HAAGA-HELIA
University of Applied Sciences

# Visually altering user controls in ASP.NET with Photoshop and CSS

Marius Cojoc

Thesis

Business Information Technology

August 2013

HAAGA-HELIA
University of Applied Sciences

**Abstract**

Busines IT

| **Author or authors**<br>Cojoc Marius | **Group or year of entry**<br>BIT 2009 |
|---|---|
| **Title of report**<br>Visually altering user controls in ASP.NET with Photoshop and CSS | **Number of report pages and attachment pages**<br>29 + 12 |
| **Teacher(s) or supervisor(s)**<br>Välimäki Juhani | |


The need for this thesis project arose after the completion of several courses on building ASP.NET webpages and applications: there seemed to be a lack of information regarding the ways how to style webpages and to give them a different look from the default. During the courses the focus was mainly on programming aspects, be them frontend, backend or XHTML, but little was mentioned about the possibilities to modify the content visually.

The purpose of this thesis project was to present an easy way to alter the default visual style of the ASP.NET user controls, such as like buttons, check boxes and input fields, using CSS. An additional objective was to produce a short tutorial on how to create a webpage from a Photoshop designed layout and develop it further with Visual Studio.

The empirical part of the thesis contained a short presentation on how to implement some changes to the user controls and how to create a working ASP.NET webpage from a given Photoshop created document.

The result of this thesis project was an ASP.NET webpage that was created from a design layout from a Photoshop document. CSS styling was applied so that the resulting webpage matches the design.



**Keywords**
Photoshop, CSS, CSS3, ASP.NET, Visual Studio, Microsoft, XHTML

# Table of contents


Glossary of terms ..... 1

1 Introduction ..... 2

1.1 Project environment ..... 2

1.2 Deliverables ..... 3

1.3 Scope ..... 3

1.4 Out of scope ..... 3

2 Research plan ..... 4

3 ASP.NET ..... 5

3.1 How does ASP.NET work ..... 5

3.2 ASP.NET MVC 3 and 4 ..... 6

3.3 New ASP.NET 4.5 features ..... 7

4 CSS ..... 8

4.1 History of CSS ..... 8

4.2 Versions ..... 9

4.3 New CSS3 features ..... 10

5 HTML 5 ..... 12

5.1 New features ..... 12

6 Photoshop ..... 14

6.1 Photoshop general interface ..... 14

7 Webpage from Photoshop PSD layout ..... 16

7.1 Photoshop slices ..... 16

7.2 Creating an HTML document from a sliced layout ..... 17

8 Styling inputs with CSS ..... 21

9 Styling radio buttons and check boxes with CSS ..... 24

10 Evaluation ..... 27

11 Further research on this topic ..... 28

12 Conclusion ..... 29

Bibliography ..... 30

Table of figures ..... 33

Attachments ..... 34

Attachment 1. Sample code for the generated HTML file from Photoshop ..... 34
Attachment 2. ASPX webpage created in Visual Studio ..... 35
Attachment 3. The second webpage with the modified inputs ..... 36
Attachment 4. CSS file used to style the second webpage ..... 37
Attachment 5. HTML code from the browser ..... 39

## Glossary of terms

| | |
|---|---|
| **Adobe Photoshop** | also referred as Photoshop, is a graphics editing software developed by Adobe Systems (Wikipedia, http://en.wikipedia.org/wiki/Adobe_Photoshop, retrieved 21.1.2012) |
| **CSS** | Cascading Style Sheet. Programing language used for defining a web-site's appearance and style |
| **HTML** | HyperText Markup Language. |
| **ASP.NET** | server-side Web application framework designed for Web development to produce dynamic Web pages. It was developed by Microsoft |
| **PSD** | PhotoShop Document. The default file type created when saving a document with Photoshop |
| **MVC** | Model View Controller |

# 1 Introduction

Websites have become part of our daily lives. From checking the weather to keeping in touch with friends and family to finding the latest news to watching videos, we use our computers, tablets or phones to access some form of webpage to get the needed information. This leads to a need for tools to present everything in the best way possible to the user, to make easier for him to read or watch online content. Such technologies exist and are evolving. HTML5 and CSS3 are the latest hot subject on the Internet and in the web development community. Web frameworks and content management systems get invested in, created and updated to give everyone a possibility to create content for the web.

The need for this project paper arose after the completion of several courses on building ASP.NET webpages and applications, where I felt there is a lack of information regarding ways to style the webpages, to give them a different look than the default. During the courses the focus was mainly on programming aspects, be them frontend, backend or XHTML, but little has been mentioned about possibilities for modifying content visually. Some mention was made of using CSS to style a simple HTML website during a first semester course. Also the research will help me understand better the way that CSS and ASP.NET work in combination.

This paper will benefit any student that is creating any ASP.NET webpages or applications during their study in HAAGA-HELIA and help them improve their webpage's visual style. The outcome can be referenced in future courses as extra reading.

## 1.1 Project environment

The tools used in this project consist of Adobe Photoshop, Visual Studio 2010, Microsoft Word and Microsoft Windows, for the software part. For the hardware part, the researcher's personal computer and HAAGA-HELIA computer labs were used. Same software was present on all computers during the project. Additionally Internet websites and books presented in the references section were consulted.

## 1.2 Deliverables

The deliverable of the project is a thesis paper, that will provide information (a tutorial) on how to modify the visual styles of an ASP.NET webpage (or web applications) using CSS and Photoshop. The paper will show how to create custom elements (whole layout of a page, backgrounds, custom radio buttons, drop down lists, etc.) and how to integrate them in the users' webpage.

Other objectives of the project are to present the reader with information about other techniques for changing a website's appearance than the possibilities presented during HAAGA-HELIA courses.

## 1.3 Scope

The scope of the project is to present an easy way to alter the default visual style of the ASP.NET user controls (like buttons, check boxes, etc.) and to present a short tutorial on how to create a website template in Photoshop and develop it further with Visual Studio. This will present the tools needed for doing so.

## 1.4 Out of scope

Out of scope of this project are explanations on how to the tools used are installed and utilized by the user. The way the website template was created within Adobe Photoshop is not presented either.

# 2 Research plan

The methods of study for this project are a combination of research and implementation methods. The theory part contains descriptions of the tools used in this project paper, and some background for them. Research is done from sources on the Internet and books to find out methods to modify the ASP.NET controls. The sources used are referenced in accordance to guidelines for further study by the reader.

The empirical part of the project paper contains a short presentation on how to implement some changes to the user controls and how to create a working ASP.NET webpage from a given Photoshop created document. A description for the method will be available along with the code and visual references (screen captures). The simpler methods (pure CSS methods) will be presented more in depth, with more intricate methods just presented as other possibilities.

# 3 ASP.NET

As described on Wikipedia, ASP.NET is a server-side Web application framework designed for Web development to produce dynamic Web pages. It was developed by Microsoft to allow programmers to build dynamic web sites, web applications and web services. (Wikipedia, 2013f)

## 3.1 How does ASP.NET work

ASP.NET is a part of Microsoft .NET Framework, so it has all the advantages of the framework's object oriented programming capabilities. Web pages created with ASP.NET are basically objects that have behaviors (the events within the page), commands (the methods in the code), and state (instantiated objects). (Bochicchio, Mostarda, & De Sanctis, 2011, p. 5)

Web Forms are the most common way to develop with ASP.NET. With Web Forms any item on a page is programmable and has events. (Bochicchio, Mostarda, & De Sanctis, 2011, p. 5)

More to the point, Bochicchio, Mostrada and De Sanctis (2011, 5) say that to use the Web Forms model, all the user has to do is place for example a *Button* object on the page, and intercept the *Click* event. Even more simply described, place an object and program its function.

The tree main actions of the web form model are: page post backs, view states and server controls. They interact with each other according to a model presented in **Figure 1**. Each HTTP request to the Web server that is mapped to the ASP.NET runtime, passes through some number of stages cantered on the processing of the post back event. The post back event is the main action that the user expects out of his request. (Esposito, 2011, p. 5)

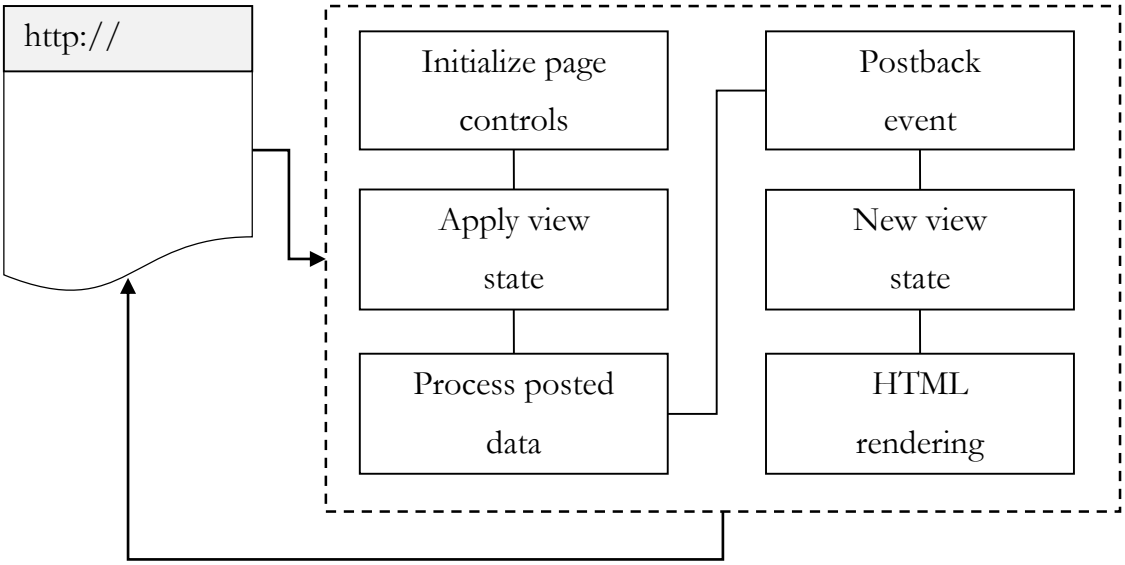


**Figure 1 - Webforms model (Esposito, 2011, p. 5)**

### 3.2 ASP.NET MVC 3 and 4

Wikipedia defines MVC as "a software architecture pattern which separates the representation of information from the user's interaction with it. The model consists of application data, business rules, logic, and functions. A view can be any output representation of data, such as a chart or a diagram. Multiple views of the same data are possible, such as a bar chart for management and a tabular view for accountants. The controller mediates input, converting it to commands for the model or view. The central ideas behind MVC are code reusability and separation of concerns." (Wikipedia, 2013c)

