Exam Engine Templates Documentation

Table of Contents

This document describes how to load the *ExamEngine* project. It then covers the design and structure of the Exam Engine Templates and associated files.

Loading the *ExamEngine* Application

Once you install Exam Engine, you will likely want to configure the *ExamEngine* project to be loaded into your favorite HTML editor. We recommend Visual Studio 2012 or later and will use that product for this document, but other editors will work as well.

Choose to "Open Project" then navigate to the location of the *ExamEngine* project (*C:\Program Files\VBTrain\Exam Engine 4\Source Solutions\ExamEngine*[1] by default) and select *ExamEngine.sln* as shown below.
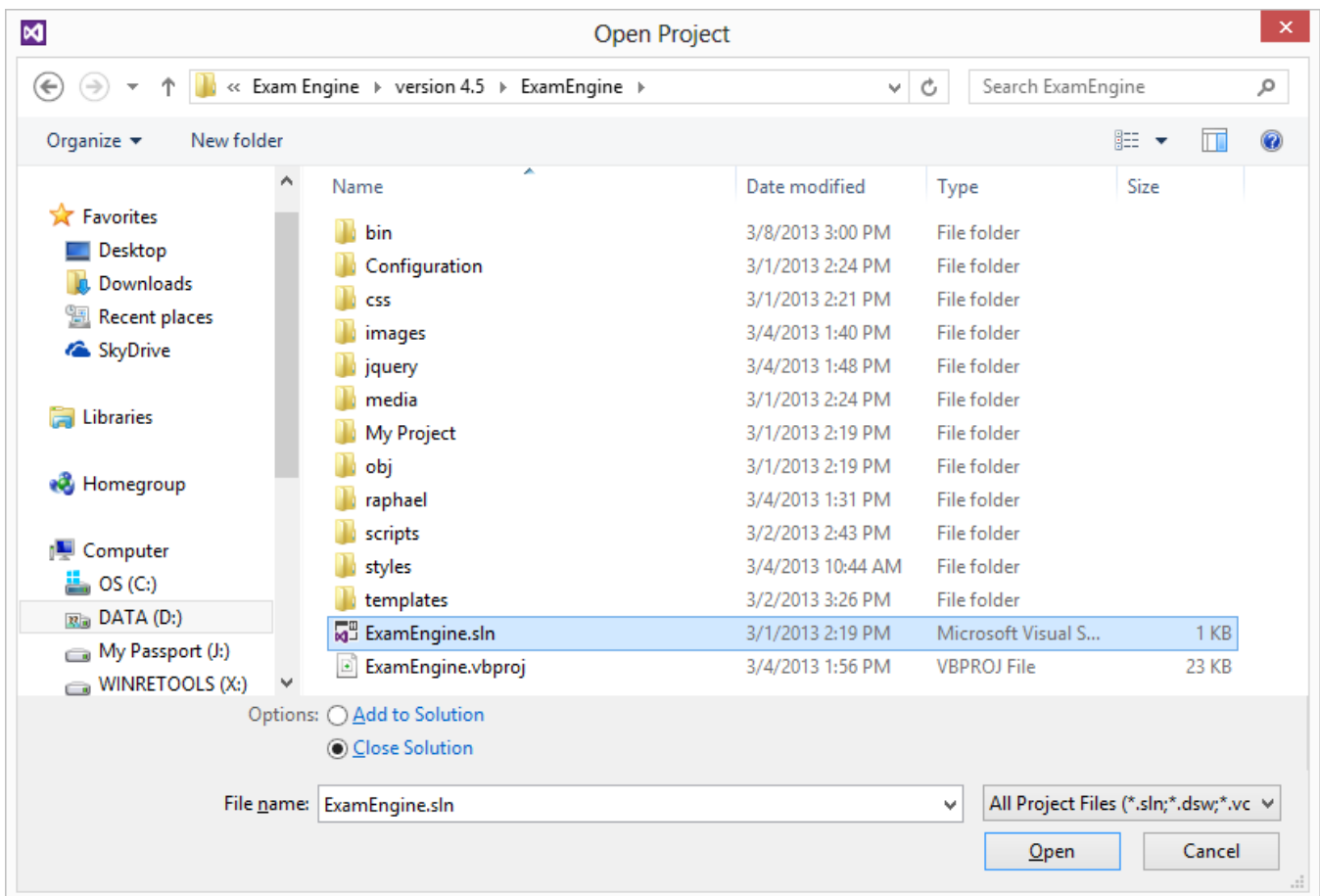
Architecture

You can look at the architecture of exams at several levels. At the highest level, the exam consists only of these key files and directories[2]: index.htm, email.htm, mediaplayer.swf, Configuration, css, images,

---

[1] This will be *C:\Program Files (x86)\VBTrain\Exam Engine 4\Source Solutions\ExamEngine* on a 64-bit operating system.
[2] If you publish to SCORM, there are additional files like imsmanifest.xml. If you publish to AICC, there are also extra files with .au, .crs, .cst, and .des descriptions. These are for packaging purposes and not relevant to the idea of templates.

jquery, media, raphael, scripts, styles, and templates. This is shown to the right. You will use Visual Studio to change styles, add/edit templates, etc. email.htm, index.htm, mediaplayer.swf and these directories are all part of the core exam files: *css, jquery, raphael, scripts, styles, and templates*. The *media* directory contains the video, audio, graphics and other media files used by your exam. The *images* directory contains any "structure" graphics such as the one shown on the optional introduction screen. The *Configuration* directory has the examengineconfig4.xml file created and edited by the *Exam Engine Configuration Editor* and possibly one or more question databases in XML format[3]. These XML databases are created with the *Exam Engine Question Editor*.

Big Picture

The figure below shows an XML version of an Exam Engine database. Depending on the settings, this can be stored in the *Configuration* directory or can be retrieved via the *Exam Engine Web Service*. The *template* (hotObjects in this case) is used to grab the correct file (*hotObjects.htm*) from the *templates* subdirectory, which is shown at the upper right. We cover templates in more detail later in the document. Other pieces of data such as *QuestionText*, *Answer1Text, Answer2Text*, etc. are used to populate the template. Since the answers are graphics for this question type, they are retrieved from the *media* directory. They are inserted into the *Image* elements defined within the template.

---
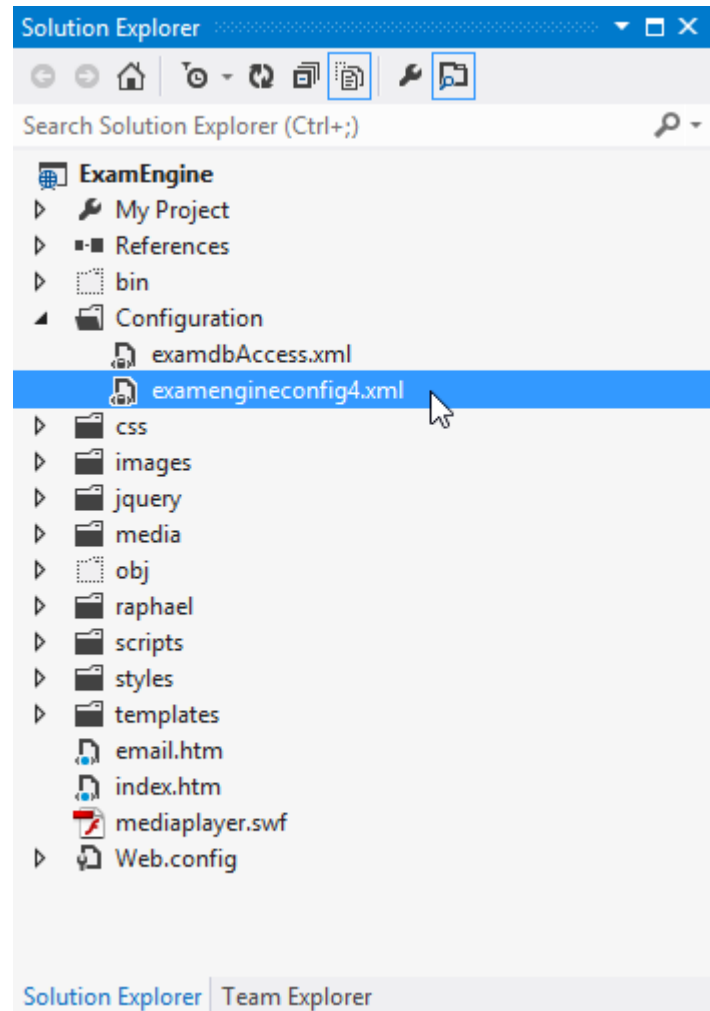
[3] You can load questions from the *ExamEngineWebService* web service. If you use that, you do not need any database files in the *Configuration*

We will now look at each of the major architecture elements in turn.
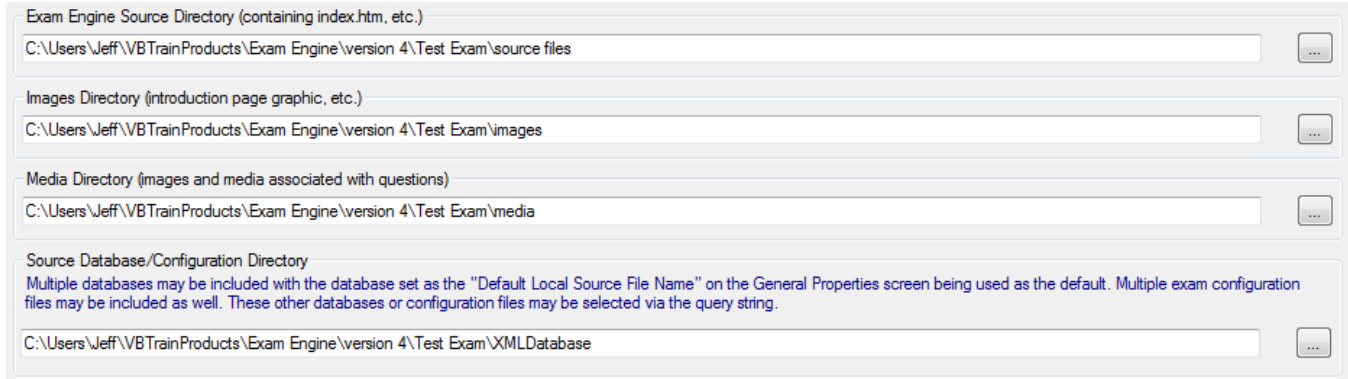
Exam Directory

As discussed above, *index.htm* and the various core files and subdirectories contain the main exam functionality with all the external content in separate directories. The *Exam Engine Configuration Editor* builds the directory structure based on the data within the *Publish Exam* screen shown below.

The source directory contains *index.htm* as well as the other non-content files and directories required by Exam Engine. This is shown in the screen capture to the right. When you update your templates or other files as described later in this document, be sure to update the copies in the source directory. The publish process then adds all the files in the selected images and media directories. It then creates the *Configuration* directory, copying *examngineconfig4.xml* and all database files into it.
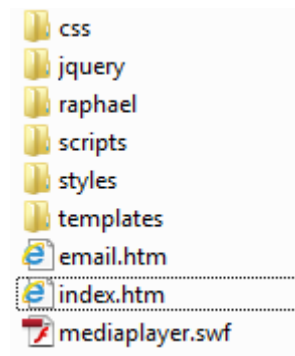
index.htm

This is analogous to the background or slide master of your exam. Viewing it in Visual Studio design mode gives a basic idea of the where items are located.
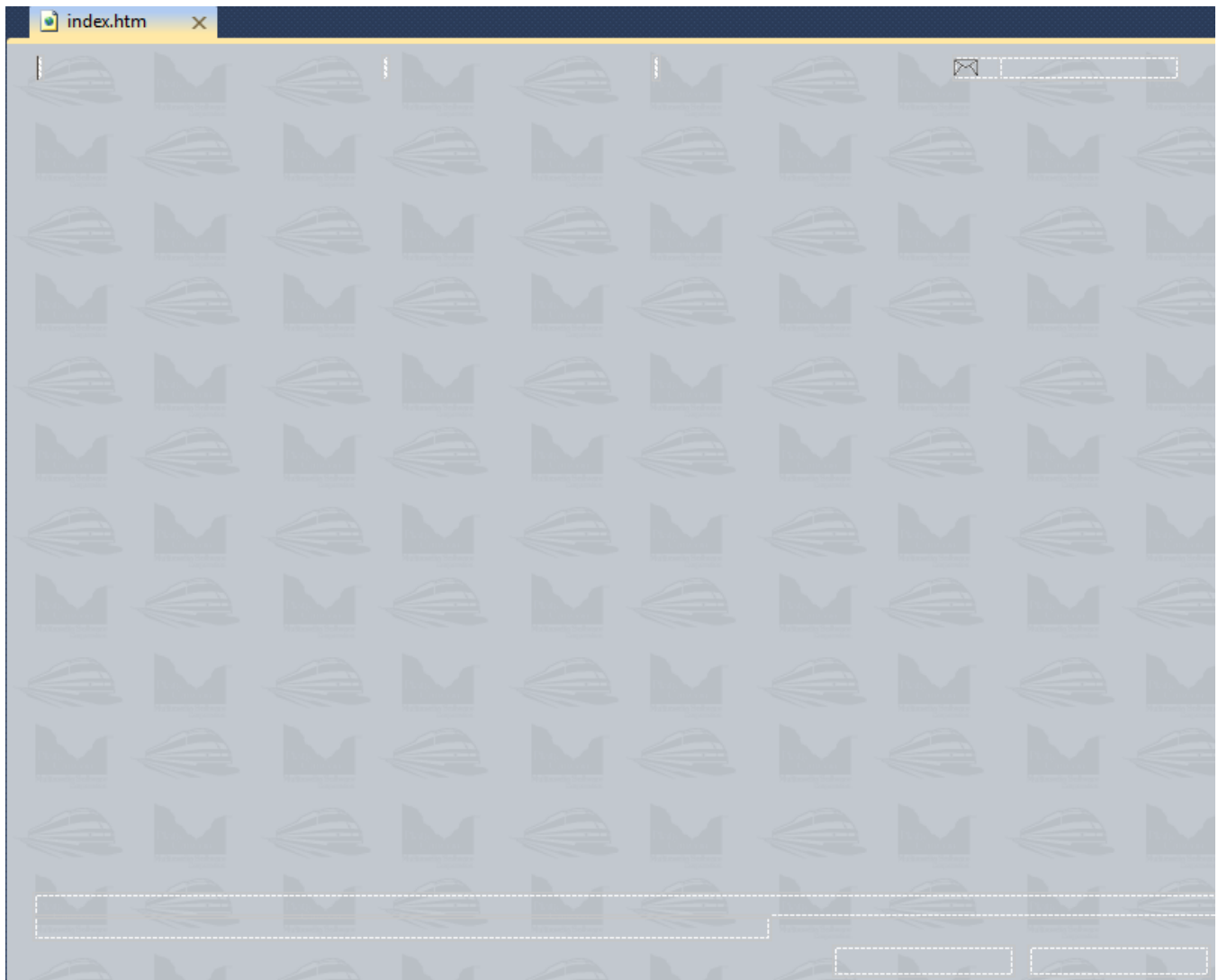
As shown in the figure below, *index.htm* contains such items as the user name, exam name, and navigation buttons. We recommend using *Visual Studio* to edit index.htm and other Exam Engine files.

matching_lines.htm

index.htm
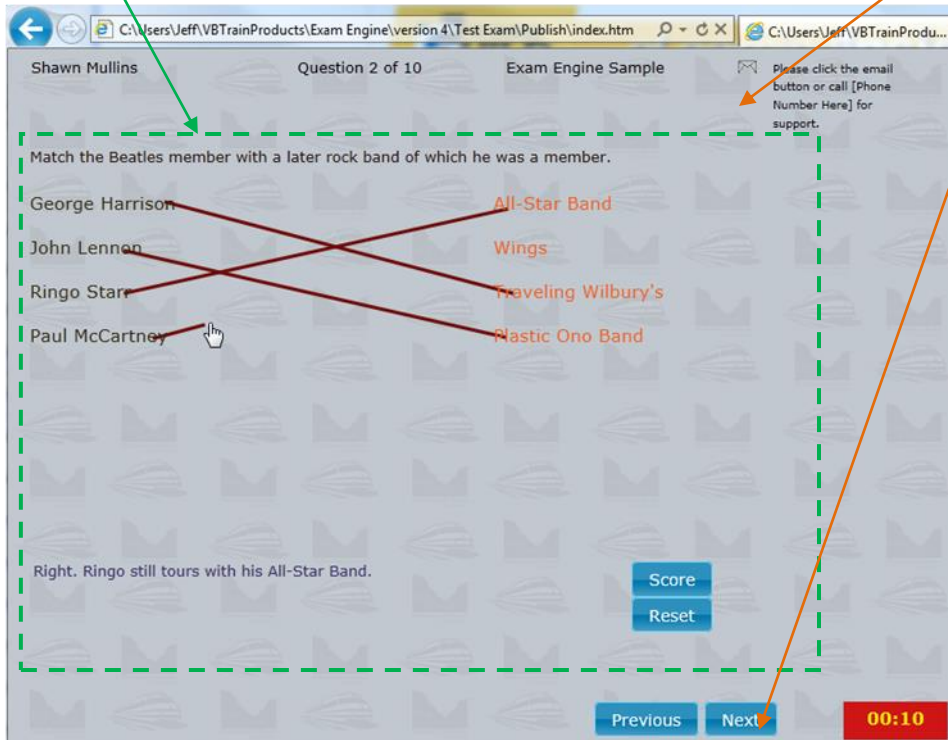
The key elements of *index.htm* are shown below. Note that one of the key features of Exam Engine 4.5 is its *responsive web design*. The styles associated with this are shown later in the document.

Item	Description

userNameLabel	This is the *span* that displays the User's Name, if that is enabled in the exam itself. Its font, color, and position are controlled by the *userNameLabel* class in *backgroundstyles.css* as shown below:
	
	``css
	#userNameLabel
	{
	  position: absolute;
	  left: 20px;
	  top: 8px;
	  color: black;
	}
	``

numQuestionsLabel	This is the *span* that displays the "Question 6 out of 12" text, if that is enabled in the exam itself. Its font, color, and position are controlled by the *numQuestionsLabel* class in *backgroundstyles.css* as shown below:

Item	Description

	``css
	#numQuestionsLabel
	{
	  position: absolute;
	  left: 250px;
	  top: 8px;
	  color: black;
	}
	``

examNameLabel	This is the *span* that displays the Exam Name, if that is enabled in the exam itself. Its font, color, and position are controlled by the *examNameLabel* class in *backgroundstyles.css* as shown below:
	
	``css
	#examNameLabel
	{
	  position: absolute;
	  left: 430px;
	  top: 8px;
	  color: black;
	}
	``

supportGrid	This is a *div* that contains the "tech support email" graphic and associated technical support text. This can be turned off as part of the exam configuration. Its size and position are controlled by the *supportGrid, supportGraphicDiv, supportLabelDiv,* and *supportMessageLabel* classes in *backgroundstyles.css* as shown below:
	
	``css
	#supportGrid
	{
	  position: absolute;
	  left: 630px;
	  top: 8px;
	  width: 150px;
	}
	
	#supportGraphicDiv
	{
	  float: left;
	  width: 25px; /* adjust if change email graphic */
	}
	
	#supportLabelDiv
	{
	  position: absolute;
	  left: 25px;
	  width: 80%; /* 120px / 150 px (context) = 80%*/
	}
	
	#supportMessageLabel
	{
	  font-size: 75%;
	}
	``

transition	This is the area of *index.htm* that contains the individual question templates such as *matching_lines.htm* in the figure above. It is an *iFrame* control that is controlled by the *transition* class in *backgroundstyles.css* as shown below. Note that the width corresponds to the width of the question templates. The height is set to 100%, so that will work with any template

Item	Description

	size. If you adjust the dimensions of the *transition* frame, be sure to adjust the question templates to match.
	
