

ASP.NET AJAX

Using C#

Student Guide

Revision 4.6

ASP.NET AJAX Using C#

Rev. 4.6

Student Guide

Information in this document is subject to change without notice. Companies, names and data used in examples herein are fictitious unless otherwise noted. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of Object Innovations.

Product and company names mentioned herein are the trademarks or registered trademarks of their respective owners.



™ is a trademark of Object Innovations.

Authors: Robert J. Oberg

Special Thanks: Marc Bueno, Ernani Cecon and Tony Ventura

Copyright ©2014 Object Innovations Enterprises, LLC All rights reserved.

Object Innovations
877-558-7246
www.objectinnovations.com

Published in the United States of America.

Table of Contents (Overview)

Chapter 1	Rich Internet Applications and AJAX
Chapter 2	Microsoft AJAX Client Library
Chapter 3	Partial Page Rendering
Chapter 4	Remote Method Calls
Chapter 5	AJAX Control Toolkit
Chapter 6	jQuery UI and JuiceUI
Appendix A	Learning Resources

Directory Structure

- **Install the course software by running the self-extractor *Install_AjaxCs_46.exe*.**
- **The course software installs to the root directory *C:\OIC\AjaxCs*.**
 - Example programs for each chapter are in named subdirectories of chapter directories **Chap01**, **Chap02** and so on.
 - The **Labs** directory contains one subdirectory for each lab, named after the lab number. Starter code is frequently supplied, and answers are provided in the chapter directories.
 - The **Demos** directory is provided for performing in-class demonstrations led by the instructor.
 - The **CaseStudy** directory contains an implementation of the AcmeBook case study, in several steps.
 - The **Images** directory contains copies of various image files.
- **Data files install to the directory *C:\OIC\Data*.**

Table of Contents (Detailed)

Chapter 1: Rich Internet Applications and AJAX	1
Desktop Applications.....	3
Web Applications.....	4
Plug-Ins	5
Client-Side Scripting.....	6
JavaScript Example.....	7
Script Code	8
JavaScript in ASP.NET.....	9
Dynamic Pages.....	10
Efficient Page Redraws.....	11
AJAX	12
Google Maps.....	13
ASP.NET AJAX	14
ASP.NET AJAX Enhancements	15
ASP.NET AJAX Control Toolkit.....	16
Demo – Accessing a Database.....	17
SQL Express LocalDB.....	18
An ASP.NET AJAX-Enabled Website.....	19
Summary	26
Chapter 2: Microsoft AJAX Client Library	27
Microsoft AJAX Components	29
AJAX Client Library	30
Using the Client Library	31
ScriptManager Control	32
Client Library Namespaces.....	33
Sys.Debug Tracing.....	34
AJAX Client Life Cycle Events.....	35
Enable Script Debugging	36
Demo: Tracing the Client Life Cycle	37
TraceConsole	41
Extending JavaScript Objects	42
Test Program for Array Extension.....	43
Test Program Code	44
Array Extensions in Client Library.....	45
Lab 2A	46
Object-Oriented JavaScript.....	47
Class.....	48
Person Class	49
Account Class	50
Test Code	51

Namespaces.....	52
Inheritance	53
Book Class	54
Initialization	55
Other Functions.....	56
User Interface.....	58
Top-Level Function	59
Shortcut Methods	60
Populating a List Box	61
JavaScript in Assemblies	62
Providing a ScriptReference	63
Demo: JavaScript in an Assembly	64
Enhanced AcmeClass Application.....	67
Lab 2B.....	68
Summary	69
Chapter 3: Partial Page Rendering.....	75
Partial Page Rendering.....	77
Partial Page Rendering Example	78
UpdatePanel Control.....	79
AJAX Extensions Controls	80
UpdatePanel Demo	81
Controlling Updates	83
Controlling Updates Example.....	84
Triggers	85
Types of Triggers.....	86
Server Initiated Updates.....	87
Server Update Example	88
A Challenge	89
Triggers Again	90
Timer Control.....	91
PageRequestManager Class.....	92
Customization Scenarios.....	93
PageRequestManager Example	94
PageRequestManger Event Handlers.....	95
UpdateProgress Control.....	97
UpdateProgress Example	98
Factors Code	99
Canceling the Async Postback.....	100
Limitations and Performance Issues	101
Lab 3	102
Summary	103
Chapter 4: Remote Method Calls.....	111
Why Remote Methods	113
Designing Remote Methods.....	114