Microsoft has its own implementation of the MVC model available in the ASP.NET framework. MVC 3 and 4 are currently used in web applications developed with ASP.NET. The latest stable release version of MVC is MVC 4. The view engines used by the two versions are Razor engine and Web forms. The default engine used is Razor, but other engines can be used as well. (Wikipedia, 2013b)

### 3.3 New ASP.NET 4.5 features

.NET Framework 4.5 was released on 15 August 2012. A set of new or improved features were added into this version (Wikipedia, 2013g). Some of them (related to this project) are:

- Support for new HTML5 form types
- support for new model binders on Web Forms. Data controls can be bound directly to data-access methods and automatically convert user input to and from .NET Framework data types
- Support for unobtrusive JavaScript in client-side validation scripts.
- Improved handling of client script through bundling and minification for improved page performance.
- Integrated encoding routines from the AntiXSS library (previously an external library) to protect from cross-site scripting attacks.
- Support for WebSockets protocol.
- Support for reading and writing HTTP requests and responses asynchronously.
- Support for asynchronous modules and handlers.
- Support for content distribution network (CDN) fallback in the ScriptManager control. (microsoft, 2013)

Another new and important feature is using .NET for building Metro style apps. A subset of the .NET Framework is available for building Metro style apps for Windows 8 using C# or Visual Basic. This subset is called .NET APIs for Metro style apps. (Wikipedia, 2013g)

# 4 CSS

CSS stands for Cascading Style Sheets and is a style sheet language used for describing the presentation semantics (look and formatting) of documents written in a mark-up language. It is generally used in styling HTML and XTML webpages, but can similarly be applied to XML documents. (Wikipedia, 2013a)

CSS is primarily designed to offer flexibility and control over separation of content in HTML documents or other mark-up languages. It can alter elements such as page or presentation layout, fonts, colors, etc. The name CSS comes from the fact that the language defines a priority to which style rule to apply first to the document, if more than one rule applies to a particular element. In this *cascade*, the priorities are calculated so the results are predictable. (Wikipedia, 2013a)

The World Wide Web Consortium (W3C) is the organization that develops and maintains the specifications for CSS since 1998. W3C also offers a CSS validating service. (Wikipedia, 2013a)

## 4.1 History of CSS

Style sheets have existed in various forms since the 1980s with the beginning of SGML (Standard Generalized Markup Language (Wikipedia)). Cascading Style Sheets were developed as a means for creating a consistent approach to providing style information for web documents. (Wikipedia, 2013a)

The need for a consistent way of providing style to web documents came along with the evolution of HTML and Web Browsers. It was needed (as it is to this day as well) to give web developers more control over site appearance, at the cost of complex HTML code. (Wikipedia, 2013a)

The foundations for what became CSS were CHSS (Cascading HTML Style Sheet) and SSP (Stream-based Style Sheet Proposal). The two were chosen out of nine applications made on the W3C web-styling mailing list. CHSS was proposed by Håkon

Wium Lie in October 1994. SSP was the own styling language of a web browser that Bert Bos was working on. Bert and Bos worked together to develop the standard now known as CSS (the *H* was dropped since the standard was developed not only for HTML documents). (Wikipedia, 2013a)

### 4.2 Versions

CSS has various levels and profiles. Each level was built and improved on the last one, with new added features. These levels are denoted as CSS 1, CSS 2, CSS 3 and CSS 4. Profiles are subsets of each level and they are built to apply to specific devices or user interfaces. There are profiles for printers, mobile phones and television sets for example. (Wikipedia, 2013a)

**CSS 1** was the first recommendation published as official by W3C, and known as CSS level 1. It was published in December 1996. It included support for: font proprieties (type faces and emphasis); text color and backgrounds; text attributes (spacing between words, lines and letters); alignment of text, images, tables and other elements; margin, border, padding and positioning for most elements; etc. W3C no longer maintains the specifications for CSS 1 Recommendation. (Wikipedia, 2013a)

**CSS 2** was published as a recommendation in May 1998. Based on CSS level 1 recommendation it included new capabilities like absolute, relative and fixed positioning of elements and z-index, the concept of media types, new font proprieties such as shadows, etc. This recommendation is no longer maintained. (Wikipedia, 2013a)

CSS 2.1 is the name given to CSS 2 revision 1, the current working recommendation of W3C for CSS. It fixes errors introduced in CSS 2, removes some poorly supported features and already implemented browser extensions to the specification. It went through various stages of completeness, with the first Candidate Recommendation in February 2004. It reverted to working draft in May 2005 followed by another Candidate recommendation in July 2007, with two updates during 2009. It again reverted to working draft in 2010 followed by the publication of the final W3C

Recommendation in June 2011, after being reviewed by the W3C Advisory committee. (Wikipedia, 2013a)

Unlike the previous levels that are a defined as a single specification, CSS 3 (or CSS level 3) is split into different documents called modules. Each of the modules adds or improves on features of CSS 2. Earliest CSS 3 drafts were published in 1999. Because of the modularity of CSS 3 (over 50 modules published as of 2012) there are different stages of stability and status for each of them. Formal recommendations have been published for four of the modules: Media queries, Name spaces, Selector level 3, Color. Some modules, like Background and Borders, have the Candidate Recommendation status. (Wikipedia, 2013a)

### 4.3 New CSS3 features

The four most important features of CSS3 and the four of the modules that are in their final recommendation status are:

- Media Queries (published in 2012) – It is possibly the most important feature of CSS3 because it allows certain conditions to be applied to style sheets, making websites more fluid and able to fit all screen sizes. Media queries allow developers to easily create styles without removing content. (1stwebdesigner.com, 2013)
- Namespaces (published in 2011) – This CSS Namespaces module defines syntax for using namespaces in CSS. It defines the `@namespace` rule for declaring a default namespace and for binding namespaces to namespace prefixes. (w3.org, 2013b)
- Selectors Level 3 (published in 2011) – This module is an addition to Level 2 selectors introduced in CSS2, and allows choosing DOM elements based on their attributes, so there is no more need to specify classes and IDs for every element. Instead the attribute field can be utilized for styling (webreference.com, 2013)
- Color (published in 2011) - This specification describes color values and properties for foreground color and group opacity. These include properties and values from CSS level 2 and new values. (w3.org, 2013a)

Other modules include new features like:

- transparency and gradients
- easy column generation for content (using column-count, column-width, column-gap, column-rule attributes)
- multiple backgrounds (allow us to use different backgrounds straight from CSS); vendor prefixes, are used for rendering certain content on specific browsers (using `-moz-` for Mozilla, `-webkit-` for Safari and Chrome, `-o-` for Opera, `-ms-` for Internet Explorer). The latest versions of modern browsers don't need all these, but under certain conditions they are used for the styles to be rendered properly
- new pseudo classes that are used to target certain elements within a page (such as `:root`, `:empty`, `:only-child`, etc.)
- text effects for generating effects for text (like shadows or overflows) (1stwebdesigner.com, 2013) (css3.info, 2013)

# 5 HTML 5

Wikipedia describes HTML5 as: "a markup language for structuring and presenting content for the World Wide Web and a core technology of the Internet. It is the fifth revision of the HTML standard (created in 1990 and standardized as HTML 4 as of 1997) and, as of December 2012, is a W3C Candidate Recommendation." It aims at making web content easier to read and access for people and consistently understood by computers and devices (via browsers or parses or other means) and it intends to replace HTML4, XHTML and DOM Level 2 HTML. (Wikipedia, 2013h)

## 5.1 New features

HTML5 is now easy to implement. The declaration for HTML document has been simplified to just `<!doctype html>`. This is easy now because HTML 5 is no longer part of SGML, but is instead a markup language all on its own. Declaring a character set has been simplified as well with just declaring a meta tag `<meta charset="UTF-8">`. (about.com, 2013)

Many other new features are visible in the multimedia area. These have been created to suid the user's needs for running heavy contend on low powered devices (such as mobile phones and tablets). That means that media content can be executed easier and faster. New attributes here are:

- `<header>, <footer>` - that make delimiting zones for headers and footers within a webpage easier
- `<audio>`, `<video>` - that delimit audio and video content
- `<embed>` - defines an area where interactive content is placed
- `<canvas>` - allows for drawing graphics using scripting (techrepublic, 2013)

New features in other areas include:

- `<input>` - new attributes have been created for the input tag, with easier validation and selection of input type (techrepublic, 2013)

- `<datalist>` - acts similar to a combo box, where the systems give suggestions, but the users can choose their own inputs. Similar behavior previously required Javascript to implement (techrepublic, 2013)
- `<contenteditable>` - specifies whether a user can edit the content of the element (HTML goodies, 2013)

Some APIs have also been introduced, like:

- Drag and drop API – allows for items to be dragged inside and to and from web pages. It also provides a simpler API for use with Mozilla-based applications and extensions.
- History API – allows for browser history manipulation, useful for interactively loading information intro web pages
- Full screen API – allows for control of a webpage displayed in full screen without the browser UI elements being displayed, useful feature for displaying web pages on mobile devices
- Pointer lock API – allows for the pointer or cursor to be locked to the content, so such applications such as games do not lose focus when the pointer reaches the window limit (Mozilla, 2013)

# 6 Photoshop

Adobe Photoshop is a graphics editing software, developed by Adobe Systems. (Wikipedia, 2013d)

The initial program was created by Thomas Knoll and his brother John Knoll in 1987 and early 1988. It was first called Display and ImagePro, later renamed Photoshop. The first version was distributed as software bundled with slide scanners sold by Barneyscan. Adobe bought the rights to distribute Photoshop in 1988 after Knoll held presentations of the program in Silicon Valley to engineers at Apple and to Russel Brown, art director at Adobe. The first version of Photohop was released in 1990 and was an Apple Computers exclusive product. (Wikipedia, 2013d)

The current commercial version of Photoshop is called Photoshop CS6. The program series was renamed to CS in 2003. Before that they were named Photoshop 1 through 7. The CS6 version is the 13th iteration of the program. It is available for PC and Macintosh computers, with both 32bit and 64bit versions. (Wikipedia, 2013e)

Photoshop is widely used in various types of activities, aside from the probably most common use, photo editing. According to the Adobe Photoshop official website, the program is used by astronomers, animators, medical professionals, science researchers, web designers and others. In web design Photoshop is very useful in the retouching of pictures, creating content such as custom buttons, page layouts and 3D artwork. (Adobe, 2013b)

## 6.1 Photoshop general interface

The Photoshop user interface has been improved with each new version of the program and any new tool addition to it. In the newer versions the interface can be customized or rearranged by choosing a different workspace.

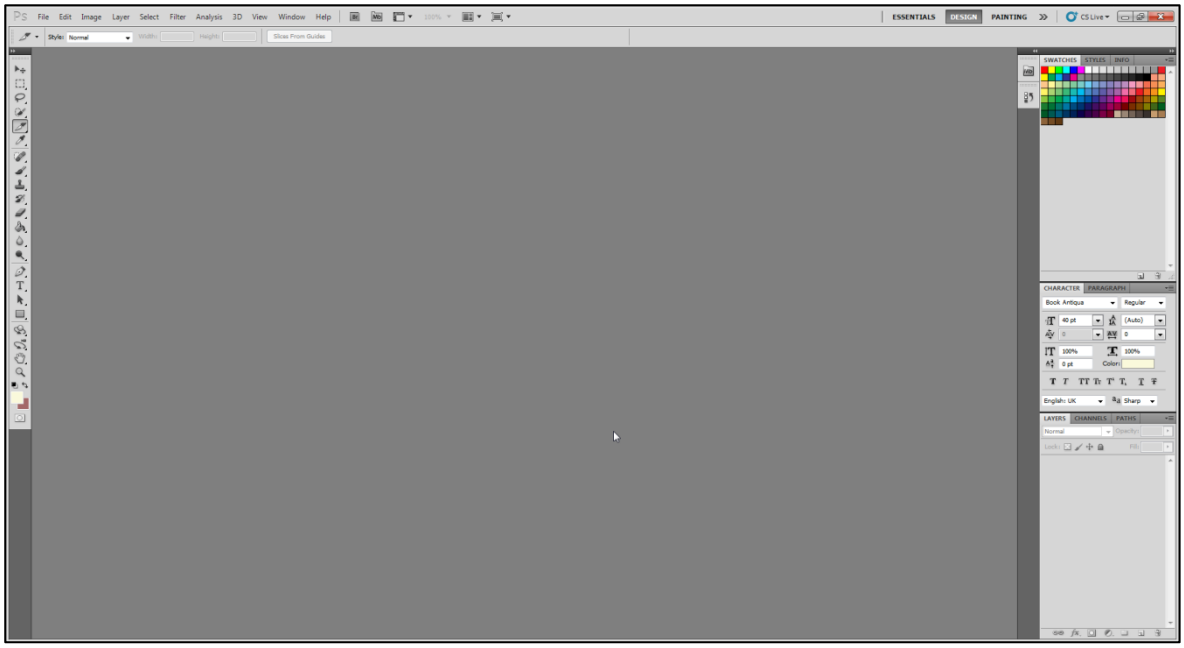

**Figure 2 – Photoshop general interface**

The toolbar on the left side of the screen contains all the tools available to the user. Tools can be accessed by clicking on them or by using predefined keyboard shortcuts. More than one tool can be available in a tool group that contains similar function tools (for example the brush tool is grouped with the pencil tool, color replacement tools and mixer brush tool). Tool groups can be identified by a small triangular notch in the lower part of the tool icon. Accessing the tools in the tool group can be done by holding the mouse left-click or by right clicking on the appropriate group.

On the left side of the screen there are panels with various uses. The most notable are the layers panel and info panel. These panels can be shown or hidden using keyboard shortcuts or by accessing the Window menu on the menu bar. Different layouts for these panels can be selected by changing the Workspace from the Window menu.

# 7 Webpage from Photoshop PSD layout

This chapter presents the Slice and Export for Web Tools, both used for exporting a webpage or website template. A simple webpage was created for this project, but the creation of it is not covered here.

The layout was saved in the PSD file format, which is the default file format for Photoshop. It stands for PhotoShop Document. PSD files store images with support for most imaging options in Photoshop, including layers with masks, transparencies, text, spot colors, etc., as opposed to JPG or GIF files that restrict such options to provide streamlined predictable functionality. (Wikipedia, 2013d)

## 7.1 Photoshop slices

The Slice tool is used to slice an image file into smaller images that are reassembled in a webpage. By slicing the image the user can assign URLs to various images to create navigability, or to individually optimize each image with its own optimization settings. It can be found in a tool group with Crop and Slice Select tool by clicking the appropriate button on the tool bar, or by pressing Shift+C. Visually the slice tool icon looks like  (Adobe, 2013a)

There are three different types of slices based on the way they are generated:

- Auto slices, that are generated automatically
- User slices, created by the user using the slice tool
- Layer based slices, created from the Layers panel

Similarly there are three types of content for slices: image, table and no image. (Adobe, 2013a)

User slices can be created in different ways.

- They can be created from Guide Lines - after Guide Lines are added to an open image or layout to delimit the content, the user selects the Slice tool and from the Options bar selects Slices From Guides

- By using the slice tool – by clicking and dragging the mouse over an area, a slice is created; by holding the Shift key the selected area will be fixed to square; by holding Alt key the selected area will be drawn from the center (Adobe, 2013a)


**Figure 3 - User slices on the website layout document**

## 7.2 Creating an HTML document from a sliced layout

After the layout document has been sliced, the slices can be modified by double clicking the wanted slice with either the Slice tool or the Slice Select tool. A window opens with the options the user can modify. These options allow the selection of the type of the slice as mentioned before, give the generated file a name, set a hyper link to it, the target file, a message text and an alternate text. Dimensions can also be set, but by default the dimensions of the slice are given. In the case of No image type slice, text can be input with HTML formatting allowed. The exported HTML page will have the text and formatting included.

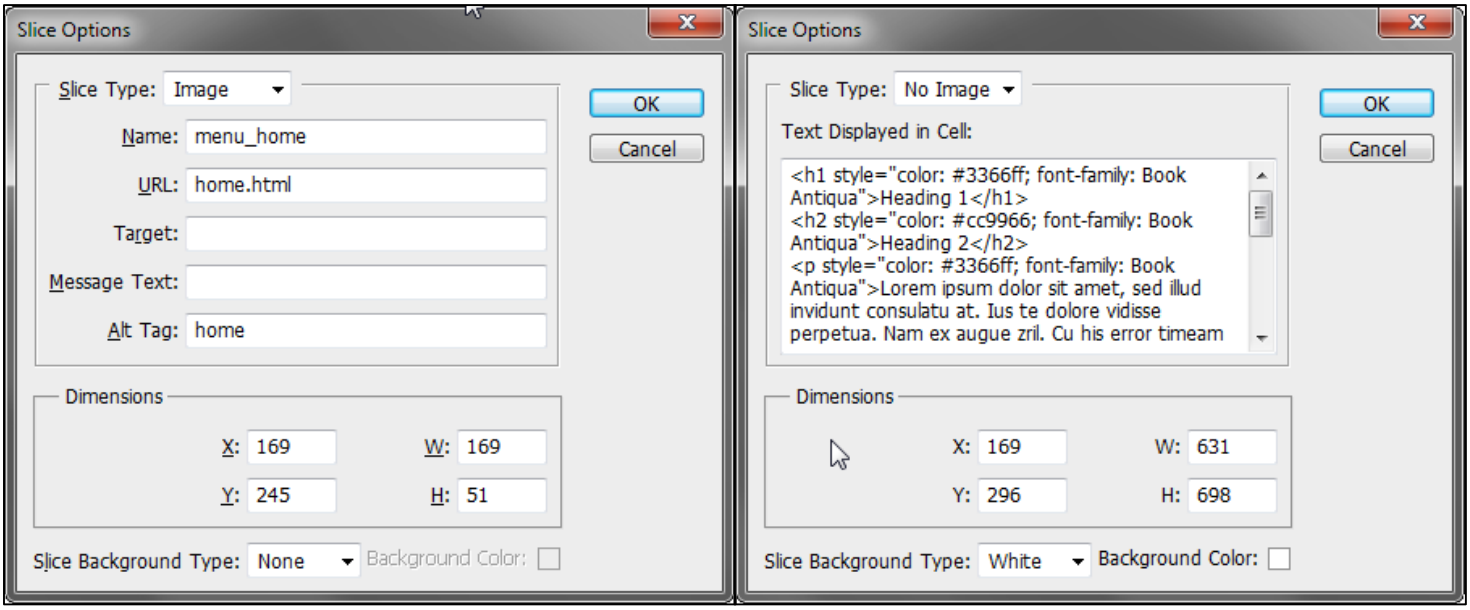


**Figure 4 - Options for Image slices and No Image slices**

Generating the HTML webpage is done by selecting Save for Web and Devices from the File menu. This opens a window where the user can select the type of generated images (JPG, GIF, PNG-8, PNG-24) and various other options not relevant at this time. For the purpose of this project PNG-24 was selected with transparency. (Adobe, 2013a)