	``css
	#transition
	{
	  position: absolute;
	  left: 8px;
	  top: 55px;
	  width: 792px;
	  height: 100%;
	  overflow: hidden; /* no scrolling */
	  border-width: 0px;
	  border-style: none;
	}
	``

Status and Navigation Buttons	The bottom of the screen (by default) contains the navigation buttons, "status" text, and the "Countdown Timer." These are all controlled by these classes in *backgroundstyles.css*. Note that there are img elements as well as input controls for each button. The img elements are used if the exam has graphical buttons and the input elements are used if the exam has *jQuery UI* buttons.
	
	``css
	#statusLabel
	{
	  position: absolute;
	  width: 475px;
	  color: black;
	  left: 8px;
	  top: 550px;
	  padding-right: 5px;
	}
	
	#previousBtn
	{
	  position: absolute;
	  left: 498px;
	  top: 550px;
	  display: none;
	}
	
	#emailResultsBtn
	{
	  position: absolute;
	  left: 558px;
	  top: 550px;
	  display: none;
	}
	
	#previousImageBtn, #emailResultsImageBtn
	{
	  position: absolute;
	  left: 558px;
	  top: 550px;
	  display: none;
	}
	``

Item	Description

	``
	#nextBtn
	{
	  position: absolute;
	  left: 593px;
	  top: 550px;
	  display: none;
	}
	
	#scoreExamBtn
	{
	  position: absolute;
	  left: 593px;
	  top: 550px;
	  display: none;
	}
	
	#nextImageBtn, #scoreExamImageBtn /* same as nextBtn since only
	one showing at a time */
	{
	  position: absolute;
	  left: 626px;
	  top: 550px;
	  display: none;
	}
	
	#exitBtn, #exitImageBtn
	{
	  position: absolute;
	  left: 708px;
	  top: 550px;
	  display: none;
	}
	
	#countdownTimer
	{
	  position: absolute;
	  left: 713px;
	  top: 550px;
	  width: 90px;
	  height: 33px;
	  text-align: center;
	  display: none;
	}
	
	#countdownTimerTextBlock
	{
	  font-weight: bold;
	  font-size: 120%;
	  color: #f1e905;
	  display: block; /* needed so margin will work */
	  margin-top: 5px;
	}
	``

AudioPlayer	All audio is played in this *div*, located by default below the navigation buttons. It is controlled by this class. ``` #audioPlayer ```

Item	Description

	``css
	{
	  position: absolute;
	  left: 0px;
	  top: 590px;
	}
	``

reviewQuestion	This is another *iFrame* control that shows review questions at the end of the exam. Again, you would want to adjust this to match the size of the templates. It is controlled by *backgroundstyles.css*:
	
	``css
	#reviewQuestion
	{
	  position: absolute;
	  left: 8px;
	  top: 640px;
	  width: 792px;
	  height: 555px;
	  overflow: hidden; /* no scrolling */
	  display: none;
	  border-width: 1px;
	  border-style: double;
	  border-color: black;
	}
	``

Here is the entire HTML of this file with associated comments.

```html
<!doctype html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title></title>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width,init
    <meta http-equiv="X-UA-Compatible" content="IE=Edge,ch
    <link href="styles/commonstyles.css" rel="stylesheet" />
    <link href="styles/backgroundstyles.css" rel="stylesheet" />
    <link href="css/start/jquery-ui-1.10.1.custom.css" rel="stylesheet" />
    <script src="jquery/jquery-1.9.1.js"></script>
    <script src="jquery/jquery-ui-1.10.1.custom.min.js"></sc
    <script src="scripts/BackgroundScript.js"></script>
    <script src="scripts/ExamAndInteractions.js"></script>
    <script src="scripts/ExamEngineSettings.js"></script>
    <script src="scripts/Question.js"></script>
    <script src="scripts/Interaction.js"></script>
    <script src="scripts/Questions.js"></script>
    <script src="scripts/scorm.js"></script>
</head>
<body>
    <span id="userNameLabel"></span>
    <span id="numQuestionsLabel"></span>
    <span id="examNameLabel"></span>
    <div id="supportGrid">
        <div id="supportGraphicDiv">
            <img id="emailImage" src="images/email.png" alt="" />
        </div>
        <div id="supportLabelDiv">
            <span id="supportMessageLabel"></span>
```

This ensures that the browser will use HTML if possible.

These *meta* tags allow us to use extended characters, ensure that iPhones and other devices don't try to "zoom out" to show our content, and makes sure that Internet Explorer uses standards mode.

commonstyles.css has items that are shared with templates. *backgroundstyles.css* has items only in index.htm.

jQuery is a standard JavaScript library. The rest of the .js files are unique to Exam Engine. Most of the functionality is in *ExamEngineSettings.js*.

This is where the question templates are displayed.

```
            </div>
        </div>
        <iframe id="transition"></iframe>
        <span id="statusLabel"></span>
        <img id="previousImageBtn" src="" alt="" />
        <img id="emailResultsImageBtn" src="" alt="" />
        <img id="nextImageBtn" src="" alt="" />
        <img id="scoreExamImageBtn" src="" alt="" />
        <img id="exitImageBtn" src="" alt="" />
        <input id="previousBtn" type="button" />
        <input id="emailResultsBtn" type="button" />
        <input id="nextBtn" type="button" />
        <input id="scoreExamBtn" type="button" />
        <input id="exitBtn" type="button" />
        <div id="countdownTimer" class="InitialCountdownColor">
            <span id="countdownTimerTextBlock"></span>
        </div>
        <div id="audioPlayer"></div>
        <div id="emaildialog-modal">
            <table class="EmailTable">
                <tr>
                    <td colspan="2">
                        <span id="EmailInstructionsLabel"></span>
                    </td>
                </tr>
                <tr>
                    <td class="EmailColumn1">
                        <span id="EmailSubjectLabel"></span>
                    </td>
                    <td class="EmailColumn2">
                        <input id="EmailSubjectBox" type="text" />
                    </td>
                </tr>
                <tr>
                    <td class="EmailColumn1">
                        <span id="CopyToEmailLabel"></span>
                    </td>
                    <td class="EmailColumn2">
                        <input id="CopyToEmailBox" type="text" />
                    </td>
                </tr>
                <tr>
                    <td class="EmailColumn1">
                        <span id="EmailMessageLabel"></span>
                    </td>
                    <td class="EmailColumn2">
                        <textarea id="EmailMessageBox" rows="3" cols="25"
class="EmailMessageEntry"></textarea>
                    </td>
                </tr>
                <tr>
                </tr>
                <tr>
                    <td class="EmailColumn1">
                        <input id="SubmitEmailBtn" type="button" />
                    </td>
                    <td class="EmailColumn2">
                        <input id="CloseEmailBtn" type="button" />
```

Note that all the positioning and formatting is controlled by *backgroundstyles.css* as explained above.

Audio is played via HTML 5 if possible. If the browser does not support HTML 5 audio, then Flash is used except in Safari, where QuickTime is used instead. The player is hidden in these cases.

This table is used by the "Email Support" dialog box.

```
                </td>
            </tr>
        </table>
    </div>
    <iframe id="reviewQuestion"></iframe>
</body>
</html>
```

This is where the review questions are shown at the end of the exam.

Styles

Virtually all of the formatting (size, position, font, etc.) of elements are controlled by style sheets located in this subdirectory. We will go through each one in turn.

backgroundstyles.css

This style sheet controls the elements in *index.htm*. We have covered many of these settings earlier in this document.

commonstyles.css

This style sheet controls *body* text and other elements that can be placed in either the index.htm or various template files.

templatestyles.css

This style sheet has common settings like the question text settings shown below as well as unique settings for particular templates. The # symbol as in *#answersDiv* means that this style goes with an element with an *id* that matches ("answersDiv" in this case).

```css
body
{
  background-color: transparent;
  overflow: hidden; /* no scrollbars */
}

video
{
  width: 100%;
  max-width: 100%;
}

/* Common to Multiple Templates */

#answersDiv
{
  left: 8px;
  top: 70px;
  width: 100%;
  position: absolute;
}

#questionText
{
  color: #220F04;
  font-size: 100%;
  left: 10px;
  top: 0px;
}
```

```css
#questionMedia, #questionGraphic
{
  left: 440px;
  top: 70px;
  position: absolute;
}

#questionGraphic
{
  display: inline;
}

#questionGraphic2
{
  display: inline;
}

#questionGraphic3
{
  display: inline;
}

#questionMedia4
{
  width: 100%;
  height: 100%;
}

#questionGraphic4
{
  display: inline;
}

#scoreImageBtn, #scoreBtn
{
  left: 490px;
  top: 340px;
}

#resetQuestionImageBtn, #resetQuestionBtn
{
  left: 490px;
  top: 370px;
}

#showAnswersImageBtn, #showAnswersBtn
{
  left: 490px;
  top: 400px;
}

#instructionsFeedbackText
{
  left: 0px;
  top: 460px;
  width: 738px;
  height: 58px;
```

```css
  position: absolute;
  color: #342B68;
  font-size: 100%;
}

#scoreBtn, #scoreImageBtn, #resetQuestionBtn, #resetQuestionImageBtn, #showAnswersBtn,
#showAnswersImageBtn
{
  cursor: pointer;
  position: absolute;
}

#scoreBtn
{
  width: 125px;
}

#resetQuestionBtn
{
  width: 125px;
}

#showAnswersBtn
{
  width: 125px;
}

#answer_1, #answer_2, #answer_3, #answer_4, #answer_5, #answer_6, #answer_7, #answer_8
{
  cursor: pointer;
}

#answer_1_imageButton, #answer_2_imageButton, #answer_3_imageButton, #answer_4_imageButton,
#answer_5_imageButton, #answer_6_imageButton, #answer_7_imageButton,
#answer_8_imageButtonButton, #answer_2_imautton, #answer_4_imageButton, #answer_5_imageButton,
#answer_6_imageButton, #answer_7_imageButton, #answer_8_imageButton
{
  float: left;
}

/* Classes */

.AnswersInitialTextColor
{
  color: #2D2805;
  z-index: 2;
}

.AnswersSelectedTextColor
{
  color: #1111F9;
}

.AnswersCorrectTextColor
{
  color: #128212;
}
```

```css
.AnswersIncorrectTextColor
{
  color: #ED2919;
}

.HiddenContent
{
  visibility: hidden;
}

.InitialImage
{
  border-width: 0;
}

.CorrectImage
{
  border-color: #128212;
  border-width: 3px;
  border-style: double;
}

.IncorrectImage
{
  border-color: #ED2919;
  border-width: 3px;
  border-style: double;
}

.SelectedImage
{
  border-color: #1111F9;
  border-width: 3px;
  border-style: double;
}
```

Note that In addition, settings applicable only to a particular template are located in the style sheet associated with that template.

Individual Template Styles

Each template has a corresponding style sheet. These can override settings like *#questionMedia* and *#instructionsFeedbackText*. They also define settings applicable only to that template. For example, *hotobjects.css* in the *styles* directory corresponds to *hotobjects.htm* in the *templates* directory. Here is an example of what that template looks like.

Here is the entire style sheet along with some comments.

```css
#answer_1, #answer_2, #answer_3, #answer_4, #answer_5, #answer_6, #answer_7, #answer_8
{
  position: absolute;
}

#answer_1
{
  left: 0px;
  top: 0px;
}

#answer_2
{
  left: 190px;
  top: 0px;
}

#answer_3
```

This sets the position of each answer image as *absolute*. Notice this supplements *templatestyles.css,* which sets their cursor to be a *pointer* (hand).

These entries set the position of each of the answer images. You will need to adjust these if your images are a different size.

```
{
  left: 0px;
  top: 160px;
}

#answer_4
{
  left: 190px;
  top: 160px;
}

#answer_5
{
  left: 380px;
  top: 0px;
}

#answer_6
{
  left: 380px;
  top: 160px;
}

#answer_7
{
  left: 570px;
  top: 0px;
}

#answer_8
{
  left: 570px;
  top: 160px;
}

#scoreImageBtn, #scoreBtn
{
  top: 398px;
}

#resetQuestionImageBtn, #resetQuestionBtn
{
  top: 428px;
}

#showAnswersImageBtn, #showAnswersBtn
{
  top: 458px;
}

#instructionsFeedbackText
{
  width: 443px;
}

#questionMedia, #questionGraphic
{
  top: 130px;
```

We need to move the various buttons (Score, Reset, and Show Answers) down compared to the default. That is why we set the *top* value.

Similarly, we need to adjust the width of the instructions and feedback so as to not overlap the buttons. We also adjust the location of any media or graphic.

Responsive Web Design

Responsive Web Design is a fairly new concept with the idea being that we want our web content to adjust to different browser capabilities and, in particular, viewport or screen sizes. So rather redirecting to a different set of content when accesses by an iPhone, iPad, or Android device with a smaller screen, we adjust our content accordingly. Ethan Marcotte coined the phrase and is one of its biggest proponents. Here are two articles that we found helpful: http://alistapart.com/article/responsive-web-design and http://unstoppablerobotninja.com/entry/fluid-images. We first learned about the concept in this excellent book: http://www.amazon.com/gp/product/B007SVJA3M/ref=oh_d__o00_details_o00__i00.

Making *Exam Engine* content responsive is challenging in the sense that you have to deal with both the background (index.htm) and the templates shown within the iFrame. But having templates rather than individual pages of training makes it quite a bit easier as there are a limited number of templates to fine-tune. The approach we used is to adjust the background elements both horizontally and vertically so they fit with both small and large screen sizes. With the templates, we only adjust them horizontally. This keeps the templates from running over the navigation buttons and so forth when they are resized[4]. Here is a *matching* template at full size:

---
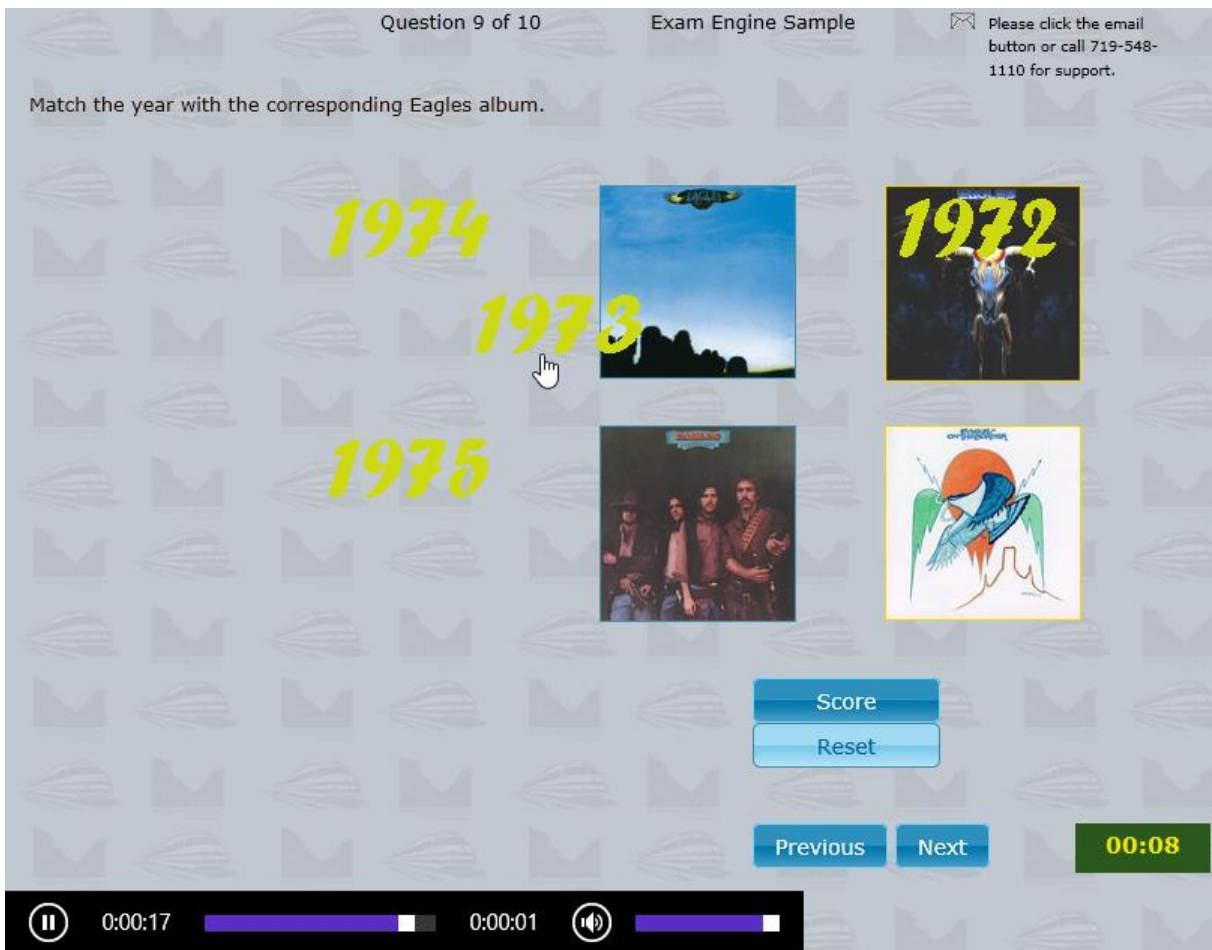
[4] This can still happen if you have long feedback. If you are deploying to smaller devices like iPhones, we recommend either limiting question feedback or adjusting *backgroundstyles.css* to further push down the navigation buttons.

Here it is again at a width of about 520 pixels:

Notice that we moved the *Previous* and *Next* buttons to the left of the screen. The status text (not shown) is now below those buttons and above the audio controls (which the items within the template have sized automatically so they still fit on the screen. The drag and drop functionality still works fine, though with this particular template the "reset" functionality only operates correctly when the user has not resized the screen while this template is displayed.

Here is the same screen at about 390 pixels. It still looks good but any smaller and we start running into problems with the support text wrapping into the question text and so forth. Notice that we have moved the exam name to its own line so that it doesn't overlap the "Question 10 of 10" text. We have also moved the "countdown timer" right next to the Score Exam button so that it doesn't get cut off.

