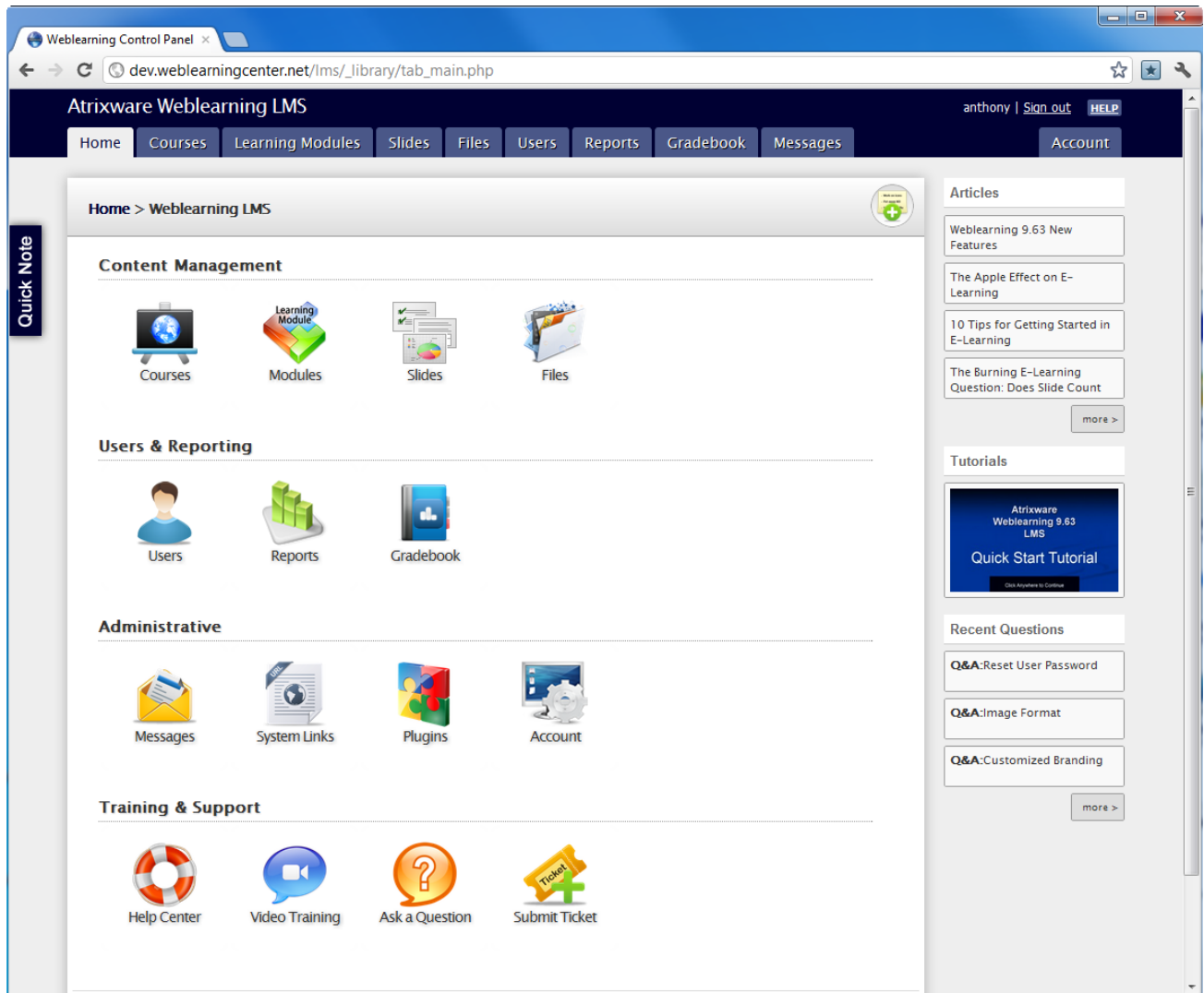


ATRIXWARE

Documentation



Weblearning 9.6 Learning Management System Dynamic Scripting Reference

Proprietary Notice

The software described in this document is a proprietary product of Atrixware, and is furnished to the user under a license for use as specified in the license agreement. The software may be used or copied only in accordance with the terms of the agreement.

Information in this document is subject to change without notice and does not represent a commitment on the part of Atrixware. No part of this document may be reproduced, transmitted, transcribed, stored in any retrieval system, or translated into any language without the express written permission of Atrixware.

Atrixware Weblearning 9.6 Dynamic Scripting Reference
Copyright © 1997-2012, Atrixware, All Rights Reserved.
Second Edition v2, June 2012
Printed in USA

Contents

What is Dynamic Scripting?	5
How does Dynamic Scripting work?	5
Dynamic Scripting Variables	12
Date/Time Variables	12
\$ <i>time</i>	12
\$ <i>hour</i>	12
\$ <i>minute</i>	12
\$ <i>second</i>	12
\$ <i>am_pm</i>	12
\$ <i>date</i>	12
\$ <i>month</i>	12
\$ <i>month_no</i>	13
\$ <i>day</i>	13
\$ <i>day_no</i>	13
\$ <i>year</i>	13
Visitor Data Variables	13
\$ <i>ip_address</i>	13
\$ <i>host_name</i>	13
\$ <i>http_referer</i>	13
\$ <i>http_referer_domain</i>	13
User Data Variables.....	14
\$ <i>user.login</i>	14
\$ <i>user.email</i>	14
\$ <i>user.firstname</i>	14
\$ <i>user.lastname</i>	14
\$ <i>user.middleinit</i>	14
\$ <i>user.start</i>	14
\$ <i>user.expire</i>	14
\$ <i>user.company</i>	14
\$ <i>user.address</i>	14
\$ <i>user.city</i>	15
\$ <i>user.state</i>	15
\$ <i>user.zip</i>	15
\$ <i>user.phone</i>	15
\$ <i>user.c1</i>	15
\$ <i>user.c2</i>	15
\$ <i>user.c3</i>	15
\$ <i>user.c4</i>	15
\$ <i>user.c5</i>	15
\$ <i>user.c6</i>	15
\$ <i>user.c7</i>	16
\$ <i>user.c8</i>	16
\$ <i>user.c9</i>	16
\$ <i>user.c10</i>	16
\$ <i>user.c11</i>	16
\$ <i>user.c12</i>	16
\$ <i>user.c13</i>	16
\$ <i>user.c14</i>	16
\$ <i>user.c15</i>	16
\$ <i>user.notes</i>	16
Usergroup Data Variables	17
\$ <i>usergroups[]</i>	17