A Web Service in an .asmx File.....	115
Demo: A Greeting Service.....	116
Demo: JavaScript Client.....	120
Registering AJAX Web Services.....	121
Calling AJAX Web Services.....	122
Running the Application.....	123
Handling Errors.....	124
Context.....	125
Modified Web Service.....	126
Using Context.....	127
Method Parameter.....	128
Using Method Parameter.....	129
Page Methods.....	130
JavaScript Object Notation.....	131
JSON Data Types.....	132
Comparing JSON and SOAP.....	133
SOAP Serialization.....	134
ScriptMethod Attribute.....	135
Lab 4A.....	136
Lab 4B.....	137
Summary.....	138
Chapter 5: AJAX Control Toolkit.....	149
AJAX Control Toolkit.....	151
Install ACT in Visual Studio.....	152
ACT Controls in Visual Studio.....	153
ACT Sample Web Site.....	154
Hello ACT.....	155
Intellisense for Properties.....	159
AjaxControlToolkit.dll.....	160
Registering AjaxControlToolkit.dll.....	161
Extender Controls.....	162
NumericUpDownExtender Control.....	163
Rating Control.....	164
Using a Style Sheet.....	165
MaskedEdit Control.....	167
PasswordStrength Control.....	169
PasswordStrength Example.....	170
Page Layout Controls.....	173
Tab Controls.....	174
Tabs Example Markup.....	175
Tab Examples Code.....	176
Accordion Control.....	177
CollapsiblePanel Control.....	178
CollapsiblePanel Markup.....	179
Popup Controls.....	180

Modal Popup Example Markup	181
Modal Popup Visual Effects	182
ModalPopup Example Code	183
ACT Controls and Web Services	184
DynamicPopulate	185
DynamicPopulate Example Markup	186
Lab 5	187
Summary	188
Chapter 6: jQuery UI and JuiceUI	203
jQuery Library	205
jQuery and Visual Studio	206
Selectors	207
jQuery Wrapper	208
jQuery Documentation and Download	209
jQuery Example	210
Running the Example	211
jQuery UI	212
jQuery UI Demo	213
Downloading jQuery UI	214
Copy Files to Script Folder	215
Markup and Code to Enter a Date	216
Using the Datepicker	217
Using a Style Sheet	219
JuiceUI	220
JuiceUI Demo	221
Obtaining JuiceUI via NuGet	223
Add JuiceUI to the Toolbox	224
Add a JuiceUI DatePicker	225
Setting Properties	227
Client-Side Interaction	228
Summary	229
Appendix A: Learning Resources.....	231

Chapter 1

Rich Internet Applications and AJAX

EVALUATION COPY
Unauthorized reproduction or distribution is prohibited.

Rich Internet Applications and AJAX

Objectives

After completing this unit you will be able to:

- **Position rich Internet applications in the spectrum of desktop and Web applications.**
- **Describe the approach of using plug-ins to implement rich-client applications and discuss the advantages and disadvantages of this approach.**
- **Describe the use of JavaScript to implement rich client applications.**
- **Discuss the importance of asynchronous communication with the server in providing a responsive user interface.**
- **Describe AJAX, or Asynchronous JavaScript and XML.**
- **Outline Microsoft's AJAX technologies and set up a test bed for ASP.NET AJAX development.**
- **Implement a simple AJAX application.**

Desktop Applications

- **There are two kinds of desktop applications:**
 - Standalone personal productivity applications running on a PC.
 - Networked applications in which the desktop PC is a client for code running on a server, possibly through multiple tiers.
- **Advantages:**
 - A rich user interface experience leveraging the advanced pointing device and graphics capabilities of the PC operating system.
 - Ability to run disconnected from a network and to make use of local computer resources, such as hard disk storage.
- **Disadvantages:**
 - The application is tied to a particular PC operating system and will not run in a different environment.
 - There is significant administrative overhead in installing desktop applications and in making sure they are kept up-to-date.
- **Also, there are configuration issues that the individual user has to cope with.**
 - Have you had a Windows application start acting strangely, perhaps because some Registry or other configuration information has become corrupted?

Web Applications

- **Web applications are accessible through virtually any browser running on any PC operating system.**
- **A Web application is trivial to deploy, as it just needs to be posted on a website, and any user can run it merely by pointing the browser to a certain URL.**
- **But traditionally Web applications have a much more primitive user interface, with much less interactivity with the user.**
- **Typical user interactions are quite simple:**
 - A page may simply display some information, with links to go to other pages.
 - A page may have a simple form which the user can fill out and submit.
 - The response is another page.