**Figure 5 - Save options**

After pressing Save another window opens with the options for naming the webpage, the format in which it will be saved and the settings for it and which slices will be selected in the output. The format options are HTML and Images (which exports both HTML file and images), Images only (exports only images) and HTML only (creates only an HTML file, with no images). The HTML and Images with XHTML settings was used for this example. The result is an HTML file containing the code and a folder where the pictures are stored. The resulting XHTML code is valid and conforms to W3C standards. (Adobe, 2013c)

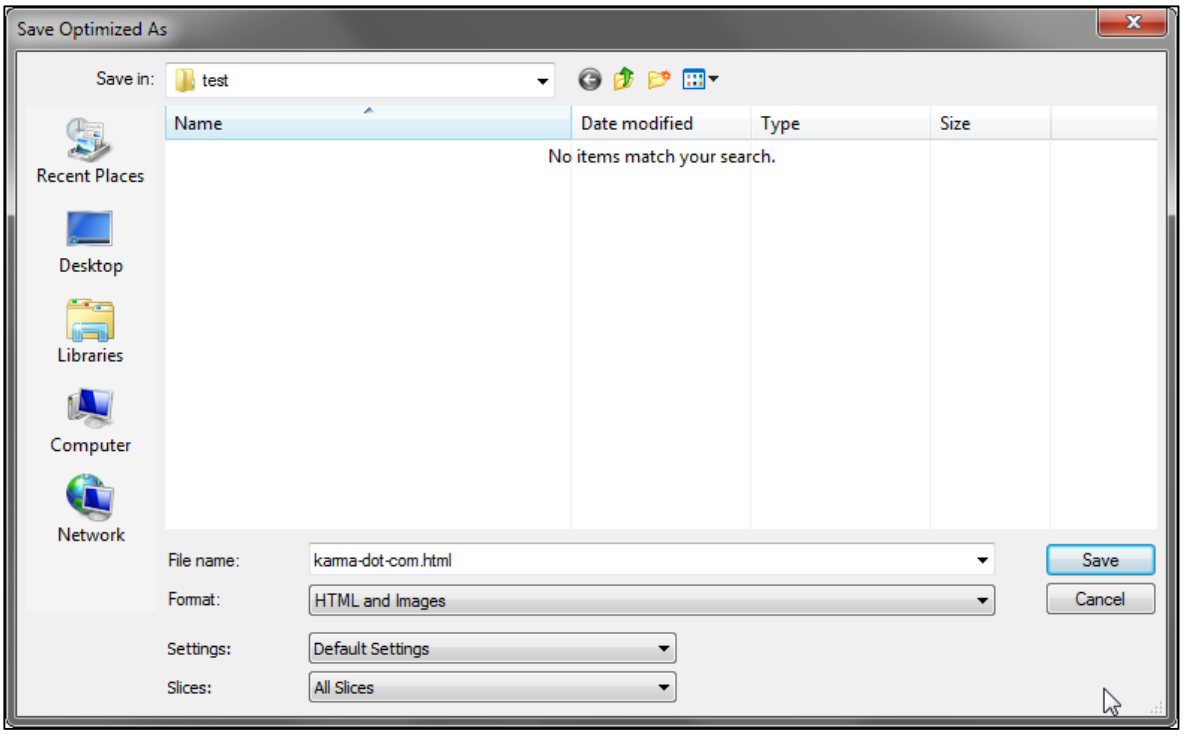


**Figure 6 - Save options**


**Figure 7 - The exported HTML file viewed in Chrome Browser**

Importing the results into a Visual Studio ASP.NET website is done by adding the generated images folder and the HTML file to the solution. After creating a new WebForms page inside the solution, the content of the HTML file generated from Photoshop can be copied into the ASPX page. From here on editing can continue in the same manner as any other ASP.NET project. More content can be added later, along with navigability and more styling.

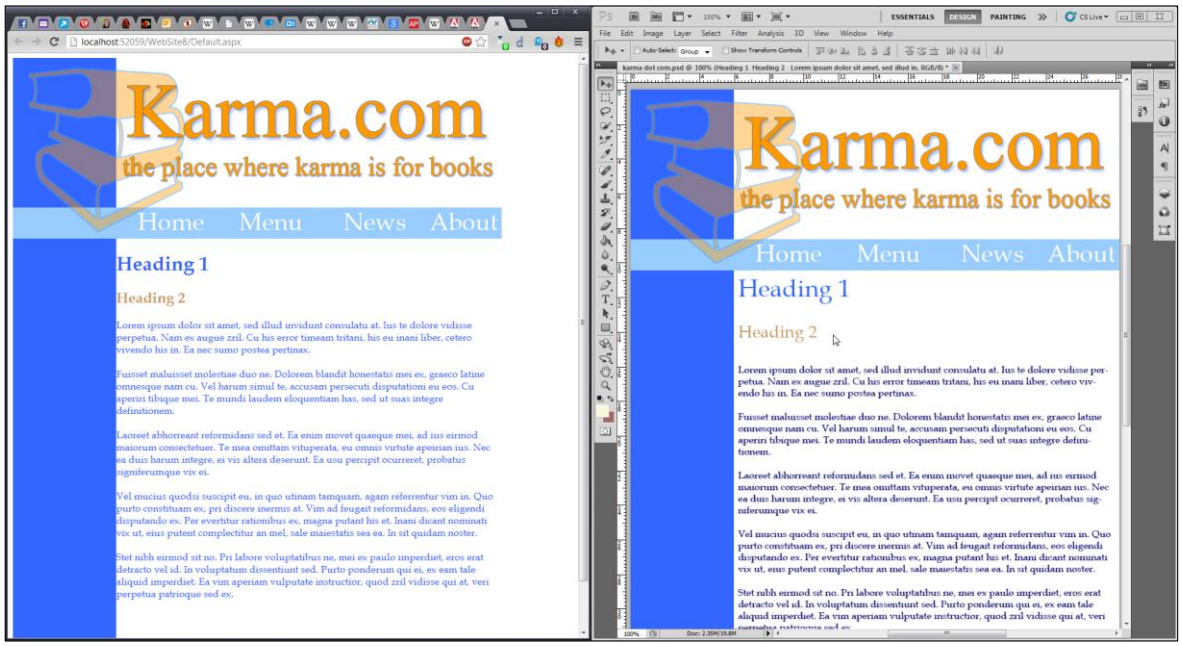


**Figure 8 - ASPX page in comparison to Photoshop document**

# 8 Styling inputs with CSS

Using new declarations introduced in CSS3, the webpage's style can be carried on from the initial design in Photoshop. Namely text boxes can have a drop shadow similar to the one in the title using the `box-shadow` property.


**Figure 9 – Default style (top) and CSS styled input boxes**

```
<tr>
    <td>
        <img id="left_column" src="images/left_column.png" width="169" height="698" alt="left column" /></td>
        <td width="631" height="698" colspan="4" bgcolor="#FFFFFF" style="vertical-align:top; padding-left:10px;">
        <h1 style="color: #3366ff; font-family: Book Antiqua">Newsletter register</h1>
        <br />
        <asp:Label ID="lbName" runat="server" CssClass="labels" Text="First name"></asp:Label>
        <asp:TextBox ID="tbName" runat="server" CssClass="inputs"></asp:TextBox>
        <br />
        <asp:Label ID="lbLastName" runat="server" CssClass="labels" Text="Last name"></asp:Label>
        <asp:TextBox ID="TextBox2" runat="server" CssClass="inputs"></asp:TextBox>
        <br />
        <asp:Label ID="lbEmail" runat="server" CssClass="labels" Text="Email address"></asp:Label>
        <asp:TextBox ID="tbEmail" runat="server" CssClass="inputs"></asp:TextBox>
        <br />
        <asp:Button ID="btRegister" runat="server" CssClass="button" Text="Register" />

    </td>
</tr>
```

**Figure 10 - Code snippet from the ASPX page**

style.css | home.aspx

```
.button {
    width:100px;
    right:20px;
    bottom:20px;
    background:#0099CC;
    color:#FFFFFF;
    font-family: Book Antiqua;
    height:30px;
    border-radius: 15px;
    border: 1p solid #999999;
    font-size: 18px;
}

.button:hover
{
    background: #cc9966;
    color: #0099CC;
}

.labels
{
   font-family: Book Antiqua;
   font-size: 22px;
   color: #3366ff;
}

.inputs
{
    width:375px;
    display:block;
    border: 1px solid #cc9966;
    height: 20px;
    box-shadow: 1px 1px 8px #0099CC;

}

.inputs:focus
{
     border: 2px solid #cc9966;
}
```

**Figure 11 - CSS code for the elements inside the page (Joice, 2012)**

CSS classes were defined for all the elements. They are properly named to describe the type of element they are modifying. Two pseudo classes were created for the button and input elements to create dynamism. A mention has to be made that a special validator package for CSS3 was installed for Visual Studio 2010, for the purpose of this project (it can be downloaded from the Microsoft Visual Studio page). By default Visual Studio has only CSS 2.1 validators installed.

