

FEMA/EMI Courseware Template Instructions

Version 2.0

March 2010



FEMA

Table of Contents

Table of Contents	1
Creating a Course: The Basic Structure	3
Installing and Using the Template Files	5
index.htm	7
indexMenu.htm.....	8
mainCode.js	9
glossary.htm.....	10
Using the Templates.....	12
MainContent.....	13
MainContentFluffRight	14
MainContentBigVideo	15
MainContentClickToRead	16
MainContentExercise	17
MainContentExercise: Check All That Apply.....	18
MainContentExercise: ColumnRow.....	21
MainContentExercise: Compare Text	25
MainContentExercise: Matching.....	27
MainContentExercise: MultiChoice	30
AudioTranscript.....	32
EndScreen	33
ExtraScreen	34
FramedText.....	35
FrameSet	36
Creating Links.....	38
Customizing Dreamweaver.....	40

This series of guidelines will show you how to use Dreamweaver to create your own course. When doing so, please note the following:

- These instructions are formatted for using Dreamweaver in either the Code or Split view. When working directly with the HTML code, you will remain in complete control of your programming and do not have to worry about Dreamweaver "guessing" that it knows what you want. It is impossible to program a course with Dreamweaver only in Design view.
- Code fragments are presented in a different (`fixed width`) font to immediately differentiate them from the descriptions above and below. In large portions of code, the Word highlight function is used to mark where you, the user, will add or alter the code accordingly.
- These instructions assume a basic understanding of both HTML and Dreamweaver.
- The templates were designed in and intended for Dreamweaver CS4, but should be applicable in earlier versions.

This section will walk you through the framework of setting up a course, the splash screen, the lesson menu, and several other minor files.

The next section will show you how to create individual screens to populate your lessons.

Note: These instructions assume that you have installed the Dreamweaver Extension (http://training.fema.gov/devres/docs/HT_2010v2_0.zip).

Creating a Course: The Basic Structure

Most courses for FEMA are set up in the same structure. There is a splash screen that leads the student to a course menu. From there, they select a lesson and move through all of that lesson's screens. While all of this looks transparent to the student, it is set up in a hierarchy of files. Before you create a course, it is best to understand how everything will fit together.

If you took the files from the main directory and turned them into a hierarchical bulleted list, a small two-lesson course might look something like this:

- index.htm
 - indexMenu.htm
 - SMPL0101000.htm
 - SMPL0101010.htm
 - SMPL0101020.htm
 - SMPL0101030.htm
 - SMPL0101040.htm
 - SMPL0101050.htm
 - SMPL0101060.htm
 - SMPL0101070.htm
 - SMPL0101080.htm
 - SMPL0102000.htm
 - SMPL0102010.htm
 - SMPL0102020.htm
 - SMPL0102030.htm
 - SMPL0102040.htm
 - SMPL0102050.htm
 - SMPL0102060.htm
 - SMPL0102070.htm
 - SMPL0102080.htm
 - SMPL0102090.htm
 - SMPL0102100.htm
 - SMPL0102110.htm
 - SMPL0102120.htm
 - SMPL0102130.htm
 - SMPL0102140.htm

The first level is the splash screen. This file should always be named index.htm.

In a multi-lesson course, the second level is the Course Menu. This file should be named indexMenu.htm. This is the screen where the student gets a list of all of the lesson titles and links to the lessons. In a single-lesson course, the index.htm page directly launches the lesson.

The third level contains the FrameSet files for each lesson. This is a file that lists all of the individual screens within a lesson, and the order in which they appear. You will learn how to create this file in the section on the FrameSet template. The last three digits of this filename are typically 000.

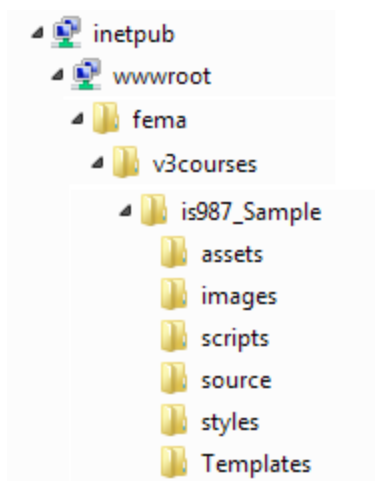
The fourth level of files is the individual pages of each lesson, and is created using the various MainContent templates.

Installing and Using the Template Files

To begin developing a new course, we recommend creating a new folder for the course and then copying all the sample files there, and then do the following:

1. Rename all the files (change **SMPL01** to the new prefix (i.e. **NIMS01**)). We use a great little utility called RenameWiz [<http://www.renamewiz.com>].
2. Create the new site in Dreamweaver.
3. Perform a Search and Replace to search the source code and change all instances of **SMPL01** to the new prefix (i.e. **NIMS01**).
4. Empty out the /assets folder except for files that start with temp* and the spacer.gif file.
5. Delete all the non-boilerplate files from the root (SMPL0101070.htm and subsequent).
(Note from the developer: I personally like to leave the frameset page, first page, and last page from the other lessons past lesson 1, just makes it easier that re-creating them from scratch).
Be sure to leave all the *.xsd and *.xml files. These are the SCORM files.

The directory structure should look something like this:



File Naming Conventions

The standard page filename is formatted like the following: SMPL0101010.htm

This filename breaks down in the following way:

SMPL0101010.htm = This can be a two- to four-character string abbreviation of the course title.

SMPL**01**01010.htm = This is a two-digit number corresponding to the course module.

SMPL01**01**010.htm = This is a two-digit number corresponding to the lesson number.

SMPL0101**010**.htm = This is a three-digit number corresponding to the page number. Use 010 for page 1, 020 for page 2, 100 for page 10, and so on. This gives you room to grow later; that way if you need to add a page in the middle of a lesson after the lesson has been programmed, you can create a filename with a number between the two existing screens. For example, let's say you have these pages:

SMPL0101010.htm

SMPL0101020.htm

SMPL0101030.htm

SMPL0101040.htm

If you needed to add a screen after page 3, you could name the file SMPL0101035.htm. Then, you would edit your FrameSet file (explained later in the documentation) to insert the new filename where it belongs:

SMPL0101010.htm

SMPL0101020.htm

SMPL0101030.htm

SMPL0101035.htm

SMPL0101040.htm

So, for instance, if your course had three modules with two lessons in each, the filename for the fifth page of the second lesson in the third module would be SMPL0302050.htm.

Other considerations include files for ancillary uses, such as 508 long descriptions, video/audio transcripts, assets, etc.

SMPL0101030**d**.htm = Use a lowercase d to denote a 508 long description <longdesc>.

SMPL0101030**text**.htm = Use to denote a transcript file.

SMPL0101030.jpg = Use the same naming convention for the assets.

SMPL0101summary.htm = Use a similar naming convention for summary pages.

Next, we will learn how to create the first- and second-level screens.

index.htm

This is the splash screen of your course. All splash screens should contain the course name and number at the top of the screen. There are two places in the code that you will need to insert this information as well:

```
<TITLE>IS-987.A - Sample Course</TITLE>
```

This is the title tag; updating it here will have the name of the course appear in the browser title bar.

```

```

You also need to update the alt tag for the graphic on the splash screen, for students who are using a screen reader or assistive device.

```
<a href="indexMenu.htm" onMouseOver="document.begin.src='images/splashBeginHi.gif';" onMouseDown="document.begin.src='images/splashBeginDn.gif';" onMouseOut="document.begin.src='images/splashBegin.gif';"></a>
```

Toward the bottom of the screen is a Begin the Training button.

If this is a multi-lesson course, the link on this button should be to: `indexMenu.htm`

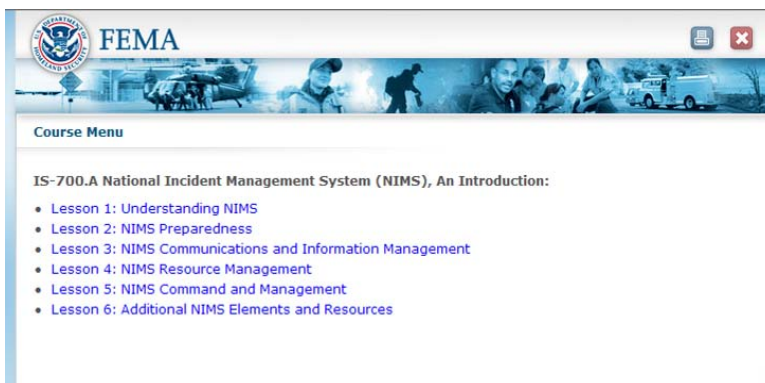
If this is a single-lesson course, the link on this button is slightly more complex. Later in this course you will learn how to create a file that lists all of the pages in a lesson, using the `FrameSet` template. You will need to use a javascript code called `LoadLesson` to call that file. If your filename is `SMPL0101000.htm`, then your code would be:

```
<a href="javascript:LoadLesson('SMPL0101000.htm')">
```



indexMenu.htm

The indexMenu.htm file serves as a course menu for your course. The setup in the indexMenu.htm file is simple; a bolded name of the course title, then a bulleted list of all the lesson names and a javascript code called LoadLesson to launch each lesson.



That section of the indexMenu.htm file should look something like this:

```
<p><strong> IS-987.A - National Incident Management System (SMPL), An
Introduction:</strong></p>
<ul>
  <li><a href="javascript:LoadLesson('SMPL0101000.htm')">Lesson 1: Understanding
SMPL</a></li>
  <li><a href="javascript:LoadLesson('SMPL0102000.htm')">Lesson 2: SMPL
Preparedness</a></li>
  <li><a href="javascript:LoadLesson('SMPL0103000.htm')">Lesson 3: SMPL
Communication and Information Management</a></li>
  <li><a href="javascript:LoadLesson('SMPL0104000.htm')">Lesson 4: SMPL Resource
Management</a></li>
  <li><a href="javascript:LoadLesson('SMPL0105000.htm')">Lesson 5: SMPL Command
and Management</a></li>
  <li><a href="javascript:LoadLesson('SMPL0106000.htm')">Lesson 6: SMPL
Additional SMPL Elements and Resources</a></li>
</ul>
```

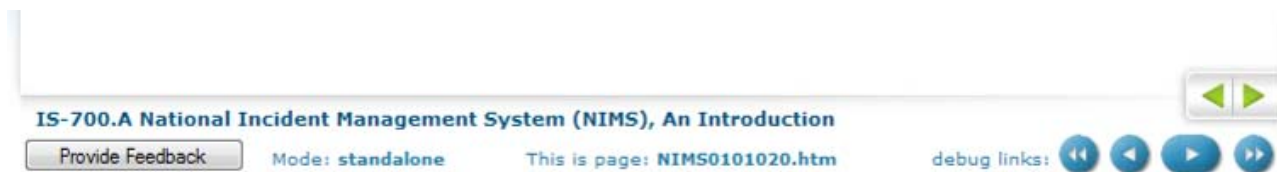
You should alter the indexMenu.htm file to list the lessons in your own course, plus update the filenames to the correct FrameSet files for each lesson.

mainCode.js

In the scripts subdirectory is mainCode.js – which holds the bulk of the javascript coding for the course. The first two lines are the only ones you will need to edit while working on your course. Do NOT alter any other part of this file.

```
var debug = 'true';  
var KCcheat = 'true';
```

To turn on either or both of these variables, let these equal true. To turn them off, let them equal false.