Plug-Ins

- **The user interface provided by standard browsers is fairly limited, and early on browsers began implementing various plug-in technologies.**
 - A plug-in is a piece of code that downloads from the server and augments the functionality of the browser.
- **Classic examples of plug-ins:**
 - Netscape Navigator plug-ins
 - Microsoft ActiveX controls
 - Java applets
- **Modern examples of the plug-in approach:**
 - Flash
 - Microsoft Silverlight¹
 - JavaFX from Sun

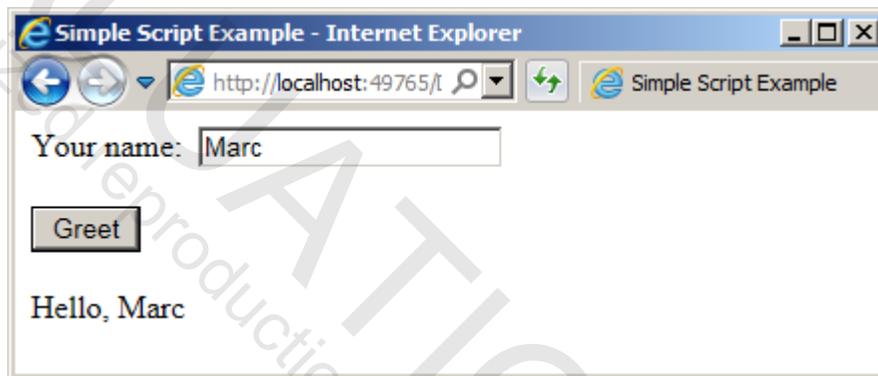
¹ Actually, Silverlight is now more than a plug-in. It can enable a full-fledged out-of-browser application. For detailed information about Silverlight, see Object Innovations' four-day course 4146, Silverlight 5 Using C#.

Client-Side Scripting

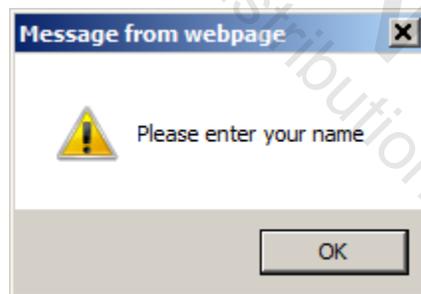
- **Another approach to enhancing Web applications is client-side scripting.**
- **HTML provides a `<script>` tag, which can be used to specify code that will execute in response to events in the browser.**
- **There are various scripting languages that can be used, but by far the most popular is *JavaScript*.**
 - JavaScript was originally introduced by Netscape.
 - Despite its name and syntax, it has nothing to do with Java!
 - Microsoft's version of JavaScript is called JScript.
- **JavaScript has been standardized by ECMA; the standard version is known as *ECMAScript*.**
 - ECMAScript Edition 5 has been approved.
 - Active work continues. See:
<http://www.ecmascript-lang.org/>
- **A scripting language for Internet Explorer is *VBScript*.**
 - But VBScript is only available on Microsoft platforms and so does not have the reach of JavaScript.

JavaScript Example

- We will use JavaScript extensively in this course, so let's look at a simple example right away.
 - In Visual Studio 2013 open the website **SimpleScript** in **C:\OIC\AjaxCs\Chap01**.
 - Run the program from within Visual Studio.



- Now try leaving the “Your name” field blank. You will see an error message displayed in a message box.



- Note that this validation occurs *completely on the client*, with no postback to the server involved.

Script Code

```
<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head><title>
  Simple Script Example
</title>
<script type="text/javascript">

  function btnGreet_onclick() {
    var name =
      document.getElementById("txtName").value;
    if (name.length == 0) {
      alert("Please enter your name");
      return false;
    }
    else
      return true;
  }

</script>
</head>
<body>
  <form method="post" action="Default.aspx"
  id="form1" onsubmit="return btnGreet_onclick();">
  ...
```

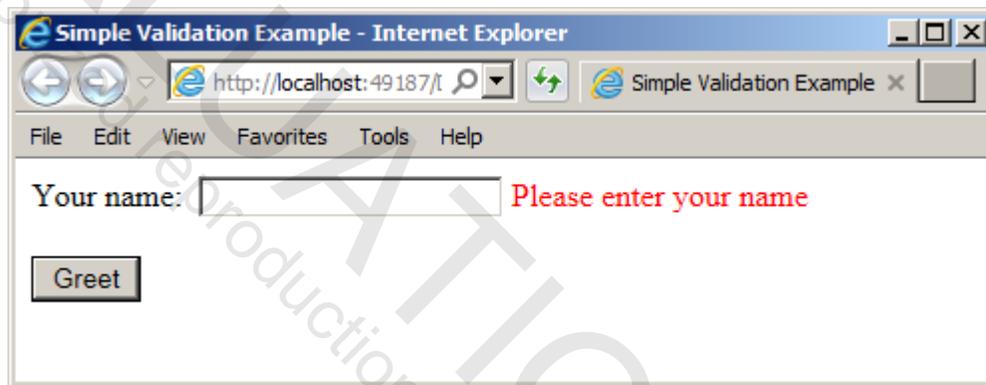
- Since the **btnGreet_onclick()** function returns **false** when the name has length zero, the form will **not** be submitted to the server.