Here is an even smaller screen size (290 pixels). We have moved the support text to its own line[5]. It now wraps and gets in the way of the question text. But some minor adjustments on the length of the text or omitting the support link altogether would take care of that. Long feedback and status text also causes problems at such a small resolution. But even without these adjustments, we've gone a long way towards making our content quite viewable at multiple screen sizes.

The general approach to making this work is to first leave the style sheets as described above alone until we get to a width below our default size (800 pixel width for the templates that come with *Exam Engine*). We then use a CSS media query to adjust from there. It is important to understand that these CSS settings then stay in place until either another media query kicks in (such as at 400 pixels or whatever) or if the user resizes her browser above the value[6].

backgroundstyles.css

There are multiple media queries. At the first one (800 pixels width), we change all *left* and *width* values to the corresponding percentage values. As discussed in the references above, this uses the all-important formula: *result = target / context*. In our case, the context is our width, typically 800 pixels. The target is the existing amount in pixels. So if we have *left: 8px;*, we change it to *left: 1%;*, since 8 / 800 = 1%. Here are the media queries with some comments in line.

```
@media screen and (max-width: 800px)
{
  #userNameLabel
  {
    left: 2.5%; /* 20px (target) / 800px (content) =
0.025 - 2.5%*/
  }

  #numQuestionsLabel
  {
    left: 31.25%;
```

```
}

#examNameLabel
{
```

---

[5] All of these screen shots have a spot for the student name, which accounts for the space at the top of the screen.
[6] Another way the media query could go away or change to a different value is if the user changes the orientation of the device, such as going from portrait to landscape.

```css
  left: 53.75%;
}

#supportGrid
{
  left: 78.75%;
  width: 18.75%;
}

#transition
{
  left: 1%;
  width: 99%;
}

#reviewQuestion
{
  left: 1%;
  width: 99%;
}

#statusLabel
{
  left: 1%;
  width: 59.375%;
}

#previousBtn
{
  left: 62.25%;
}

#emailResultsBtn
{
  left: 69.75%;
}

#previousImageBtn, #emailResultsImageBtn
{
  left: 70.75%;
}

#nextBtn
{
  left: 74.125%;
}

#scoreExamBtn
{
  left: 74.125%;
}

#nextImageBtn, #scoreExamImageBtn
{
  left: 78.25%;
}

#exitBtn, #exitImageBtn
```

```css
  {
    left: 88.5%;
  }

  #countdownTimer
  {
    left: 89.125%;
    width: 11.25%;
  }
}

@media screen and (max-width: 768px)
{
  #statusLabel
  {
    width: 235px;
  }

  #previousBtn
  {
    left: 250px;
  }

  #emailResultsBtn
  {
    left: 310px;
  }

  #previousImageBtn, #emailResultsImageBtn
  {
    left: 318px;
  }

  #nextBtn, #scoreExamBtn
  {
    left: 345px;
  }

  #nextImageBtn, #scoreExamImageBtn
  {
    left: 378px;
  }

  #exitBtn, #exitImageBtn
  {
    left: 460px;
  }

  #countdownTimer
  {
    left: 465px;
    width: 80px;
  }
}

@media screen and (max-width: 570px)
{
```

We switch the *left* and *width* of the status label and the various buttons to a fixed pixel amount so that they won't move further or get too thin.

```
#supportGrid
{
  left: 2.5%;
  top: 30px;
  width: 100%;
}

#statusLabel
{
  width: 100%;
  top: 600px;
}

#previousBtn
{
  left: 0px;
}

#emailResultsBtn
{
  left: 60px;
}

#previousImageBtn, #emailResultsImageBtn
{
  left: 68px;
}

#nextBtn, #scoreExamBtn
{
  left: 95px;
}

#nextImageBtn, #scoreExamImageBtn
{
  left: 128px;
}

#exitBtn, #exitImageBtn
{
  left: 210px;
}

#countdownTimer
{
  left: 215px;
  width: 80px;
}

#audioPlayer
{
  top: 640px;
}
}

@media screen and (max-width: 490px)
{
```

We move the support text and the status label their own lines. We also adjust the various buttons (*Next*, *Previous*, etc.) to fit at the reduce screen width.

```
#examNameLabel
{
  left: 2.5%;
  top: 30px;
}
```

We now put the exam name on its own line and further adjust the positions of the labels and buttons.

```
#supportGrid
{
  top: 60px;
}

#transition
{
  top: 85px;
}

#statusLabel
{
  top: 630px;
}

#previousBtn
{
  top: 580px;
}

#emailResultsBtn
{
  top: 580px;
}

#previousImageBtn, #emailResultsImageBtn
{
  top: 580px;
}

#nextBtn
{
  top: 580px;
}

#scoreExamBtn
{
  top: 580px;
}

#nextImageBtn, #scoreExamImageBtn
{
  top: 580px;
}

#exitBtn, #exitImageBtn
{
  top: 580px;
}

#countdownTimer
{
```
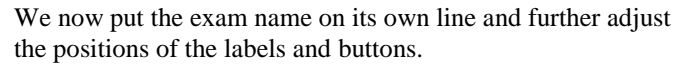
```css
    top: 580px;
  }

  #audioPlayer
  {
    top: 670px;
  }
}

@media screen and (max-width: 400px)
{

  #numQuestionsLabel
  {
    left: 2.5%;
    top: 30px;
  }

  #examNameLabel
  {
    top: 60px;
  }

  #supportGrid
  {
    top: 90px;
  }

  #transition
  {
    top: 105px;
  }
}
```

The *templatestyles.css* and the individual style sheets for the templates have similar media queries. Of particular interest is the handling of images and media that are designed to fill a particular portion of the screen. Here is the media query from *hotobjects.css*.

```css
@media screen and (max-width: 791px)
{
  #answer_1, #answer_2, #answer_3, #answer_4, #answer_5, #answer_6, #answer_7, #answer_8
  {
    max-width: 23.75%;
  }

  #answer_2
  {
    left: 23.75%;
  }

  #answer_4
  {
    left: 23.75%;
  }

  #answer_5
  {
```

```css
    left: 47.5%;
  }

  #answer_6
  {
    left: 47.5%;
  }

  #answer_7
  {
    left: 71.25%;
  }

  #answer_8
  {
    left: 71.25%;
  }

  #instructionsFeedbackText
  {
    width: 55.375%;
  }
}
```

We use the *max-width* setting to ensure that the images (answer_1, answer_2, etc.) scale down as we reduce the screen width. On an 800 pixel width screen, the images are designed to be a maximum of 190 pixels wide. So we use our *result = target / context* formula to come up with *23.75% = 190 / 800*. Note that this query starts at *791* pixels rather than *800*. That is because the *iFrame* itself is *792* pixels wide and testing revealed that we needed a 1-pixel offset.

css

This directory is used by *jQuery UI* buttons. Exam Engine uses the *Start* theme by default. That is why there is a *start* subdirectory in the *\css* directory. Inside that directory is jquery-ui-1.10.1.custom.css. You can see the *Start* theme at http://tinyurl.com/dbqg2t. You can use a different theme by going to http://jqueryui.com/themeroller and downloading your own theme. If your .css name is different than jquery-ui-1.10.1.custom.css (for example there is a newer version of jQuery UI), search the project and update all references to the new .css file name and directory. Be sure to put its images in the \css\<theme name>\images directory. For example, this is the link inside choice.htm:

```html
<link href="../css/start/jquery-ui-1.10.1.custom.css" rel="stylesheet" />
```

If you changed to the *Sunny* theme, you would change this and similar references to:

```html
<link href="../css/sunny/jquery-ui-1.10.1.custom.css" rel="stylesheet" />
```

Graphical/jQuery UI Buttons

Buttons are either graphical buttons consisting of graphics from the *images* directory or *jQueryUI* buttons controlled by the *\css\<theme name>* directory as discussed above.

Answer Graphics

Many of the question templates can have graphics associated with each of the question states: initial, selected, correct, and incorrect. These graphics are located in the *media* directory. Exam Engine determines which graphics to use by first looking at the various answer graphics optionally set in the *Exam Engine Question Editor* as shown below.

Initial Answer Graphic*	canyon_normal.png
Selected Answer Graphic*	canyon_selected.png
Correct Answer Graphic*	canyon_correct.png
Incorrect Answer Graphic*	canyon_incorrect.png

If these are left blank, the "Default Answer Graphics" set in the *Exam Engine Configuration Editor* are used:

Default Answer Graphics (used to load associated style in ExamEngineStyles.css)

Initial State:	train_normal.png
Selected State:	train_selected.png
Correct State:	train_correct.png
Incorrect State:	train_incorrect.png

Question Templates

Question and other templates are stored in the *templates* directory as shown to the right. Each is an .htm file that contains both the HTML and the code (JavaScript). You won't normally need to edit the code. Each template is discussed in detail later in this document.

Content and Databases

Content is stored in an Access, XML, or SQL Server database. If the exam is published as a *Local Data Source* with the *Exam Engine Configuration Editor,* then the *Exam Engine Question Editor* is used to write an XML version of the database. That is put in the "Source Database/Configuration Directory" referenced when publishing the exam. There can be multiple databases and configuration files in this directory since the database to be used can be set with the *dbName* query string parameter and the configuration file can be set in the *configFile* query string parameter[7]. If you are working with the ExamEngine project directly for testing purposes, both the main database (examdbAccess.xml) and configuration file (examengineconfig4.xml) are located in the *Configuration* directory as shown to the left.

Individual Templates

We will now go through each individual template, mostly question templates but also the other templates such as the intro and login pages. As mentioned above, we recommend using *Visual Studio* for editing templates.

If you want to add a template named "xyz," you would follow these steps.

1. Right-click on the templates directory within *Visual Studio* and select Add – New Item… as shown below.

---
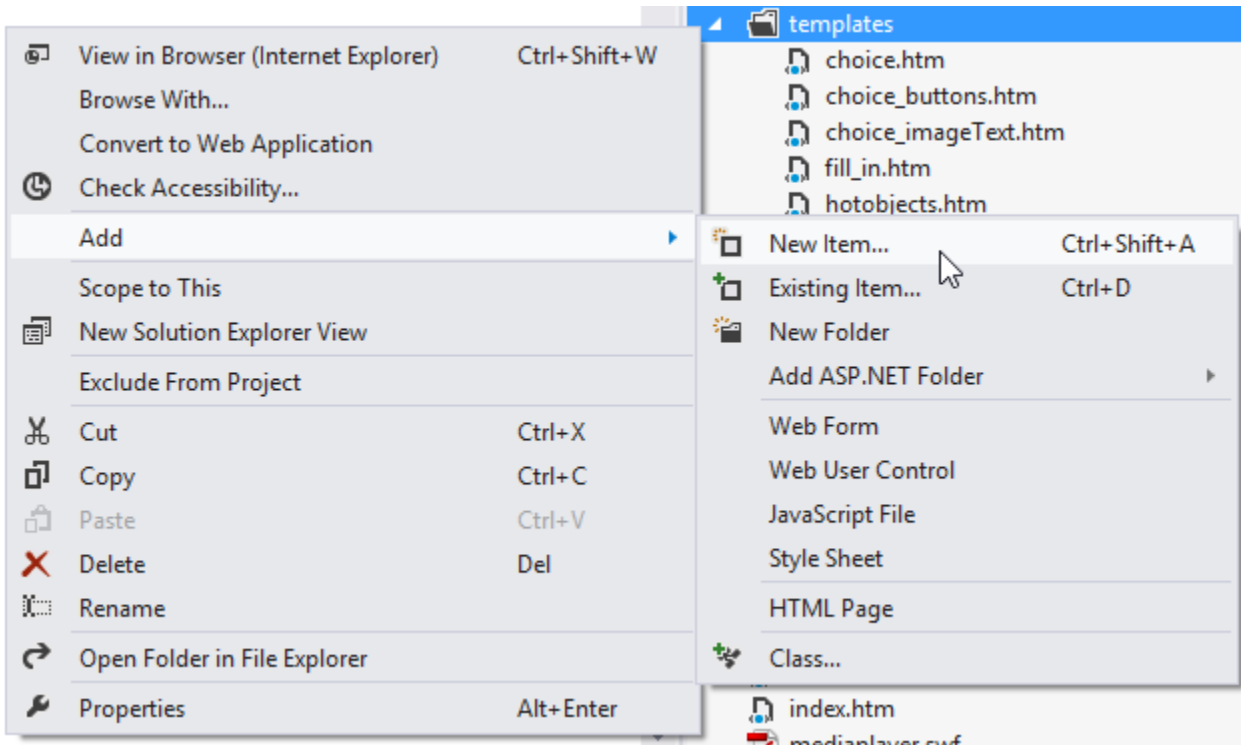
2. Select *HTML Page* and give it a name (xyz.htm). This is shown below.
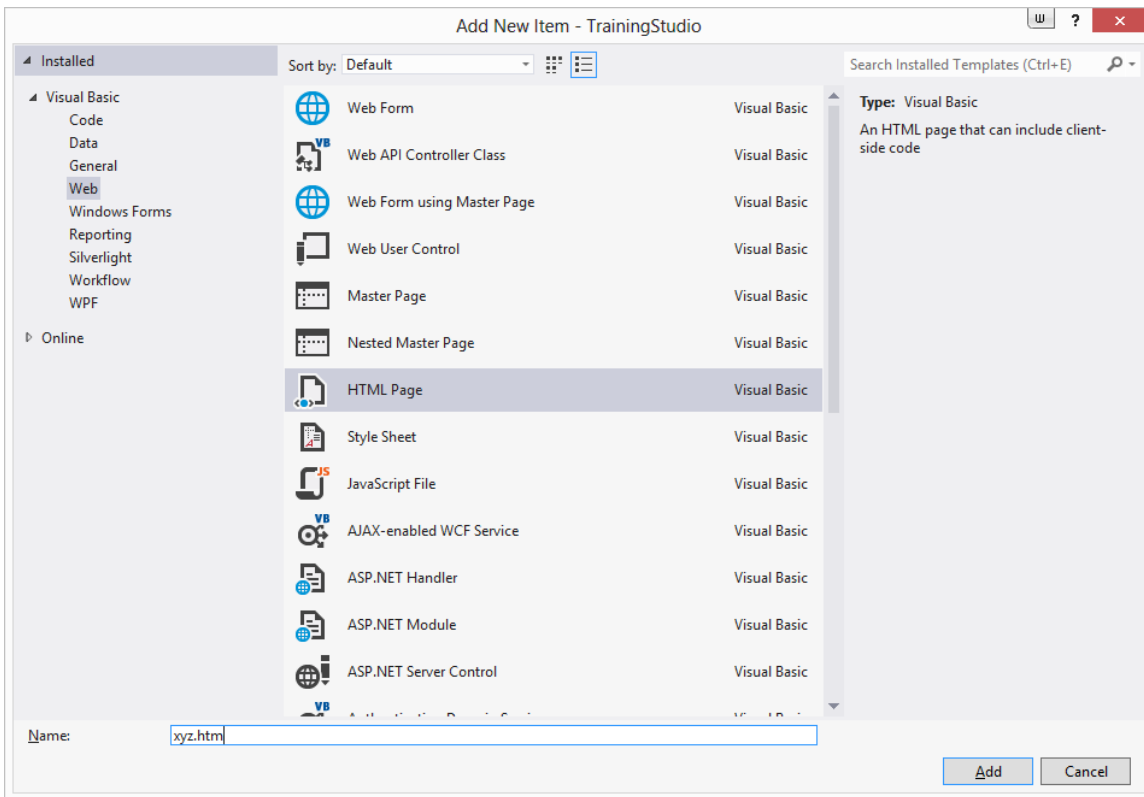
3. Copy an existing template like *choice.htm* and paste its contents into *xyz.htm*. You can then edit it as needed.

You would then *build* the project and copy the *xyz.htm* file from the *templates* directory to the same directory in your "Exam Engine Source" directory. Copy any other changed files as well

We will now go through each template in turn.

choice

This question template implements a multiple choice question with the choices being checkboxes. The template in *Visual Studio* is shown below:
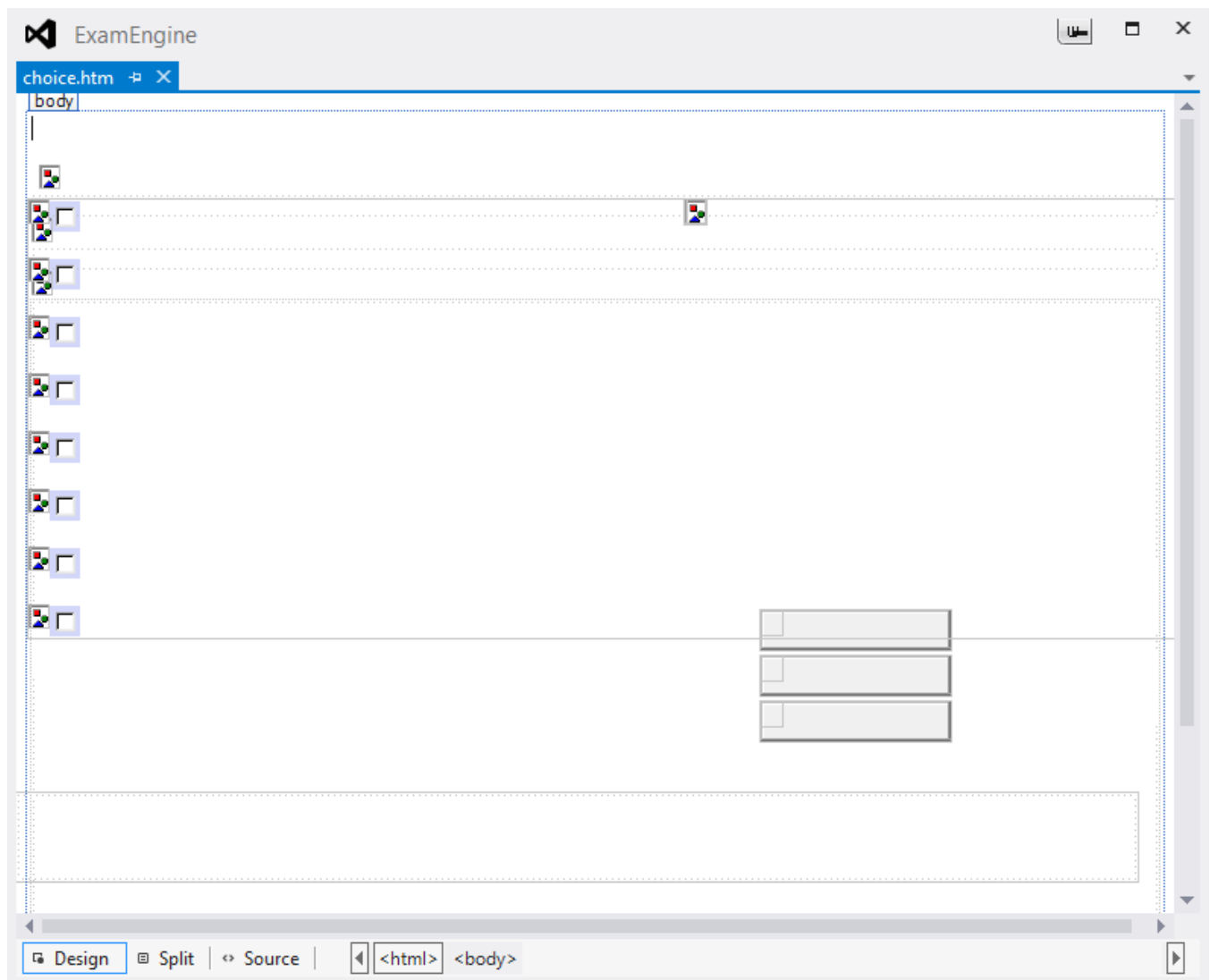
Notice how this and other template layouts are controlled by the *templatestyles.css* and individual style sheets such as *choice.css*. The *Question Text* is at the top. The checkboxes and optional images are below that within a *div*. To the right of that are the *questionGraphic* (img), and *questionMedia* (*div* that gets populated with an HTML5 media player, a Windows Media Player, a Flash Player, or a QuickTime