- Other Courses User Is Enrolled In Data Variables 17
 - \$othercourses[].name* 17
 - \$othercourses[].start* 17
 - \$othercourses[].expire* 17
 - \$othercourses[].login_url* 17
 - \$othercourses[].enroll_url* 17
- User Activity Data Variables 18
 - \$activity[].title* 18
 - \$activity[].completions* 18
 - \$activity[].date_last_complete* 18
 - \$activity[].time_last_complete* 18
 - \$activity[].total_time* 18
- Quiz Data Variables (Graded Quizzes) 18
 - \$quiz[].title* 18
 - \$quiz[].completions* 18
 - \$quiz[].date_last_complete* 18
 - \$quiz[].time_last_complete* 19
 - \$quiz[].highest_score* 19
 - \$quiz[].lowest_score* 19
 - \$quiz[].average_score* 19
 - \$quiz[].passing_score* 19
 - \$quiz[].total_time* 19
- Course Data Variables 19
 - \$course.name* 19
 - \$course.id* 19
 - \$course.login_url* 19
 - \$course.enroll_url* 20
 - \$course.enroll_code* 20
 - \$course.list_type* 20
 - \$course.order_modules* 20
 - \$course.passing_score* 20
 - \$course.admin_name* 20
 - \$course.admin_email* 20
- Using Dynamic Scripting Code..... 21**
 - Conditional Statements 21
 - Using if...else...elseif* 21
 - Comparison Operators* 22
 - Looping through Arrays 22
 - Dynamic Scripting Modifiers 24
 - capitalize* 24
 - lower* 24
 - replace* 24
 - spacify* 25
 - Upper* 25
- Additional Information 26**

What is Dynamic Scripting?

Dynamic Scripting is a technology available inside Online Courses created inside Weblearning that enable you to write scripts (*code*) to do things dynamically as opposed to statically.

For the most part, the *Section Requirements* feature will enable you to set up the most frequently used rules for when something should or should not be shown, and they are the preferred method.

However, if you find yourself needing more precise control, then the *Dynamic Scripting* functionality will give you more precise control over how to control when something gets displayed and shown (*and how it will be shown*).

How does Dynamic Scripting work?

Dynamic Scripting is available for use inside your course designs from the Advanced Page Editor (*to go there, click **Courses** tab > **Manage Courses** > **Manage** (desired course), then in the **Content and Layout Tools** section, click **Edit** icon for page you want to edit, then click the **Advanced Editor** button*):

Courses > Course 101 > Course Materials Page

Page Caption

Course Materials

Course Content Sections

+ Add a New Content Section
Content Sections contain the content that will be placed on this page.