The declarations used that come from CSS3:

- `border-radius` – adds rounded corners to elements, in this case the button element (w3schools, 2013a)
- `box-shadow` – adds shadows to elements, in this case the text box elements (w3schools, 2013b)

The declarations work after testing with Google Chrome browser. Other browsers were not tested. According to W3Schools definitions for the two properties, they should work with all latest versions of the browsers. However since CSS3 has not become a final standard (or final recommendation, according to W3C) the user can use the `-webkit-border-radius` and `-moz-border-radius` declarations to ensure compatibility with specific browsers, using the same settings.

# 9 Styling radio buttons and check boxes with CSS

This chapter presents a way to change the style of Check boxes and Radio buttons is presented. The style carries on for both of the controls with the usage of `border-radius` and `box-shadow` declarations, but others have been used as well.

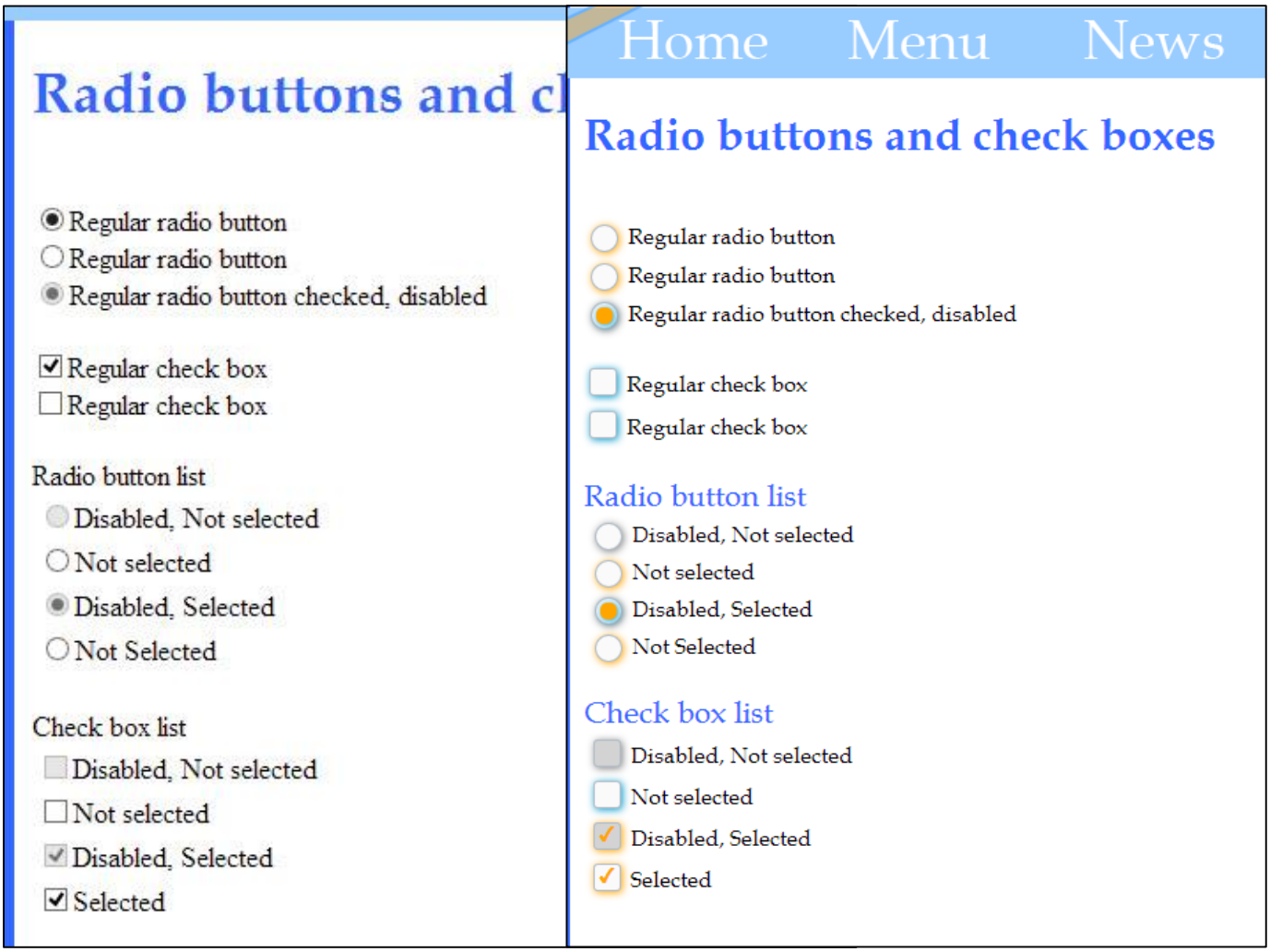


**Figure 12 - The default styled radio buttons and check boxes (left) and the styled versions (right)**

In this case attribute selectors and pseudo classes have been used. These allow us to give the same style to all the elements of the same type (radio buttons and check boxes) within any page of the website or web application. Attribute selectors are useful for styling elements without classes or IDs. (w3schools, 2013c). The attribute selector in this case is the `input` element. `type="radio"` and `type="checkbox"` are the attributes used to select which inputs are modified. (w3schools, 2013d)

CSS3 give us more flexibility in modifying the style of checkboxes and radio buttons by using the `:checked` `:enabled` and `:disabled` selectors. These combined with CSS2 `:after` and CSS1 `:active` selectors give us the possibility to alter the controls. (w3schools, 2013e)

Additionally styling the labels that checkboxes and radio buttons are often followed by, can be styled (separate from the other labels in the website) by using the adjacent sibling combinator `input[type="radio"] + label` and `input[type="checkbox"] + label`. What this does in this case is select the `label` element that immediately follows an `input` element. (w3.org, 2013c)

```css
input[type="radio"]
{
    -webkit-appearance: none;
    background-color: #fafafa;
    border: 1px solid #b1bcc7;
    box-shadow: 1px 1px 8px orange;
    padding: 9px; border-radius: 50px;
    display: inline-block;
    position: relative;
}

input[type="radio"] + label
{
    vertical-align: super;
    font-family: Book Antiqua;
    margin-left: 4px;
}

input[type="radio"]:checked
{
    border: 1px solid #b1bcc7;
    box-shadow: 1px 1px 8px orange;
}

input[type="radio"]:checked:after
{
    content: " ";
    width: 12px; height: 12px;
    border-radius: 50px;
    position: absolute;
    top: 3px; left: 3px; background: orange;
    box-shadow: 1px 1px 8px #0099CC;
    text-shadow: 0px;
    font-size: 32px;
}

input[type="radio"]:active, input[type="radio"]:checked:active
{
    box-shadow: 1px 1px 8px #0099CC;
}

input[type="radio"]:disabled
{
    box-shadow: 1px 1px 8px gray;
}
```

```css
input[type="checkbox"]
{
    -webkit-appearance: none;
    background-color: #fafafa;
    border: 1px solid #b1bcc7;
    box-shadow: 1px 1px 8px #0099CC;
    padding: 9px;
    border-radius: 3px;
    display: inline-block;
    position: relative;
}

input[type="checkbox"]:disabled
{
   box-shadow: 1px 1px 8px gray;
   background: lightgray;
}

input[type="checkbox"] + label
{
    vertical-align: super;
    font-family: Book Antiqua;
    margin-left: 4px;
}

input[type="checkbox"]:active, input[type="checkbox"]:checked
{
    box-shadow: 1px 1px 8px orange;
}

input[type="checkbox"]:checked:after
{
    content: '\2714';
    font-size: 14px;
    position: absolute;
    top: 0px;
    left: 3px;
    color: orange;
}
```

**Figure 13 - CSS code for radio buttons styling (left) and check boxes (right) (Simpson, 2013)**

The CSS3 declarations used here:

- `-webkit-appearance` – changes the appearance of an element to look like a standard user interface element. In this case the `none` value takes away any default browser style so it can be freely modified like a regular element. The `-webkit-` vendor specific property is used here as an alternative since it is not yet supported by any major browser (w3schools, 2013f) (Simpson, 2013)
- `border-radius` – adds rounded corners to elements, in this case the button element (w3schools, 2013a)
- `box-shadow` – adds shadows to elements, in this case the text box elements (w3schools, 2013b)
- `text-shadow` – adds one or more shadows to a text (w3schools, 2013g)

Another notable declaration here is `content:"\2714"`. It is present from CSS2 and changes the content of the input to the Unicode character 2714, Heavy check mark. (fileformat.info, 2013)

Like the previous chapter, the testing was done using Chrome browser. According to w3schools website the `appearance` propriety is not officially supported by any modern browser, but using the vendor specific tags ensures the desired effect. Results may be different in other browsers. (w3.org, 2013e)

# 10 Evaluation

The result of this project is a ASP.NET webpage that was created from a design layout from a Photoshop document. CSS styling was applied so that the resulting webpage matches the design. This can be a real life situation where an application developer receives a layout from a graphic designer and needs to create a website out of it.

The results of this project should be valid from the standpoint of the technologies used (CSS and Photoshop as well as ASP.NET are widely used around the world in the web application development life cycle), and can be used for a long time or until other easier methods arise. It is possible to achieve these results using other web development frameworks, because the layout exported from Photoshop is XHTML code and CSS can be applied to any kind of HTML, XHTML or XML code.

The project was completed in the allocated amount of time and in accordance to the project plan. The chosen research methods are both commonly used and have been well suited to this project. Using these methods I have learned how to create an ASPX webpage from a layout designed in Photoshop and how CSS can help style the webpages to keep the design consistent throughout all webpages.

