ContentBlock Web method to display the content block in “processed” format. Here is the code that displays that section of the page.

```
strHTML = strHTML & "<tr><td><b>Processed HTML (or XML) from ContentBlock
Web Service: </b></td></tr>"
      strHTML = strHTML & "<tr><td>" & strProcessedHtml & "</td></tr>"
      strHTML = strHTML & "</table>"
```

Earlier in the news.aspx.vb file, the `strProcessedHtml` variable was defined as `strProcessedHtml = wsCms400WebService.ContentBlock(iId)`. So, the `strProcessedHtml` variable uses the `wsCms400WebService.ContentBlock` method to retrieve content in processed format.

Here is the resulting section of that page. As you can see, the content appears as within a browser.

RC International Announces the RC Cheetah

January 14, 2001 - RC International, a leading RC racing company, has announced the availability of its new flagship product, the RC Cheetah. After nearly two years of development the RC Cheetah is designed to compete at the highest level in the RC racing arena. The excitement level of the hard core enthusiasts can be felt at all the demonstrations of the RC Cheetah.

> *"The RC Cheetah price/performance ration is simply incredible"*, stated Bob Dickerson, RC International VP of engineering. *"The RC Cheetah has been devastating its competition in all the pre-release races and we expect that to continue..."*.

The major industry magazines are agreeing with Mr. Dickerson's assessment. RC Magazine has tested the RC Cheetah head-to-head with the competition's best and they concluded:

> *"There is no comparison between the RC Cheetah and its nearest competitor. In speed, handling, and price the RC Cheetah out shines them all"*.

The RC Cheetah is now available at all RC International distributors and major Hobby stores. Please contact RC International to locate a local dealer.

Creating a Page that Provides a Web Service

Below are the typical steps you would follow to create your own page that provides a Web Service.

1. Decide which content blocks you want to provide. Here are some examples:
   - a specific content block (for example, `wsCms400WebService.ContentBlock(2)` would display content block 2 regardless of the URL parameters)
   - all blocks in a folder (the GetListSummary method's first parameter is folder)
   - all blocks in a collection (the Collection method's first parameter is collection ID)
2. Decide what information you want to provide about those blocks, and how to present it. Review"CMS400.NET Methods" on page 555 to find the type of content and the kind of information that is available about it.

   For example, if you want to display XML data about content (such as content title, date modified, date created), use a method that begins with "Get," such as GetContentBlock, GetCollection, and GetListSummary.

3. Create a new folder within the CMS400Example folder. This folder and some of its subfolders have files that are needed to run Web Services. Place all of your files within the new folder.
4. Using a development tool that supports Web Service integration (for example, Visual Studio .NET), create a .NET (.aspx) file that manages page layout. The file must refer to a Visual Basic or C# file (via a `CodeBehind` command), which manages the page's functionality.
5. Add to each .NET project a Web Reference to the WSDL file. (by default, *`webroot`*`/CMS400Example/Workarea/webservices/ContentWS.asmx`). The environment then builds the appropriate proxy class to access the Web Service's methods. You can use the proxy class within the application to invoke those methods. CMS400.NET sample .aspx pages demonstrate how to do this.
6. Provide to your consumers a URL to the .NET (.aspx) file.

Syndicating Web Content as RSS

This section explains how to add RSS syndication to Ektron CMS400.NET for ASP.NET. RSS syndication allows you to prepare Ektron CMS400.NET content for reading by RSS news feed aggregators.

To make this task easier, Ektron provides sample files to illustrate RSS syndication. You should customize the files to meet your needs.

What is RSS Syndication?

RSS stands for Really Simple Syndication, a common method of syndicating Web content as an XML document. There are several versions of RSS. Ektron has implemented version 2.0.

In Ektron CMS400.NET, you use RSS to prepare a collection list or a list summary for reading by an *aggregator*. An aggregator consumes an RSS feed and displays it to the user. (An aggregator is similar in function to a portal.) There are several popular aggregators, including:

- AmphetaDesk (http://www.disobey.com/amphetadesk/)
- NewzCrawler (http://www.newzcrawler.com/).

The RSS Specification

The RSS specification is a document: http://backend.userland.com/rss. Ektron has implemented the specification, but you can customize the RSS classes to support advanced features.