The first, debug, turns on the screen numbers at the bottom of the screen, as well as offers up additional back and next buttons (plus buttons to jump to the start and end of the lesson). While working on a course you may find it helpful to have this function turned on. This will allow reviewers to know instantly what screen they are on, as well as speed up the ability to move through the lesson to look at a specific page. Having debug turned on also allows reviewers to click on the "Submit Feedback" button to leave a note about a specific page.

Instructions: Beside the following list, indicate whether each statement is consistent with NIMS. Click on the Submit button when you are finished.

Statements	NIMS Is	NIMS Is Not
Scalable so it may be used for all incidents (from day-to-day to large-scale)	<input checked="" type="radio"/>	<input type="radio"/>
A response plan used exclusively during incidents requiring Federal coordination	<input type="radio"/>	<input checked="" type="radio"/>
Doctrine followed by those who have highly specialized, tactical response training	<input type="radio"/>	<input checked="" type="radio"/>
A set of preparedness concepts and principles for all hazards	<input checked="" type="radio"/>	<input type="radio"/>
An incident organization chart based on the Incident Command System	<input type="radio"/>	<input checked="" type="radio"/>

The second, KCcheat, automatically enables the next button on Knowledge Check/Exercise screens. While working on a course you may find it helpful to have this function turned on, to speed moving through a lesson. This also highlights or fills in the correct answer, to assist reviewers. Firefox does not support the highlighting of form elements.

glossary.htm

All FEMA courses contain a glossary. The glossary file is set up with an A to Z list of letters at the top of the screen that serve as bookmarks to definitions beginning with that letter, and then all of the terms in alphabetical order.

In the list of bookmarks, each letter that has terms beginning with that letter should receive a link that looks like the following:

```
<a href="#A">A</a>
```

If there are no terms beginning with that letter, do not add a bookmark.

At the start of each group of terms beginning with a specific letter, you will need to create an anchor link to let the Web site know that letter's section begins there. An anchor should look like this:

```
<a name="A"></a>
```

After you have concluded a letter's terms, you may want to include a "Back to Top" button and a dividing line before starting the next letter's terms. (If you have a particularly short glossary, this step can sometimes be skipped; use your own best judgment.) The code for the button and dividing line should look like this:

```
<p><a href="#top"></a></p>  
<hr>
```

The start of a glossary could look something like this:

```
<p align="center"><a href="#A">A</a> | <a href="#B">B</a> | C | <a href="#D">D</a> |  
E | <a href="#F">F</a> | <a href="#G">G</a> | <a href="#H">H</a> | <a href="#I">I</a>  
| J | <a href="#K">K</a> | <a href="#L">L</a> | <a href="#M">M</a> | <a  
href="#N">N</a> | O | <a href="#P">P</a> | Q | <a href="#R">R</a> | <a  
href="#S">S</a> | <a href="#T">T</a> | U | V | W | X | Y | Z </p>
```

```
<a name="A"></a>
```

```
<p><b>Alternate Project:</b> When the Tribe determines it not to be in the public  
interest to repair a damaged facility, funds may be used to repair or expand other  
Tribal facilities, construct new facilities, or fund hazard mitigation measures.  
Federal funding will be limited to 75 percent of the Federal share of the approved  
estimate of eligible costs or 90 percent for publicly owned facilities on unstable  
soil.
```

```
<p><b>Application Period:</b> The 30-day period following designation of the area in  
which the damage is located during which the Tribe's Authorized Representative (TAR)
```



submits a completed Request for Public Assistance (RPA) form to the Federal Emergency Management Agency (FEMA) Regional Director.

<p></p>

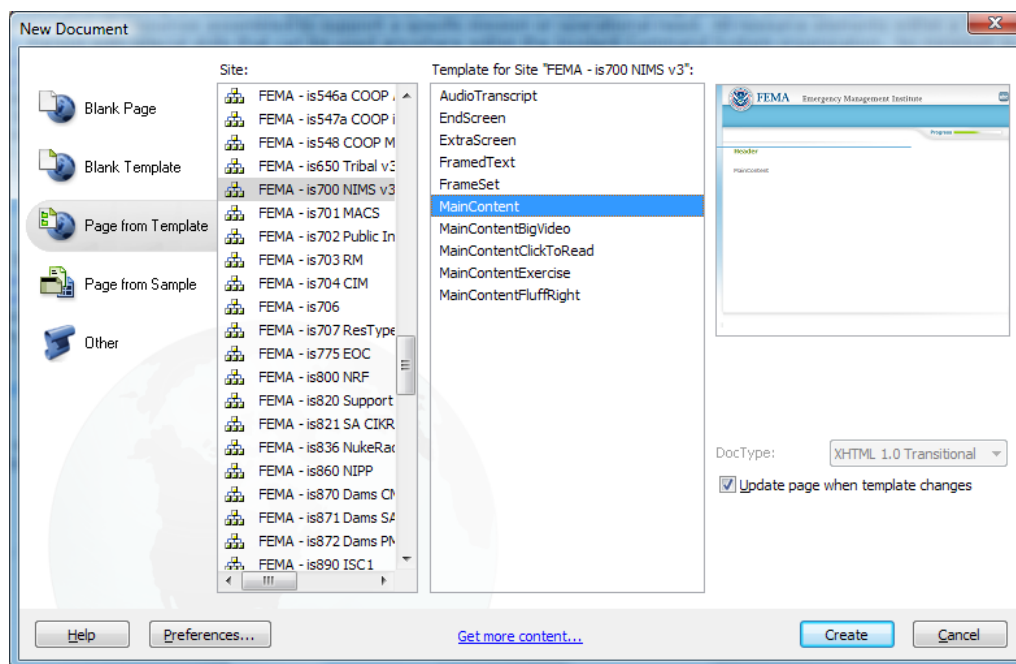
<hr>

<p>Betterment: Any work performed by the Tribe that exceeds restoring a facility to its pre-disaster design. Betterment also includes repairs or replacements to Tribal real property that are necessary to mitigate a pre-disaster condition that the Tribe had a pre-disaster duty to mitigate or correct.

Etc...

Using the Templates

If you have never created a file in Dreamweaver using a template, the process is fairly easy. Tell Dreamweaver to create a new page (Ctrl-N, or go to File > New).



- In the first column, select **Page from Template**.
- In the second column, select the site you are currently working within.
- In the third column, select the template you wish to use. Click on **Create** and Dreamweaver will create a new file for you using that template.

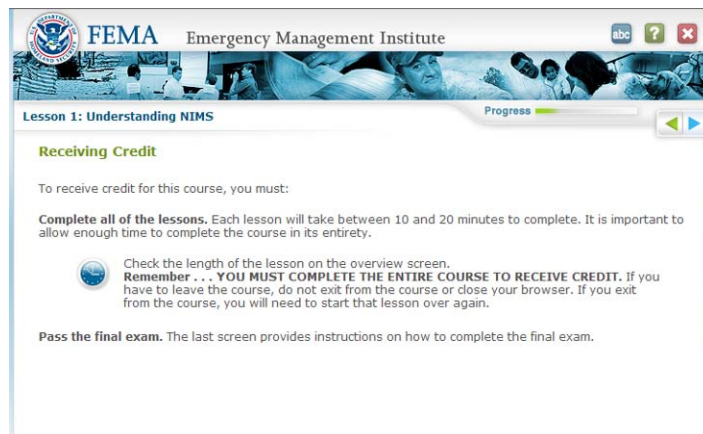
If, down the line, you need to change something on all screens using a template, you can update that template file (located in the Templates directory) and Dreamweaver will automatically update all files using that specific template.

Next, we'll go through each template style to learn what they do and how you will use them.

MainContent

Almost all of the screens you will create in a course use one of the MainContent family of screens. While the plain MainContent screen itself is not the most frequently used, all the other screens are a variant of this initial building block. After explaining the plain MainContent screen, we will then go through all of the variants.

The plain MainContent screen is good for screens with particularly wide graphics, Flash animations, or screens that simply have no graphics.



```
<div id="content" align="left"><a name="content"></a><h1><!-- InstanceBeginEditable name="PageHeader" -->Header<!-- InstanceEndEditable --></h1>
```

This portion of the code controls the individual title of this screen. Replace the placeholder "Header" with your title.

```
<!-- InstanceBeginEditable name="MainContent" --><p>MainContent</p><!-- InstanceEndEditable -->
```

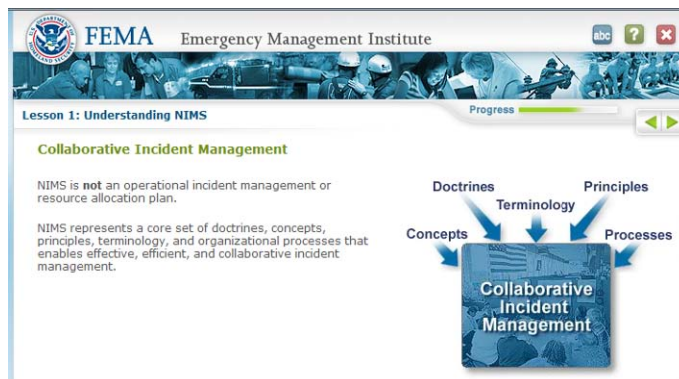
This section of the screen contains the actual teaching content of the course. Replace the placeholder "MainContent" here with the actual text of the screen.

```
<!-- InstanceBeginEditable name="PageInstructions" -->Click on Next to continue.<!-- InstanceEndEditable -->
```

At the bottom of each screen is optional informational text telling the user what to do to advance to the next screen, for screens that are out of the ordinary. If you need to change this for a single screen, you can update it here.