- **We assume you have a basic knowledge of JavaScript.**

- See Object Innovations' course 3201, Introduction to JavaScript and jQuery, for a one-day introduction.

JavaScript in ASP.NET

- **ASP.NET uses JavaScript under the hood a great deal in order to achieve many of its effects.**
- **As an example, consider the same program using the *RequiredFieldValidator* server control**
 - See **SimpleValidation** in the **Chap01** directory.



- You may examine the generated HTML/JavaScript using View | Source in Internet Explorer from the menu bar (which is not shown by default).

...

```
function ValidatorOnSubmit() {  
    if (Page_ValidationActive) {  
        return ValidatorCommonOnSubmit();  
    }  
    else {  
        return true;  
    }  
}
```

Dynamic Pages

- **The original Web primarily served static and non-modifiable HTML pages.**
 - These pages were primarily text with a little graphics, but as bandwidth increased Web pages became more complex, with far more graphics, streaming audio, and video.
- **Beyond serving static pages, Web applications were developed that allowed the server to create pages on the fly.**
 - Popular server-side technologies included CGI (common gateway interface), ASP (Active Server Pages), JSP (JavaServer Pages) and PHP and so on.
- **In the late 1990s Dynamic HTML (DHTML) was introduced by the browser vendors.**
 - The World Wide Web Consortium (W3C) standardized a Document Object Model (DOM) for page content.
 - This means that browsers can expose the whole content of a page through a read/write object model.
- **Hence, it is now feasible to dynamically alter the appearance of a Web page through client-side code.**
- **Another modern aspect of Web page development is the use of HTML 5, which promotes stricter syntax than traditional HTML.**
 - Visual Studio will generate HTML 5 compliant pages for us.

Efficient Page Redraws

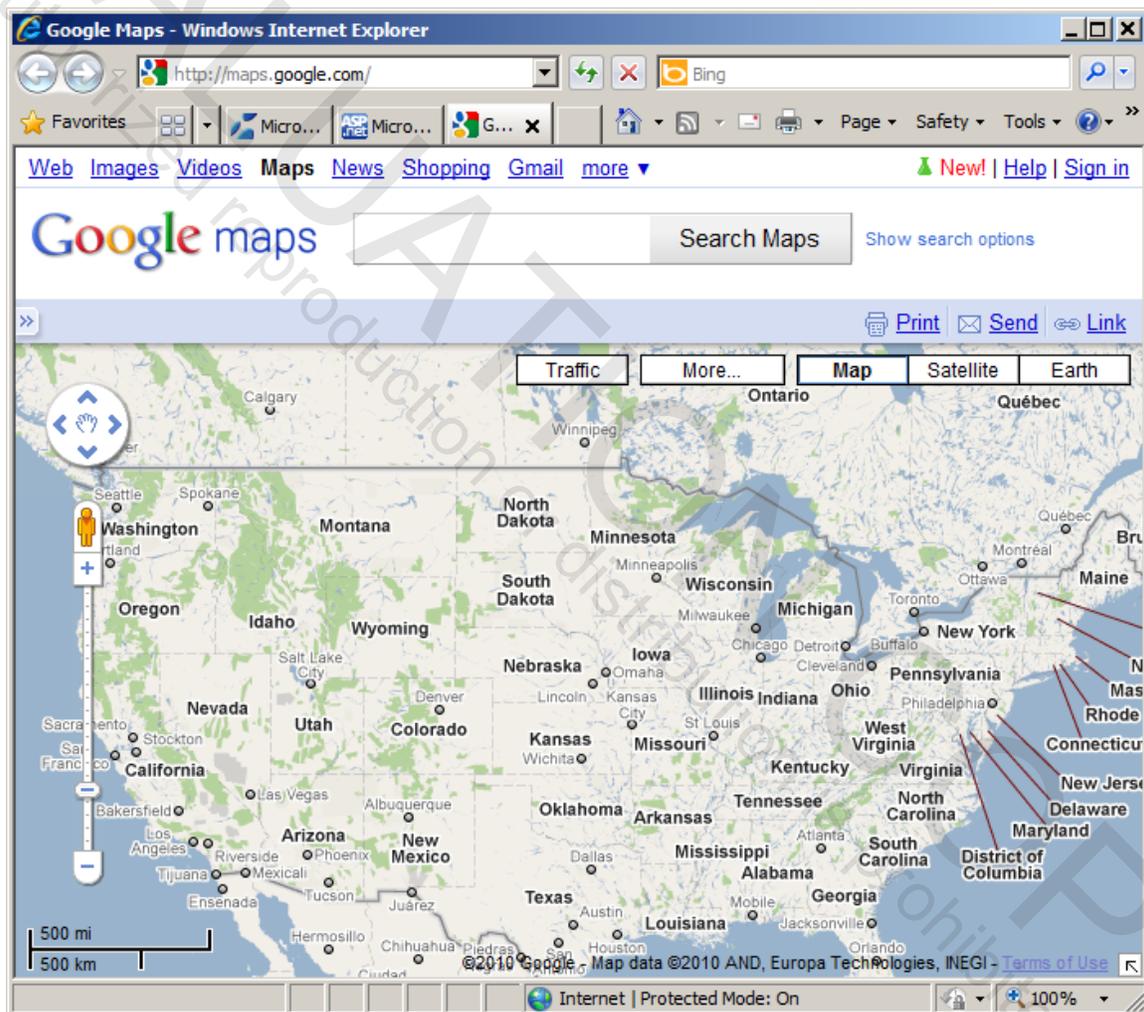
- **The normal way a page is updated is by bringing a whole new page down from the server.**
 - This works fine for simple text and a little graphics.
 - But what about complex pages with significant graphics, elaborate data in tables, and so on?
- **A more efficient approach is to bring down only the data that changes, and let the client dynamically update the page through DHTML/DOM.**
- **Technology to do this, called *Remote Scripting (RS)* was introduced by Microsoft in 1997.**
 - RS originally used a Java applet and later an ActiveX object `XMLHttpRequest`.
- **Modern browsers implement the *XMLHttpRequest* object directly, providing a uniform approach to obtaining selected data from the server to facilitate partial page redraws.**
 - Mozilla has always used `XMLHttpRequest`.
 - Internet Explorer before 7.0 used an ActiveX object, but from 7.0 on implements `XMLHttpRequest` directly.