RSS Syndication in Ektron CMS400.NET

Ektron CMS400.NET allows you to syndicate content by using .NET. Syndicating via .NET requires that you have the Microsoft .NET framework installed on your computer.

With Ektron CMS400.NET, you can syndicate a collection and a list summary via RSS.

Syndicating via .NET

Architecture of RSS Syndication

aspx pages

You can use an aspx page as a URL to dynamically create and return RSS documents.To display an RSS feed on a Web site, the site administrator identifies the URL of the aspx page. The aggregator determines how to use the document's elements, such as <title>, <description> and <link>, to display the syndicated content.

The sample aspx templates (`RSS_ListSummary.aspx and RSS_Collection.aspx`) are installed by default to `localhost/CMS400Example` folder. They use the WebService.ContentWS class to retrieve the collection or list summary data. Then, they create CollectionToRSS20 or ListSummaryToRSS20. These classes fill the RSS 2.0 data classes and serialize them to an XML document, which can be returned as an HTTP response.

Since these pages are part of the CMS400.NET sample, you should modify them or create your own aspx pages for your Web site. Place your custom pages in the `yourwebsite/WebServices/` folder so that the web.config file is within the path.

Retrieving a Collection or List Summary as an RSS Feed

The left frame of the corporate.aspx page is hardcoded to retrieve the News Collection. The right frame retrieves a ListSummary.

Creating Your Own aspx Collection Page

To create your own collection page, copy the collection object to an RSS 2.0 object and serialize it. To accomplish this, use these methods.

```
ConvertToRSS(ByVal result CMS400WS.CollectionResult, ByVal
```

```
title As String, ByVal description As String) As RSS20.rss


Serialize(Optional ByVal stream As System.IO.Stream = Nothing,
Optional ByVal o As Object = Nothing)
```

Example

```
Dim objRSSConverter As New CollectionToRSS20(Request, Response)
objRSSConverter.ConvertToRSS(objResult,
objResult.CollectionTitle, objResult.CollectionDescription)
objRSSConverter.Serialize()
```

Remarks

The ConvertToRSS method returns the RSS 2.0 object so that you can fill in or modify additional fields as desired before calling Serialize. The Serialize method has optional parameters that let you send the RSS to a stream other than HttpResponse.

You can add code to serialize a different object (such as an RSS 1.0 data class) or to run an XSLT on the serialized XML document before returning it as the HTTP response.

Example

```
Dim objRSSConverter As New CollectionToRSS20(Request, Response)
Dim objRSS20 As rss.RSS20.rss =
objRSSConverter.ConvertToRSS(objResult,
objResult.CollectionTitle, objResult.CollectionDescription)
objRSS20.channel.language = "en-us"
objRSSConverter.Serialize()
```

Creating Your Own aspx List Summary Page

To create your own list summary page, copy a list summary object to an RSS 2.0 object and serialize it. To accomplish this, use these methods.

```
ConvertToRSS(ByVal result As CMS400WS.ListSummaryResult, ByVal
title As String, ByVal description As String) As RSS20.rss

Serialize(Optional ByVal stream As System.IO.Stream = Nothing,
Optional ByVal o As Object = Nothing)
```

Example

```
Dim objRSSConverter As New ListSummaryToRSS20(Request, Response)
objRSSConverter.ConvertToRSS(objResult, "News", "The latest news")
objRSSConverter.Serialize()
```

Remarks

The ConvertToRSS method returns the RSS 2.0 object so that you can fill in or modify additional fields as desired before calling Serialize. The Serialize method has optional parameters that let you send the RSS to a stream other than HttpResponse.

You can add code to serialize a different object (such as an RSS 1.0 data class) or to run an XSLT on the serialized XML document prior to returning it as the HTTP response.

Example

```
Dim objRSSConverter As New ListSummaryToRSS20(Request, Response)
Dim objRSS20 As rss.RSS20.rss =
objRSSConverter.ConvertToRSS(objResult, "Title", "Description")
objRSS20.channel.language = "en-us"
objRSSConverter.Serialize()
```

Learning About Visual Studio .NET

This section introduces some background information about using Microsoft's Visual Studio .NET. For more information, use the help feature installed with Visual Studio.NET and Microsoft's developer Center (http://msdn.microsoft.com/vstudio/).