player as appropriate) to hold the various possible media and graphics. Below that are the *Score*, *Reset*, and *Show Answers* buttons (img or input versions are used depending on whether the exam is using graphical or *jQuery UI* buttons). At the bottom is a *span* showing the instructions initially and the feedback once an answer is selected and/or the question is scored. Hidden are additional graphic and media objects if needed. If the question has multiple correct answers, each checkbox turns the color of the *AnswersSelectedTextColor* when selected. They then turn the color of the *AnswersCorrectTextColor* or *AnswersIncorrectTextColor* once the question is scored, depending on whether the answer is correct. Here is an example of how this question looks at runtime.
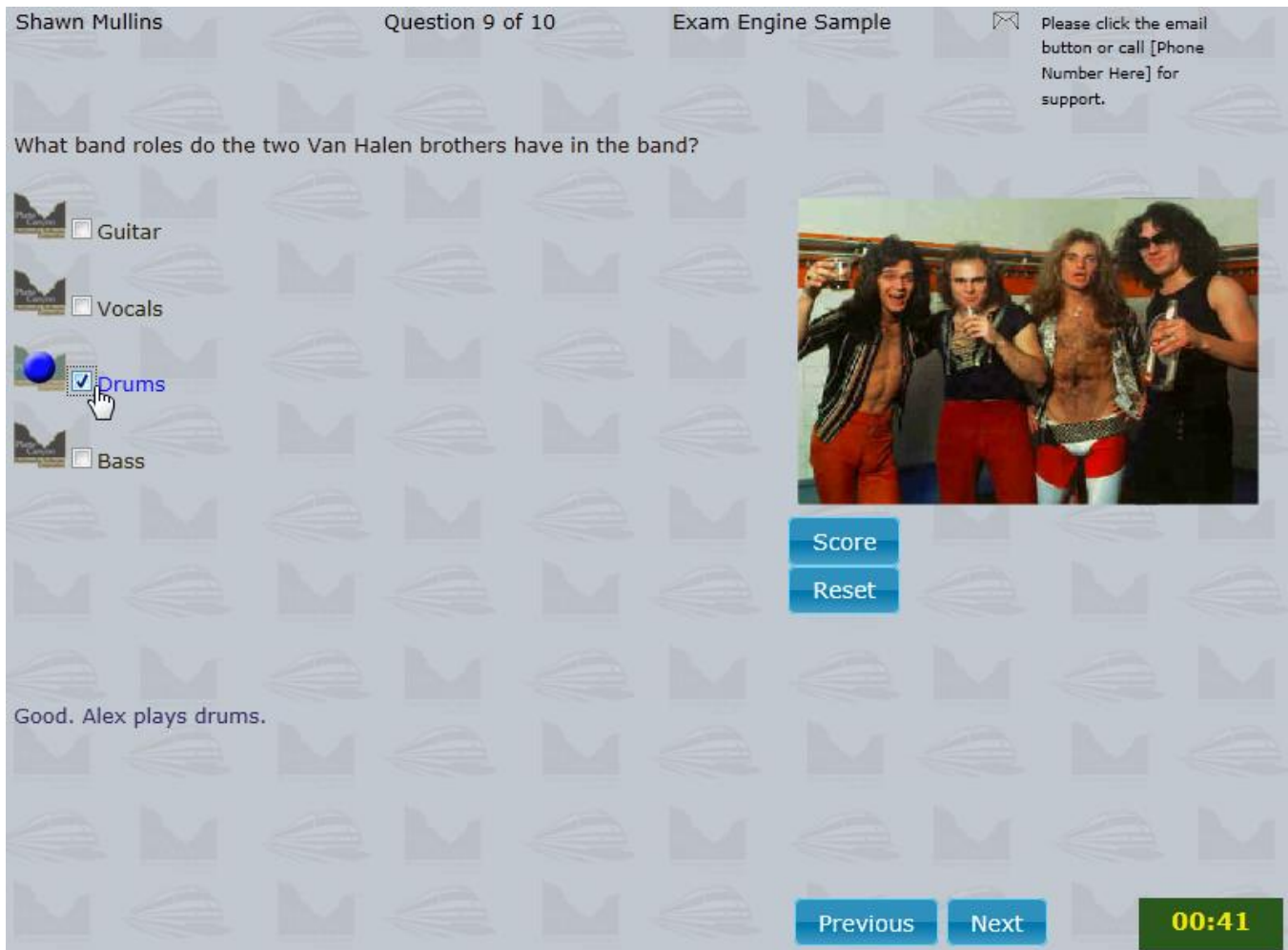
choice_buttons

This question template implements a multiple choice question with the choices being normal input buttons. The template in *Visual Studio* is shown below:
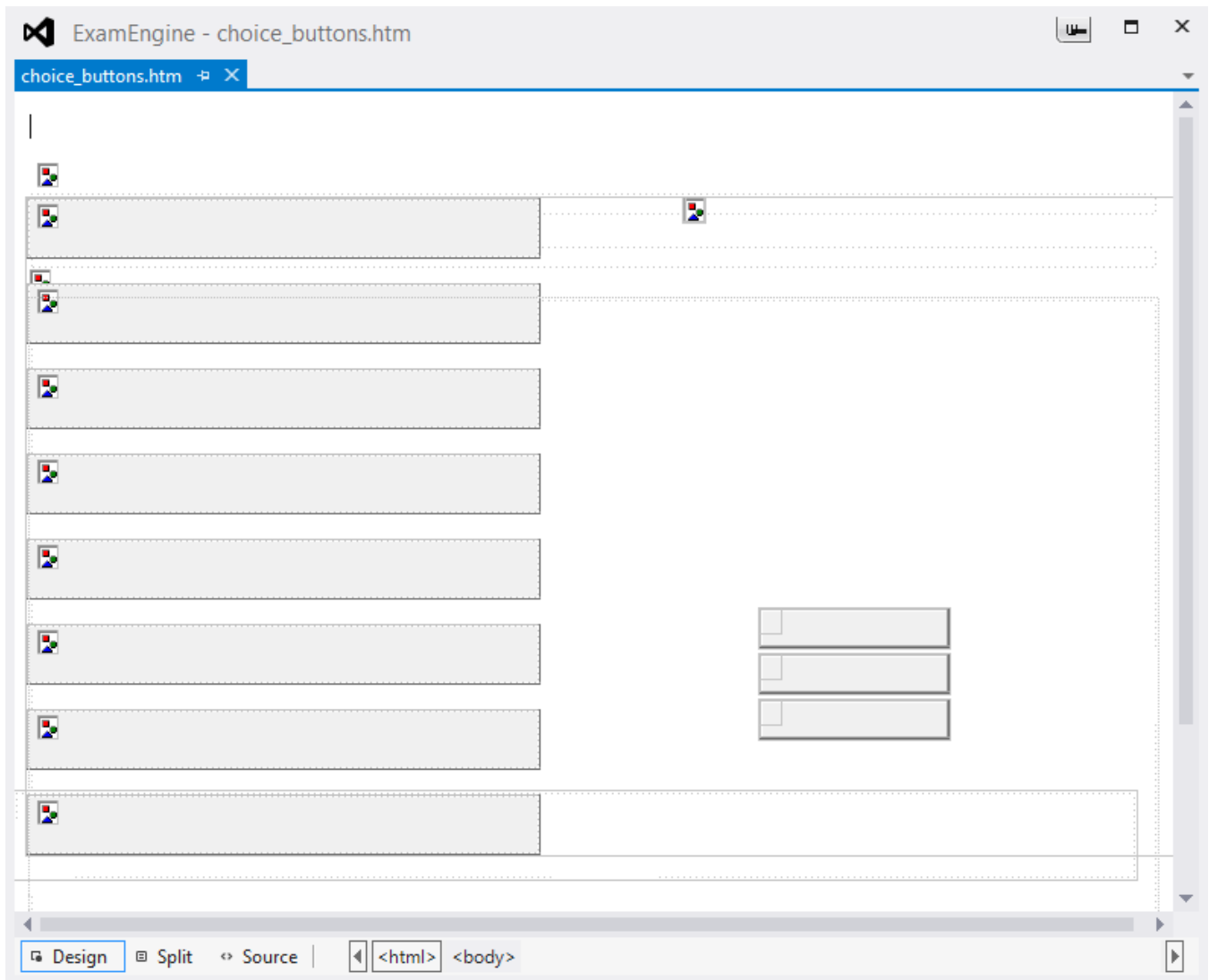
The layout is similar to the *choice* template. Here is an example of how this question looks at runtime.

choice_imageText

This question template is the same as *choice* except that the question text is an image ("Initial Graphic 1"). The normal question text label is hidden but should still be entered in the *Exam Engine Question Editor* since it is displayed on the scoring page and sent back to the LMS and the *Exam Engine Web Service*. The template in *Visual Studio* is shown below:
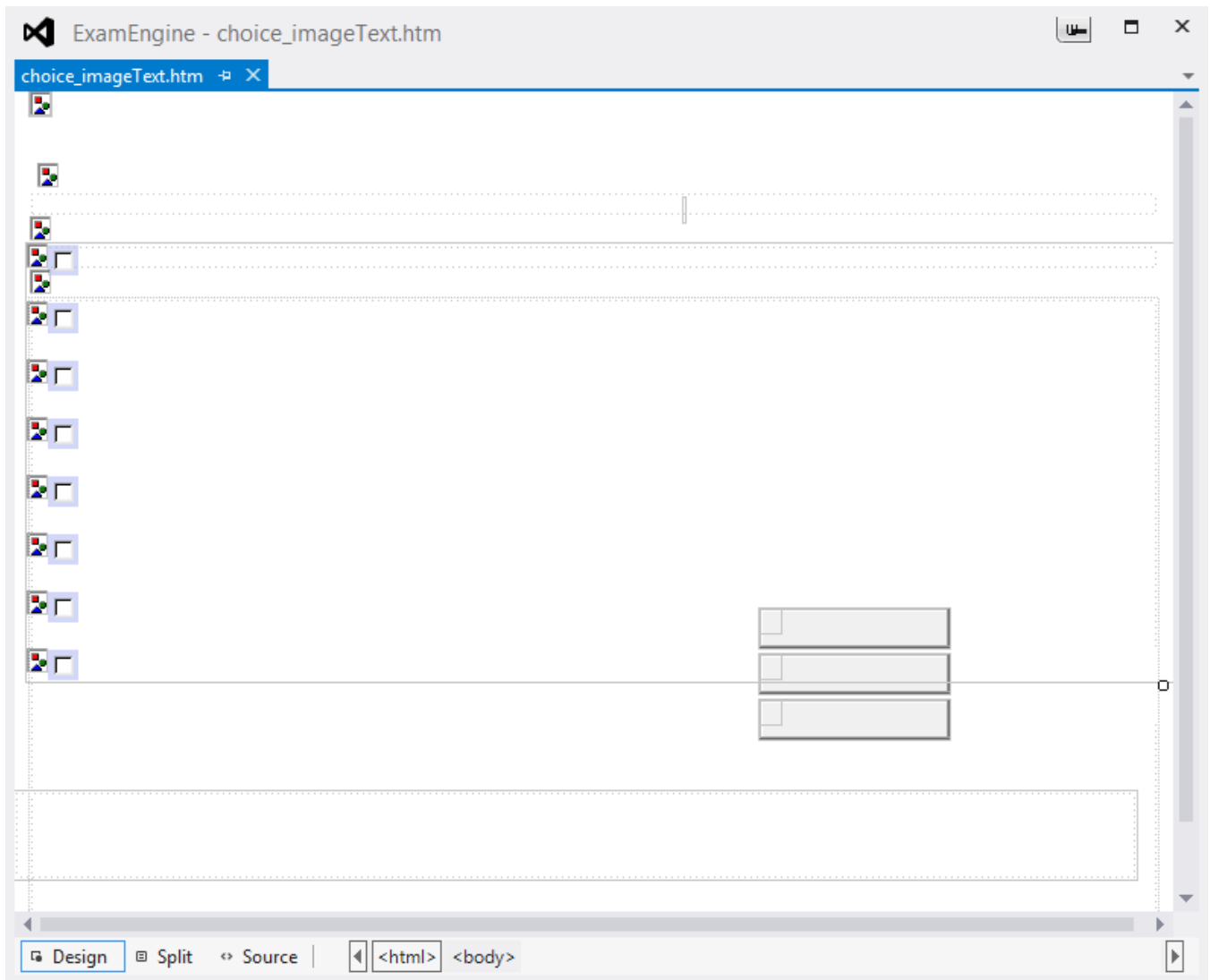
Here is an example of how this question looks at runtime.

fill_in

This question template implements a fill-in-the-blank question. The template in *Visual Studio* is shown below:
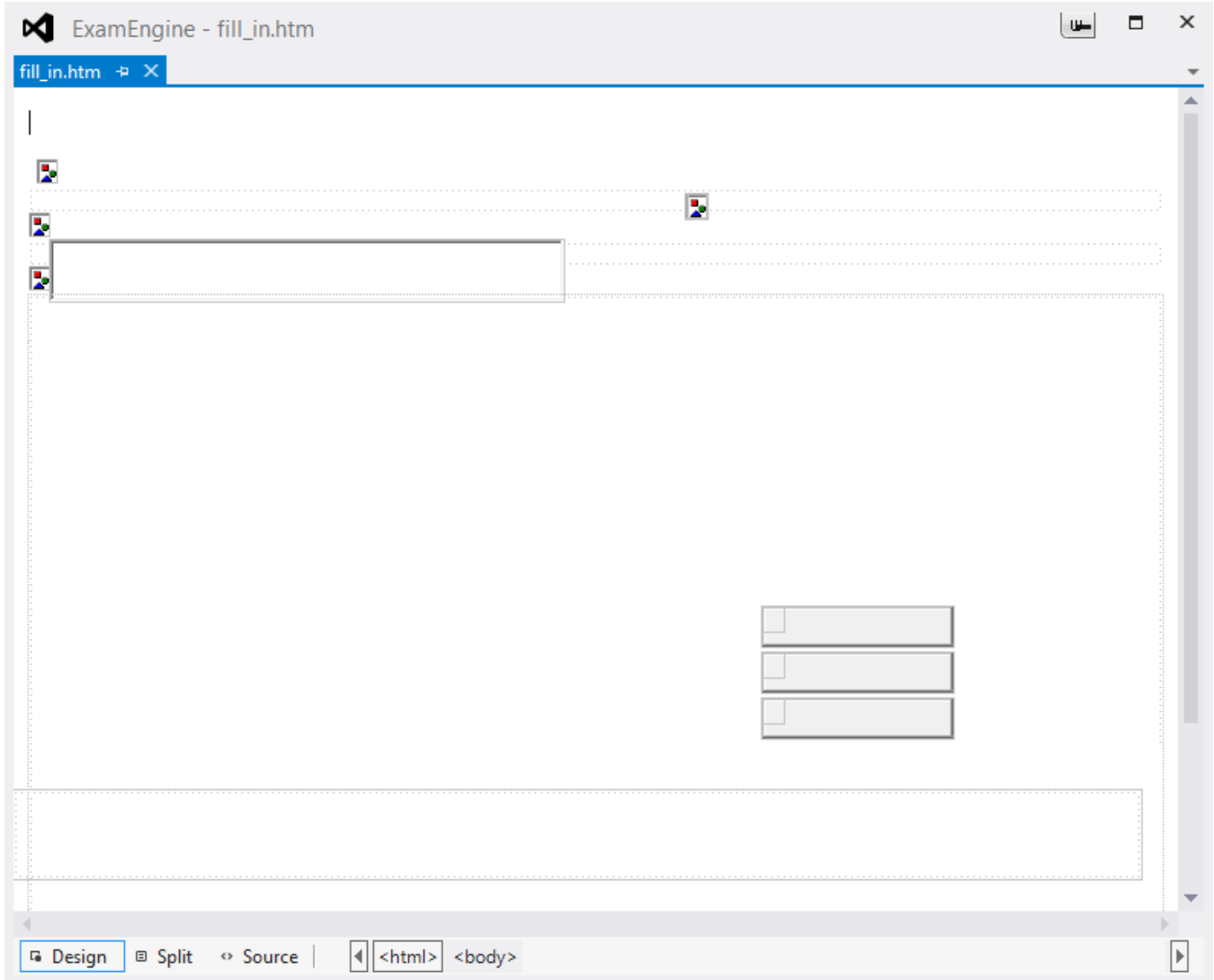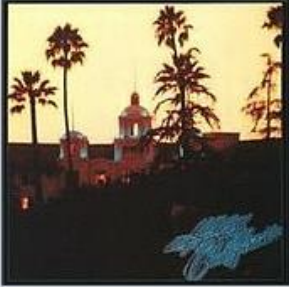
The layout is similar to the *choice* template except there is a single *input* rather than buttons to click. The user can press the Enter key or click the *Score* button to score the question. The text turns the color of the *AnswersCorrectTextColor* or *AnswersIncorrectTextColor*, depending on whether the answer is correct. Here is an example of how this question looks at runtime.

