THE EVOLUTION OF WEB DEVELOPMENT

The Web has now existed for almost three decades. In that time, the way websites look and work has changed dramatically. The way people *create* websites has also evolved. Today web pages can be written by hand (perhaps with the help of a design tool such as Adobe Dreamweaver), or they can be *programmed* using any one of a number of powerful platforms.

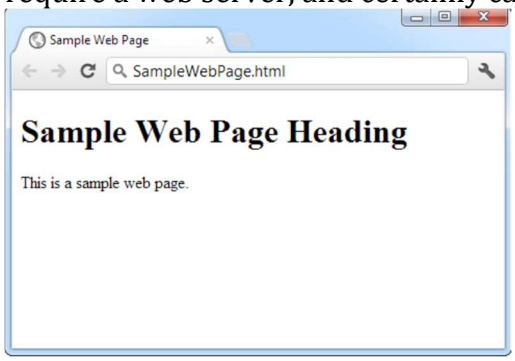
ASP.NET is Microsoft's web programming toolkit. It's a part of .NET, a cluster of technologies that are designed to help developers build a variety of applications. Developers write the code in one of several core .NET languages, such as C#, which is the language you'll use in this course.

Basic HTML

It would be difficult to describe early websites as web *applications*. Instead, the first generation of websites often looked more like brochures, consisting mostly of fixed HTML pages that needed to be updated by hand. A basic HTML page is a little like a word-processing document—it contains formatted content that can be displayed on your computer, but it doesn't actually *do* anything. The following example shows HTML at its simplest, with a document that contains a heading and a single line of text:

```
<!DOCTYPE html>
<html>
<head>
<title>Sample Web Page</title>
</head>
<body>
<h1>Sample Web Page Heading</h1>
<p>This is a sample web page.</p>
</body>
</html>
```

The Figure 1-1 below shows the simple HTML page in a browser. This is just a fixed file (named SampleWebPage.htm) that contains HTML content. It has no interactivity, doesn't require a web server, and certainly can't be considered a web application.

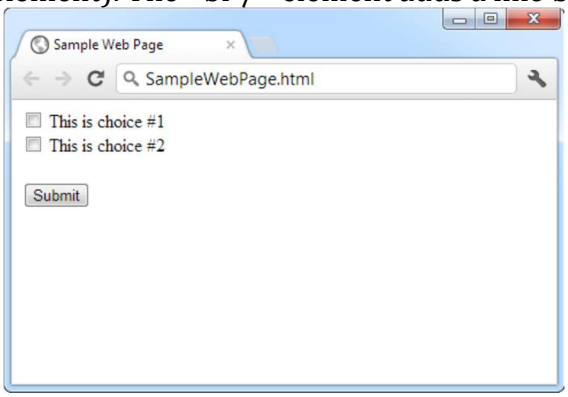


HTML Forms

HTML 2.0 introduced the first seed of web programming with a technology called *HTML forms*. HTML forms expand HTML so that it includes not only formatting tags but also tags for graphical widgets, or *controls*. These controls include common ingredients such as drop-down lists, text boxes, and buttons. Here's a sample web page created with HTML form controls:

```
<!DOCTYPE html>
<html>
<head>
<title>Sample Web Page</title>
</head>
<body>
<form>
<input type="checkbox" />
This is choice #1<br />
<input type="checkbox" />
This is choice #2<br /><br />
<input type="submit" value="Submit" />
</form>
</body>
</html>
```

In an HTML form, all controls are placed between the <form> and </form> tags. The preceding example includes two check boxes (represented by the ☐ element) and a button (represented by the <input type="submit"/> element). The
 element adds a line break between lines.



HTML forms allow web developers to design standard input pages. When the user clicks the Submit button on the page shown in the Figure all the data in the input controls (in this case, the two check boxes) is patched together into one long string of text and sent to the web server. On the server side, a custom application receives and processes the data. In other words, if the user selects a check box or enters some text, the application finds out about it after the form is submitted.

Amazingly enough, the controls that were created for HTML forms more than ten years ago are still the basic foundation that you'll use to build dynamic ASP.NET pages! The difference is the type of application that runs on the server side. In the past, when the user clicked a button on a form page, the information might have been e-mailed to a set account or sent to an application on the server that used the challenging Common Gateway Interface (CGI) standard. Today you'll work with the much more capable and elegant ASP.NET platform.

ASP.NET

Early web development platforms had two key problems. First, they didn't always scale well. As a result, popular websites would struggle to keep up with the demand of too many simultaneous users, eventually crashing or slowing to a crawl. Second, they provided little more than a bare-bones programming environment. If you wanted higher-level features, such as the ability to authenticate users or read a database, you needed to write pages of code from scratch. Building a web application this way was tedious and error-prone.

To counter these problems, Microsoft created higher-level development platforms—first ASP and then ASP.NET. These technologies allow developers to program dynamic web pages without worrying about the low-level implementation details. Even better, ASP.NET is stuffed full of sophisticated features, including tools for implementing security, managing data, storing user-specific information, and much more. And amazingly enough, it's even possible to program an ASP.NET page without knowing anything about HTML (although a little bit of HTML smarts will help you build your pages more quickly and effectively).

Server-Side and Client-Side Programming

ASP.NET is designed first and foremost as a *server-side* programming platform. That means that all ASP.NET code runs on the web server. When the ASP.NET code finishes running, the web server sends the user the final result—an ordinary HTML page that can be viewed in any browser.

Server-side programming isn't the only way to make an interactive web page. Another option is *client-side* programming, which asks the browser to download the code and execute it locally, on the client's computer. Just as there are a variety of server-side programming platforms, there are also various ways to perform client-side programming, from snippets of JavaScript code that can be embedded right inside the HTML of a web page, to plug-ins such as Adobe Flash and Microsoft Silverlight