The sources used are reliable. The book authors have years of experience in developing web applications with ASP.NET and Microsoft technologies, and have written other books on the subject. The web sources from Wikipedia are written and maintained by specialists and contributors around the world. The others are the responsibility of their original authors.

# 11 Further research on this topic

Further research can be done to improve the techniques presented in this thesis project. Likewise the impact of newer versions of ASP.NET, Photoshop and CSS can provide easier ways of achieving the wanted results. Other ideas can include an empirical research into how CSS and Photoshop can increase creativity or workflow in web application design. Also a more in depth look at how different browsers render the CSS styles and what compatibility issues can arise with the yet unfinished CSS3 modules specifications.

Another research can be done as well to find out how a website project created with MVC can be fully altered since it gives full control over all HTML code and complete control over CSS scripting. (Esposito, 2011, s. 21)

In the lines of this research paper a continuation of it can investigate what or how other user controls can be modified as well as modifications that can be done to ASP.NET Web Controls of Web Forms.

# 12 Conclusion

From the business stand point, CSS and Photoshop along with ASP.NET or any other web development platform or framework, are needed tools for developing websites or web applications that keep the user connected thus in turn providing revenue in some form or another. The more users a business's website gets the more profits they can bring. Users will not spend time on websites that do not attract them visually or that are hard to read or navigate.

This project aimed to find out a way to create a webpage with ASP.NET from a Photoshop document. This method is used in real life web development. It is common for a web designer to create the layouts of a website or web application in Photoshop and then ship it to the developers (or programmers) who add the needed code and functionality. This can apply to the students of HAAGA-HELIA that take part in the web application development courses. After reading this document they can get an idea of what happens when they may face the situation of creating a webpage from a readymade design.

In short and in conclusion CSS and Photoshop are tools that can help individuals or companies create their desired web pages or web applications in the way they have envisioned it. This project presented these tools and their history, and two ways in which they can be used.

# Bibliography


1stwebdesigner.com. (2013, 05 26). *CSS3 introduction.* Retrieved from 1stwebdesigner.com: http://www.1stwebdesigner.com/css/css3-introduction/

about.com. (2013, 05 26). *What's new in HTML5.* Retrieved from webdesign.about.com: http://webdesign.about.com/od/html5/a/html_5_whats_new.htm

Adobe. (2013a, 05 11). *Adobe Photoshop - Slicing webpages.* Retrieved 05 11, 2013, from help.adobe.com: http://help.adobe.com/en_US/photoshop/cs/using/WSfd1234e1c4b69f30ea53e41001031ab64-7570a.html

Adobe. (2013b, 05 01). *Photoshop.* (Adobe) Retrieved 05 01, 2013, from photoshop.com: http://www.photoshop.com/products/photoshop/who#

Adobe. (2013c, 05 13). *Photoshop - HTML options for Slices.* Retrieved 05 133, 2013, from help.adobe.com: http://help.adobe.com/en_US/photoshop/cs/using/WSfd1234e1c4b69f30ea53e41001031ab64-7539a.html

Bochicchio, D., Mostarda, S., & De Sanctis, M. (2011). *ASP.NET 4.0 In Practice.* Manning Publications.

css3.info. (2013, 05 26). *CSS3 Previews.* Retrieved from css3.info: http://www.css3.info/preview/

Esposito, D. (2011). *Programming Microsoft ASP.NET 4.* Microsoft Press.

fileformat.info. (2013, 06 14). *Unicode Character 'HEAVY CHECK MARK' (U+2714).* Retrieved from fileformat.info: http://www.fileformat.info/info/unicode/char/2714/index.htm

HTML goodies. (2013, 05 26). *Wnat's new with HTML5.* Retrieved from HTML goodies: http://www.htmlgoodies.com/article.php/3875431#fbid=9eafZaV7c-W

Joice, J. (2012, 09 04). *Style Web Forms Using CSS.* Retrieved 05 13, 2013, from sitepoint.com: http://www.sitepoint.com/style-web-forms-css/

microsoft. (2013, 05 26). *What's new in the .NET framework version 4.5.* Retrieved from msdn.microsoft.com: http://msdn.microsoft.com/en-us/library/ms171868.aspx#web

Mozilla. (2013, 26 05). *Web developer guid*. Retrieved from Mozilla: https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/HTML5

Simpson, J. (2013, 06 14). *Creating Custom Form Checkboxes and Radio Buttons with Just CSS!* Retrieved from insterthtml.com: http://www.inserthtml.com/2012/06/custom-form-radio-checkbox/

techrepublic. (2013, 05 26). *10 new HTML5 tags you need to know about*. Retrieved from techrepublic.com: http://www.techrepublic.com/blog/10things/10-new-html5-tags-you-need-to-know-about/3219

w3.org. (2013a, 05 26). *CS Color module level 3*. Retrieved from w3.org: http://www.w3.org/TR/css3-color/

w3.org. (2013b, 05 25). *CSS namespaces module*. Retrieved from w3.org: http://www.w3.org/TR/css3-namespace/#intro

w3.org. (2013c, 06 14). *w3.org*. Retrieved from CSS3 Selectors - Sibling combinators: http://www.w3.org/TR/css3-selectors/#sibling-combinators

w3.org. (2013e, 06 26). *Syntax and data types - Vendor-specific extensrions*. Retrieved from w3.org: http://www.w3.org/TR/CSS21/syndata.html#vendor-keywords

w3schools. (2013a, 05 13). *Border radius property*. Retrieved 05 13, 2013, from W3Schools: http://www.w3schools.com/cssref/css3_pr_border-radius.asp

w3schools. (2013b, 05 13). *Box shadow property*. Retrieved 05 13, 2013, from W3Schools: http://www.w3schools.com/cssref/css3_pr_box-shadow.asp

w3schools. (2013c, 06 12). *CSS attribute selectors*. Retrieved from w2schools.com: http://www.w3schools.com/css/css_attribute_selectors.asp

w3schools. (2013d, 06 14). *HTML forms*. Retrieved from w3schools: http://www.w3schools.com/html/html_forms.asp

w3schools. (2013e, 06 14). *CSS selectors*. Retrieved from w3schools: http://www.w3schools.com/cssref/css_selectors.asp

w3schools. (2013f, 04 14). *CSS3 appearance property*. Retrieved from w3schools: http://www.w3schools.com/cssref/css3_pr_appearance.asp

w3schools. (2013g, 06 14). *CSS3 text-shadow property*. Retrieved from w3schools: http://www.w3schools.com/cssref/css3_pr_text-shadow.asp

webreference.com. (2013, 05 26). *Top 10 new features in CSS3*. Retrieved from webreference.com: http://www.webreference.com/authoring/css3/index.html

Wikipedia. (2013a, 05 03). *CSS*. Retrieved 05 02, 2013, from wikipedia.com: https://en.wikipedia.org/wiki/Cascading_Style_Sheets

Wikipedia. (2013b, 05 09). *Microsoft MVC*. Retrieved 05 09, 2013, from wikipedia.com: http://en.wikipedia.org/wiki/ASP.NET_MVC_Framework

Wikipedia. (2013c, 05 09). *MVC*. Retrieved 05 09, 2013, from wikipedia.com: http://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller

Wikipedia. (2013d, 05 01). *Photoshop - Wikipedia*. Retrieved 05 01, 2013, from wikipedia.com: http://en.wikipedia.org/wiki/Adobe_Photoshop

Wikipedia. (2013e, 05 01). *Photoshop Versions*. Retrieved 05 01, 2013, from wikipedia.com: http://en.wikipedia.org/wiki/Adobe_Photoshop_version_history

Wikipedia. (2013f, 04 20). *ASP.NET*. Retrieved from wikipedia: http://en.wikipedia.org/wiki/ASP.NET

Wikipedia. (2013g, 05 26). *.NET Framework version history*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/.NET_Framework_version_history#.NET_Framework_4.5

Wikipedia. (2013h, 05 26). *HTML5*. Retrieved from Wikipedia: http://en.wikipedia.org/wiki/HTML5

# Table of figures

Figure 1 - Webforms model (Esposito, 2011, p. 5) ..... 6
Figure 2 – Photoshop general interface ..... 15
Figure 3 - User slices on the website layout document ..... 17
Figure 4 - Options for Image slices and No Image slices ..... 18
Figure 5 - Save options ..... 18
Figure 6 - Save options ..... 19
Figure 7 - The exported HTML file viewed in Chrome Browser ..... 20
Figure 8 - ASPX page in comparison to Photoshop document ..... 20
Figure 9 – Default style (top) and CSS styled input boxes ..... 21
Figure 10 - Code snippet from the ASPX page ..... 21
Figure 11 - CSS code for the elements inside the page (Joice, 2012) ..... 22
Figure 12 - The default styled radio buttons and check boxes (left) and the styled versions (right) ..... 24
Figure 13 - CSS code for radio buttons styling (left) and check boxes (right) (Simpson, 2013) ..... 25