Grid Layout vs. Flow Layout

NOTE The following definitions are from Visual Studio Help:

Grid Layout - Absolute positioning attributes are inserted into elements that are added, and updated in elements that are moved. Elements can be dragged across the Design view surface. The positioning grid and Snap to Grid are available.

Flow Layout - Elements are added without absolute positioning attributes. Web browsers arrange elements in the order that they occur on the page, from top to bottom. You cannot drag elements across the Design view surface or use the positioning grid.

Grid layout is the default, which means that all controls drawn to the Web form in the designer window will have absolute positioning. Here is an example.

```
<body MS_POSITIONING="GridLayout">
 <form id="Form1" method="post" runat="server">
  <asp:Button id="Button1" style="Z-INDEX: 101; LEFT: 160px; POSITION:
absolute; TOP: 80px" runat="server" Text="Button"></asp:Button>
  <asp:Button id="Button2" style="Z-INDEX: 102; LEFT: 480px; POSITION:
absolute; TOP: 88px" runat="server" Text="Button"></asp:Button>
  <asp:DataGrid id="DataGrid1" style="Z-INDEX: 103; LEFT: 208px; POSITION:
absolute; TOP: 152px" runat="server"></asp:DataGrid>
 </form>
</body>
```

In Grid layout, you can position your controls like a WYSIWYG editor with no knowledge of HTML. However, because absolute positioning is not rendered consistently by all browsers, the page layout can be flexible based on the size of other controls on the page, and the Web browser window.

When other controls are dynamically populated, such as a DataGrid, controls that appear beneath it in the Web form would be obscured if they were positioned absolutely at design time. In addition, when utilizing globalization of pages with different languages, the size of text areas can vary and cause obstructions.

Customizing the Ektron CMS400.NET Workarea

Customizing Validation

Customizing Validation Options

In the cms_config.aspx file, you can specify a set of validation options for plain text and calculation field types. (Data in other field types cannot be validated.) For these field types, you can assign standard and custom validation checks. The checks are applied when data is inserted into one of the field types, and when the user tries to save a Data Design document whose fields have validation attributes.

You can modify the standard options and enter your own criteria for each field. You can also establish dependencies between fields. For example, a value is only required for a field if a certain Checkbox field is checked.

Validation Elements in the Configuration Data

The default configuration file includes standard validation options for plain text and calculation field types.

```
<validation name="calculation">
   <choice name="none" treeImg="text">
      <caption localeRef="dlgNV8n" />
      <schema datatype="string" />
```

and

```
<validation name="plaintext" visible="false">
   <choice name="none" treeImg="text">
      <caption localeRef="dlgNV8n" />
```

The validation sections let you control the drop-down list of validation choices for a field type (for example, Plain Text). Here is an overview of that section of the cms_config.aspx file.

```
<datadesign>
    <validation> (0 or more)
        <choice> (0 or more)
            <caption>
            <schema>
            <calculate>
<regexp> OR
```

```
            <script> OR
            <xpath>
        <validate>
            <regexp> OR
            <script> OR
            <xpath>
    <errormessage> (optional)
```

Details are provided below.

Validation Attributes

Attribute	Description	Possible values
visible	Controls whether the **Validation** field appears on the field's Properties dialog.	true, false
enabled	Controls whether the **Validation** field is active or “grayed out” on the field's Properties dialog. If set to **true**, the **Validation** field is active; if **false**, it appears but is grayed out.	true, false
name	The field type to which the validation configuration data applies. A separate `<validation>` element must exist for each field type.	plaintext, calculation

Validation Sub-elements

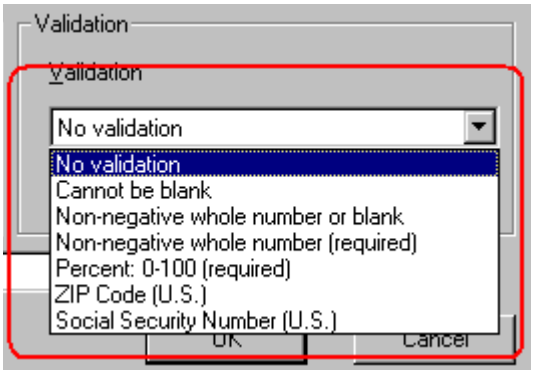
Element	Description	For more information, see
Choice	The calculation or validation expression for standard fields	”Choice Sub-element” on page 584
Custom	The calculation or validation expression for custom fields	”Defining Custom Validation” on page 589

Choice Sub-element

Every item in the validation drop-down list must be defined within a set of `<choice>` tags. Within the `<choice>` tags, you define a

caption, a schema, and either a calculation or validation expression.