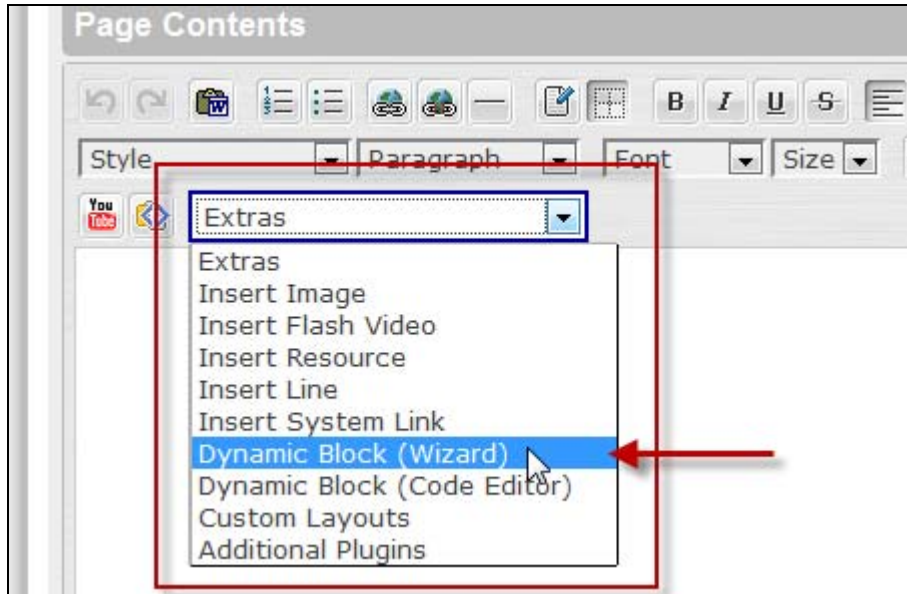
Course Progress Bars

Learning Modules Section

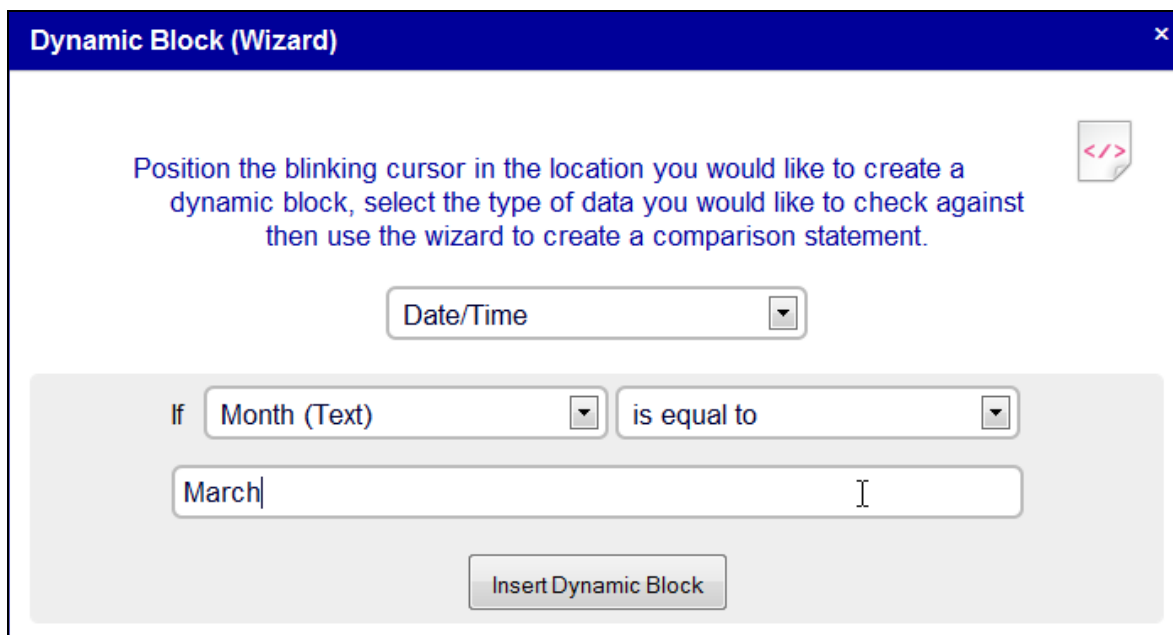
Open this page in the Advanced Editor
The advanced editor allows for full control of how the content of this page is displayed to the user.

There are two ways you can utilize *Dynamic Scripting*, both of which are available from the **Extras** menu in the Page Contents editor.

The easier-to-use option is **Dynamic Block (Wizard)**:

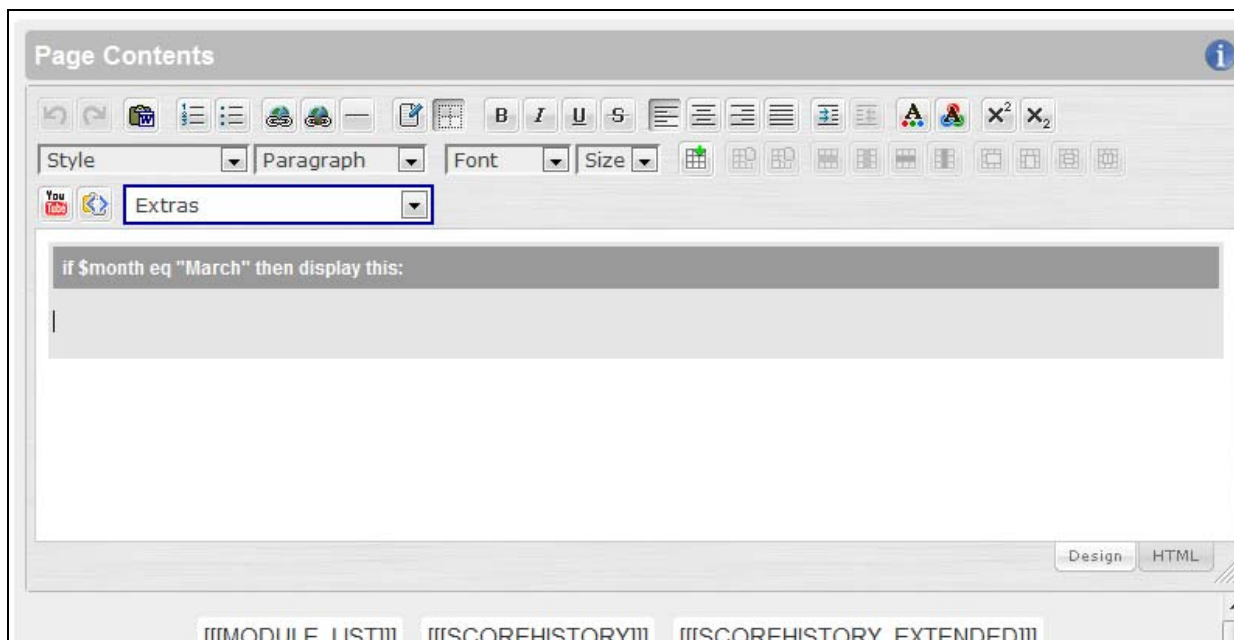


When you choose this wizard, you will not have nearly the number of possibilities you would using the code editor (*discussed next*), but when you use the wizard, you will not have to worry about writing any code:

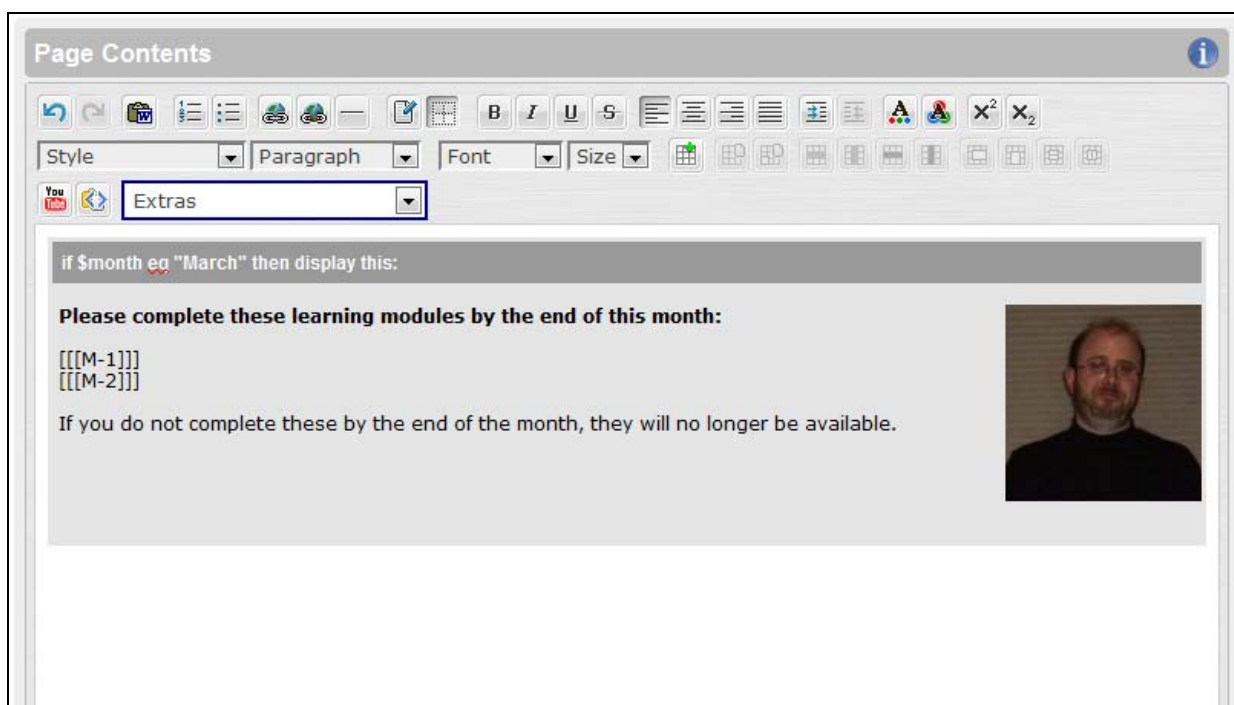


In the above example, I am creating a *dynamic block* which will display only if the month is March.

When I click the Insert Dynamic Block button, it gives me the following in the Page Contents editor:

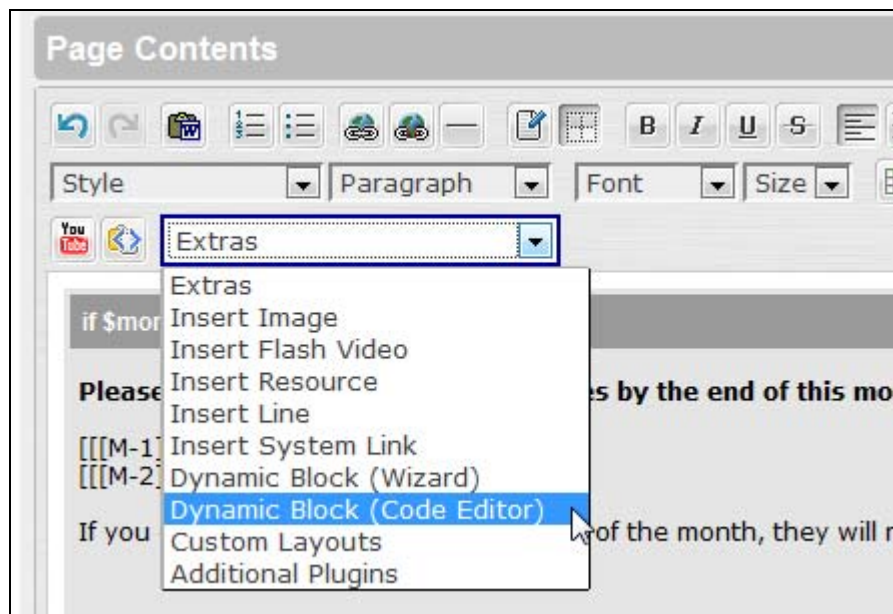


So anything I enter in the light-gray area will be shown only if the month is March:



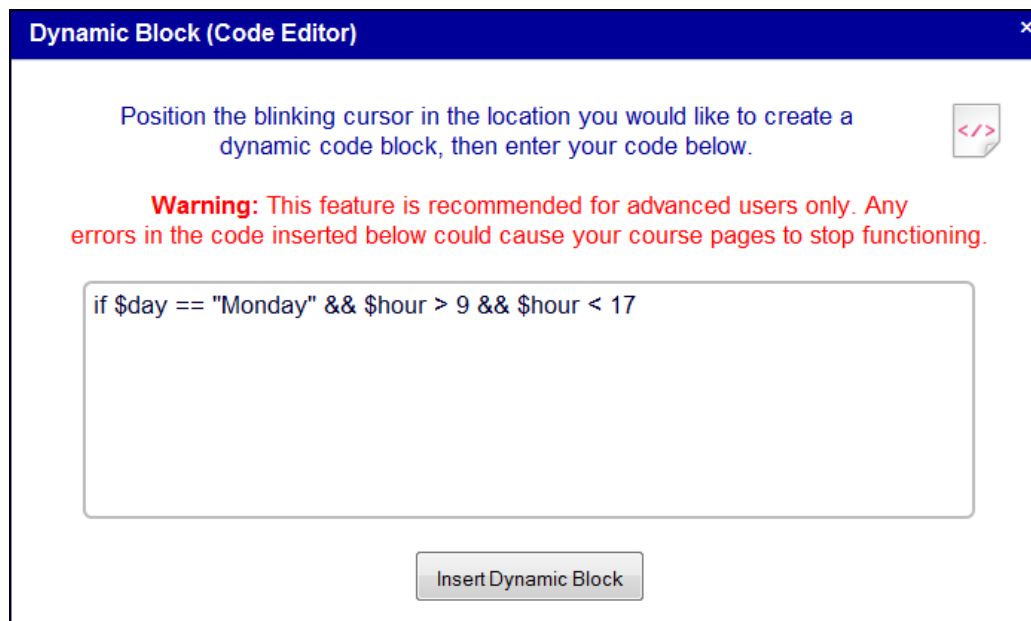
When using the wizard, you can create conditional blocks based on the date, the time of day, visitor information (*IP address*), user record data, and course data. You can also nest blocks inside other blocks (*so you can do things like if today is Monday and its after 9am and before 5pm*), although sophisticated and complex evaluations like that are best handled using the *code editor*.