hotObjects

This question template is basically a multiple choice question where the answers are images rather than text. It is distinguished from the *hotSpot* template in that the images in *hotObjects* start off clear (technically an *opacity* of 100%). When the user selects an answer, the image gets grayed out (opacity of 50%) and rotates (by default 10° but configurable in the individual exam). The template in *Visual Studio* is shown below:
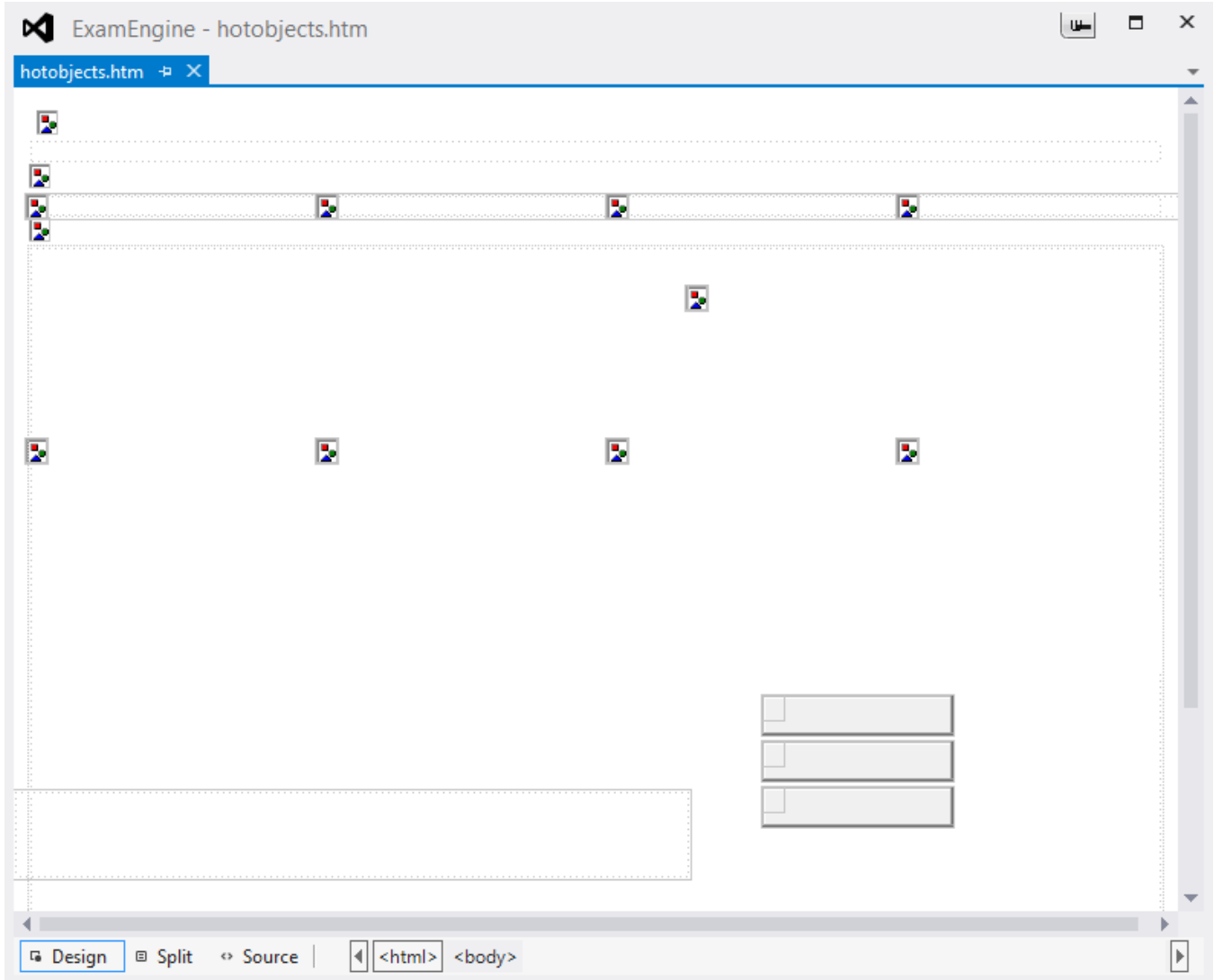
The layout is similar to the *choice* template except there the images are spread out in a grid over the entire second row (which is taller than other templates). The *media* controls are thus moved to the third row. If there is more than one correct answer, the images a style of *selectedImage*, which changes their border color and thickness. They then get a style of *correctImage* or *incorrectImage* once the question is scored, depending on whether the answer is correct. Here is an example of how this question looks at runtime.

hotSpot

This question template is basically a multiple choice question where the answers are images rather than text. It is distinguished from the *hotObjects* template in that the images in *hotSpot* start off grayed out (*opacity* of 50%). When the user selects an answer, the image turns clear (opacity of 100%) but does not rotate. The template in *Visual Studio* is shown below:
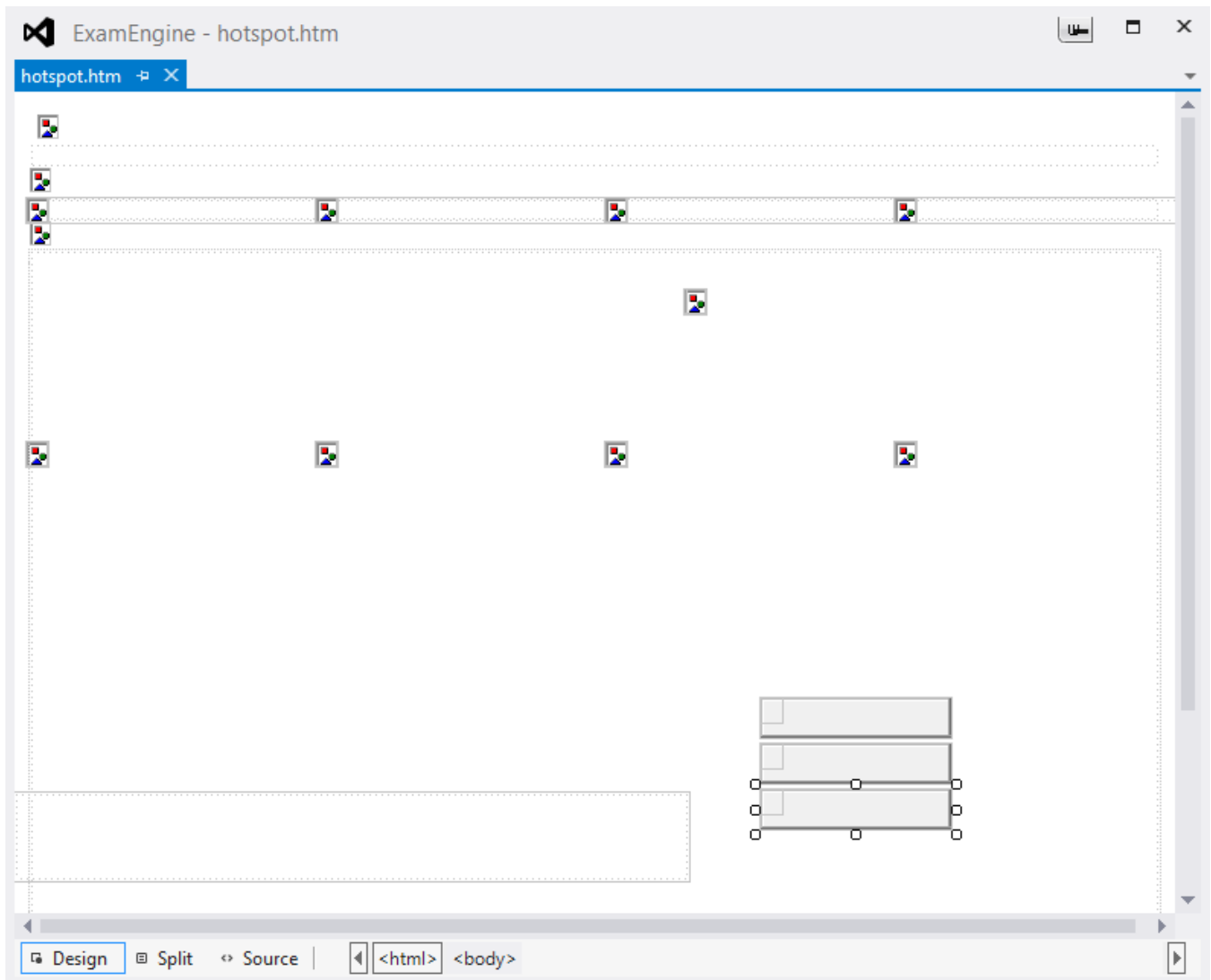
The layout is similar to the *hotObjects* template. Here is an example of how this question looks at runtime.

hotSpot_imageText

This question template is the same as *hotSpot* except that the question text is an image ("Initial Graphic 1"). The normal question text label is hidden but should still be entered in the *Exam Engine Question Editor* since it is displayed on the scoring page and sent back to the LMS and the *Exam Engine Web Service*. It is distinguished from the *hotObjects* template in that the images in *hotSpot* start off grayed out (*opacity* of 50%) but does not rotate. When the user selects an answer, the image turns clear (opacity of 100%). The template in *Visual Studio* is shown below:
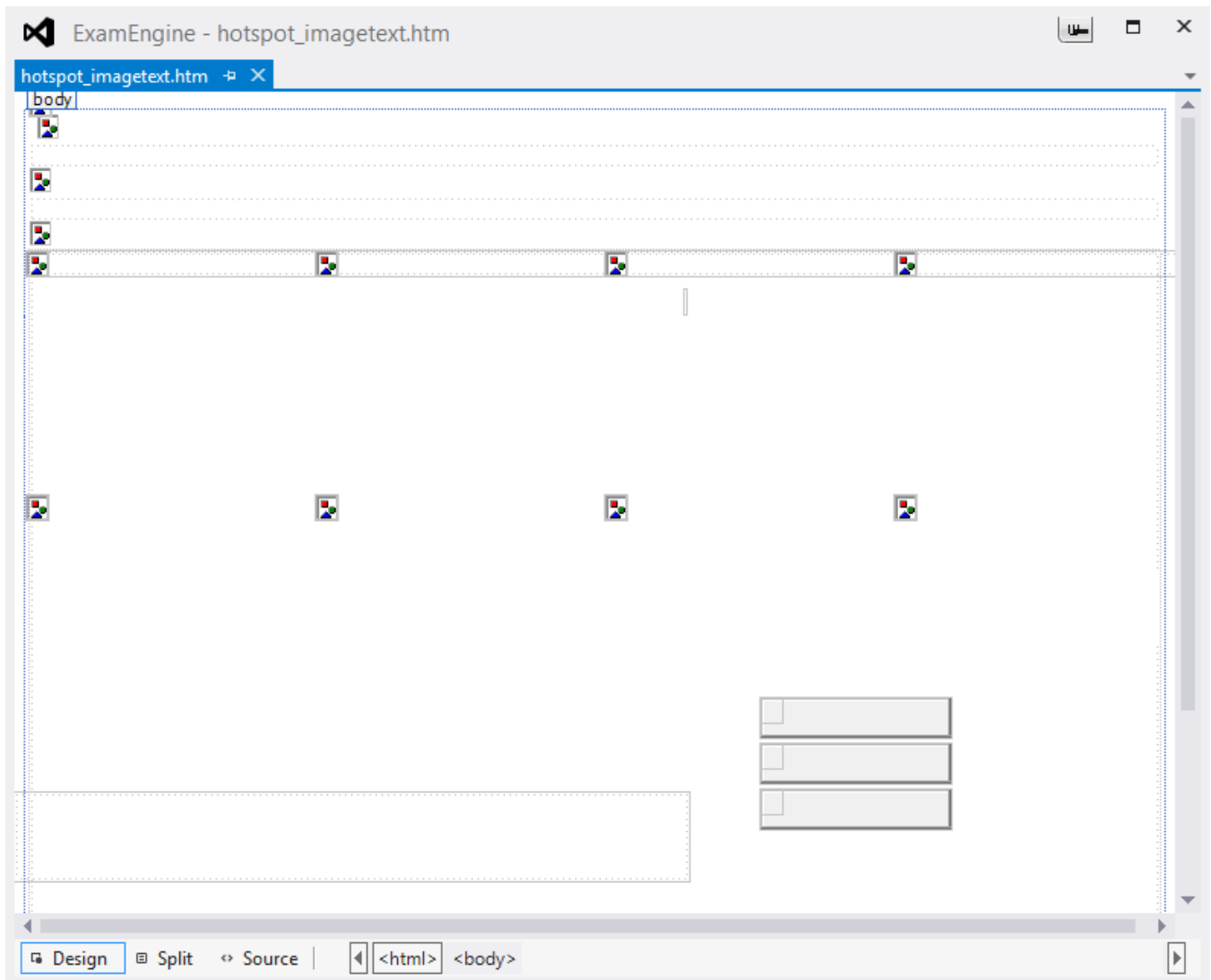
The layout is similar to the *hotObjects* template. Here is an example of how this question looks at runtime.

intro

This is not a question template but rather the page that implements the optional introduction page. The template has space for a user-configurable graphic and text. The template in *Visual Studio* is shown below:
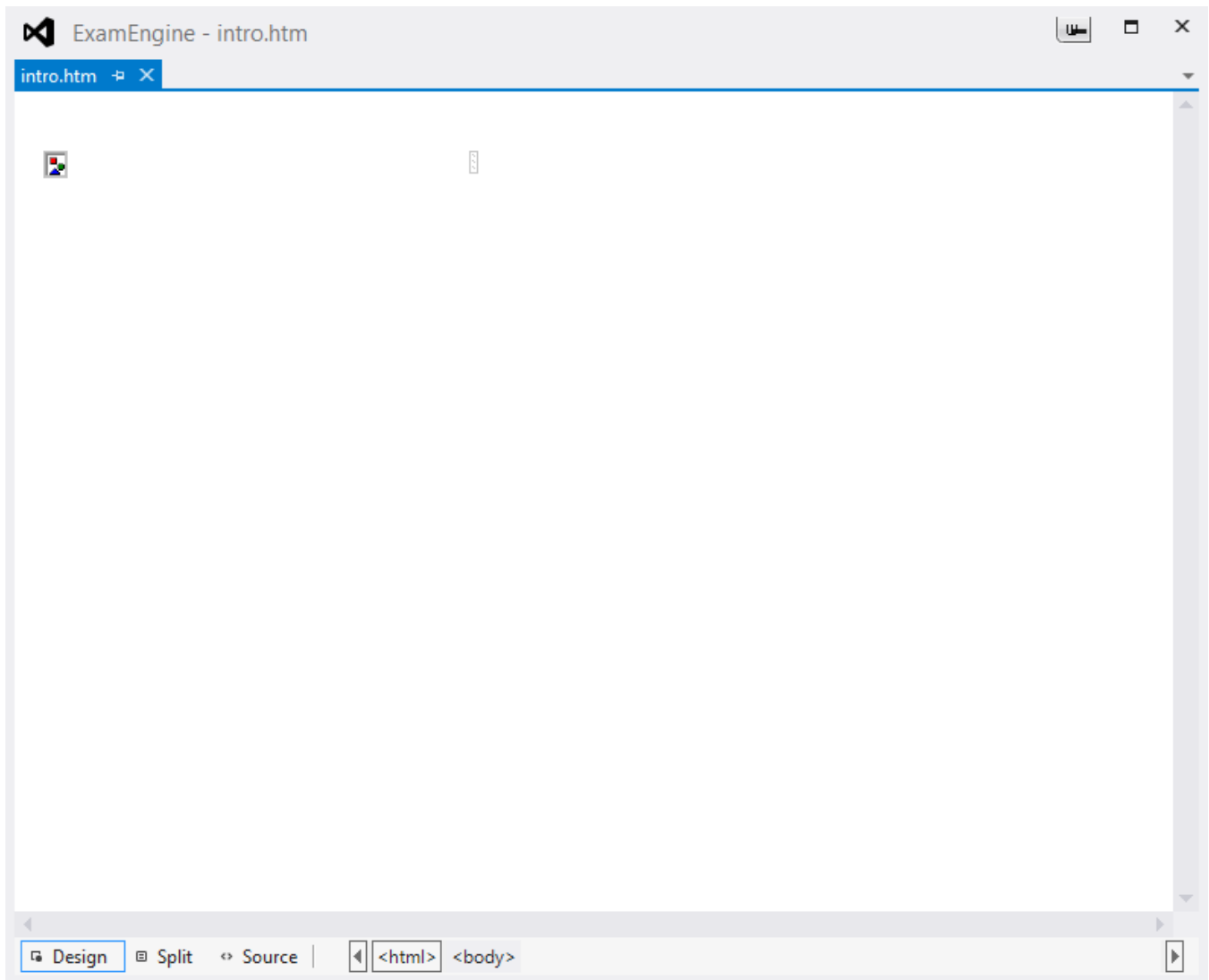
The layout is basic with a single *Image* control and a single *Text*. Here is an example of how this template looks at runtime.
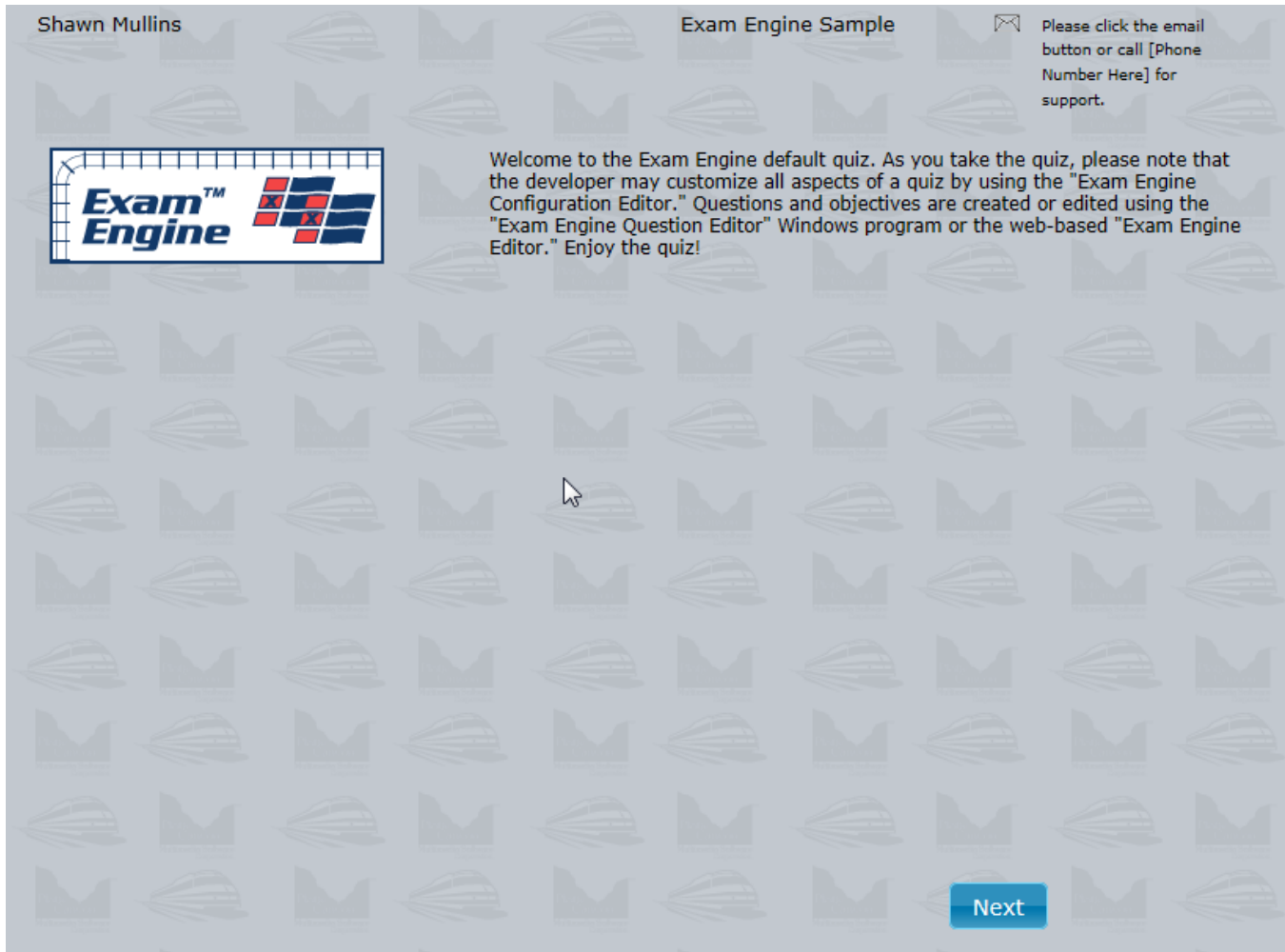
login