The default validation choices appear below.



Choice Attributes

Attribute	Description
name	Assign a new name to each choice.
treeImg	The icon to display for this field in the Select Field or Group dialog. See "Icons on the Select Field or Group Screen" on page 595. Unlike toolbar icons, you cannot create your own icons. Note: This attribute only applies to the Data Design of an XML configuration -- it does not apply to HTML forms.

Choice Sub-Elements

Element	Description	For more information, see
caption	Defines the displayed text for this choice. The attributes and description are the same as `<caption>` elements for commands.	"Caption" on page 335
schema	Defines a W3C XML schema (WXS) definition for this choice. The definition may be a simple type defined by the `datatype` attribute and/or a WXS fragment. Note: This attribute only applies to the Data Design of an XML configuration -- it does not apply to HTML forms.	"Defining a Schema Fragment" on page 586
calculate	Defines an expression which normalizes a value prior to checking validation	"Defining a Calculation" on page 587
validate	Defines an expression that determines if a value is valid	"Defining Validation" on page 589

Defining a Schema Fragment

When defining a schema fragment, the "xs" namespace prefix is required for WXS tags. Also, the WXS fragment must be valid for inclusion in an xs:restriction (simple type). That is,

```
<xs:simpleType>
    <xs:restriction>
        ...fragment...
    </xs:restriction>
</xs:simpleType>
```

Simple Data Types

The simple data types are defined by W3C XML Schema definition language 1.0.

NOTE The datatype value should *not* include a namespace prefix. For example datatype="string" is correct; datatype="xs:string" is incorrect

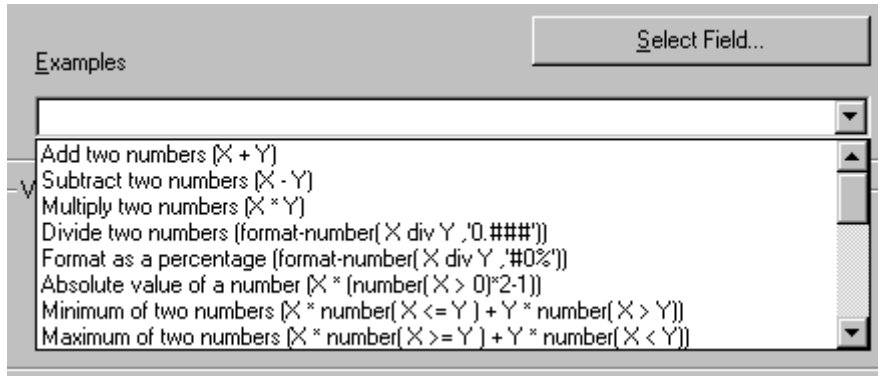
Examples

- Using only a datatype attribute
```
<schema datatype="string"/>
```
- Using only a schema fragment
```
<schema>
    <xs:simpleType>
        <xs:union memberTypes="xs:nonNegativeInteger">
            <xs:simpleType>
                <xs:restriction base="xs:string">
                    <xs:length value="0"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:union>
    </xs:simpleType>
</schema>
```
- Using a datatype attribute *and* a schema fragment
```
<schema datatype="string">
    <xs:minLength value="1"/>
</schema>
```

Defining a Calculation

Use the `<calculate>` element to define an expression that normalizes a value prior to checking validation. For example, a calculation can truncate digits in a decimal, remove excess white space, or capitalize text. The expression must return a value of the same data type and format as the original value.

Several sample calculations are delivered with **Ektron CMS400.NET**. They appear when the user clicks the **Examples** drop-down on the Calculated Field dialog. These calculations are explained in the “Explanation of Examples” section of the “Using the Data Designer” chapter of the Administrator Manual.



You can only define one expression for `<calculate>` and `<validate>`.

The expression element choices are listed below. Use whichever language is easiest.