MainContentFluffRight

This template is the most commonly used out of all the MainContent family of templates. On these screens, the student will have text on the left side of the screen, and a graphic on the right side of the screen. The difference is in the second editable section on this screen.



```
<table width="100%" border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td valign="top" width="100%"><p>MainContent</p></td>
    <td valign="top" style="padding-left:25px;"></td>
  </tr>
</table>
```

This section of the code contains both the screen's teaching text, plus the fluff graphic, all contained within a single table.

The first cell of the table is straightforward; it starts with a single paragraph with the placeholder "MainContent." Paste your new screen text over top "MainContent."

The second cell of the table holds a temporary graphic. Built into the code is a style that provides 25 pixels of padding on the left-hand side of the cell. This keeps the text from running into the edge of the image and looks visually more appealing. This way there is also no need to manually add a large border of empty space on your graphics. By using a table to format the page, it keeps text from wrapping underneath the graphics.

When you replace the placeholder graphic (assets/temp.jpg), also make sure to update the alt text for your new graphic. Placeholder graphics are assigned a "tempGraphic" alt text by default.

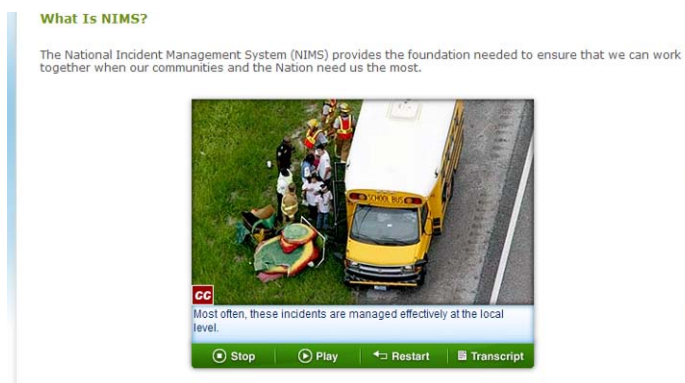
If you need to have the graphic on the left-hand side of the screen, simply swap the two cells in the source code; then, change the "padding-left:25px;" to "padding-right:25px;" so that the buffer between the text and the image is in the correct position.

Note: Although HTML tables aren't normally desired for layout, there are instances like this that make them indispensable. CSS cannot properly format this sort of layout, where the fluff graphic may be any width and having the text not wrap under the graphic is desired.

MainContentBigVideo

MainContentBigVideo is used to house a Flash-based video in the center of the screen, ready for the student to view. As with all of the "MainContent" family of screens, it is the second section of code that is different.

Because most of the code will not need to change, only the sections you will alter are reproduced below.



```
<!-- InstanceBeginEditable name="MainContent" --><p>MainContent</p>
```

You may want to have text above the video, serving as an introductory sentence or two. If so, replace the MainContent placeholder with your new text. Otherwise, delete this placeholder.

```
<object id="audioMovie" data="assets/filename.swf" type="application/x-shockwave-flash" width="375" height="267">  
  <param name="movie" value="assets/filename.swf" />  <param name="quality" value="high"/></object>
```

This is the code for the Flash video itself. The player is set up for a video at 375x267 pixels. The template will insert a placeholder filename of "assets/filename.swf" for your video. Simply swap out this filename for the correct name of your video. When you swap out the placeholder for the real filename, make sure you change it in **both** locations within the code.

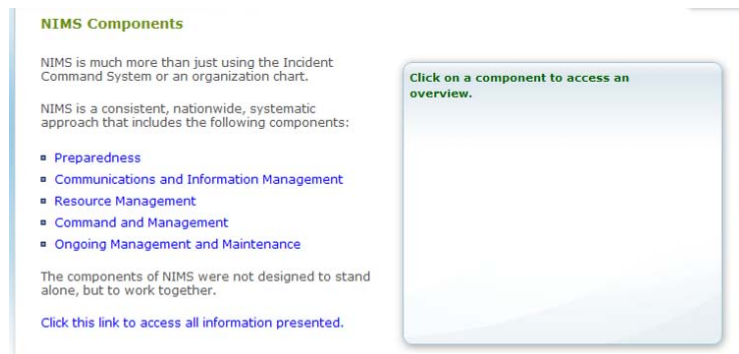
```
<td><a href="JavaScript:SendMeSummary('SMPL0102030text.htm')"  
onmouseover="document.video4.src='images/vidholderBigHi_09.gif';"  
onmouseout="document.video4.src='images/vidholderBig_09.gif';"  
onmousedown="document.video4.src='images/vidholderBigDn_09.gif';"></a></td>
```

The last portion of this screen you will need to update is the link on the Transcript button. The link for the transcript button uses a piece of JavaScript called SendMeSummary. Simply change the filename within the parentheses and apostrophes to the correct filename containing the transcript.

508 Note: You will want to insert a “stop” frame in the beginning of the Flash timeline so that the media does not start automatically. EMI does not want the media to “talk over” screen reading devices such as JAWS. HT normally inserts it after the preloader, but before the audio begins.

MainContentClickToRead

MainContentClickToRead is used to create a screen where students click on links on the left-hand side of the screen, to update text that appears on the right-hand side of the screen. The text that appears in the iframe on the right-hand side of the screen is created using the FramedText template.



As with all of the "MainContent" family of screens, it is the second section of code that is different. Because most of the code will not need to change, only the sections you will alter are reproduced below.

Left-Hand Code:

```
<table width="100%" border="0" cellpadding="0" cellspacing="0">
  <tr>
    <td valign="top" width="100%"><p>MainContent</p>
    <!-- <a href="javascript:showInIframe('filenamef1.htm');">link</a> ---->
      <a
href="javascript:SendMeSummary('filenametext.htm')">Click this link to access all
information presented.</a>
```

Generally, you will want to put your content in the MainContent placeholder space. There are two important pieces of code that are provided and will need to be applied to your content. The first is the code for links that will call up the new text in the right-hand frame. Immediately below the MainContent placeholder is a commented-out sample link, using a piece of javascript code named showInFrame. Change filenamef1.htm for each link to the name of the file that you want to appear in the right-hand frame.

Right-Hand Code:

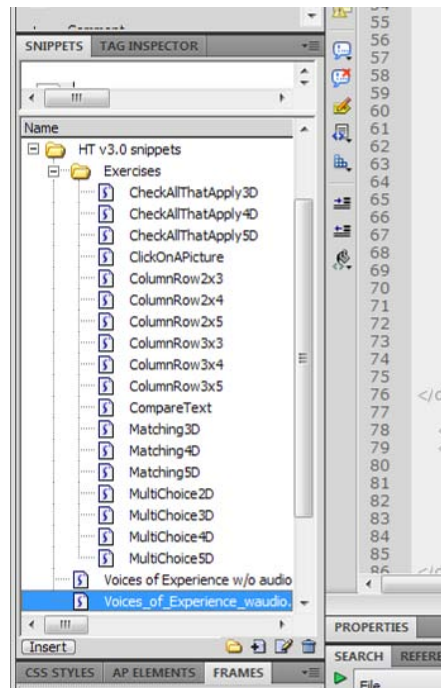
```
<iframe src="filenamef0.htm" name="definitions" width="312" height="270"
frameborder="0" scrolling="Auto" id="definitions" title="definitions" alt="Scroll
through these definitions to learn more."></iframe>
```

This is the only piece of code you will need to update for the right-hand frame. Swap out the filenamef0.htm placeholder name with the name of the FramedText screen that you wish to initially appear on this page when the user first sees it. Generally, that opening screen should include some sort of instruction to let the user to know they need to click links on the left-hand side for the frame to change.

MainContentExercise

MainContentExercise is almost identical to MainContent, with the one exception being that "Click on Next to continue." at bottom of the screen is instead pre-filled in with "Complete the exercise to continue."

You must first install the Dreamweaver extension HT_2010v3_1.mxp to load the snippets we are about to discuss.



When using the MainContentExercise screen, you will need to use the pre-generated exercise Dreamweaver Snippets to add in the code for a Knowledge Check. Click on the Snippets tab (next to Files and Assets in Dreamweaver's file tree panel), and from there select **HT v3.0 Snippets**. Each style of snippet will be described in its own page, following.

MainContentExercise: Check All That Apply

When using an exercise that allows the user to select multiple answers (or distracters), use the Check All That Apply snippet. Already set up are ones that use three distracters, four distracters, and five distracters.

In the MainContent placeholder, you should place instructions and the question for the user. There is an "instruction" class already set up which is used as a standard look for the opening statement on the screen.

```
<tr>
  <td valign="top" width="20"><input type="checkbox" name="Quiz"
    value="1" id="Quiz1"></td>
  <td valign="top" width="100%"><label for="Quiz1"></label></td>
</tr>
```

The distracter text should be inserted between the start and end of the label tag. The best way to do this is copy the text from the storyboards and paste it into the MainContent placeholder, and then highlight and drag the distracter text down into the table cell. Verify that the label tags are still intact. A completed distracter should look like this:

```
<tr>
  <td valign="top" width="20"><input type="checkbox" name="Quiz"
    value="1" id="Quiz1"></td>
  <td valign="top" width="100%"><label for="Quiz1">This is the first possible
  answer</label></td>
</tr>
```

If you need more than five distracters, copy and paste the existing code for a single distracter after all of the pre-generated code, then increase the number accordingly. For example, if you needed a sixth distracter the code would be:

```
<tr>
  <td valign="top" width="20"><input type="checkbox" name="Quiz"
    value="6" id="Quiz6"></td>
  <td valign="top" width="100%"><label for="Quiz6"></label></td>
</tr>
```

After you have set up your distracters, you need to update the correct answer and feedback.

```
<script language="JavaScript" src="scripts/FeedbackCA.js"
type="text/javascript"></script>
<script language="JavaScript" type="text/javascript">
var Feedback = "" // you will need to "escape" all double quotes in this area with
back slashes. (i.e. ...he said. \"this is great\")
var vCorrectAnswer = "" // enter 1s and 0s as checked and unchecked (i.e. if items 2
and 3 should only be checked, enter 011)
var vTriesAllowed = 2
showCorrect();
</script>
```

var Feedback is where you will paste the text that a user receives when the Knowledge Check is over, regardless of whether they got it right or wrong. This should include correct HTML formatting (paragraph tags, list tags, etc.), but all the HTML must be on a single line. If you have a particularly detailed set of feedback, you may want to paste it into a blank screen, then cut and paste the HTML between the quotation marks.

var vCorrectAnswer is where you set the correct answer. The answer is formatted as a series of 0s and 1s, where a 0 means that the box should not have been checked off, while a 1 means that the box should have been checked off. For instance, if the correct answer was checking off items 1 and 3, but not checking off item 2, you would enter 101 between the quotes.

var vTriesAllowed lets the code know how many tries the user gets before they are told the correct answer and are allowed to move on to the next screen. For this type of exercise, two tries are traditionally allowed.