AJAX

- **Although remote scripting has been around for a long time, it became popular in 2005 with the introduction of the acronym *AJAX*.**
 - Originally introduced in the Java community, AJAX is now popular in .NET, PHP, and other platforms.
- **AJAX stands for *Asynchronous JavaScript and XML*.**
 - The asynchronous nature of the request to the server means that the user can continue to interact with the browser-based application while waiting for the data.
 - Despite the XML in the name, data sent to and from the server does not have to be XML documents.
- **Remote scripting enables out-of-band HTTP requests.**
 - The normal mechanism for submitting HTTP requests results in the return of an entire page.
 - An out-of-band request can be tailored to what is most suitable for the task at hand, and will normally call for much less data to be sent.
 - An AJAX framework provides a **proxy** that can be called from JavaScript code to issue the out-of-band HTTP requests.
- **The other key ingredient that makes the technology work is an updatable DOM.**
 - Client code does a partial update of the page.

Google Maps

- **A killer application for AJAX was Google maps.**
 - The client-side functionality has tremendous performance, zooming in and out, moving the map about, and so on.
 - Out-of-band HTTP requests bring back only the needed data.



ASP.NET AJAX

- **Although it is quite feasible to create AJAX-enabled Web applications without any special tools, it is much easier with an AJAX framework.**
 - ASP.NET AJAX is a free framework from Microsoft that greatly simplifies creating AJAX applications using ASP.NET and Visual Studio 2013.
 - It is bundled into the .NET 4.5.1 release of ASP.NET with Visual Studio 2013, and it is also available as a separate download for various versions of ASP.NET.
 - ASP.NET AJAX provides both server-side and client-side services, as well as integration with Visual Studio 2013.
- **Some of the key features include:**
 - Performing significant parts of a Web page's processing in the browser
 - Controls to simplify AJAX programming
 - Partial-page updates
 - Client integration with forms authentication and user profiles.
 - Calls to Web Services from client script
- **Visual Studio enhancements include:**
 - A new group AJAX Extensions in the Toolbox, such as ScriptManager and UpdatePanel controls.

ASP.NET AJAX Enhancements

- **With .NET 4.5.1 there are a number of important enhancements for AJAX applications.**
 - Website: <http://www.asp.net/ajax>.