# Attachments

## Attachment 1. Sample code for the generated HTML file from Photoshop

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>karma dot com</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
</head>
<body bgcolor="#FFFFFF">
<!-- Save for Web Slices (karma dot com.psd) -->
<table id="Table_01" width="800" border="0" cellpadding="0" cellspacing="0">
   <tr>
   <td>
   <img id="header_left" src="images/header_left.png" width="169" height="245" alt="header left" /></td>
   <td colspan="4">
   <a href="home.html">
   <img id="header_right" src="images/header_right.png" width="631" height="245" border="0" alt="header
right" /></a></td>
   </tr>
   <tr>
   <td>
   <img id="menu_blank" src="images/menu_blank.png" width="169" height="51" alt="menu" /></td>
   <td>
   <a href="home.html">
   <img id="menu_home" src="images/menu_home.png" width="169" height="51" border="0" alt="home" /></a></td>
   <td>
   <a href="menu.html">
   <img id="menu_menu" src="images/menu_menu.png" width="169" height="51" border="0" alt="menu" /></a></td>
   <td>
   <a href="news.html">
   <img id="menu_news" src="images/menu_news.png" width="169" height="51" border="0" alt="news" /></a></td>
   <td>
   <a href="about.html">
   <img id="menu_about" src="images/menu_about.png" width="124" height="51" border="0" alt="about"
/></a></td>
   </tr>
   <tr>
   <td>
   <img id="left_column" src="images/left_column.png" width="169" height="698" alt="left column" /></td>
   <td width="631" height="698" colspan="4" bgcolor="#FFFFFF"><h1 style="color: #3366ff; font-family: Book
Antiqua">Heading 1</h1>
<h2 style="color: #cc9966; font-family: Book Antiqua">Heading 2</h2>
<p style="color: #3366ff; font-family: Book Antiqua">Lorem ipsum dolor sit amet, sed illud invidunt
consulatu at. Ius te dolore vidisse perpetua. Nam ex augue zril. Cu his error timeam tritani, his eu inani
liber, cetero vivendo his in. Ea nec sumo postea pertinax.
<br/><br/>
Fuisset maluisset molestiae duo ne. Dolorem blandit honestatis mei ex, graeco latine omnesque nam cu. Vel
harum simul te, accusam persecuti disputationi eu eos. Cu aperiri tibique mei. Te mundi laudem eloquentiam
has, sed ut suas integre definitionem.
<br/><br/>
Laoreet abhorreant reformidans sed et. Ea enim movet quaeque mei, ad ius eirmod maiorum consectetuer. Te mea
omittam vituperata, eu omnis virtute apeirian ius. Nec ea duis harum integre, ei vis altera deserunt. Ea usu
percipit ocurreret, probatus signiferumque vix ei.
<br/><br/>
Vel mucius quodsi suscipit eu, in quo utinam tamquam, agam referrentur vim in. Quo purto constituam ex, pri
discere inermis at. Vim ad feugait reformidans, eos eligendi disputando ex. Per evertitur rationibus ex,
magna putant his et. Inani dicant nominati vix ut, eius putent complectitur an mel, sale maiestatis sea ea.
In sit quidam noster.
<br/><br/>
```

```
Stet nibh eirmod sit no. Pri labore voluptatibus ne, mei ex paulo imperdiet, eros erat detracto vel id. In
voluptatum dissentiunt sed. Purto ponderum qui ei, ex eam tale aliquid imperdiet. Ea vim aperiam vulputate
instructior, quod zril vidisse qui at, veri perpetua patrioque sed ex. </p></td>
   </tr>
   <tr>
   <td colspan="5">
   <img id="bottom_bar" src="images/bottom_bar.png" width="800" height="33" alt="bottom bar" /></td>
   </tr>
</table>
<!-- End Save for Web Slices -->
</body>
</html>
```

## Attachment 2. ASPX webpage created in Visual Studio

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs" Inherits="_Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>karma dot com</title>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
</head>
<body bgcolor="#FFFFFF">
<!-- Save for Web Slices (karma dot com.psd) -->
<table id="Table_01" width="800" border="0" cellpadding="0" cellspacing="0">
   <tr>
   <td>
   <img id="header_left" src="images/header_left.png" width="169" height="245" alt="header left" /></td>
   <td colspan="4">
   <a href="home.html">
   <img id="header_right" src="images/header_right.png" width="631" height="245" border="0" alt="header
right" /></a></td>
   </tr>
   <tr>
   <td>
   <img id="menu_blank" src="images/menu_blank.png" width="169" height="51" alt="menu" /></td>
   <td>
   <a href="home.html">
   <img id="menu_home" src="images/menu_home.png" width="169" height="51" border="0" alt="home" /></a></td>
   <td>
   <a href="menu.html">
   <img id="menu_menu" src="images/menu_menu.png" width="169" height="51" border="0" alt="menu" /></a></td>
   <td>
   <a href="news.html">
   <img id="menu_news" src="images/menu_news.png" width="169" height="51" border="0" alt="news" /></a></td>
   <td>
   <a href="about.html">
   <img id="menu_about" src="images/menu_about.png" width="124" height="51" border="0" alt="about"
/></a></td>
   </tr>
   <tr>
   <td>
   <img id="left_column" src="images/left_column.png" width="169" height="698" alt="left column" /></td>
   <td width="631" height="698" colspan="4" bgcolor="#FFFFFF" style="vertical-align:top; padding-left:
10px;">
<h1 style="color: #3366ff; font-family: Book Antiqua">Heading 1</h1>
<h2 style="color: #cc9966; font-family: Book Antiqua">Heading 2</h2>
<p style="color: #3366ff; font-family: Book Antiqua">Lorem ipsum dolor sit amet, sed illud invidunt
consulatu at. Ius te dolore vidisse perpetua. Nam ex augue zril. Cu his error timeam tritani, his eu inani
liber, cetero vivendo his in. Ea nec sumo postea pertinax.
<br/><br/>
```

```
Fuisset maluisset molestiae duo ne. Dolorem blandit honestatis mei ex, graeco latine omnesque nam cu. Vel
harum simul te, accusam persecuti disputationi eu eos. Cu aperiri tibique mei. Te mundi laudem eloquentiam
has, sed ut suas integre definitionem.
<br/><br/>
Laoreet abhorreant reformidans sed et. Ea enim movet quaeque mei, ad ius eirmod maiorum consectetuer. Te mea
omittam vituperata, eu omnis virtute apeirian ius. Nec ea duis harum integre, ei vis altera deserunt. Ea usu
percipit ocurreret, probatus signiferumque vix ei.
<br/><br/>
Vel mucius quodsi suscipit eu, in quo utinam tamquam, agam referrentur vim in. Quo purto constituam ex, pri
discere inermis at. Vim ad feugait reformidans, eos eligendi disputando ex. Per evertitur rationibus ex,
magna putant his et. Inani dicant nominati vix ut, eius putent complectitur an mel, sale maiestatis sea ea.
In sit quidam noster.
<br/><br/>
Stet nibh eirmod sit no. Pri labore voluptatibus ne, mei ex paulo imperdiet, eros erat detracto vel id. In
voluptatum dissentiunt sed. Purto ponderum qui ei, ex eam tale aliquid imperdiet. Ea vim aperiam vulputate
instructior, quod zril vidisse qui at, veri perpetua patrioque sed ex. </p>
</td>
   </tr>
   <tr>
   <td colspan="5">
   <img id="bottom_bar" src="images/bottom_bar.png" width="800" height="33" alt="bottom bar" /></td>
   </tr>
</table>
<!-- End Save for Web Slices -->
</body>
</html>
```

## Attachment 3. The second webpage with the modified inputs

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="home.aspx.cs" Inherits="_Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<title>karma dot com</title>
<link rel="stylesheet" type="text/css" href="Style.css" />
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
</head>
<body bgcolor="#FFFFFF">
<form id="form1" runat="server">
<!-- Save for Web Slices (karma dot com.psd) -->
<table id="Table_01" width="800" border="0" cellpadding="0" cellspacing="0">
   <tr>
   <td>
   <img id="header_left" src="images/header_left.png" width="169" height="245" alt="header left" /></td>
   <td colspan="4">
   <a href="home.html">
   <img id="header_right" src="images/header_right.png" width="631" height="245" border="0" alt="header
right" /></a></td>
   </tr>
   <tr>
   <td>
   <img id="menu_blank" src="images/menu_blank.png" width="169" height="51" alt="menu" /></td>
   <td>
   <a href="home.html">
   <img id="menu_home" src="images/menu_home.png" width="169" height="51" border="0" alt="home" /></a></td>
   <td>
   <a href="menu.html">
   <img id="menu_menu" src="images/menu_menu.png" width="169" height="51" border="0" alt="menu" /></a></td>
   <td>
   <a href="news.html">
```

```
   <img id="menu_news" src="images/menu_news.png" width="169" height="51" border="0" alt="news" /></a></td>
   <td>
   <a href="about.html">
   <img id="menu_about" src="images/menu_about.png" width="124" height="51" border="0" alt="about"
/></a></td>
   </tr>
   <tr>
   <td>
   <img id="left_column" src="images/left_column.png" width="169" height="698" alt="left column" /></td>
   <td width="631" height="698" colspan="4" bgcolor="#FFFFFF" style="vertical-align:top; padding-left:10px;">
<h1 style="color: #3366ff; font-family: Book Antiqua">Newsletter register</h1>
<br />
<asp:Label ID="lbName" runat="server" CssClass="labels" Text="First name"></asp:Label>
<asp:TextBox ID="tbName" runat="server" CssClass="inputs"></asp:TextBox>
<br />
<asp:Label ID="lbLastName" runat="server" CssClass="labels" Text="Last name"></asp:Label>
<asp:TextBox ID="TextBox2" runat="server" CssClass="inputs"></asp:TextBox>
<br />
<asp:Label ID="lbEmail" runat="server" CssClass="labels" Text="Email address"></asp:Label>
<asp:TextBox ID="tbEmail" runat="server" CssClass="inputs"></asp:TextBox>
<br />
<asp:Button ID="btRegister" runat="server" CssClass="button" Text="Register" />

        </td>
   </tr>
   <tr>
   <td colspan="5">
   <img id="bottom_bar" src="images/bottom_bar.png" width="800" height="33" alt="bottom bar" /></td>
   </tr>
</table>
<!-- End Save for Web Slices -->
    </form>
</body>
</html>
```

## Attachment 4. CSS file used to style the second webpage