The more advanced option is **Dynamic Block (Code Editor)**:

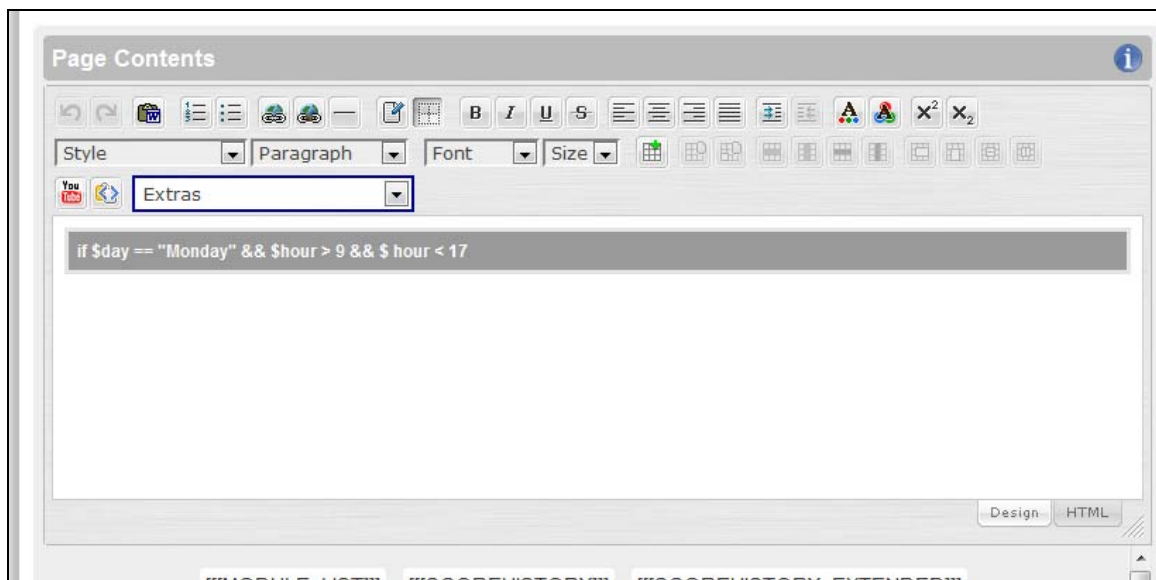


When you choose the code editor, you are essentially writing script code. This reference describes all of the variables (*dynamic values*) you have access to, as well as some example code you can use. In addition, you will also find links to the full scripting language (*Smarty*) which is a 3rd party template language being used for the *Dynamic Scripting* feature.

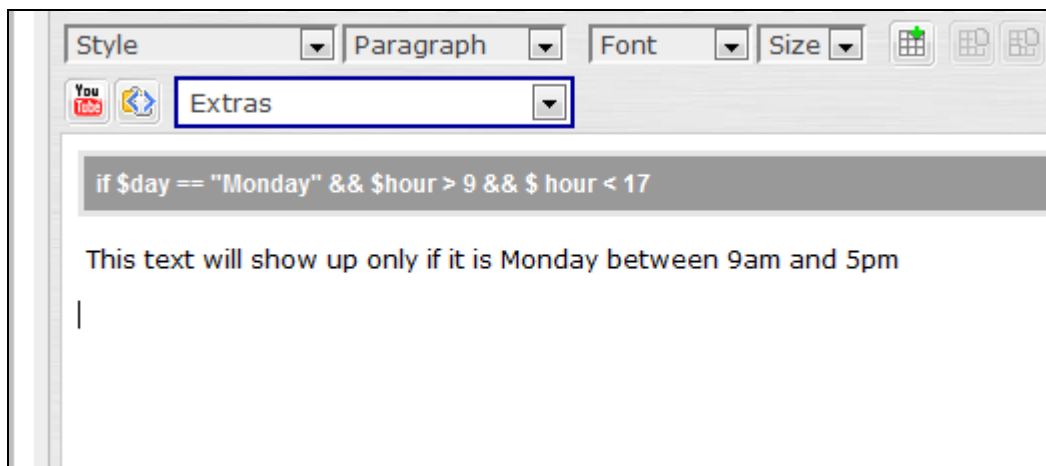
As a quick example, let's use the scenario I mentioned in the wizard that would be more efficient using the code wizard (*if today is Monday and its after 9am and before 5pm*). Here is the initial code to add to start that block:



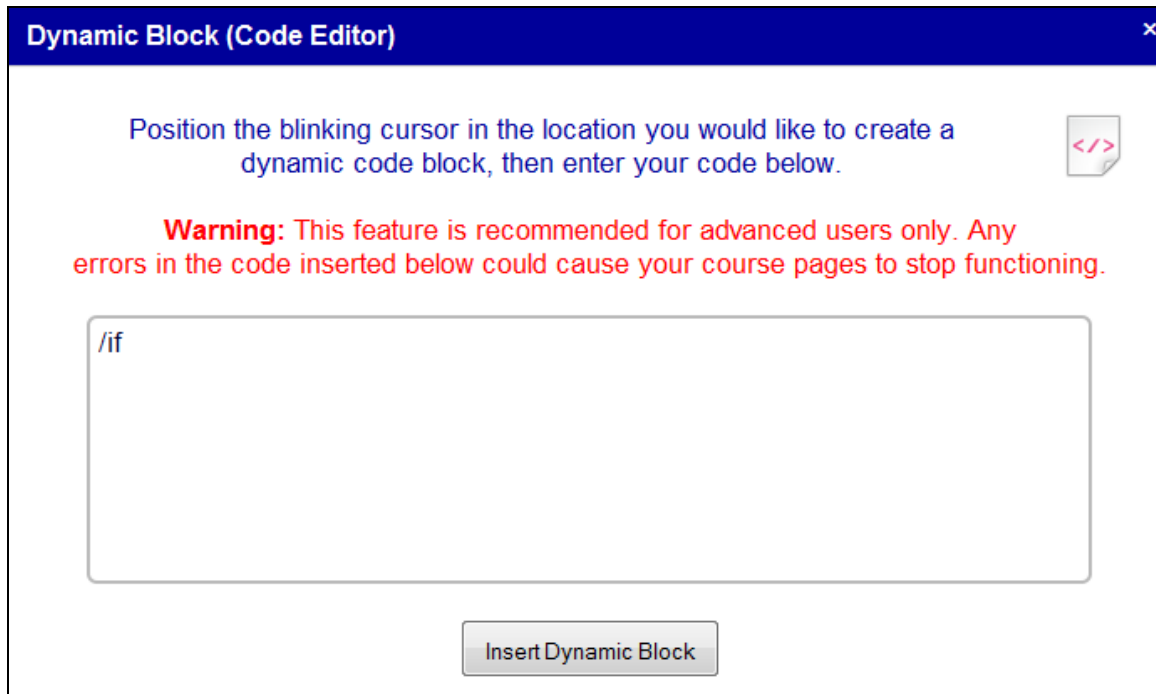
When you click the **Insert Dynamic Block** button, you should see the following:



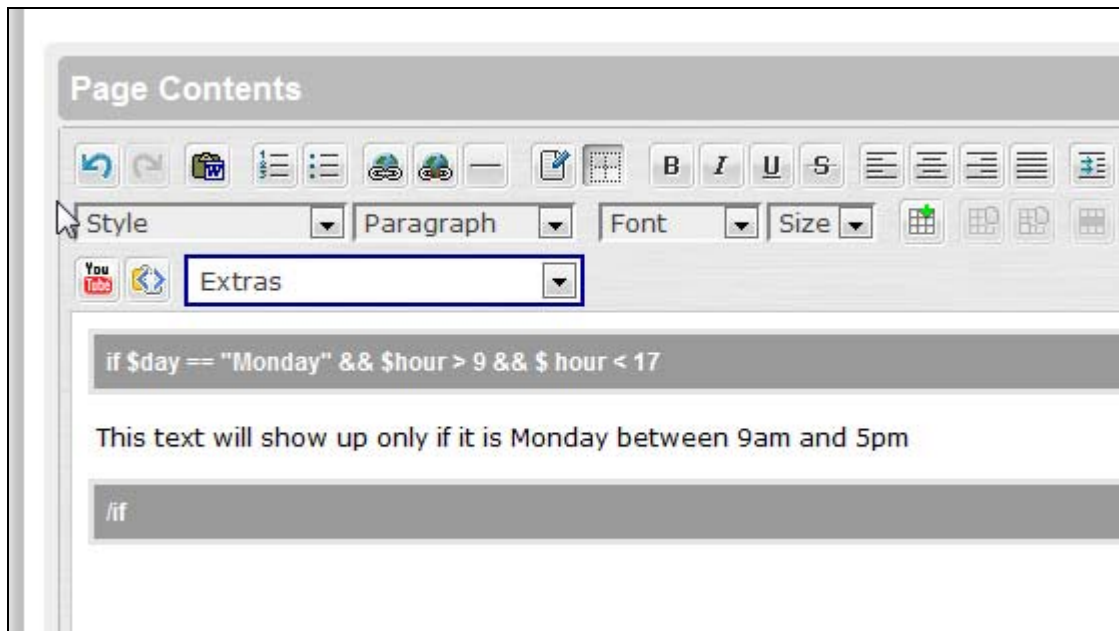
Next, we can add some content:



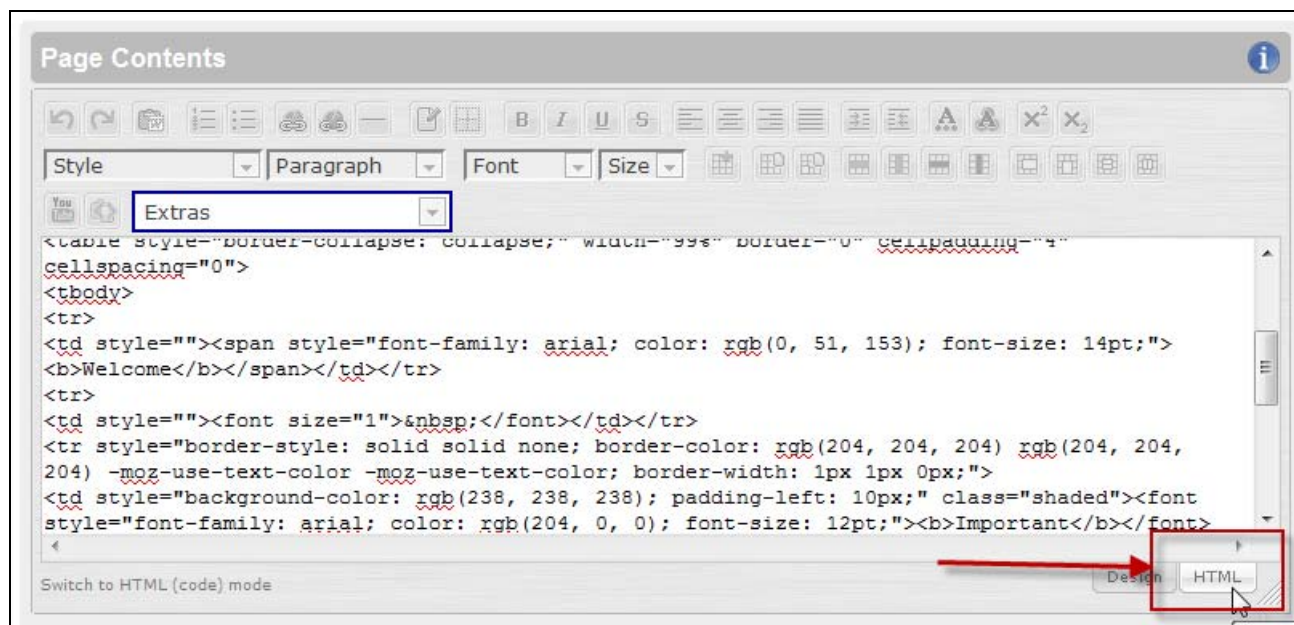
As it stands now, **this will not work**. Since we are writing the code, in this case, we need to end the block as well, which is accomplished by writing another piece of code using the *code editor*:



Clicking the **Insert Dynamic Block** button will render this:



There is actually a 3rd way to enter *dynamic scripting* code, which is to go into the HTML code of the page contents, and enter the code manually.



If you choose this route, you will need to make sure you surround all of your *dynamic scripting* code with code starting and ending tags, which are as follows:

Starting tag: `<!--CODE{`

Ending tag: `}-->`

For example:

Today is `<!--CODE{$day}-->`

In the above example, what would render to the user (*assuming today was Monday*) would be:

Today is Monday

Dynamic Scripting Variables

There are a wide variety of *Dynamic Scripting* variables available for use in your courses, although some may not be applicable to the enrollment and/or login pages of your courses.

You will notice that all variable names start with a **\$** symbol.

This section outlines all the *Dynamic Scripting* variables that are available.

Date/Time Variables

\$time

Returns the current time (*server time*) in #:## AM/PM format (3:12 AM)

\$hour

Returns the current hour (*server time*) as a number from 0 to 23

\$minute

Returns the current minute (*server time*) as 00 to 59

\$second

Returns the current seconds (*server time*) as 00 to 59

\$am_pm

Returns AM or PM

\$date

Returns current date (*server date*) in MM-DD-YYYY

\$month

Returns current month (*server month*) as a string (*January, February, etc.*).

\$month_no

Returns current month (*server month*) as a number from 1 – 12.

\$day

Returns current day (*server day*) as a string (*Monday, Tuesday, etc.*).

\$day_no

Returns current day (*server day*) as a number from 1 – 31.

\$year

Returns current year (*server year*) as a 4-digit number (*2010, 2011, etc.*).

Visitor Data Variables**\$ip_address**

Returns the visitors IP address (*67.23.153.34*)

\$host_name

Returns visitors host name (*where they are coming from*)

\$http_referer

Returns visitors referring page, meaning what page they were on before they came to this page. **This does not work when the user is using Internet Explorer.** (*http://www.somesite.com/index.htm*)

\$http_referer_domain

Returns domain name portion of the visitors referring page, meaning what page they were on before they came to this page. **This does not work when the user is using Internet Explorer.** (*www.somesite.com*)

User Data Variables

\$user.login

Returns users login id (*ie: username*)

\$user.email

Returns users email address

\$user.firstname

Returns users first name

\$user.lastname

Returns users last name

\$user.middleinit

Returns users middle initial

\$user.start

Returns users start date

\$user.expire

Returns users expiration date

\$user.company

Returns users company

\$user.address

Returns users address

\$user.city

Returns users city

\$user.state

Returns users state

\$user.zip

Returns users zip code

\$user.phone

Returns users phone number

\$user.c1

Returns users custom 1 field

\$user.c2

Returns users custom 2 field

\$user.c3

Returns users custom 3 field

\$user.c4

Returns users custom 4 field

\$user.c5

Returns users custom 5 field

\$user.c6

Returns users custom 6 field

\$user.c7

Returns users custom 7 field

\$user.c8

Returns users custom 8 field

\$user.c9

Returns users custom 9 field

\$user.c10

Returns users custom 10 field

\$user.c11

Returns users custom 11 field

\$user.c12

Returns users custom 12 field

\$user.c13

Returns users custom 13 field

\$user.c14

Returns users custom 14 field

\$user.c15

Returns users custom 15 field

\$user.notes

Returns user notes

Usergroup Data Variables

Note: this variable is an array – see the section on iterating through arrays in this documentation.

\$usergroups[]

An array containing each usergroup this user is a member of.

Other Courses User Is Enrolled In Data Variables

Note: this variable is an array – see the section on iterating through arrays in this documentation.

\$othercourses[].name

Name of the course

\$othercourses[].start

Start date of the course

\$othercourses[].expire

Expire date of the course

\$othercourses[].login_url

Login URL of the course

\$othercourses[].enroll_url

Enrollment URL of the course

User Activity Data Variables

Note: this variable is an array – see the section on iterating through arrays in this documentation.

\$activity[].title

The title of the graded quiz, presentation, or practice quiz

\$activity[].completions

Number of times the item has been completed

\$activity[].date_last_complete

Date of last completion

\$activity[].time_last_complete

Time of last completion

\$activity[].total_time

Total time (*in minutes*) spent on the item (*across all attempts*)

Quiz Data Variables (Graded Quizzes)

Note: this variable is an array – see the section on iterating through arrays in this documentation.