ASP.NET Ajax: Enhanced Interactivity and Responsiveness



Ajax Control Toolkit

Add AJAX functionality by doing more with JavaScript



jQuery

Add Ajax functionality to your ASP.NET application



Juice UI

Supercharge your website without writing JavaScript



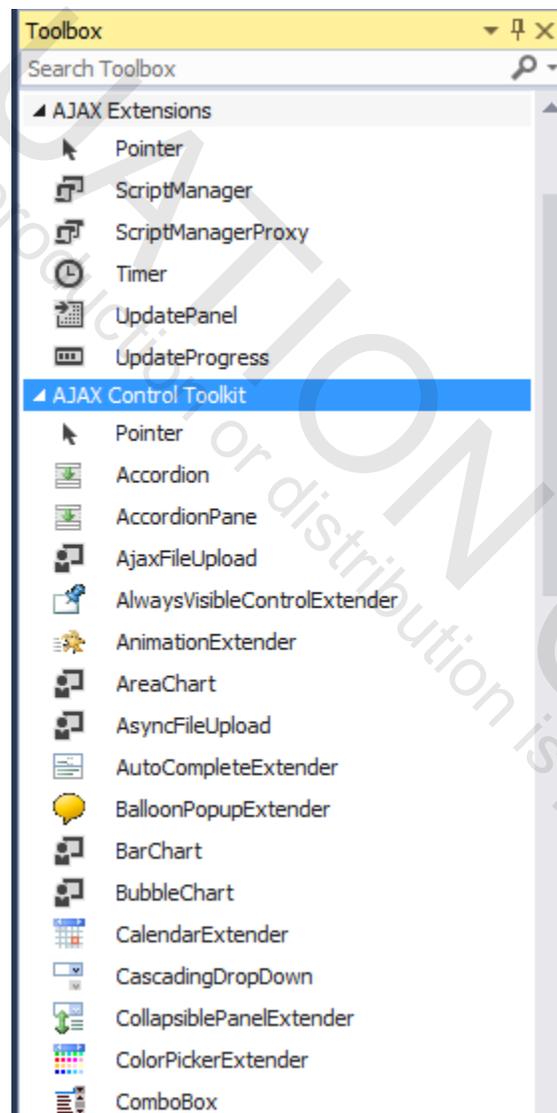
Microsoft Ajax CDN

Improve the performance of your Ajax applications

- The AJAX Control Toolkit adds many more controls for AJAX applications.
- jQuery is a very popular open source JavaScript library that now comes bundled with ASP.NET 4.5.
- Juice UI is an open-source collection of WebForms components that brings jQuery UI Widgets to your project. This enables you to leverage the power of jQuery while working with familiar code in your ASP.NET projects.
- The Microsoft AJAX CDN (Content Delivery Network) caches popular JavaScript libraries on servers around the world. The result is a significant improvement in the performance of JavaScript applications.

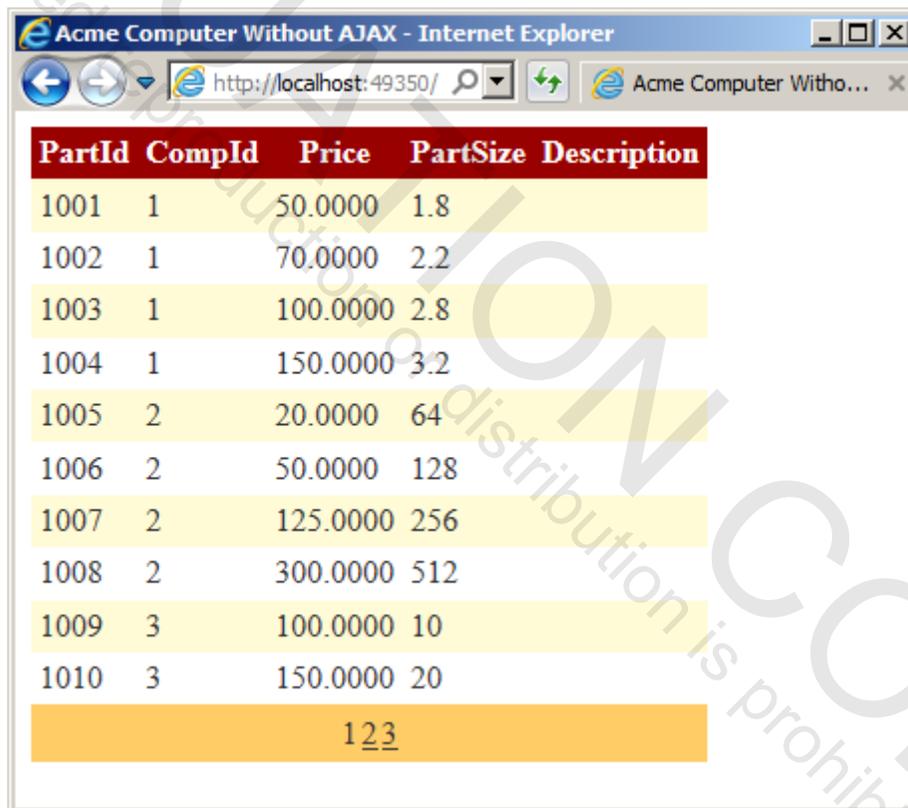
ASP.NET AJAX Control Toolkit

- A key piece of Microsoft's AJAX technology is the **ASP.NET AJAX Control Toolkit**.
 - Download: <http://www.codeplex.com/AjaxControlToolkit>.
- This will install a larger number of useful controls in the Visual Studio Toolbox. See Chapter 5.