This is not a question template but rather the page that implements the optional login page. If enabled, this page is displayed if the exam is not run from a Learning Management System. The template has space for a login message, a single *input*, a graphical or *jQuery UI* login button (controlled by the *loginImageBtn* or *loginBtn* classes in templatestyles.css), and a status *Text* for messages like "Logging in is required." The template in *Visual Studio* is shown below:
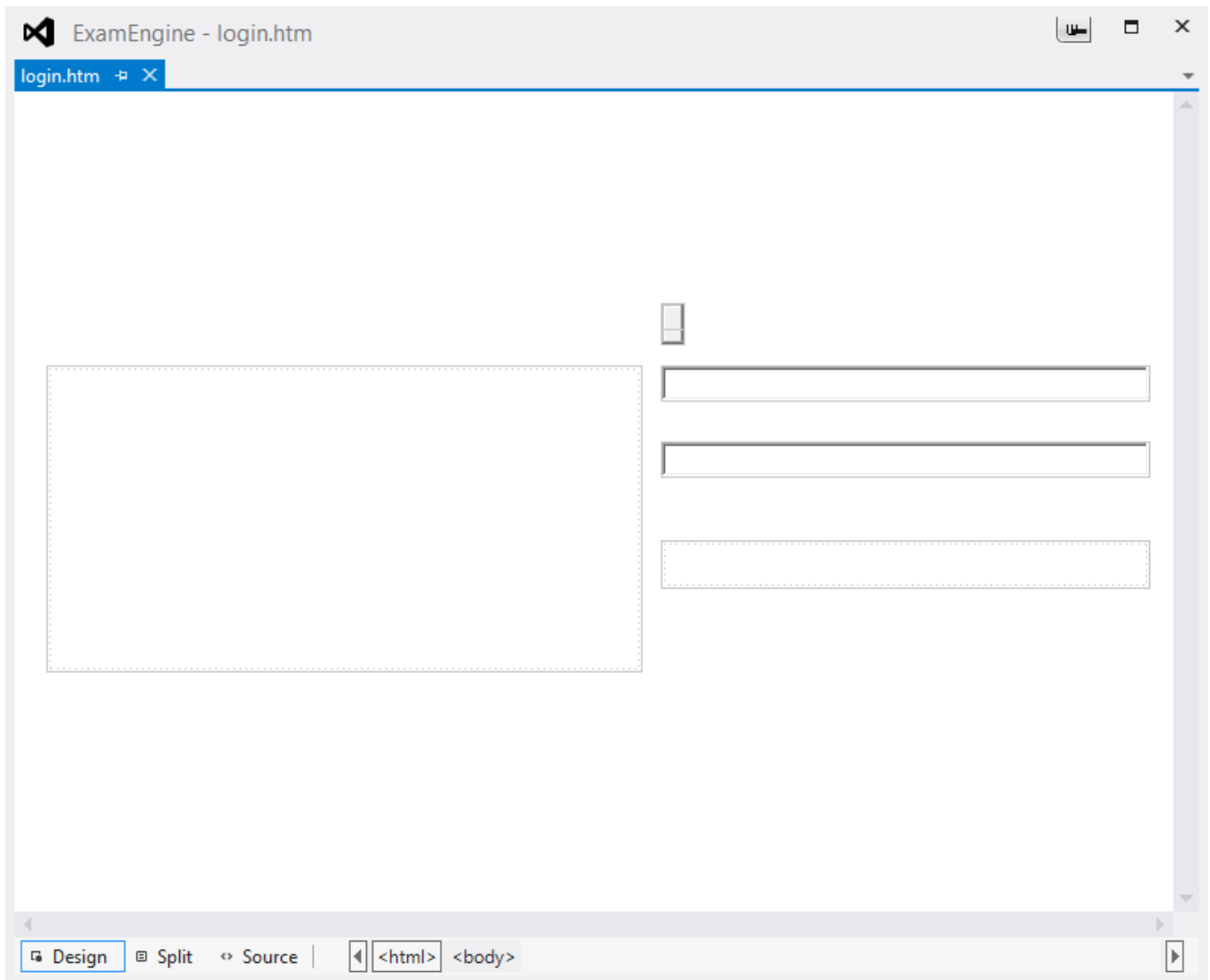
Here is an example of how this template looks at runtime.
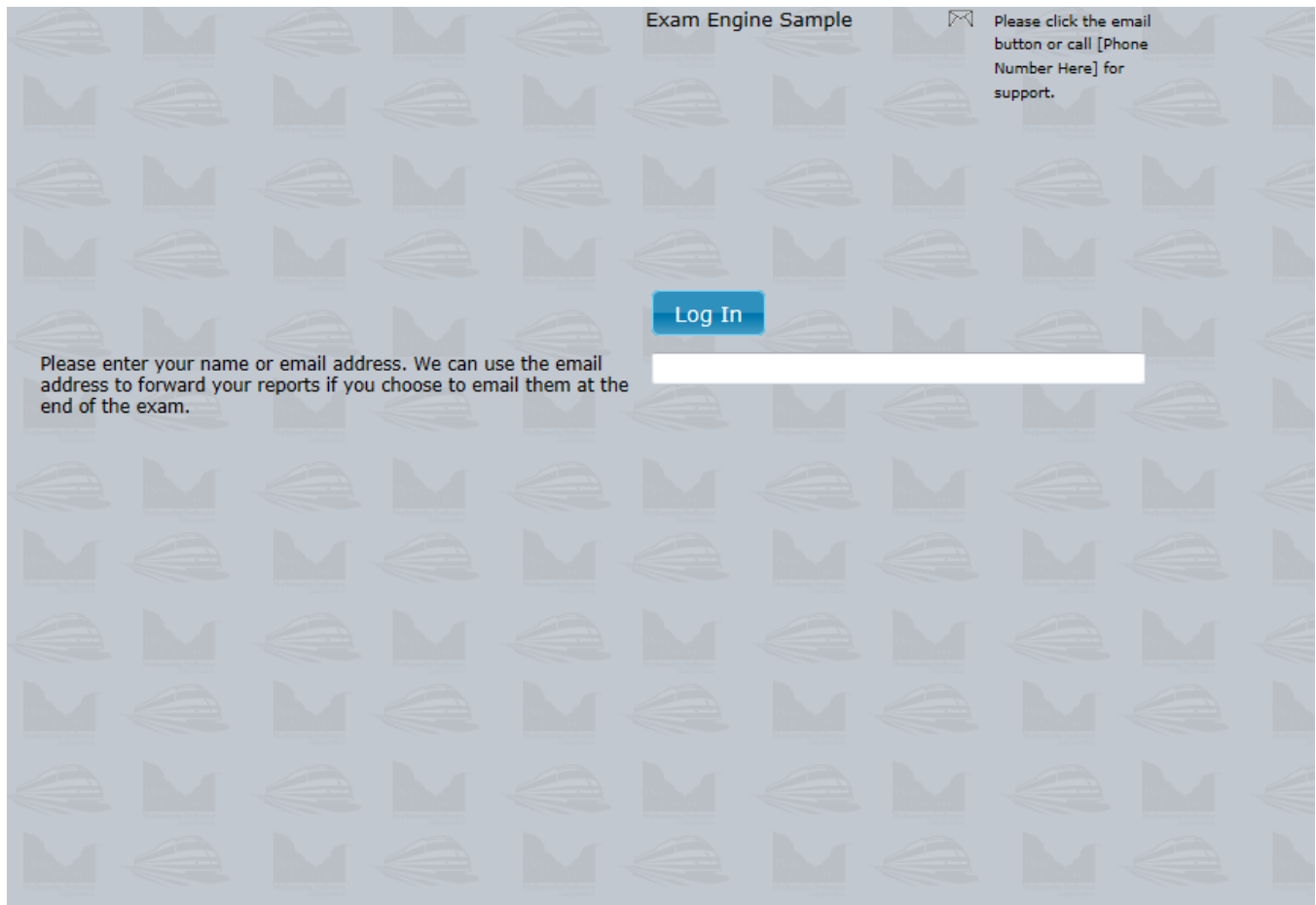
If the exam is configured to validate logins and passwords, there will be an additional field for the password as shown below:
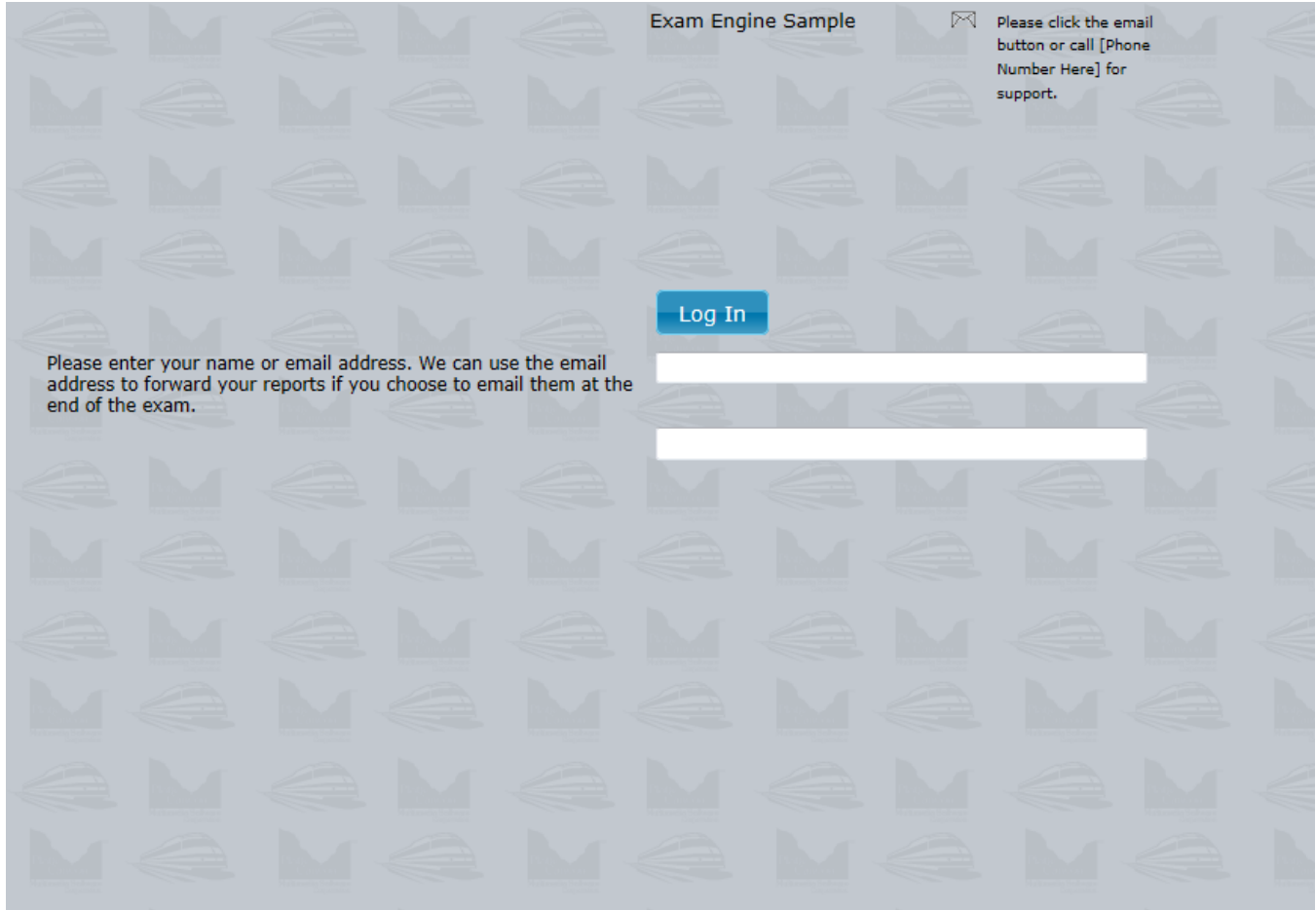
matching

This question template is a "matching" question where both the answers and "targets" are images. It is distinguished from the *matching_lines* template in that the *matching* template involves the user dragging the answers onto the targets whereas the *matching_lines* template involves the user drawing lines between the answer text and the target text. This template uses *jQuery UI* for "drag and drop" functionality. That is why this JavaScript library is linked:

```
<script src="../jquery/jquery-ui-1.10.1.custom.min.js" type="text/javascript"></script>
```

The styles associated with "drag and drop" are in *../css/start/jquery-ui-1.10.1.custom.css* along with the other *jQueryUI* styles.
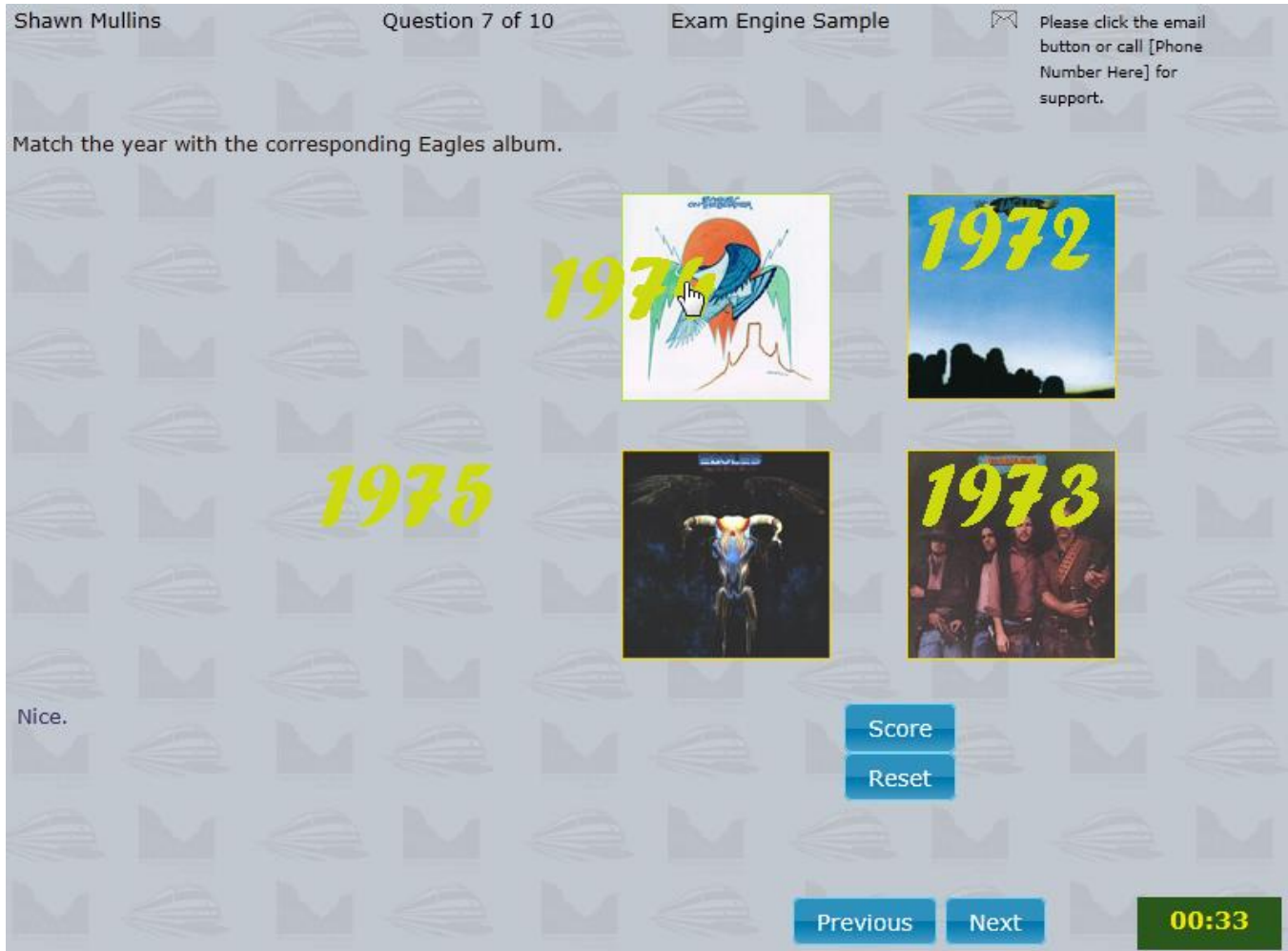
The template in *Visual Studio* is shown below:
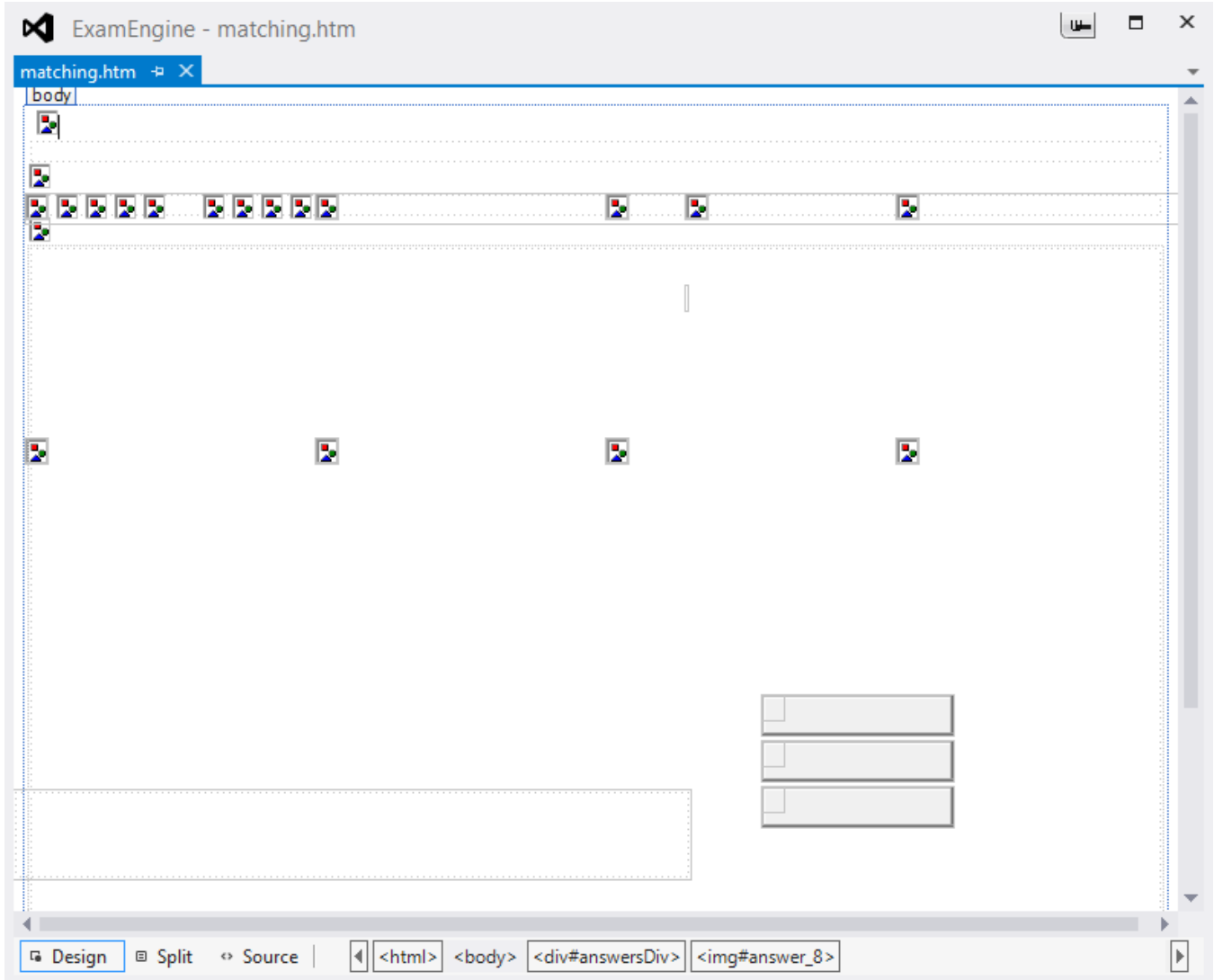
The layout is similar to the *hotObjects* template except that there is a *div* that covers the entire middle of the screen. Both the answer and target images are inside the div so that the drag and drop can be implemented. The *media* controls are again moved to the bottom part of the screen. Here is an example of how this question looks at runtime.