```
.button {
    width:100px;
    right:20px;
    bottom:20px;
    background:#0099CC;
    color:#FFFFFF;
    font-family: Book Antiqua;
    height:30px;
    border-radius: 15px;
    border: 1p solid #b1bcc7;
    font-size: 18px;
}

.button:hover
{
    background: #cc9966;
    color: #0099CC;
}

.labels
{
   font-family: Book Antiqua;
   font-size: 22px;
   color: #3366ff;
}
```

```
.inputs
{
    width:375px;
    display:block;
    border: 1px solid #cc9966;
    height: 20px;
    box-shadow: 1px 1px 8px #0099CC;

}

.inputs:focus
{
     border: 2px solid #b1bcc7;
}

input[type="checkbox"]
{
    -webkit-appearance: none;
    background-color: #fafafa;
    border: 1px solid #b1bcc7;
    box-shadow: 1px 1px 8px #0099CC;
    padding: 9px;
    border-radius: 3px;
    display: inline-block;
    position: relative;
}

input[type="checkbox"]:disabled
{
   box-shadow: 1px 1px 8px gray;
   background: lightgray;
}

input[type="checkbox"] + label
{
    vertical-align: super;
    font-family: Book Antiqua;
    margin-left: 4px;
}

input[type="checkbox"]:active, input[type="checkbox"]:checked
{
    box-shadow: 1px 1px 8px orange;
}

input[type="checkbox"]:checked:after
{
    content: "\2714";
    font-size: 14px;
    position: absolute;
    top: 0px;
    left: 3px;
    color: orange;
}

input[type="radio"]
{
    -webkit-appearance: none;
    background-color: #fafafa;
    border: 1px solid #b1bcc7;
    box-shadow: 1px 1px 8px orange;
    padding: 9px; border-radius: 50px;
    display: inline-block;
    position: relative;
```

```
}

input[type="radio"] + label
{
    vertical-align: super;
    font-family: Book Antiqua;
    margin-left: 4px;
}

input[type="radio"]:checked
{
    border: 1px solid #b1bcc7;
    box-shadow: 1px 1px 8px orange;
}

input[type="radio"]:checked:after
{
    content: " ";
    width: 12px; height: 12px;
    border-radius: 50px;
    position: absolute;
    top: 3px; left: 3px; background: orange;
    box-shadow: 1px 1px 8px #0099CC;
    text-shadow: 0px;
    font-size: 32px;
}

input[type="radio"]:active, input[type="radio"]:checked:active
{
    box-shadow: 1px 1px 8px #0099CC;
}

input[type="radio"]:disabled
{
    box-shadow: 1px 1px 8px gray;
}
```

## Attachment 5. HTML code from the browser

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head><title>
   karma dot com
</title><link rel="stylesheet" type="text/css" href="Style.css" /><meta http-equiv="Content-Type"
content="text/html; charset=iso-8859-1" /></head>
<body bgcolor="#FFFFFF">
    <form method="post" action="buttons.aspx" id="form1">
<div class="aspNetHidden">
<input type="hidden" name="__VIEWSTATE" id="__VIEWSTATE"
value="/wEPDwUKMTE0MDM0NjYwMg9kFgICAw9kFgQCDQ8QZGQWAQICZAIRDxBkZBYCAgICA2QYAQUeX19Db250cm9sc1JlcXVpcmVQb3N0Q
mFja0tleV9fFgsFDFJhZGlvQnV0dG9uMQUMUmFkaW9CdXR0b24xBQxSYWRpb0J1dHRvbjIFDFJhZGlvQnV0dG9uMgUJQ2hlY2tCb3gxBQlDa
GVja0JveDIFD0NoZWNrQm94TGlzdDEkMAUPQ2hlY2tCb3hMaXN0MSQxBQ9DaGVja0JveExxc3QxJDIFD0NoZWNrQm94TGlzdDEkMwUPQ2hlY
2tCb3hMaXN0MSQzIm28GGnXDFPItphp+x9FiupS/FxF3c+u+py5VcoRxuw=" />
</div>

<div class="aspNetHidden">

   <input type="hidden" name="__EVENTVALIDATION" id="__EVENTVALIDATION"
value="/wEdAA+vK2MjRJpn3QelMVuS8pbiFVr6/W0Z4NrOqo0co6hqfyzEK45lA3svH8n3IOFu9v+2g/YEJ4dcN2xg003C94WjoRCr7j10B
NyMBx5ZdwO4h6ZjaI7GfrUUpG0b4t2bCmdohxz32fARXjg/IBk4m2RdYWDs2R2slY7Dys5fIRJdmFTBfebxm8pyH9XHbGTyqD696czEPkjQD
UaYXPe1HpyHlhAmT0ES7q40xVVD01hDVmUgvfSOipW6li3hh9stZcawmRNupW5HFY5PrwHDOUfeUT0qdyYrXk4KHcdtoNilhtK24ndwjjpRk
x6LzjheqiSLYgXOPel85THhLKgai0nZ8/AgG3bu1JVqqrCm6OoZJA==" />
```

```
</div>
<!-- Save for Web Slices (karma dot com.psd) -->
<table id="Table_01" width="800" border="0" cellpadding="0" cellspacing="0">
   <tr>
   <td>
   <img id="header_left" src="images/header_left.png" width="169" height="245" alt="header left" /></td>
   <td colspan="4">
   <a href="home.html">
   <img id="header_right" src="images/header_right.png" width="631" height="245" border="0" alt="header
right" /></a></td>
   </tr>
   <tr>
   <td>
   <img id="menu_blank" src="images/menu_blank.png" width="169" height="51" alt="menu" /></td>
   <td>
   <a href="home.html">
   <img id="menu_home" src="images/menu_home.png" width="169" height="51" border="0" alt="home" /></a></td>
   <td>
   <a href="menu.html">
   <img id="menu_menu" src="images/menu_menu.png" width="169" height="51" border="0" alt="menu" /></a></td>
   <td>
   <a href="news.html">
   <img id="menu_news" src="images/menu_news.png" width="169" height="51" border="0" alt="news" /></a></td>
   <td>
   <a href="about.html">
<img id="menu_about" src="images/menu_about.png" width="124" height="51" border="0" alt="about" /></a></td>
   </tr>
   <tr>
   <td>
   <img id="left_column" src="images/left_column.png" width="169" height="698" alt="left column" /></td>
   <td width="631" height="698" colspan="4" bgcolor="#FFFFFF" style="vertical-align:top; padding-left:10px;">
<h1 style="color: #3366ff; font-family: Book Antiqua">Radio buttons and check boxes</h1>
<br/>
<input id="RadioButton1" type="radio" name="RadioButton1" value="RadioButton1" /><label
for="RadioButton1">Regular radio button</label>
<br/>
<input id="RadioButton2" type="radio" name="RadioButton2" value="RadioButton2" /><label
for="RadioButton2">Regular radio button</label>
<br/>
<span class="aspNetDisabled"><input id="RadioButton3" type="radio" name="RadioButton3" value="RadioButton3"
checked="checked" disabled="disabled" /><label for="RadioButton3">Regular radio button checked,
disabled</label></span>
<br/><br/>
<input id="CheckBox1" type="checkbox" name="CheckBox1" /><label for="CheckBox1">Regular check box</label>
<br/>
<input id="CheckBox2" type="checkbox" name="CheckBox2" /><label for="CheckBox2">Regular check box</label>
<br /><br />
<span id="Label1" class="labels">Radio button list</span>
<table id="RadioButtonList1">
   <tr>
   <td><span class="aspNetDisabled"><input id="RadioButtonList1_0" type="radio" name="RadioButtonList1"
value="Disabled, Not selected" disabled="disabled" /><label for="RadioButtonList1_0">Disabled, Not
selected</label></span></td>
   </tr><tr>
   <td><span><input id="RadioButtonList1_1" type="radio" name="RadioButtonList1" value="Not selected"
/><label for="RadioButtonList1_1">Not selected</label></span></td>
   </tr><tr>
   <td><span class="aspNetDisabled"><input id="RadioButtonList1_2" type="radio" name="RadioButtonList1"
value="Disabled, Selected" checked="checked" disabled="disabled" /><label for="RadioButtonList1_2">Disabled,
Selected</label></span></td>
   </tr><tr>
   <td><span><input id="RadioButtonList1_3" type="radio" name="RadioButtonList1" value="Not Selected"
/><label for="RadioButtonList1_3">Not Selected</label></span></td>
   </tr>
</table>
<br />
<span id="Label2" class="labels">Check box list</span>
```

```
<table id="CheckBoxList1">
   <tr>
   <td><span class="aspNetDisabled"><input id="CheckBoxList1_0" type="checkbox" name="CheckBoxList1$0"
disabled="disabled" value="Disabled, Not selected" /><label for="CheckBoxList1_0">Disabled, Not
selected</label></span></td>
   </tr><tr>
   <td><span><input id="CheckBoxList1_1" type="checkbox" name="CheckBoxList1$1" value="Not selected" /><label
for="CheckBoxList1_1">Not selected</label></span></td>
   </tr><tr>
   <td><span class="aspNetDisabled"><input id="CheckBoxList1_2" type="checkbox" name="CheckBoxList1$2"
checked="checked" disabled="disabled" value="Disabled, Selected" /><label for="CheckBoxList1_2">Disabled,
Selected</label></span></td>
   </tr><tr>
   <td><span><input id="CheckBoxList1_3" type="checkbox" name="CheckBoxList1$3" checked="checked"
value="Selected" /><label for="CheckBoxList1_3">Selected</label></span></td>
   </tr>
</table>
</td>
   </tr>
   <tr>
   <td colspan="5">
   <img id="bottom_bar" src="images/bottom_bar.png" width="800" height="33" alt="bottom bar" /></td>
   </tr>
</table>
<!-- End Save for Web Slices -->
</form>
</body>
</html>
```