\$quiz[].title

Title of the quiz

\$quiz[].completions

Number of times quiz has been completed & scored

\$quiz[].date_last_complete

Date of last completion for the quiz

\$quiz[].time_last_complete

Time of last completion for the quiz

\$quiz[].highest_score

Highest score achieved for the quiz

\$quiz[].lowest_score

Lowest score achieved for the quiz

\$quiz[].average_score

Average score for quiz (*average is based on all attempts*)

\$quiz[].passing_score

The passing / required score for quiz

\$quiz[].total_time

Total time (in minutes) spend on quiz (*across all attempts*)

Course Data Variables**\$course.name**

Returns the name of current course

\$course.id

Returns the ID of current course

\$course.login_url

Returns the login URL of current course

\$course.enroll_url

Returns the enrollment URL of current course

\$course.enroll_code

Returns the enrollment code of current course

\$course.list_type

Returns the listing type of current course (*either no, enroll, or login*)

\$course.order_modules

Returns either **yes** or **no** indicating whether or not the course is configured to force users to take learning modules in order

\$course.passing_score

Returns the passing score for the current course

\$course.admin_name

Returns the course admin's login name for the current course

\$course.admin_email

Returns the course admin's email address for the current course

Using Dynamic Scripting Code

The following section outlines some sample code snippets on how to perform simple tasks within the Dynamic Scripting systems' own internal scripting language.

Conditional Statements

Conditional Statements allow you to control how or what your course displays depending on the value of one or more template variables.

Using if...else...elseif

The *if...else* statement allows you to evaluate whether or not a statement is **true** or **false** and, depending upon the result, determine what gets displayed. Usually the condition is an expression that uses a comparison operator to compare one value to another.

Example 1:

```
<!--CODE{if $day == "Friday"}-->
    Get ready for the weekend!
<!--CODE{else}-->
    It's a long way to the weekend
<!--CODE{/if}-->
```

Example 2:

```
<!--CODE{if $day == "Friday"}-->
    Get ready for the weekend!
<!--CODE{else if $day == "Wednesday" || $day == "Thursday"}-->
    Just a few more days until the weekend
<!--CODE{else}-->
    It's a long way to the weekend
<!--CODE{/if}-->
```

Comparison Operators

When comparing two values, the following comparison operators are available to you:

Qualifier	Alternates	Syntax Example	Meaning
==	eq	\$a eq \$b	equals
!=	ne, neq	\$a neq \$b	not equals
>	gt	\$a gt \$b	greater than
<	lt	\$a lt \$b	less than
>=	gte, ge	\$a ge \$b	greater than or equal
<=	lte, le	\$a le \$b	less than or equal
===		\$a === 0	check for identity
!	not	not \$a	negation (unary)
%	mod	\$a mod \$b	modulus
is [not] div by		\$a is not div by 4	divisible by
is [not] even		\$a is not even	[not] an even number (unary)
is [not] even by		\$a is not even by \$b	grouping level [not] even
is [not] odd		\$a is not odd	[not] an odd number (unary)
is [not] odd by		\$a is not odd by \$b	[not] an odd grouping

Looping through Arrays

Several *Dynamic Scripting* variables are actually an array containing several values, rolled into a single variable.

To display that array of values to the user, you'll need to loop through the array and display each item individually.

Example 1: Looping through Usergroups

```

<table>
<!--CODE{section name=item loop=$usergroups}-->
  <tr>
    <td>Group Name:</td>
    <td><!--CODE{$usergroups[item]}--></td>
  </tr>
<!--CODE{/section}-->
</table>

```

Example 2: Looping through user activity

```

<table>
<!--CODE{section name=item loop=$activity}-->
  <tr>
    <td><b>Title:</b></td>
    <td><b><!--CODE{$activity[item].title}--></b></td>
  </tr>
  <tr>
    <td>Completions:</td>
    <td><!--CODE{$activity[item].completions}--></td>
  </tr>
  <tr>
    <td>Last Completion Date:</td>
    <td><!--CODE{$activity[item].date_last_complete}--></td>
  </tr>
  <tr>
    <td>Last Completion Time:</td>
    <td><!--CODE{$activity[item].time_last_complete}--></td>
  </tr>
  <tr>
    <td>Total Time:</td>
    <td><!--CODE{$activity[item].total_time}--></td>
  </tr>
  <tr>
    <td>&nbsp;</td>
    <td>&nbsp;<td>
  </tr>
<!--CODE{/section}-->
</table>

```

Dynamic Scripting Modifiers

Dynamic Scripting modifiers allow you to easily control the appearance/display of the values held by the *Dynamic Scripting* variables outlined previously in this manual.

Normally, to display a variable in a course you would simply insert the name of the *Dynamic Scripting* variable between *code tags*.

Example:

```
<!--CODE{$site_name}--> = My website
```

Dynamic Scripting modifiers enable you to control how the data is displayed to the user. Below is a listing of commonly used modifiers.

capitalize

Capitalizes the first letter of every word

i.e.

```
<!--CODE{$site_name|capitalize}--> = My Website
```

lower

Makes an entire string lower-case

i.e.

```
<!--CODE{$site_name|lower}--> = my website
```

replace

A simple search-and-replace on a string

i.e.

```
<!--CODE{$site_name|replace:'website':'course'}--> = My course
```


spacify

Inserts a space between every character in a variable

i.e.

```
<!--CODE{$site_name|spacify}--> = My website
```

Upper

Converts a string to uppercase

i.e.

```
<!--CODE{$site_name|upper}--> = MY WEBSITE
```

Additional Information

The Weblearning *Dynamic Scripting* system is powered by the *Smarty Template Engine*. For additional documentation on using templates, tips and tricks, and more examples you can visit the Smarty website at <http://www.smarty.net/>

Here are some useful links that were valid at the time this documentation was written:

Smarty Online Documentation:

<http://www.smarty.net/manual/en/>

Smarty Example Code:

http://www.smarty.net/sampleapp/sampleapp_p1.php

Smarty Frequently Asked Questions

<http://smarty.incutio.com/?page=SmartyFrequentlyAskedQuestions>