A completed Knowledge Check might look like this:

```
<p class="instruction">Instructions: Click on ALL the correct answers. Then, click on Submit.</p>
<p>What colors do you typically see in the sky?</p>
<form name="form1" action="">
<table width="95%" border="0" cellspacing="0" cellpadding="3">
  <tr>
    <td valign="top" width="20"><input type="checkbox" name="Quiz" value="1" id="Quiz1"></td>
    <td valign="top" width="100%"><label for="Quiz1">Blue</label></td></tr>
  <tr>
    <td valign="top" width="20"><input type="checkbox" name="Quiz" value="2" id="Quiz2"></td>
    <td valign="top" width="100%"><label for="Quiz2">White</label></td></tr>
  <tr>
    <td valign="top" width="20"><input type="checkbox" name="Quiz" value="3" id="Quiz3"></td>
    <td valign="top" width="100%"><label for="Quiz3">Green</label></td></tr>
  <tr>
    <td>&nbsp;</td><td valign="top"><a href="javascript:Clicked()" onmouseover="document.submitme.src='images/buttonSubmitHi.gif';" onmouseout="document.submitme.src='images/buttonSubmit.gif';" onmousedown="document.submitme.src='images/buttonSubmitDn.gif';"></a></td></tr>
</table>
<div id="feedbackDiv"></div>
</form>

<script language="JavaScript" src="scripts/FeedbackCA.js" type="text/javascript"></script>
<script language="JavaScript" type="text/javascript">
var Feedback = "<p>You would typically see <strong>blue</strong> and <strong>white</strong> in the sky.</p><p>Green is normally found on the ground, not
```

```
in the sky.</p>" // you will need to "escape" all double quotes in this area with
back slashes. (i.e. ...he said. \"this is great\")
var vCorrectAnswer = "110" // enter 1s and 0s as checked and unchecked (i.e. if items
2 and 3 should only be checked, enter 011)
var vTriesAllowed = 2
showCorrect();
</script>
```

MainContentExercise: ColumnRow

The ColumnRow exercise snippet is used when you have a series of statements, with two or more columns to then check off as possible responses. For example:

Are each of these items an animal, a vegetable, or a mineral?

Item	Animal	Vegetable	Mineral
Platinum			
Border Collie			
Eggplant			

This type of setup is also used for a series of true-or-false questions, since you can stack multiple questions onto a single page.

Snippets are set up for options using two or three categories, and with three, four, or five questions/statements.

In the MainContent placeholder, you should place instructions and the question for the user. There is an "instruction" class already set up which is used as a standard look for the opening statement on the screen.

First, you need to add in the column headings.

```
<tr>
  <th scope="col">Statement</th>
  <td width="1"></td>
  <th scope="col"></th><!-- Edit the column heading to suit your needs -->
  <th scope="col"></th><!-- Edit the column heading to suit your needs -->
  <th scope="col"></th><!-- Edit the column heading to suit your needs -->
</tr>
```

Simply add in the names of the column headings between the <th> tags, where noted by the comments.

Next, you will need to add in your questions/statements, plus update the title tags for the radio buttons.

```
<tr class=rowColor1>
  <td width="100%" valign="top"></td>
  <td></td>
  <td width="50" valign="top" align="center"><input type="radio" name="Quiz1"
    id="Quiz1a" value="1" title=""></td>
  <td width="50" valign="top" align="center"><input type="radio" name="Quiz1"
    id="Quiz1b" value="2" title=""></td>
  <td width="50" valign="top" align="center"><input type="radio" name="Quiz1"
    id="Quiz1c" value="3" title=""></td>
</tr>
```

The statement goes between the first set of <td> tags. Then, update the title tag for each button to show both the statement and the potentially selected answer. An updated version might look like this:

```
<tr class=rowColor1>
  <td width="100%" valign="top">Platinum</td>
  <td></td>
  <td width="50" valign="top" align="center"><input type="radio" name="Quiz1"
    id="Quiz1a" value="1" title="Platinum: An animal"></td>
  <td width="50" valign="top" align="center"><input type="radio" name="Quiz1"
    id="Quiz1b" value="2" title="Platinum: A vegetable"></td>
  <td width="50" valign="top" align="center"><input type="radio" name="Quiz1"
    id="Quiz1c" value="3" title="Platinum: A mineral"></td>
</tr>
```

If you need additional questions or items, you can copy and paste the existing code to add on additional rows. The rows as programmed alternate colors, with the <tr> tags having either a class of rowColor1 or rowColor2. When adding additional rows, make sure you continue to alternate those colors.

After you have set up your questions/statements, you need to update the correct answer and feedback.

```
<script language="JavaScript" src="scripts/FeedbackCR.js"
type="text/javascript"></script>
<script language="JavaScript" type="text/javascript">
var Feedback = "" // you will need to "escape" all double quotes in this area with
back slashes. (i.e. ...he said. \"this is great\")
var vCorrectAnswer = "" //enter a 1, 2, or a 3 for each row (i.e. if the first row's
answer is the 2nd column, enter a 2)
var vTriesAllowed = 2
showCorrect();
</SCRIPT>
```

var Feedback is where you will paste the text that a user receives when the Knowledge Check is over, regardless of whether they got it right or wrong. This should include correct HTML formatting (paragraph tags, list tags, etc.), but all the HTML must be on a single line. If you have a particularly detailed set of feedback, you may want to paste it into a blank screen, then cut and paste the HTML between the quotation marks.

var vCorrectAnswer is where you set the correct answer. The answer is formatted as a series of numbers, in order, of which options are correct. For instance, if the correct set of answers was the third column for row 1, the first column for row 2, and the second column for row 3, you would enter 312 between the quotes.

var vTriesAllowed lets the code know how many tries the user gets before they are told the correct answer and are allowed to move on to the next screen. For this type of exercise, two tries are traditionally allowed.

A completed Knowledge Check might look like this:

```
<p class="instruction">Instructions: Select the correct category for each item. Then,
click on Submit.</p>
<p>Are each of these items an animal, a vegetable, or a mineral?</p>
```

```
<form name="form1">
<!-- if the first column header is something besides "Statement", edit the alt tag
below -->
  
<table width="99%" border="0" cellspacing="0" cellpadding="3">
  <tr>
    <th scope="col">Item</th>
    <td width="1"></td>
    <th scope="col">Animal</th>
    <!-- Edit the column heading to suit your needs -->
    <th scope="col">Vegetable</th>
    <!-- Edit the column heading to suit your needs -->
    <th scope="col">Mineral</th>
    <!-- Edit the column heading to suit your needs -->
  </tr><!-- Edit the title tags below to match the headings -->
  <tr class=rowColor1>
    <td width="100%" valign="top">Platinum</td>
    <td></td>
    <td width="50" valign="top" align="center"><input type="radio" name="Quiz1"
id="Quiz1a" value="1" title="Platinum: An animal"></td>
    <td width="50" valign="top" align="center"><input type="radio" name="Quiz1"
id="Quiz1b" value="2" title="Platinum: A vegetable"></td>
    <td width="50" valign="top" align="center"><input type="radio" name="Quiz1"
id="Quiz1c" value="3" title="Platinum: A mineral"></td>
  </tr>
  <tr class=rowColor2>
    <td width="100%" valign="top">Border Collie</td>
    <td></td>
    <td width="50" valign="top" align="center"><input type="radio" name="Quiz2"
id="Quiz2a" value="1" title="Border Collie: An animal"></td>
    <td width="50" valign="top" align="center"><input type="radio" name="Quiz2"
id="Quiz2b" value="2" title="Border Collie: A vegetable"></td>
    <td width="50" valign="top" align="center"><input type="radio" name="Quiz2"
id="Quiz2c" value="3" title="Border Collie: A mineral"></td>
  </tr>
  <tr class=rowColor1>
    <td width="100%" valign="top">Eggplant</td>
    <td></td>
    <td width="50" valign="top" align="center"><input type="radio" name="Quiz3"
id="Quiz3a" value="1" title="Eggplant: An animal"></td>
    <td width="50" valign="top" align="center"><input type="radio" name="Quiz3"
id="Quiz3b" value="2" title="Eggplant: A vegetable"></td>
    <td width="50" valign="top" align="center"><input type="radio" name="Quiz3"
id="Quiz3c" value="3" title="Eggplant: A mineral"></td>
  </tr>
</tr>
```



```

        <td valign="top" align="right" colspan="5"><a href="javascript:Cliked()"
onmouseover="document.submitme.src='images/buttonSubmitHi.gif';"
onmouseout="document.submitme.src='images/buttonSubmit.gif';"
onmousedown="document.submitme.src='images/buttonSubmitDn.gif';"></a></td>
    </tr>
</table>
<div id="feedbackDiv"></div>
</form>
<script language="JavaScript" src="scripts/FeedbackCR.js"
type="text/javascript"></script>
<script language="JavaScript" type="text/javascript">
var Feedback = "<p>Platinum is a mineral. A border collie is an animal. Eggplant is a
vegetable.</p>" // you will need to "escape" all double quotes in this area with
back slashes. (i.e. ...he said. \"this is great\")
var vCorrectAnswer = "312" //enter a 1, 2, or a 3 for each row (i.e. if the first
row's answer is the 2nd column, enter a 2)
var vTriesAllowed = 2
showCorrect();
</SCRIPT>

```

MainContentExercise: Compare Text

Compare Text is the simplest of the exercise snippets to use; it lets the student type an answer, then compares the student's response to a sample answer.

In the MainContent placeholder, you should place instructions and the question for the user. There is an "instruction" class already set up which is used as a standard look for the opening statement on the screen.

```
<p><strong><label for="compareMe">QuestionStatementGoesHere</label></strong></p>
```

Replace the QuestionStatementGoesHere placeholder with your own question.

```
<script language="JavaScript" src="scripts/FeedbackCompare.js"
type="text/javascript"></script>
<script language="JavaScript" type="text/javascript">
var FBheader = "Compare your response to the suggested answers below. <br>Remember to
click on the Continue button to proceed. " // Feel free to edit this text
var Feedback = "<p><strong>Suggested Response</strong></p>" // this is where you add
the text to be compared to. You will need to "escape" all double quotes in this area
with back slashes. (i.e. ...he said. \"this is great\")
</SCRIPT>
```

var FBheader contains a standard header in the feedback area; because there is no "correct answer" grading, this serves as a replacement for the "Correct" and "Incorrect" headers that normally exist. If you want to use a different statement at the start of the feedback, you can replace it here.

var Feedback is where you place your sample/suggested response for the student.