matching_lines

This question template is a "matching" question where both the answers and "targets" are text. It is distinguished from the *matching* template in that the *matching* template involves the user dragging the answers onto the targets whereas the *matching_lines* template involves the user drawing lines between the answer text and the target text. This template uses the *Raphael.js* open source library to implement the lines. You can learn more about it at http://raphaeljs.com. That is the reason for this JavaScript library:

```
<script src="../raphael/raphael.js" type="text/javascript"></script>
```

The template in *Visual Studio* is shown below:
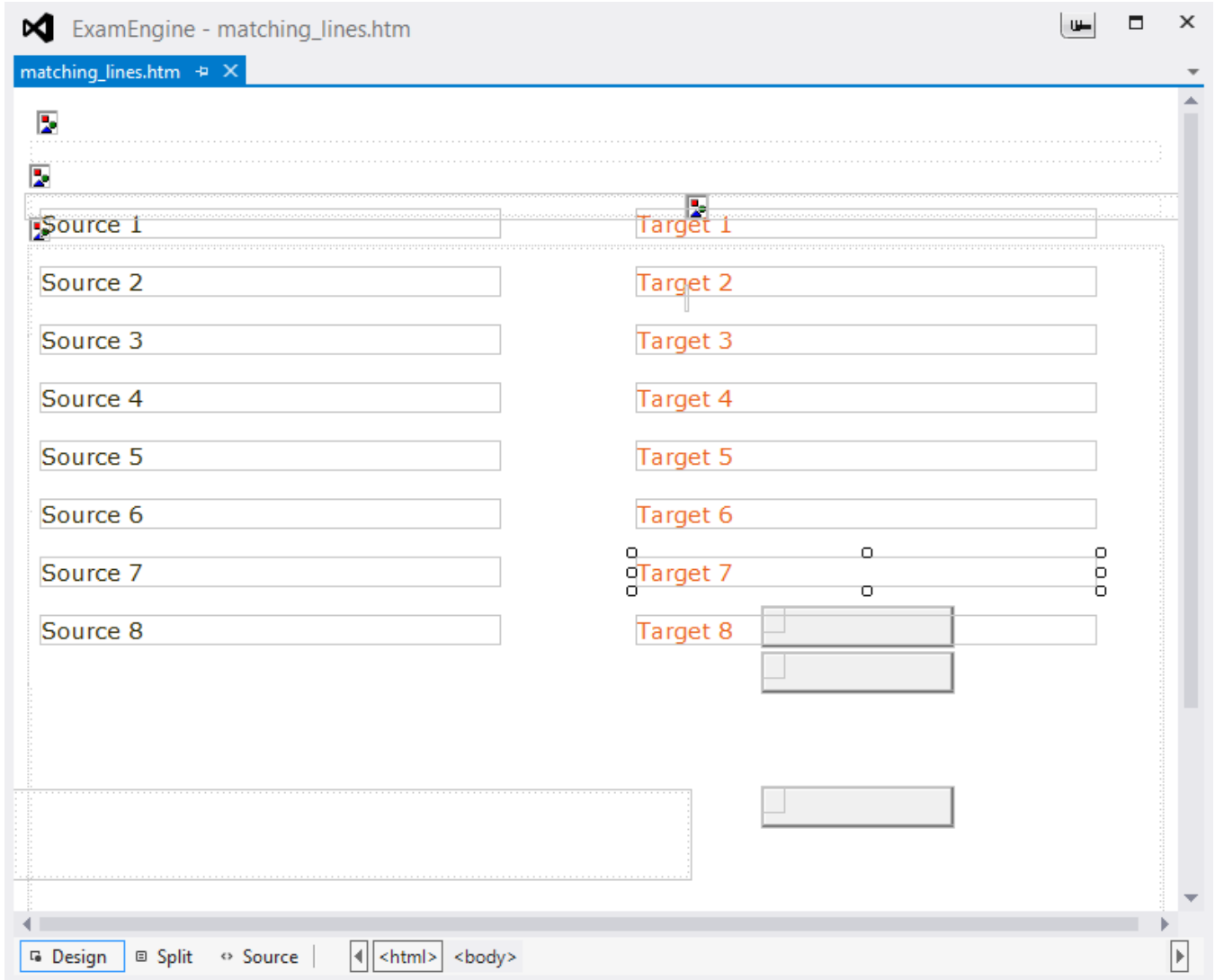
The layout is identical to the *matching* template except that there are *Text* components for the answers and targets rather than images. Note that the color of the targets is controlled by the *TargetsInitialTextColor* class in *templatestyles.css*. Both the answer and target blocks are inside the div so that the drawing can be implemented. Here is an example of how this question looks at runtime.
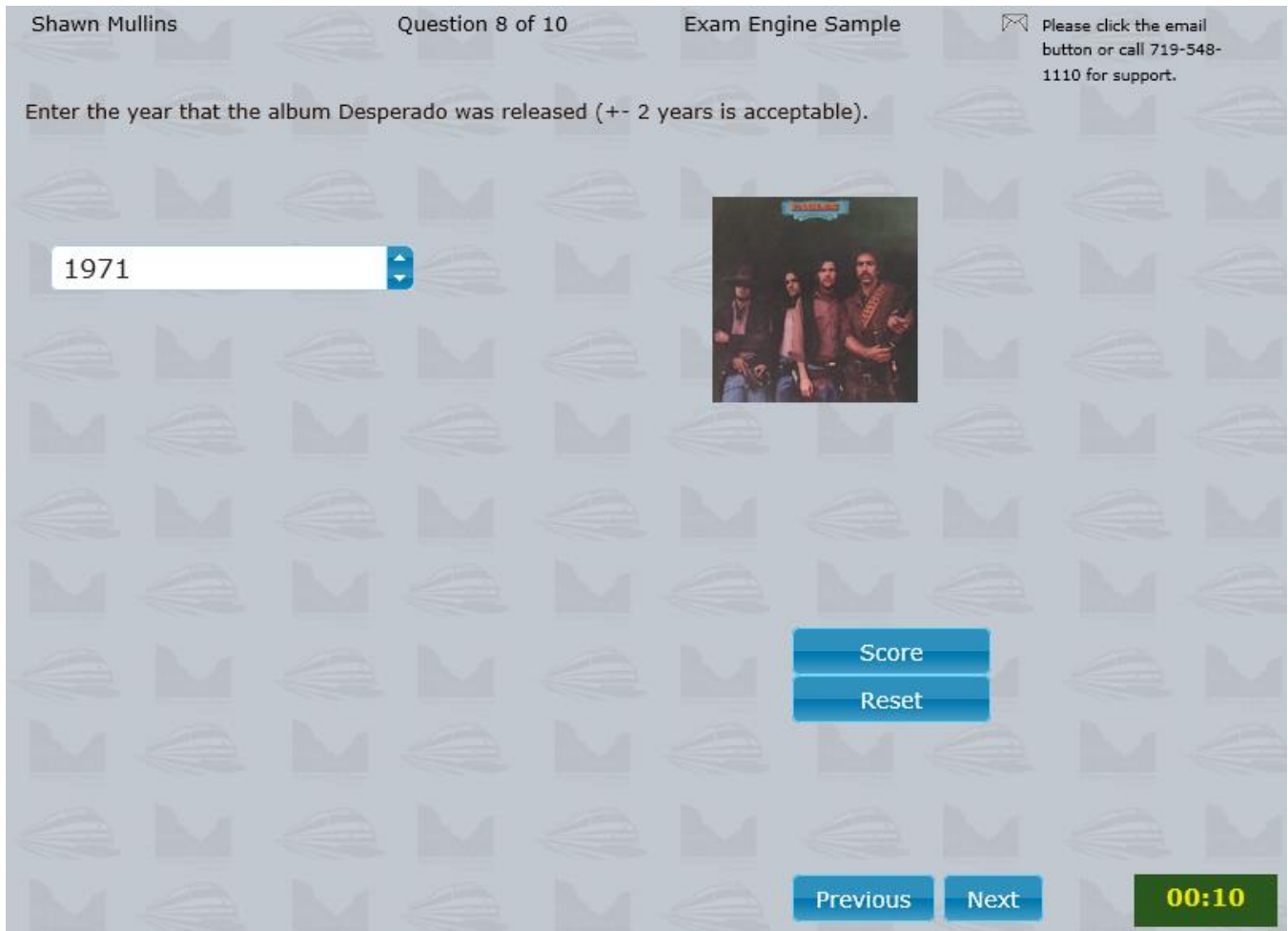
Shawn Mullins   Question 8 of 10   Exam Engine Sample   ✉   Please click the email
                                                                                                    button or call [Phone
   Number Here] for
                                                                                                    support.

Match the Beatles member with a later rock band of which he was a member.

John Lennon   Traveling Wilbury's

George Harrison   Wings

Paul McCartney   All-Star Band

Ringo Starr   Plastic Ono Band

Good. The "Ono" refers to Yoko Ono.

Score

Reset

Previous   Next   00:29

numeric

This question template implements a numeric question. It uses the *jQueryUI spinner* control.

The template in *Visual Studio* is shown below:
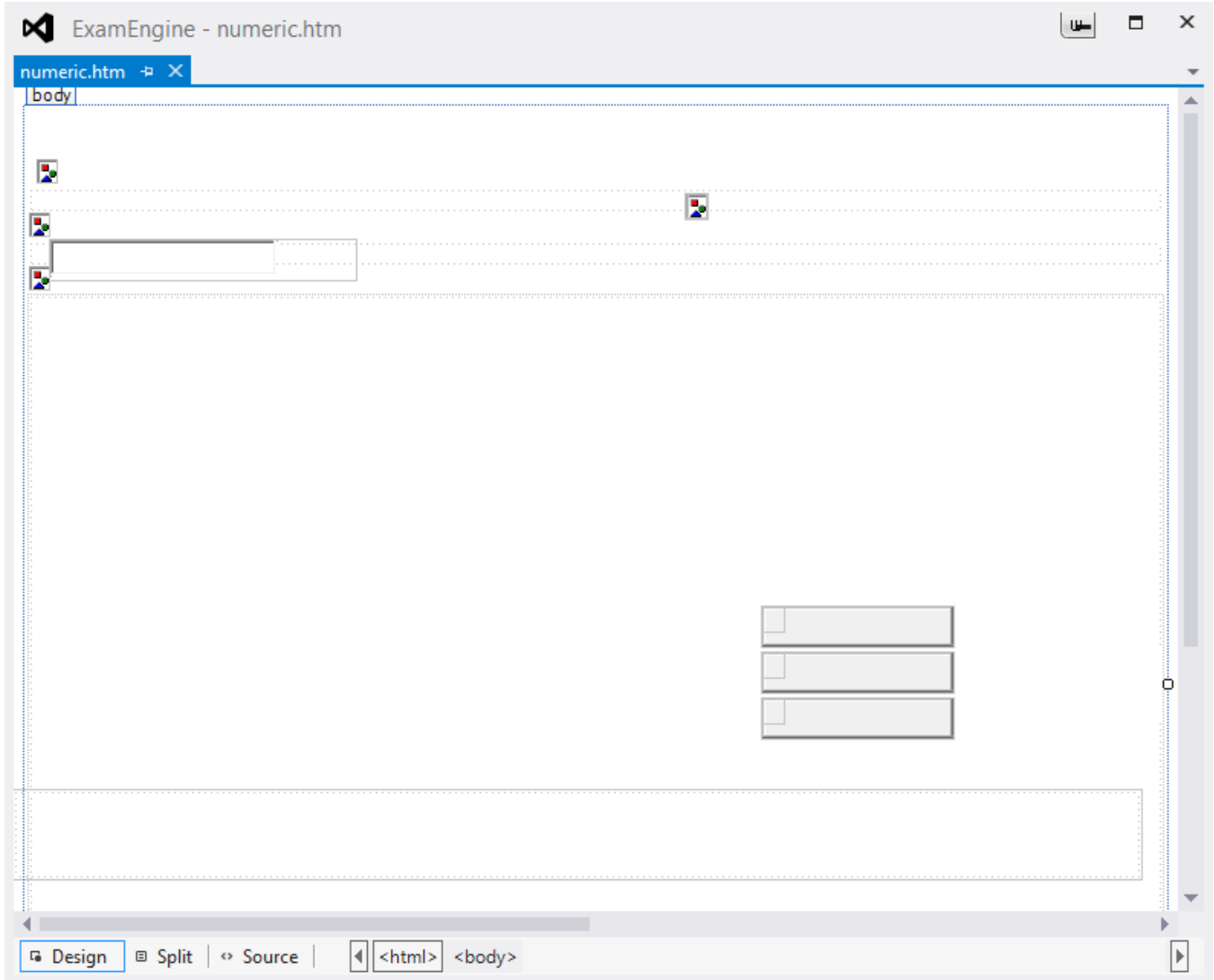
The layout is similar to the *fill_in* template. The question is set up with an initial value as well as a min and max value for when the user clicks the up or down buttons. The user can click the *Score* button to score the question. The value turns the color of the *AnswersCorrectTextColor* or *AnswersIncorrectTextColor* class, depending on whether the answer is correct. Here is an example of how this question looks at runtime.

scoreexam

This is not a question template but rather the page that is displayed at the end of the exam. The grid showing the exam results can be hidden by the exam developer if desired. Note that the "View Question" link (if selected in the exam) will show a locked version of the question as it was answered at the bottom of the screen. The template in *Visual Studio* is shown below:
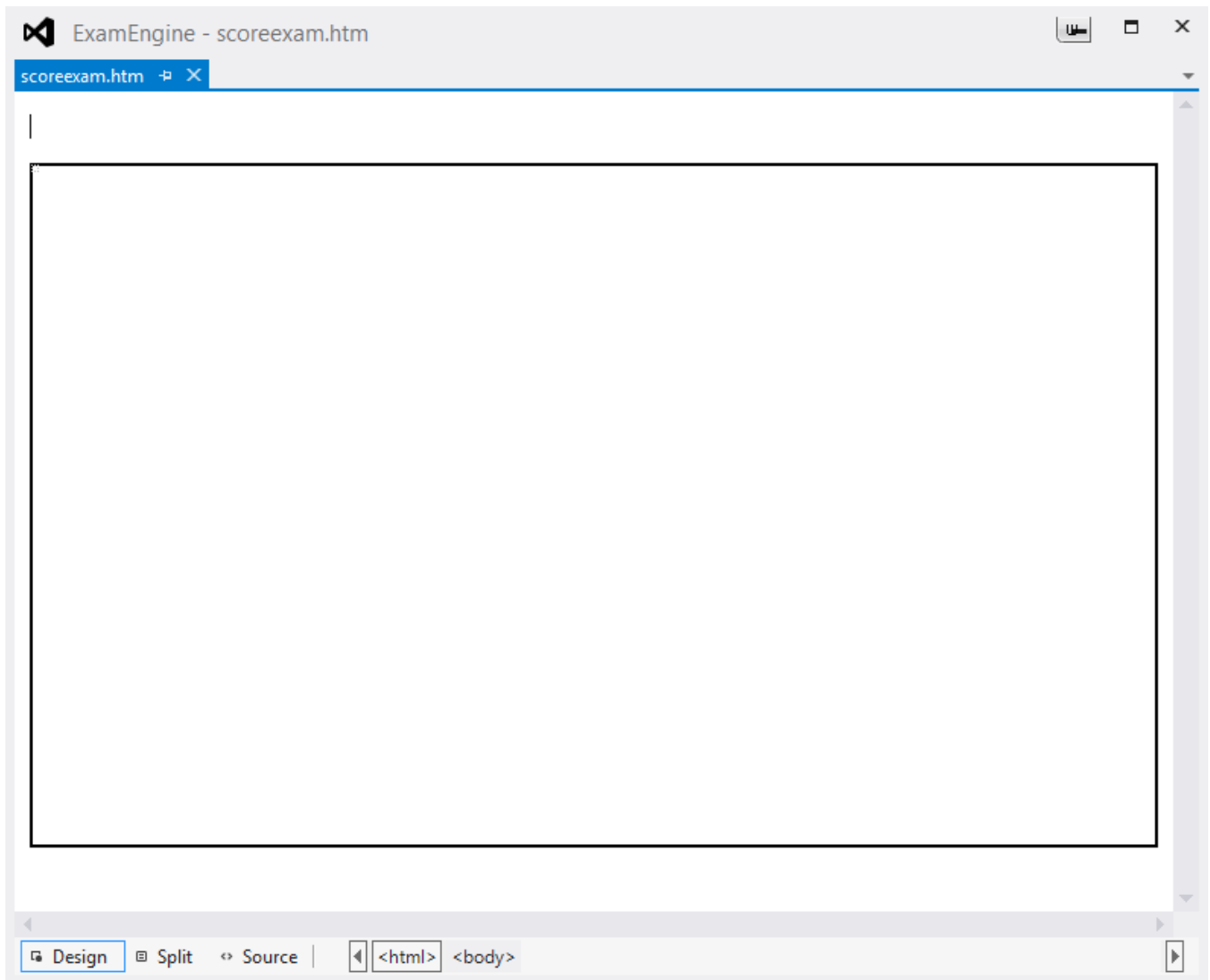
Here is an example of how this template looks at runtime.

Shawn Mullins

Exam Engine Sample

✉ Please click the email button or call [Phone Number Here] for support.