- Regular Expression (JScript)
- JavaScript
- XPATH

Expression	Description
Regular Expressions (JScript)	
<regexp>	Defines a regular expression supported by JScript. You can define a regular expression *either* between the tags *or* using attributes. If between the tags, the expression must begin with a slash (/) character. The g, i, and m flags are allowed. For example: `<regexp>/\S+/</regexp>`.
<regexp patternings-exp-pattern”	A regular expression pattern (without the “/” chars).
<regexp global=”true|false”	Specifies whether the pattern matches only the first occurrence or all occurrences within the text. This corresponds to the g flag.
<regexp ignorecase=”true|false”	Specifies if the match is case-sensitive. This corresponds to the “i” flag.
<regexp multiline=”true|false”	Specifies if the match, when using ^ and $, is applied to each line in text that has multiple lines. This corresponds to the “m” flag.
<regexp wholeline=”true|false”	Specifies whether the pattern applies to the whole text or not. This is the same as placing “^(“at the beginning of the pattern, and “)$” at the end of the pattern.
JavaScript	
<script value=”javascript-expression”	Specifies a JavaScript expression to be evaluated. The field's value is available in a property named 'this.text'. For example `<script value="this.text.toUpperCase()"/>`

Expression	Description
XPATH (*See Also:* "Learn More about XPath" on page 596)	
<xpath select="xpath-expression"	Specifies an XPath expression to be evaluated. The field's value is available using ".".

Defining Validation

Use the `<validate>` element to define an expression that determines if a value is valid. The expression must return a Boolean (true or false) result. For example:

```
<validate>
      <regexp>/^\d+$/</regexp>
/validate>
```

To construct the `<validate>` element, use the same expression element options as `<calculate>`. *See Also:* "Defining a Calculation" on page 587

Defining an Error Message

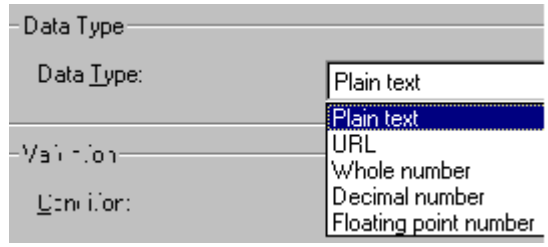
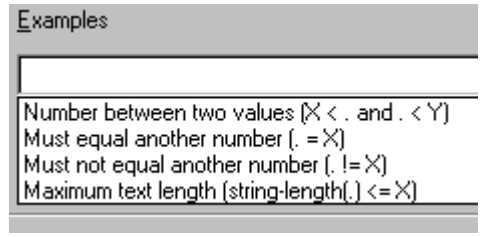
Use the `<errormessage localeRef="id">` element to define a message to display when the data is not valid. For example, "Must be a number between 1 and 10, inclusive."

The text may be within the `<errormessage>` tags or referenced using localeRef into the localeNNNN.xml file.

Defining Custom Validation

NOTE This section only applies to the Data Design of an XML configuration -- it does not apply to HTML forms.

Use the `<custom>` element to change the standard values that appear in the following fields of the Custom Validation dialog.

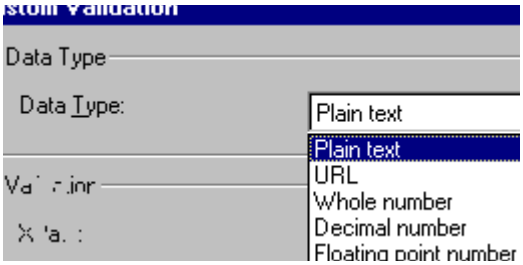
Field	Screen image
Data Type	
Examples	

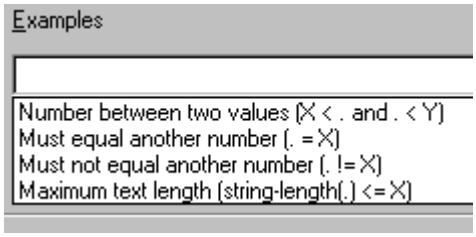
To change the list of options, modify the `<custom>` element of the cms_config.aspx file. This section describes the `<custom>` element's attributes and child elements.