A finished screen might look like this:

```
<p class="instruction">Instructions: Read the statement below, then type your answer
in the provided area. When you are done, click on Submit.</p>
<p><strong><label for="compareMe">What are a few of your favorite
things?</label></strong></p>
<form name="form1" action="">
<table width="500" border="0" cellspacing="0" cellpadding="3" align="center">
  <tr>
    <td valign="top"><textarea id="compareMe" cols="70" rows="8" name="response"
onfocus="if(document.form1.response.value=='Type your answer in this
area.'){document.form1.response.value=''}">Type your answer in this
area.</textarea></td>
  </tr>
  <tr>
    <td valign="top" align="center"><a href="javascript:ScoreMe()"
onmouseover="document.submitme.src='images/buttonSubmitHi.gif';"
onmouseout="document.submitme.src='images/buttonSubmit.gif';"
onmousedown="document.submitme.src='images/buttonSubmitDn.gif';"></a></td>
</tr>
</table>
<div id="feedbackDiv"></div>
</form>

<script language="JavaScript" src="scripts/FeedbackCompare.js"
type="text/javascript"></script>
<script language="JavaScript" type="text/javascript">
var FBheader = "Compare your response to the suggested answers below. <br>Remember to
click on the Continue button to proceed. " // Feel free to edit this text
var Feedback = "<p><strong>Suggested Response</strong></p><ul>
<li>Well-written HTML code</li> <li>Interesting online courses</li> <li>Chocolate
cake</li> </ul>" // this is where you add the text to be compared to. You will need
to "escape" all double quotes in this area with back slashes. (i.e. ...he said.
\"this is great\")
</SCRIPT>

```

MainContentExercise: Matching

The Matching exercise snippet lets the student match terms or phrases with potential answers. The Matching snippet comes with templates offering three, four, or five distracters/answers.

In the MainContent placeholder, you should place instructions and the question for the user. There is an "instruction" class already set up which is used as a standard look for the opening statement on the screen. This section is also where you should include the terms that the student will use to match with the distracters below. Each term should be numbered, so that the user will then type the number in the appropriate space next to the distracter. Only use numbers as identifiers; letters, words, or phrases should not be typed into the boxes.

```
<tr>
  <td valign="top" width="20"><input type="text" name="Quiz" id="Quiz1" size="1"
maxlength="1"></td>
  <td width="100%"><label for = "Quiz1"></label></td> <!-- paste the distracter
between the label tags, don't edit anything else in the table -->
</tr>
```

The distracter should be inserted between the two <label> tags. If you need more than five distracters, copy and paste this code in additional times, then update "Quiz1" to have the correct number in its place.

```
<script language="JavaScript" src="scripts/FeedbackMA.js"
type="text/javascript"></script>
<script language="JavaScript" type="text/javascript">
var Feedback = "" // you will need to "escape" all double quotes in this area with
back slashes. (i.e. ...he said. \"this is great\")
var vCorrectAnswer = "" // enter the sequence the distracters should be in (i.e.
"312" specifies that the second item comes first, then the third item...)
var vTriesAllowed = 2
showCorrect()
</script>
```

var Feedback is where you will paste the text that a user receives when the Knowledge Check is over, regardless of whether they got it right or wrong. This should include correct HTML formatting (paragraph tags, list tags, etc.), but all the HTML must be on a single line. If you have a particularly detailed set of feedback, you may want to paste it into a blank screen, then cut and paste the HTML between the quotation marks.

var vCorrectAnswer is where you set the correct answer. The answer is formatted as the order of numbers to form the correct answer. So if the first distracter's answer is 1, the second answer is 3, and the third answer is 2, you would type 132 inside the quotes.

var vTriesAllowed lets the code know how many tries the user gets before they are told the correct answer and are allowed to move on to the next screen. For this type of exercise, two tries are traditionally allowed.

A completed exercise might look like this:


```
var vCorrectAnswer = "2413" // enter the sequence the distracters should be in (i.e.
"312" specifies that the second item comes first, then the third item...)
var vTriesAllowed = 2
showCorrect()
</script>
```

MainContentExercise: MultiChoice

The MultiChoice exercise snippet gives the student a multiple-choice question and answer. MultiChoice snippets are already set up for questions with two, three, four, or five answers/distracters.

In the MainContent placeholder, you should place instructions and the question for the user. There is an "instruction" class already set up which is used as a standard look for the opening statement on the screen.

```
<tr>
  <td valign="top" width="20%"><input type="radio" title="" name="Quiz" value="1"
    id="Quiz1" /></td>
  <td width="100%"><label for="Quiz1"></label></td>
</tr>
```

Each distracter is placed between the <label> tags. Additionally, you need to update the title="" value on the radio button to include both the question and possible answer.

If you need more than five distracters, copy and paste a table row at the end, and then make sure to update the value="1" as well as the two "Quiz1" data so that the number is increased to reflect what number answer it is.

```
<script language="JavaScript" src="scripts/FeedbackMC.js"
type="text/javascript"></script>
<script language="JavaScript" type="text/javascript">
var Feedback = "" // you will need to "escape" all double quotes in this area with
back slashes. (i.e. ...he said. \"this is great\")
var FeedbackWrong = "<strong>Your answer is not correct.</strong> You now have a
second chance to answer correctly."
var vCorrectAnswer = "" // enter the item number of the correct distracter
var vTriesAllowed = 2
showCorrect();
</script>
```

var Feedback is where you will paste the text that a user receives when the Knowledge Check is over, regardless of whether they got it right or wrong. This should include correct HTML formatting (paragraph tags, list tags, etc.), but all the HTML must be on a single line. If you have a particularly detailed set of feedback, you may want to paste it into a blank screen, then cut and paste the HTML between the quotation marks.

var vCorrectAnswer is where you set the correct answer. The answer is the number attached to that distracter/option.

var vTriesAllowed lets the code know how many tries the user gets before they are told the correct answer and are allowed to move on to the next screen. For this type of exercise, if there are only two possible answers (often a True or False type of question), only one try is allowed. Otherwise, two tries are traditionally allowed.

A completed version of this exercise might look like this:

```
<p class="instruction">Instructions: Read the question below, then select the correct answer. When you are done, click on Submit.</p>
```

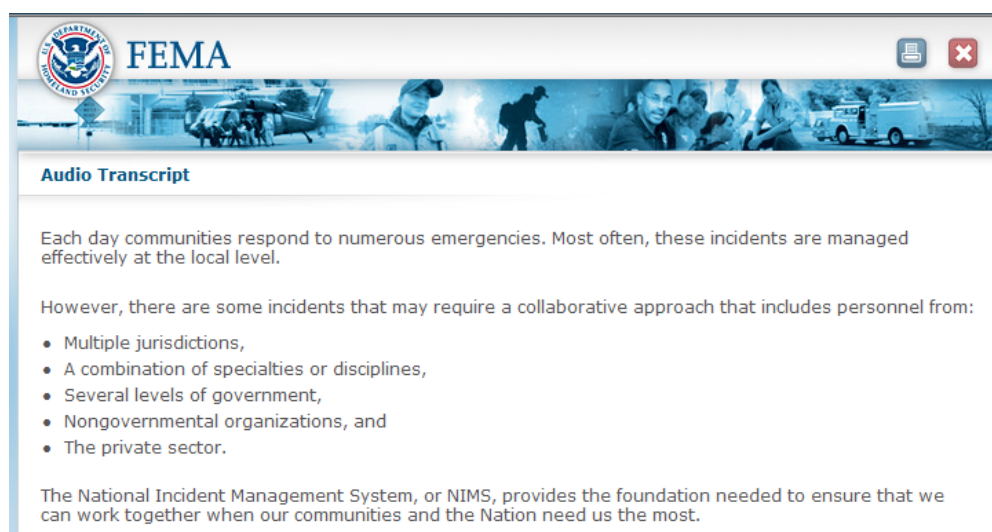
```
<p>The atmosphere is primarily:</p>
```

```
<form name="form1" action="">
<table width="95%" border="0" cellspacing="0" cellpadding="3">
  <tr>
    <td valign="top" width="20"><input type="radio" title="The atmosphere is primarily: Carbon" name="Quiz" value="1" id="Quiz1" /></td>
    <td width="100%"><label for="Quiz1">Carbon</label></td>
  </tr>
  <tr>
    <td valign="top" width="20"><input type="radio" title="The atmosphere is primarily: Nitrogen" name="Quiz" value="2" id="Quiz2" /></td>
    <td width="100%"><label for="Quiz2">Nitrogen</label></td>
  </tr>
  <tr>
    <td valign="top" width="20"><input type="radio" title="The atmosphere is primarily: Oxygen" name="Quiz" value="3" id="Quiz3" /></td>
    <td width="100%"><label for="Quiz3">Oxygen</label></td>
  </tr>
  <tr>
    <td>&nbsp;</td>
    <td valign="top"><a href="javascript:Clicked()"
onmouseover="document.submitme.src='images/buttonSubmitHi.gif';"
onmouseout="document.submitme.src='images/buttonSubmit.gif';"
onmousedown="document.submitme.src='images/buttonSubmitDn.gif';"></a></td>
  </tr>
</table>
<div id="feedbackDiv"></div>
</form>
```

```
<script language="JavaScript" src="scripts/FeedbackMC.js"
type="text/javascript"></script>
<script language="JavaScript" type="text/javascript">
var Feedback = "<p>The atmosphere is primarily <strong>oxygen</strong>.</p>" // you
will need to "escape" all double quotes in this area with back slashes. (i.e. ...he
said. \"this is great\")
var FeedbackWrong = "<strong>Your answer is not correct.</strong> You now have a
second chance to answer correctly."
var vCorrectAnswer = "3" // enter the item number of the correct distracter
var vTriesAllowed = 2
showCorrect();
</script>
```


Audio Transcript

When creating pop-up transcripts of audios and videos in a course, you must always include the audio transcript in an HTML file, so that people using screen readers can still receive the same material as those who are listening to the audio. Additionally, some students may learn better by reading a transcript rather than listening.

A screenshot of a FEMA Audio Transcript pop-up window. The window has a header with the FEMA logo and the text "FEMA". Below the header is a banner image showing emergency responders. The main content area is titled "Audio Transcript" and contains the following text:

Each day communities respond to numerous emergencies. Most often, these incidents are managed effectively at the local level.