Demo – Accessing a Database

- As a demonstration of using ASP.NET AJAX let's create an AJAX enabled website to access the AcmeComputer database.
 - This database is provided as a file **AcmeComputer.mdf** in the **OIC\Data** directory, so there is no database setup.
 - The chapter directory contains a website **AcmeNoAjax** to access the database without use of AJAX.



The screenshot shows a web browser window titled "Acme Computer Without AJAX - Internet Explorer". The address bar displays "http://localhost:49350/". The main content area contains a table with the following data:

PartId	CompId	Price	PartSize	Description
1001	1	50.0000	1.8	
1002	1	70.0000	2.2	
1003	1	100.0000	2.8	
1004	1	150.0000	3.2	
1005	2	20.0000	64	
1006	2	50.0000	128	
1007	2	125.0000	256	
1008	2	300.0000	512	
1009	3	100.0000	10	
1010	3	150.0000	20	

Below the table, there is a yellow bar containing the text "1 2 3", which appears to be a pagination control.

SQL Express LocalDB

- **This course uses the LocalDB version of SQL Server 2012, which is installed automatically with Visual Studio 2013.**
- **LocalDB is an improved version of SQL Server 2012 Express intended for use by developers.**
 - It is easy to install and requires no management.
 - It provides the same API as full-blown SQL Server.
 - You can access a SQL Server 2013 database by specifying the filename of the database in your connection string.

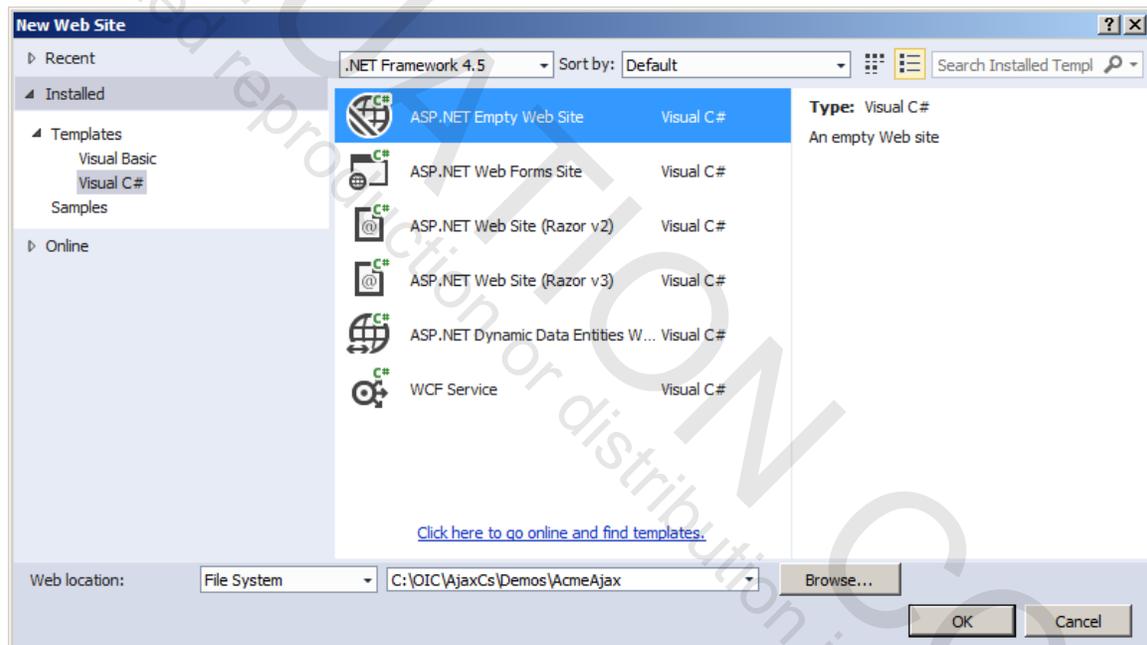
```
connectionString="Data Source=(LocalDB)\v11.0;  
  AttachDbFilename=C:\OIC\Data\AcmeComputer.mdf;  
  Integrated Security=True;Connect Timeout=30;"
```

- **LocalDB is only available for SQL Server 2012 databases.**
- **If you attempt to connect to an earlier version database file you will be given an opportunity to convert the database to SQL Server 2012.**
 - The database will then no longer be accessible to earlier versions of SQL Server.
- **LocalDB does not create any database services.**
 - A LocalDB process is created as a child process of the application that invokes it. It is stopped automatically a few minutes after the last connection is closed.

An ASP.NET AJAX-Enabled Website

- **Now let's create from scratch a version of this website that is ASP.NET AJAX enabled.**

1. In Visual Studio Express 2013 for Web² create a new website using File | New Web Site.
2. Choose ASP.NET Empty Web Site as the template, browse to the **Demos** directory, and assign the name **AcmeAjax** to your new website. Click OK.