Exam Results

Question Text	Score	Min Score	Max Score	Selected Answer(s)	Correct Answer(s)	Objective(s) Covered	Time Spent (seconds)	Number of Tries	View Question
Put the groups in chronological order of the date of their first album with the earliest at the top.	1	0	1	Beatles->1 | Led Zeppelin->2 | Eagles->3 | Van Halen->4	Beatles->1 | Led Zeppelin->2 | Eagles->3 | Van Halen->4	Beatles_Info | Eagles_Info | Led_Zeppelin_Info | Van_Halen_Info	11	1	Review
Which album had the hit "Dreams" on it?	1	0	1	5150.png	5150.png	Van_Halen_Info | American_Rock_and_Roll	5	1	Review
Select the album with the popular song "Jump."	1	0	1	1984.png	1984.png	Van_Halen_Info | American_Rock_and_Roll	1	1	Review
Match the Beatles member with a later rock band of which he was a member.	0.5	0	1	George Harrison->Traveling Wilbury's | Ringo Starr->All-Star Band	John Lennon->Plastic Ono Band | Paul McCartney->Wings | George Harrison->Traveling Wilbury's | Ringo Starr->All-Star Band	Beatles_Info | English_Rock_and_Roll	9	1	Review
What was Robert Plant's age when Led Zeppelin I was released (+- 1 year is acceptable)?	1	0	1	21	21	Led_Zeppelin_Info | English_Rock_and_Roll	4	1	Review
				John Lennon->Rhythm Guitar | Paul	John Lennon->Rhythm Guitar | Paul				

Normalized test score = 95%. Raw test score = 9.5 out of a maximum score of 10. The minimum possible score was 0. The score by objective was Beatles_Info: 83%. Eagles_Info: 100%. Led_Zeppelin_Info: 100%. Van_Halen_Info: 100%. American_Rock_and_Roll: 100%. English_Rock_and_Roll: 90%. .

Email Results   Exit

sequencing

This question template is a "sequencing" question where the user drags the buttons into the correct order vertically. It is distinguished from the *sequencing_images* template in that the *sequencing* template uses buttons with text and the *sequencing_images* uses images that have to be dragged into the right order. This template uses *jQueryUI* for "drag and drop" functionality. The template in *Visual Studio* is shown below:
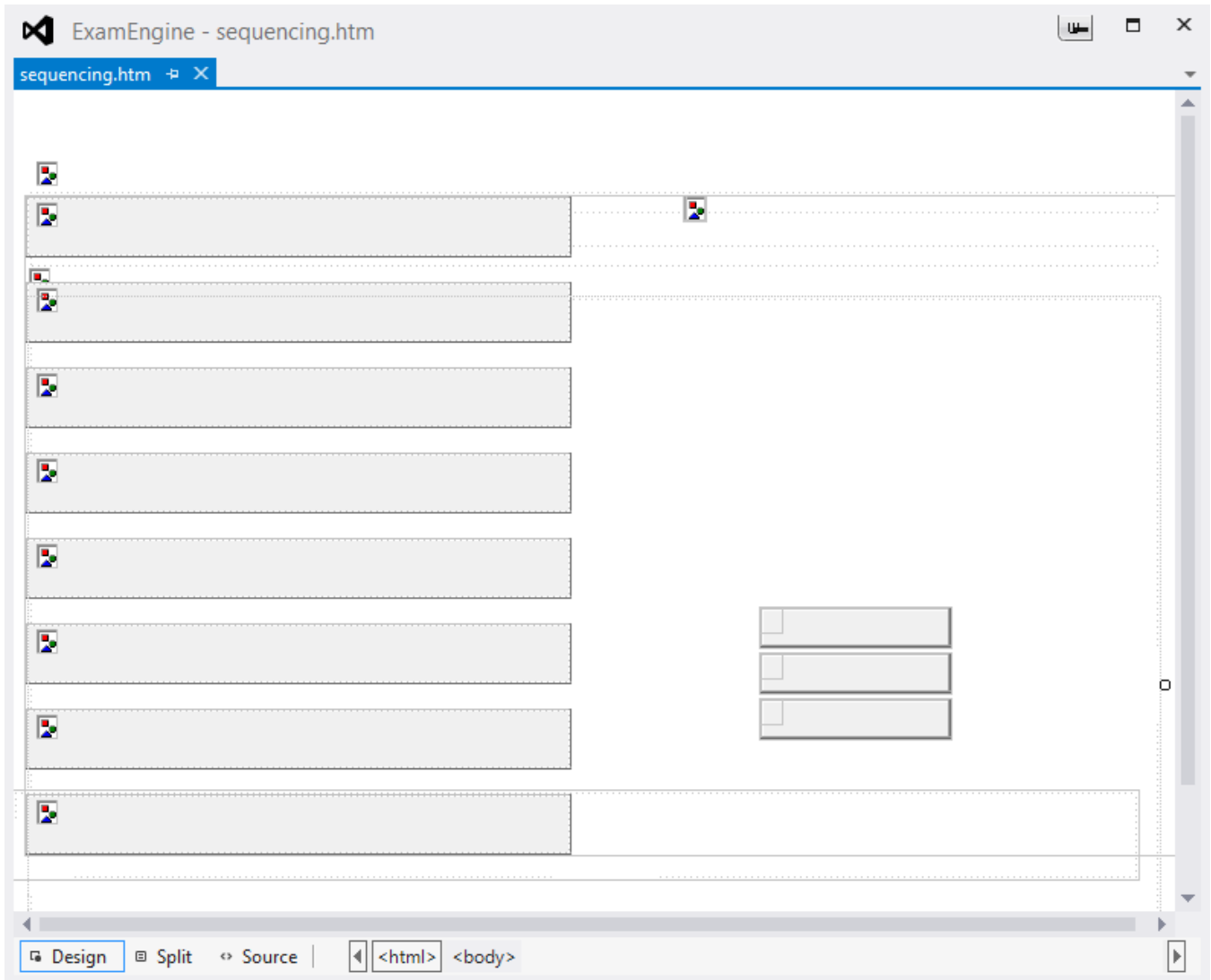
The layout is similar to the *choice_buttons* template. The buttons turn the style of *AnswersCorrectTextColor* or *AnswersIncorrectTextColor* once the question is scored, depending on whether the answer is in the correct position. Here is an example of how this question looks at runtime.
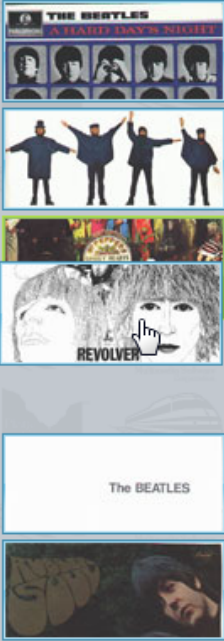
sequencing_images

This question template is a "sequencing" question where the user drags images into the correct order vertically. It is distinguished from the *sequencing* template in that the *sequencing* template uses buttons with text and the *sequencing_images* uses images that have to be dragged into the right order. This template uses *jQueryUI* for "drag and drop" functionality. The template in *Visual Studio* is shown below:
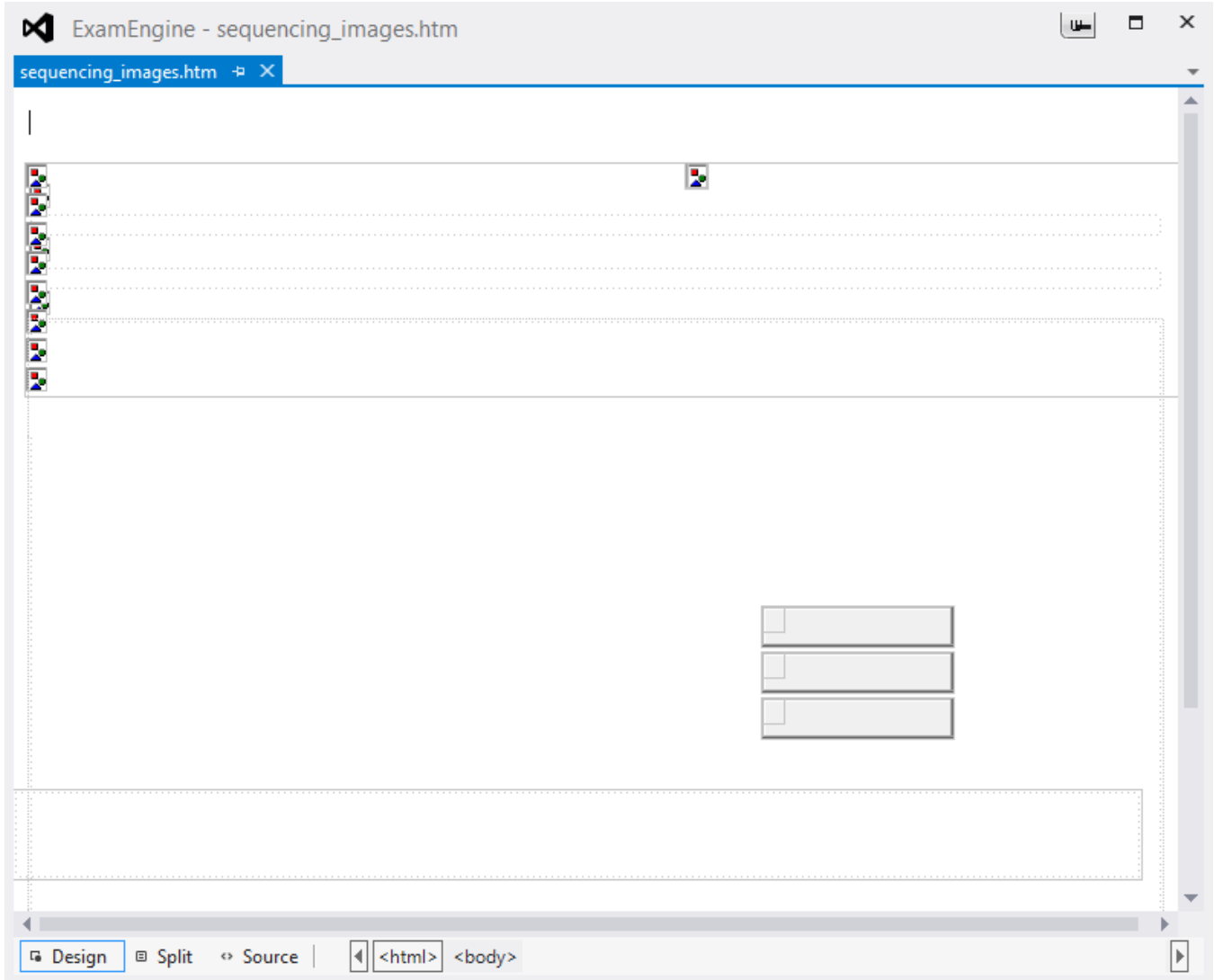
The layout is similar to the *sequencing* template. Notice that the *Question Text* and *Instructions/Feedback* have been down to give more vertical area for the images. The images get a style of *correctImage* or *incorrectImage* once the question is scored, depending on whether it is in the correct position. Here is an example of how this question looks at runtime.

Shawn Mullins   Question 1 of 10   Exam Engine Sample   Please click the email button or call [Phone Number Here] for support.

Put the albums below in chronological order with the earliest release at the top.

Score

Reset

Drag the items into the correct order.

Previous   Next   00:27

true_false

This question template implements a true-false question with the choices being radio buttons. The template in *Visual Studio* is shown below:
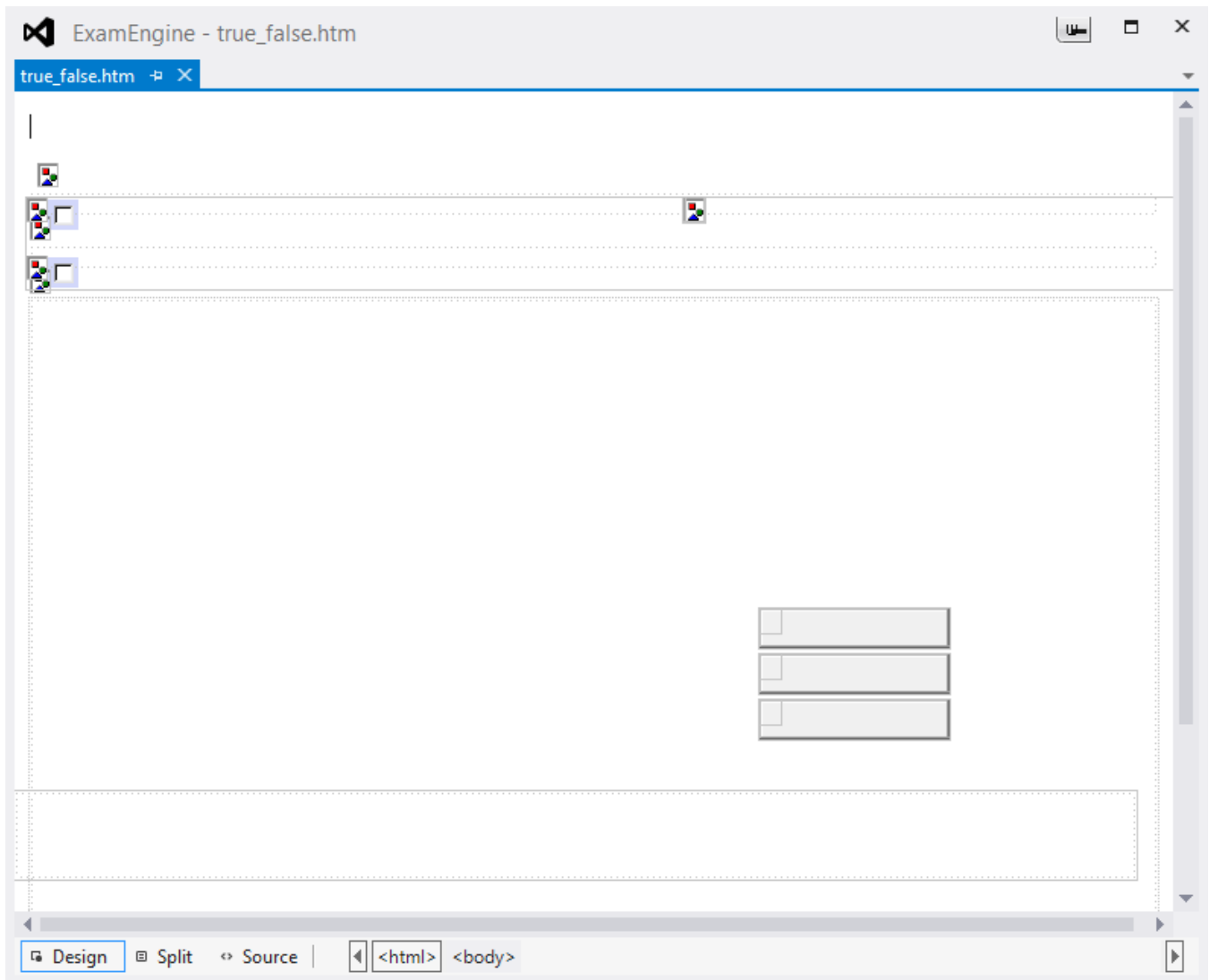
The layout is similar to the *choice* template except that there are only two answers. Here is an example of how this question looks at runtime.

Template Layout & Media

As discussed earlier in this document, templates are shown in the area defined by the *transition iFrame* of the *index.htm* file. Each template corresponds to a *template* selected by the user of the *Exam Engine Question Editor* application and stored in the database. When a student goes to a new question, the exam determines the template name and loads that template file, which is then responsible for loading its content, images, and media. Templates are typically independent of content and user interface as shown in the various question template screen captures above. For graphics and media, Exam Engine has logic to detect the type of media being played/displayed. It checks the browser and if the browser can play the media file natively, it loads those tags into the designated *div* in the template. If the *Add Multiple Sources for Media* box on the *Questions & Media* tab of the *Exam Engine Configuration Editor – General Properties* screen is checked, Exam Engine will load the various formats for the different browsers as well as a traditional Windows Media Player, Flash Player, or QuickTime player for older browsers. For example, the screen capture below shows the question in Internet Explorer 10. Notice the standard video controls given by the browser.

Here is how the XML database looks for that question:

```xml
<Questions>
    <QuestionId>1</QuestionId>
    <templateId>5</templateId>
    <InteractionId>Beatles_Bass_Player</InteractionId>
    <QuestionText>Who was the bass player for the Beatles?</QuestionText>
    <QuestionMedia>beatles1.mp4</QuestionMedia>
```

Here is how the actual HTML is rendered to the browser:
<div id="questionMedia">
  <video autoplay="autoplay" controls="controls">
    <source type="video/mp4; codecs='avc1.42E01E, mp4a.40.2'" src="../media/beatles1.mp4" ,="">
    <source type="application/ogg" src="../media/beatles1.ogv" ,="">
    <source type="video/webm; codecs='vp8, vorbis'" src="../media/beatles1.webm" ,="">
    <source type="application/ogg" src="../media/beatles1.ogg" ,="">
    <object id="WMPPlayer"
  codeBase="http://activex.microsoft.com/activex/controls/mplayer/en/nsmp2inf.cab#version=6,0,0,0"

```
        classid="clsid:22D6F312-B0F6-11D0-94AB-0080C74C7E95" width="100%" align="left"
    type="application/x-oleobject" height="100%">
        <param name="Filename" value="../media/beatles1.wmv">
        <param name="AutoStart" value="True">
        <param name="Volume" value="-500">
        <param name="StretchToFit" value="True">
        <param name="showControls" value="True">
    </object>
  </video>
</div>
```

The other file formats are for browsers like Firefox that don't accept the mp4 video type or older browsers that need the Windows Media Player to play the video. Note that you must have the other media files (beatles1.ogv, beatles1.webm, beatles1.ogg, and beatles1.wmv) to use this feature. Here is a screen capture of the same page in Internet Explorer 8. Notice how it uses the Windows Media Player instead.

All templates have *img* and *div* tags corresponding to the "Initial Graphic 1" or "Initial Media 1" values in a prominent place. The remaining *img* and *div* graphic and media tags are not as prominent and are

typically in the bottom, right grid location to preserve screen real estate. So if you need to play two videos at once, for example, you will need to edit your template(s). Note that the developer of the exam can pick which one of the media elements to use for playing question feedback.

Another issue related to media is the showing of audio controls. When Exam Engine detects an older (pre-HTML 5) browser, it will play audio with no controls. However, if it detects HTML 5 media capabilities, it will look to the *Show Audio Controls (HTML 5 Audio)* box on the *Questions & Media* tab of the *Exam Engine Configuration Editor – General Properties* screen. If this is checked, then the default browser audio controls will be shown. This can be helpful for the user. The audio controls are shown below the navigation buttons, but you can adjust this with the *#audioPlayer* style and media queries in *backgroundstyles.css* as discussed earlier in this document. Note that the real estate taken up by the various browsers vary. Here are some screen shots.

Internet Explorer 10

Exam Engine Templates Documentation

Firefox 19

Exam Engine Templates Documentation

Chrome 25