However, there are some incidents that may require a collaborative approach that includes personnel from:

- Multiple jurisdictions,
- A combination of specialties or disciplines,
- Several levels of government,
- Nongovernmental organizations, and
- The private sector.

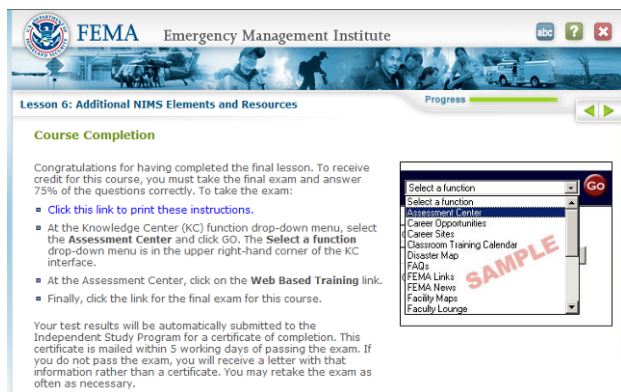
The National Incident Management System, or NIMS, provides the foundation needed to ensure that we can work together when our communities and the Nation need us the most.

<p>TextStartsHere</p>

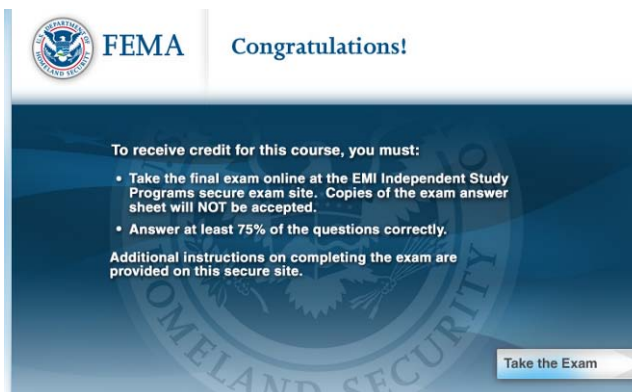
Simply replace the TextStartsHere placeholder with the text of your transcript; the rest of the page is already set up for you.

EndScreen

The final screen of a course developed for EMI should use the EndScreen template. This is a standard screen that will provide a link to the final exam, as well as remind them that they need to score 75 percent or higher to pass. This page will dynamically determine whether it is being served up on EMI's FEKC or not and provide the appropriate screen. If on the FEKC, they will see instructions on how to take the test and end the course. If not on the FEKC, they will be given a link to EMI's test site available to everyone.



or



There is only one piece of code that will need to change:

```
<script type="text/javascript">
```

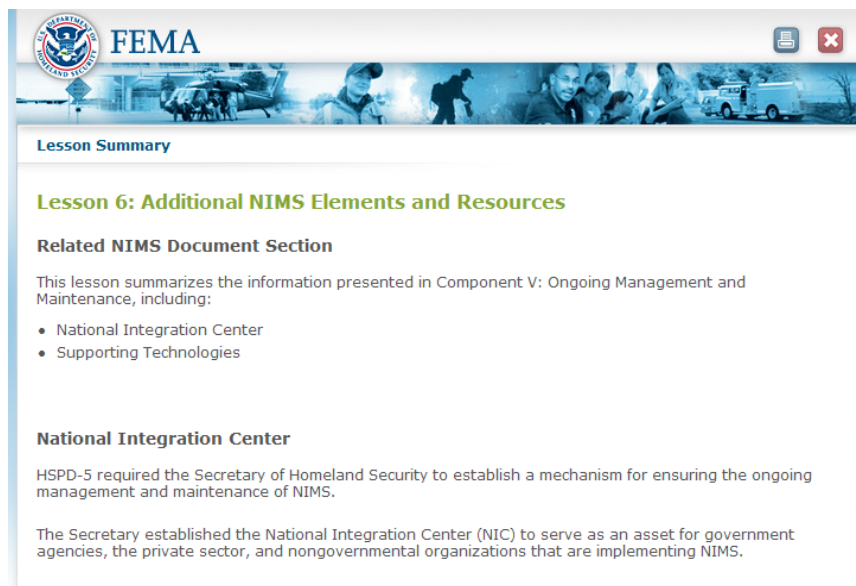
```
var ExamID = 'is987' // this is the id that comes after the  
http://training.fema.gov/EMIWeb/IS/examnotice.asp?eid=isXX
```

```
</script>
```

Your course manager will provide you an IS number for your course, which will be used as part of the URL for the final exam. Substitute your number for the provided placeholder in the code. Please note that most exam links will still require the "is" prefix as part of the ExamID; do not delete it.

ExtraScreen

The ExtraScreen template is one of the most commonly used templates in a course for EMI. ExtraScreen is used for pop-up screens providing additional information, plus lesson and course summaries.



There are three sections on an ExtraScreen template that you will need to alter, although the first two will possibly feature the same information.

```
<!-- InstanceBeginEditable name="doctitle" -->  
<title>PageTitle</title>  
<!-- InstanceEndEditable -->
```

```
<div id="PageTitle"><span class="heading1"><!-- InstanceBeginEditable  
name="PageHeading" -->PageHeading<!-- InstanceEndEditable --></span></div>
```

The first section contains a placeholder for the title of the screen. This is what will appear in the Internet browser's title bar. Replace the PageTitle placeholder with your own title for the screen.

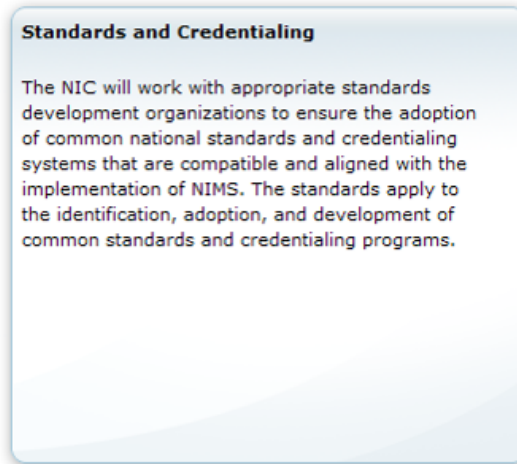
The second section contains a placeholder for the heading of the screen, and will appear at the top of the screen. Generally, this will be the same title that you included in the PageTitle placeholder. Replace PageHeading with your own heading.

```
<p>TextStartsHere</p>
```

The third section contains a placeholder for the body of the screen's text. Replace the text here with your own information.

FramedText

FramedText is a template used to create screens that appear in the MainContentClickToRead template, on the right-hand side.



`<p>ContentText</p>`

Replace the ContentText placeholder with your own text for each FramedText screen that you create.

You will probably want to use the Instruction style for the initial FramedText screen on a MainContentClickToRead page.

FrameSet

The FrameSet template is the most complex basic template in a FEMA course. You use FrameSet templates to create a file for each lesson, containing a list of all of the pages in the lesson, as well as information on the lesson number and title.

The resulting page is then linked to from the Lesson List, and calls up all the pages for the lesson. If down the line you need to add, remove, or rearrange the order of screens in a lesson, simply alter this listing of filenames.

A FrameSet page must be edited using the Code view; Split or Design view in Dreamweaver will not function properly.

Before you can use the Template, you must first edit the FrameSet.dwt file itself to include some important information: the title of the course, the IS number of the course, and whether this is a single-lesson course or a multi-lesson course.

The following are examples of where you will need to alter the master Template file (FrameSet.dwt) for a new course, using "IS-235 – Emergency Planning" as an example.

```
<title>IS-987 - Sample Course</title>

var CourseTitle = "IS-235 - Sample Course "; //the title of the course to be
displayed
var CourseISnumber = "is235-SampleCourse "; //short course nomenclature for the HT
client comments site (no spaces)
var ShowLessonTitle = "true"; // true for multi-lesson courses, false for single
lesson courses.
```

If you are programming a single-lesson course, make sure to change the last to "False".

Once you have made these alterations to your FrameSet.dwt file, you can begin using the template.

```
var LessonNum = '1'; //the numeric lesson number
var LessonTitle = 'This is the Lesson Title'; // the title of this lesson
```

Update the lesson number and title placeholders with the correct information for this file's lesson.

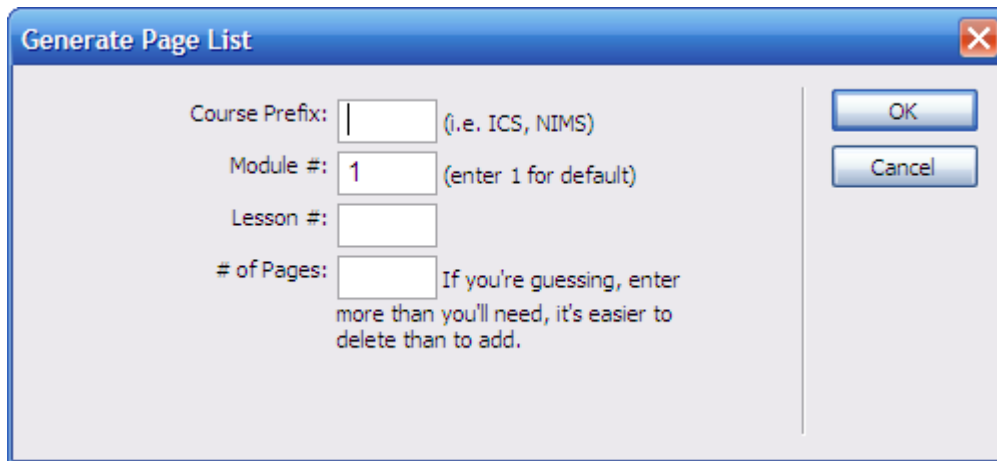
The next section is easiest to delete where marked, and then generate using the "Generate Page List" function. It is a list of pages, in order, that make up the entire lesson:

```
// use the FEMA "Generate Page List" extension
// totally replace the Javascript code below (see comment on where to stop)

var scoPages = new Array (
  "",
  "SMPL0101010.htm",
```

```
"SMPL0101020.htm",  
"SMPL0101030.htm",  
"SMPL0101040.htm"  
  
// IMPORTANT!: check to make sure there isn't a comma after the last item above  
  
); // Completes the array that will hold the content pages
```

The Generate Page List function looks like this:



Course Prefix: This is the start of every filename. In the above example, the prefix would be **SMPL**.

Module #: This determines the first two numbers of the filename. Most FEMA courses use **1** for all lessons.

Lesson #: This determines the next two numbers of the filename. You do not need to include a zero at the start of this number; if it is the third lesson, a **3** is all you need to enter.