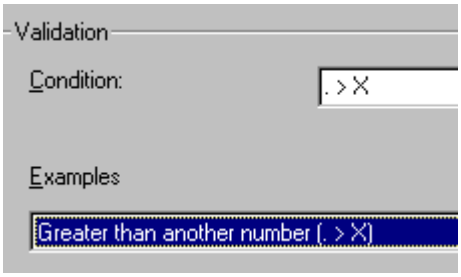
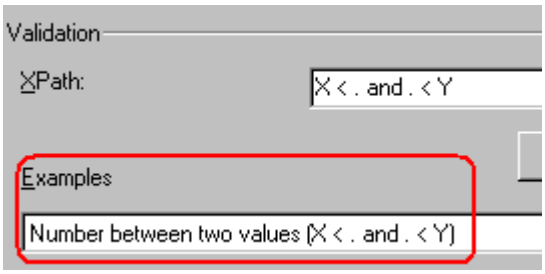
See Also: "selections" on page 352, "listchoice" on page 345

```
<datadesign>
    <validation>
        :
        <custom> (optional tag)
            <caption>
            <selections name="datatype">
                <listchoice> (0 or more)
            <selections name="examples">
                <listchoice> (0 or more)
```

Element or Attribute	Description
<custom>	Specifies basic data types available when customizing validation. The types are defined in the `<listchoice>` elements (see below).

Element or Attribute	Description
<custom visible="true|false"	Controls whether the **Custom Validation** field appears on the Properties dialog.
<custom enabled="true|false"	Controls whether the **Custom Validation** field is active or "grayed out" on the Properties dialog. If set to **true**, the **Custom Validation** field is active; if **false**, it is grayed out.
<caption localeRef="id">	Specifies text to display in the validation drop-down list. The default caption is "**(Custom)**".
Selections element for Data Type field *See Also:* "selections" on page 352	
<selections name="datatype">	The name must be `datatype`.
<selections enabled="true|false"	Controls whether the drop-down list is active or "grayed out" on the Properties dialog.
<selections visible="true|false"	Controls whether the drop-down list appears on the Properties dialog.
Listchoice element for Data Type field *See Also:* "listchoice" on page 345	
<listchoice>	Defines the values in the Custom Validation dialog Data Type drop-down list.
<listchoice value="simple-data-type"	Enter each data type that should appear in the Custom Validation dialog's Data Type drop-down list.  The simple data types are defined by W3C XML Schema definition language 1.0. *See Also:* "Defining a Schema Fragment" on page 586

Element or Attribute	Description
<listchoice treeImg="id"	Specifies the icon to display for this field in the Select Field or Group dialog. See "Icons on the Select Field or Group Screen" on page 595. Unlike toolbar icons, you cannot create your own.
<listchoice localeRef="id"	The text that describes this Data Type on the Custom Validation dialog. This element can refer to a string in the localeNNNN.xml file. Or, you can enter the string between the `<listchoice>` tags.
<listchoice default="true|false"	Use this attribute to indicate the default choice in the Custom Validation dialog's Data Type drop-down list.
Selections element for Examples field *See Also:* "selections" on page 352	
<selections name="examples">	The name must be "examples".
<selections enabled="true|false"	Determines if the **Examples** drop-down list and label are active or "grayed out" in the Custom Validation dialog. 
<selections visible="true|false"	Determines if the **Examples** drop-down list and label appear.
Listchoice Element for Examples field *See Also:* "listchoice" on page 345	
<listchoice>	Defines the values in the Examples drop-down list.

Element or Attribute	Description
<listchoice value="xpath-expression-example"	The XPath expression appears in the **Examples** drop-down list of the Custom Validation dialog. 
<listchoice localeRef="id" -	The text that describes the examples on the Custom Validation dialog.  This element can refer to a string in the localeNNNN.xml file. Or, you can enter the string between the `<listchoice>` tags.

Saving Invalid Documents

Use the `publishinvalid` attribute of the `<standard>` element to determine if an invalid document should be saved. This attribute lets you decide if a user can save an XML document when the data in one or more fields does not satisfy the validation criteria.

In Data Entry mode, an example would be if a field requires a non-negative whole number, but the user does not insert a value in that field.

If the `publishinvalid` attribute's value is `true`, content is not checked for validity when it is saved.

If `false`, the content is checked for validity during a save. If the content is valid, it is saved. If it is invalid, the user is notified, and a

custom script can be created to allow the content to be saved or prevent it.