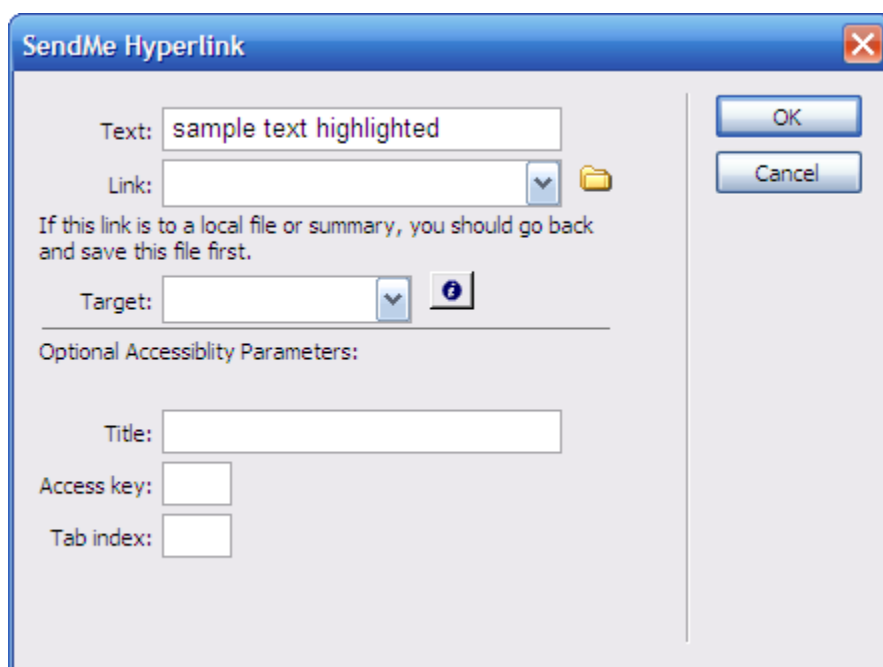
of Pages: This determines how many filenames it will auto-generate for you in the listing of pages. If you are not sure, it's always easier to add in a higher number and then delete the extra pages afterwards.

Once you've filled out all the spaces, click on OK and it will add in all of the new code for you.

Creating Links

You will occasionally need to link to additional screens or files from within your course. To assist in this process, there are javascript functions to put links in specially sized windows, and (if connecting to a nongovernmental site) to provide a disclaimer.

Included as part of the templates is a SendMe Hyperlink function. To use the function, select the text you wish to have linked, and then click on the function's button in the taskbar. Doing so will display the following menu.



The image shows a dialog box titled "SendMe Hyperlink". It has a standard Windows-style title bar with a close button (X) in the top right corner. The dialog is divided into several sections. At the top, there is a "Text:" label followed by a text input field containing "sample text highlighted". Below this is a "Link:" label followed by a text input field and a folder icon. A note below the link field reads: "If this link is to a local file or summary, you should go back and save this file first." Underneath is a "Target:" label followed by a dropdown menu and an information icon (i). Below the target field is a section titled "Optional Accessibility Parameters:" which includes three input fields: "Title:", "Access key:", and "Tab index:". On the right side of the dialog, there are two buttons: "OK" and "Cancel".

Text: This box contains the text you wish to have the link attached to. If you have not highlighted text already, it will be blank; you can then type in the text you wish associated with this link.

Link: The destination of the link. If this is a HTML page or file that is part of the site, you can click on the folder icon and select the destination using the file browser. Otherwise, type in the URL for the link.

Target: This drop-down menu offers three options: Summary/Extra Page, External URL, and Local File.

- **Summary/Extra Page:** You should use this option if you are linking to a lesson/course summary, or an additional details page. These pages are created using the ExtraScreen template.
- **External URL:** You should use this option if you are linking to another Web site.

- **Local File:** You should use this option if you are linking to a local asset, such as a PDF or DOC file.

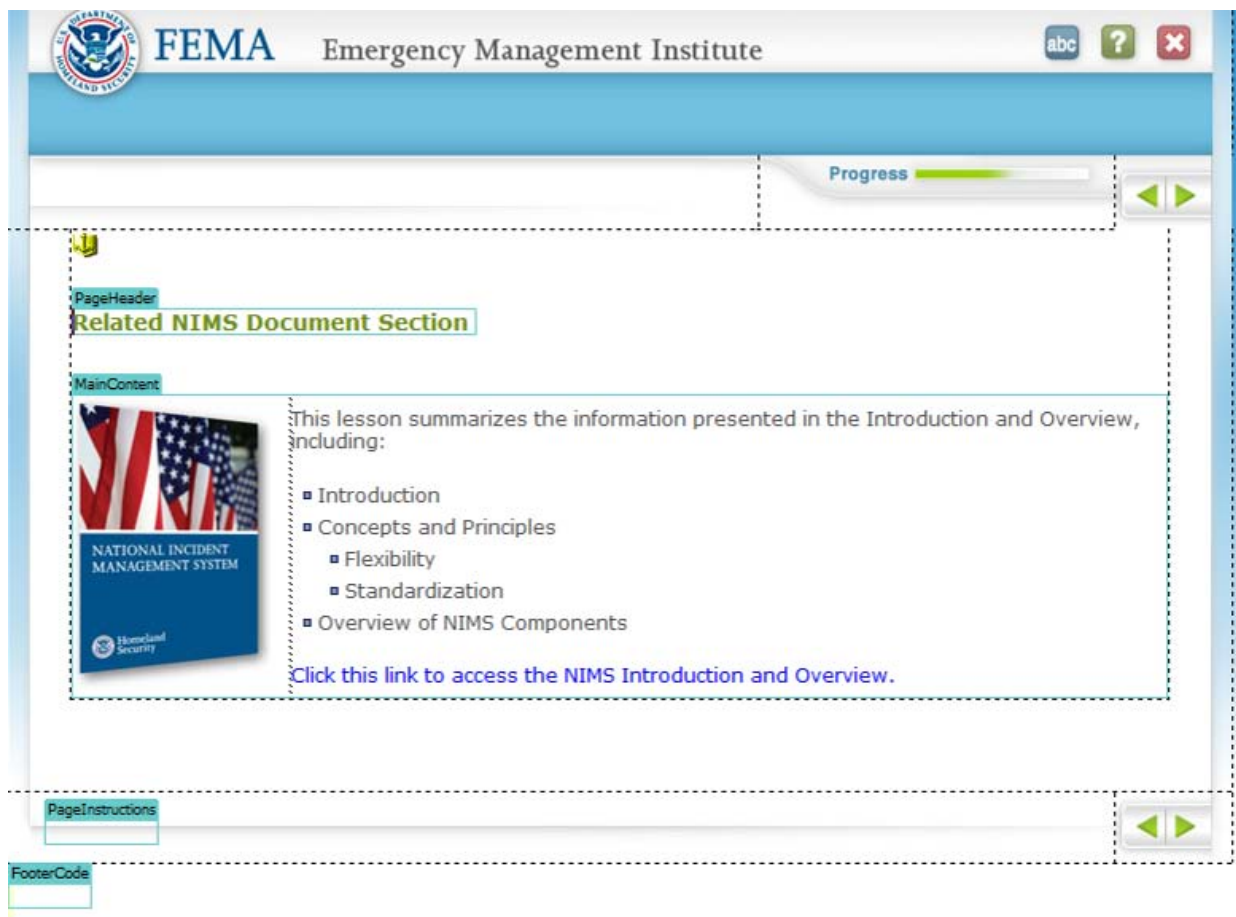
If you would prefer to code your links manually, the code for each is as follows:

- **Summary/Extra Page:** `javascript:SendMeSummary('filename.htm')`
- **External URL:** `javascript:SendMeExternalURL('http://www.website.gov')`
- **Local File:** `javascript:SendMeLocalFile('filename.pdf')`

Customizing Dreamweaver

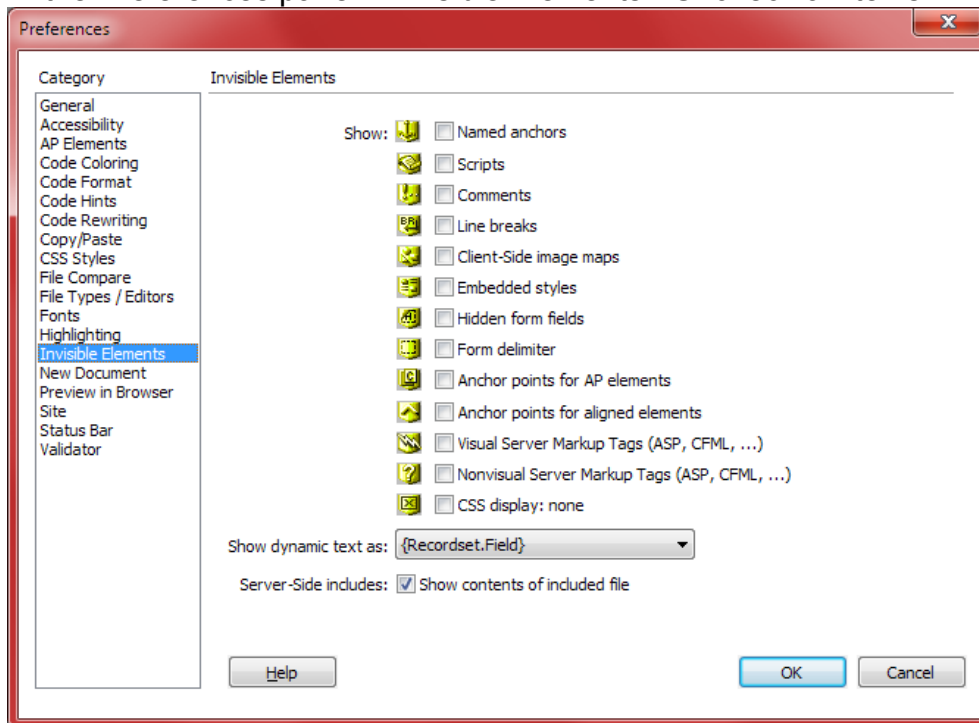
The default installation of Dreamweaver is not the most conducive for production and ease of use. The following are hints to make it better.

You will notice below that Dreamweaver shows you all of the borders for DIV tags, labels for editable regions, and graphical placeholders for hidden elements. It is a bit messy.