NOTE If the `publishinvalid` attribute is not defined in the `<standard>` element, the default is **true** so that Ektron CMS400.NET is backwards compatible with previous releases.

Custom Script that Handles Saving Invalid Files

To determine whether or not an invalid XML document should be saved, write a client-side script that is called when invalid content is found. The routine should ask if the user wants to save the invalid content. The script should return `True` to save the content, or `False` to abort the save.

Below is an example of such a script.

```
<script language="JavaScript1.2">
<!--
eWebEditPro.instances["MyEditor1"].onerror = myOnErrorHandler;

 function myOnErrorHandler()
{
   if (EWEP_STATUS_INVALID == this.status && "save" == this.event.source)
   {
     var strMsg = "Content is invalid.";
     strMsg += "\nError Code: " + this.event.reason;
     strMsg += "\nError Reason: " + this.event.message;
     alert(strMsg);
     return false; // prevent save
   }
}
//-->
</script>
```

The onerror Event

If the content is invalid, an onerror event is generated. (*See Also:* ”Event: onerror” on page 174) This event provides two additional properties for the event object when it fires:

- reason (a numeric error code)
- message (text describing the error)

As with the regular onerror event, the `source` property is available. You can display the values of these properties in an error message that informs the user why the document is not valid.

NOTE You can localize the invalidContent message. See "Modifying the Language of eWebEditPro" on page 258.

Calculated Fields

A Calculated Field lets you include values that are calculated, typically from values in other fields. For example, you can multiply two field values. The equation used is an XPath expression.

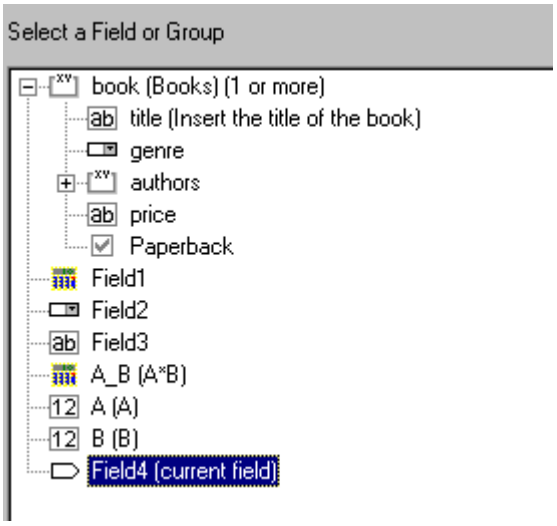
You can validate a Calculated Field, just like the Plain Text Field. For instance, the total of a series of numbers should be less than 100.

For more information about calculated fields, see the **Ektron CMS400.NET** Administrator Manual chapter "Using the Data Designer."

Icons on the Select Field or Group Screen

NOTE This section only applies to the Data Design of an XML configuration -- it does not apply to HTML forms.

Use the following icons to denote a field on the Select A Field or Group Screen (illustrated below). These are the valid values for the `treeImg` attribute.



Icon	Name
	calculation
	calendar
	checkbox
	droplist
	fieldset
	hidden
	hyperlink
	number
	password
	picture
	richarea
	text
	textbox

Learn More about XPath

Xpath Operators

Content copied from `www.w3schools.com/xpath`.

Operator	Description	Example	Result
Numerical expressions perform arithmetic operations on numbers. XPath converts each operand to a number before performing an arithmetic expression.			
+	addition	6 + 4	10
-	subtraction	6 - 4	2
*	multiplication	6 * 4	24
div	division	8 div 4	2
mod	division remainder	5 mod 2	1
Equality/ greater/ less than expressions test equality between two values			
=	equals	price = 9.80	true (if price is 9.80)
!=	is not equal	price! = 9.80	false
<	less than	price < 9.80	false (if price is 9.80)
< =	less than or equal to	price <= 9.80	true
>	greater than	price > 9.80	false
>=	greater than or equal to	price >= 9.80	true
Boolean expressions compare two values			
or	or	price = 9.80 or price = 9.70	true (if price is 9.80)
and	and	price <=9.80 and price = 9.70	false
Identifying a path and element			

Operator	Description	Example	Result
.	the current element	. > 100	true if element exceeds 100
..	the current element's parent	count(../*)	counts the number of elements at the same level as the current element
Grouping and separating			
[ ]	predicate	../*[0]	../*[0] - returns the parent's first child element
|	specify multiple elements	sum(X | Y | Z)	If X=1 and Y =2 and Z=3, sum(X | Y | Z) = 6

Xpath Functions

Content copied from `www.w3schools.com/xpath`.