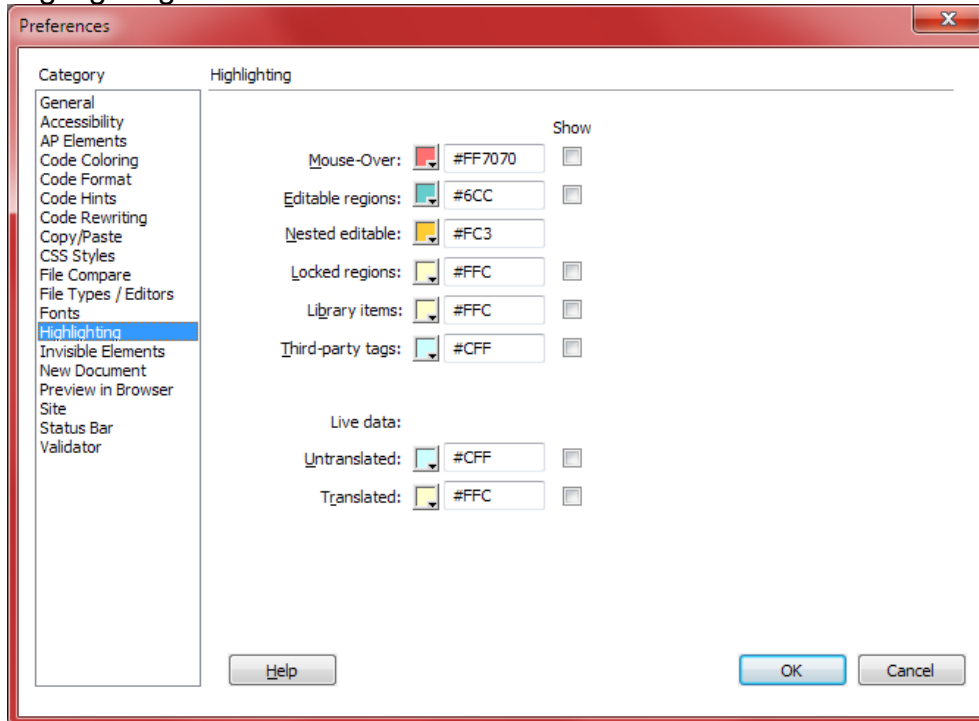


Note: The blue bar at the top will be where the four random image bars show when displayed in a browser.

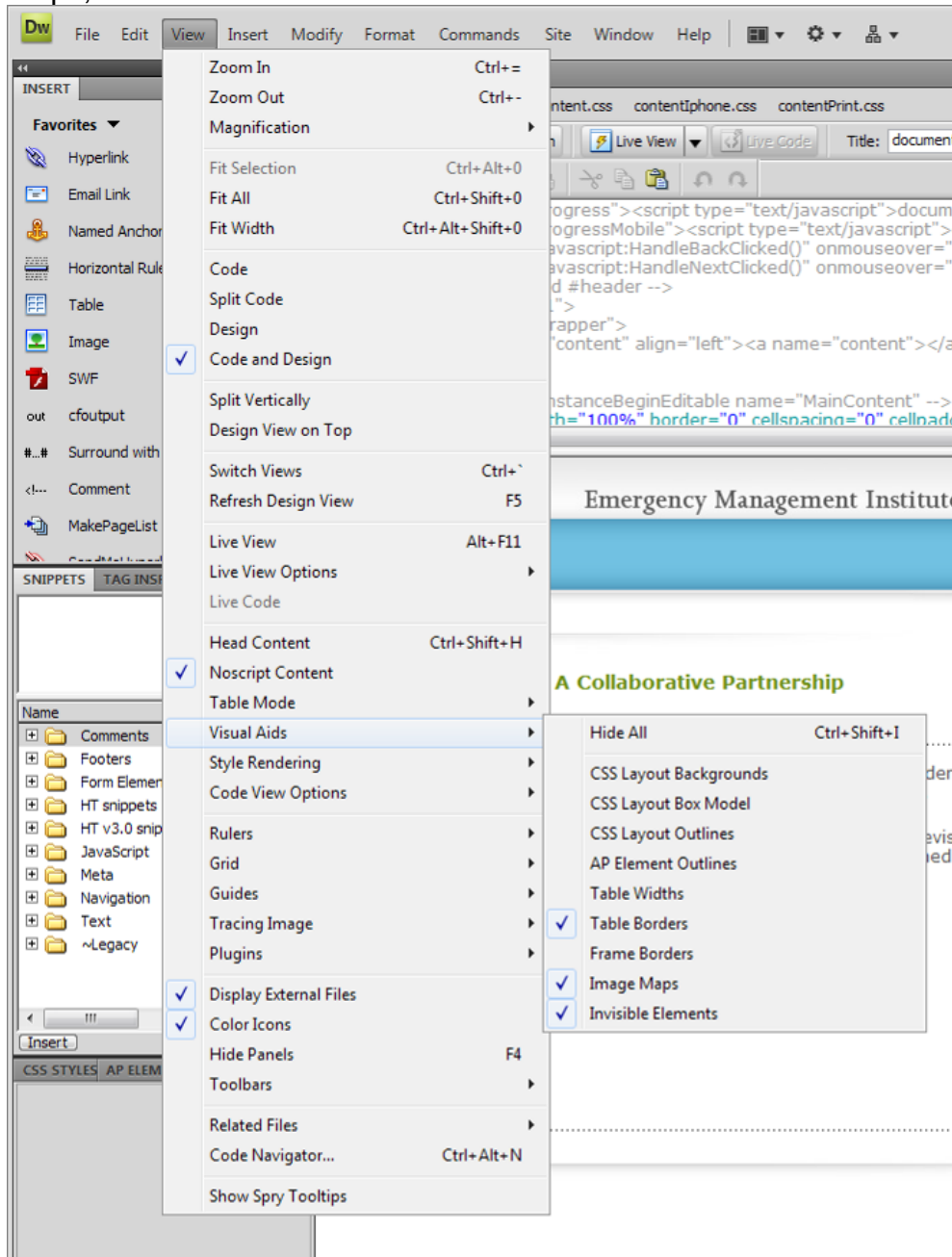
In the Preferences panel - Invisible Elements - Uncheck all items



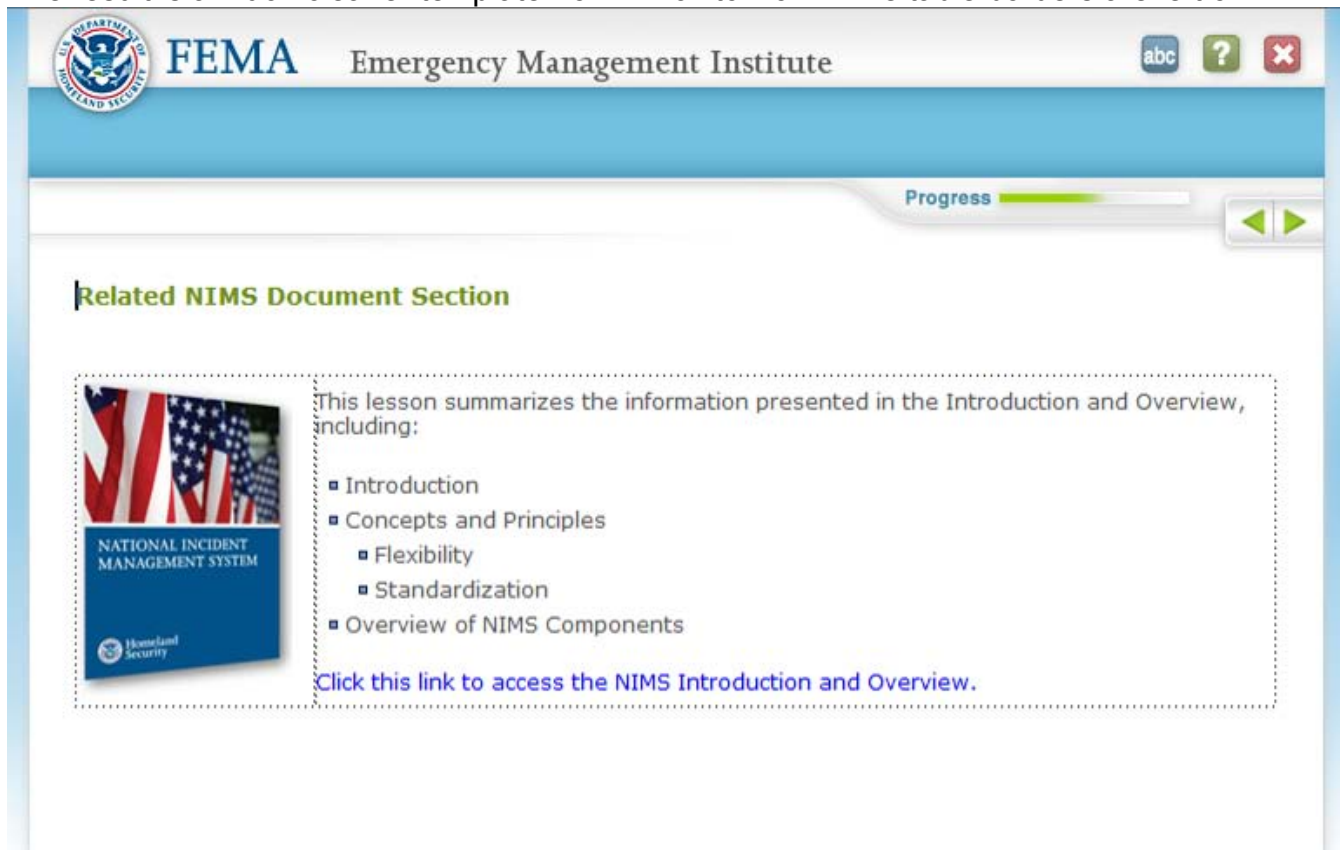
Highlighting - Uncheck all items



From the View menu item, choose Visual Aids and uncheck all except for Table Borders, Image Maps, and Invisible Elements



The result is a much cleaner template from which to work. The table borders are left on.



The screenshot shows a web interface for FEMA's Emergency Management Institute. At the top left is the FEMA logo, and at the top right are navigation icons for 'abc', a question mark, and a close button. Below the header is a blue progress bar with the word 'Progress' and a green indicator. The main content area is titled 'Related NIMS Document Section' in green. It features a thumbnail image of a document cover titled 'NATIONAL INCIDENT MANAGEMENT SYSTEM' with the 'Homeland Security' logo. To the right of the thumbnail, a text block states: 'This lesson summarizes the information presented in the Introduction and Overview, including:'. Below this text is a bulleted list of topics: Introduction, Concepts and Principles (with sub-bullets for Flexibility and Standardization), and Overview of NIMS Components. At the bottom of the list is a blue link: 'Click this link to access the NIMS Introduction and Overview.'