Operator	Description	Syntax	Example
last	Returns the position number of the last node in the processed node list	number=last()	
position	Returns the position in the node list of the node that is currently being processed	number=position()	
count	Returns the number of nodes in a node-set	number=count(node-set)	
name	Returns the name of a node	string=name(node)	
string	Converts the value argument to a string	string(value)	string(314) Result: '314'
concat	Returns the concatenation of all its arguments	string=concat(val1, val2, ..)	concat('The',' ','XML') Result: 'The XML'

Operator	Description	Syntax	Example
starts-with	Returns true if the first string starts with the second string. Otherwise, it returns false.	bool=starts-with(string,substr)	starts-with('XML','X') Result: true
contains	Returns true if the second string is contained within the first string. Otherwise, it returns false.	bool=contains(val, substr)	contains('XML','X') Result: true
substring-after	Returns the part of the string in the string argument that occurs after the substring in the substr argument	string=substring-after(string,substr)	substring-after('12/10','/') Result: '10'
substring-before	Returns the part of the string in the string argument that occurs before the substring in the substr argument	string=substring-before(string,substr)	substring-before('12/10','/') Result: '12'
substring	Returns a part of the string in the string argument	string=substring(string,start,length)	substring('Beatles',1, 4) Result: 'Beat'
string-length	Returns the number of characters in a string	number=string-length(string)	string-length('Beatles') Result: 7
normalize-space	Returns the whitespace-normalized version of a passed string. All leading and trailing whitespace is stripped, and all sequences of whitespace get combined to one single space.	normalize-space('string')	normalize-space(' some text ') would return some text
translate	normalize	string=translate(value,string1,string2)	• translate ('12:30','30','45') Result: '12:45' • translate ('12:30','03','54') Result: '12:45' • translate ('12:30','0123','abcd') Result: 'bc:da'

Operator	Description	Syntax	Example
boolean	Converts the value argument to Boolean and returns true or false	bool=boolean(value)	
not	Returns true if the condition argument is false, and false if the condition argument is true	bool=not(condition)	not(false())
true	Returns true	true()	number(true()) Result: 1
false	Returns false	false()	number(false()) Result: 0
lang	Returns true if the language argument matches the language of the xsl:lang element. Otherwise, it returns false.	bool=lang(language)	
number	Converts the value argument to a number	number=number(value)	number('100') Result: 100
sum	Returns the total value of a set of numeric values in a node-set	number=sum(nodeset)	sum(/cd/price)
floor	Returns the largest integer that is not greater than the number argument	number=floor(number)	floor(3.14) Result: 3
ceiling	Returns the smallest integer that is not less than the number argument	number=ceiling(number)	ceiling(3.14) Result: 4
round	Rounds the number argument to the nearest integer	integer=round(number)	round(3.14) Result: 3

XPath References

To learn more about XPath, check these Web pages.

Topic	URL
W3C Spec - XPath 1.0 spec	`http://www.w3.org/TR/xpath`

Topic	URL
Tutorial	www.w3schools.com/xpath/default.asp
Quick reference card	www.mulberrytech.com/quickref/XSLTquickref.pdf

Additional Information

Supporting Documents

You can find more detailed information about Ektron CMS400.NET in the following manuals:

- Ektron CMS400.NET Administrator Manual
- Ektron CMS400.NET User Manual
- Ektron CMS400.NET Quick Start Manual
- Ektron CMS400.NET Setup Manual

Support

If you are experiencing trouble with any aspect of Ektron CMS400.NET, please contact the Ektron Support Team via our Web Site at

http://www.ektron.com/support/cms400_support.aspx

Sales

For questions and answers about purchasing Ektron CMS400.NET from Ektron, contact the sales department by sending an e-mail to:

sales@ektron.com

Information

For general information about Ektron and its products, you may send an e-mail to:

info@ektron.com

Ektron, Inc.
5 Northern Blvd., Suite 6
Amherst, NH 03031
Email: sales@ektron.com
Tel: +1 603-594-0249
Fax: +1 603-594-0258