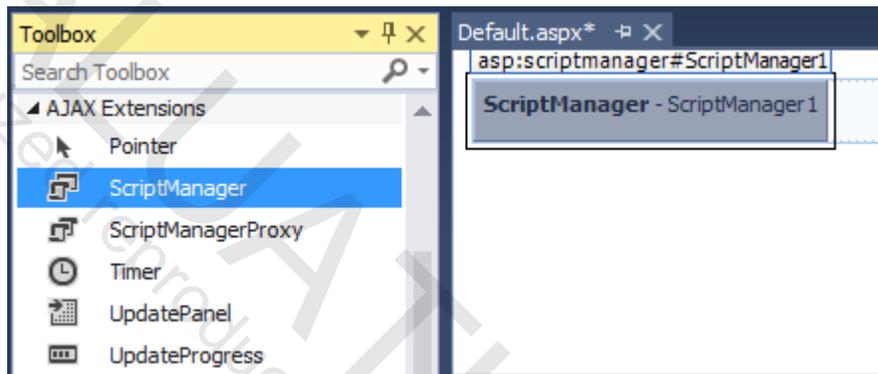
- An earlier version of ASP.NET AJAX had a special “ASP.NET AJAX-Enabled Web Site” template, but that is neither present nor needed with Visual Studio 2013.

3. In Solution Explorer, right-click over the website and choose Add | Web Form from the context menu.

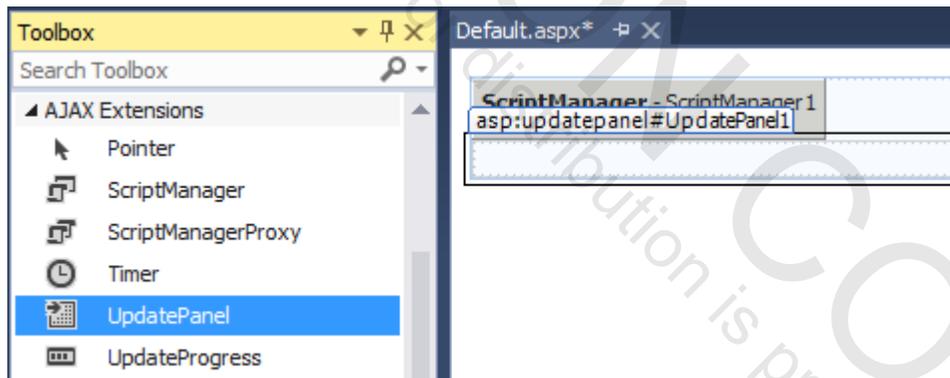
² Of course you could use a higher edition such as Visual Studio Professional 2013.

ASP.NET AJAX Demo (Cont'd)

4. In the Specify Name for Item dialog that comes up, accept the suggested name **Default.aspx**. This will automatically place code in a separate file **Default.aspx.cs**.
5. Switch to Design view and drag a ScriptManager control from the AJAX Extensions group onto the page³.



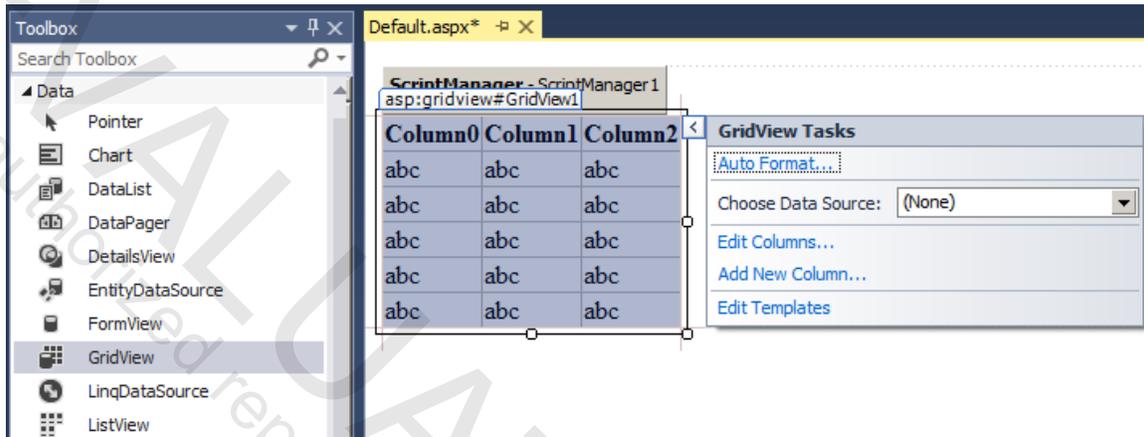
6. Drag an UpdatePanel control onto the page.



³ A ScriptManager control was automatically provided by the previous ASP.NET AJAX-Enabled Web Site template, the only distinguishing feature of this template.

ASP.NET AJAX Demo (Cont'd)

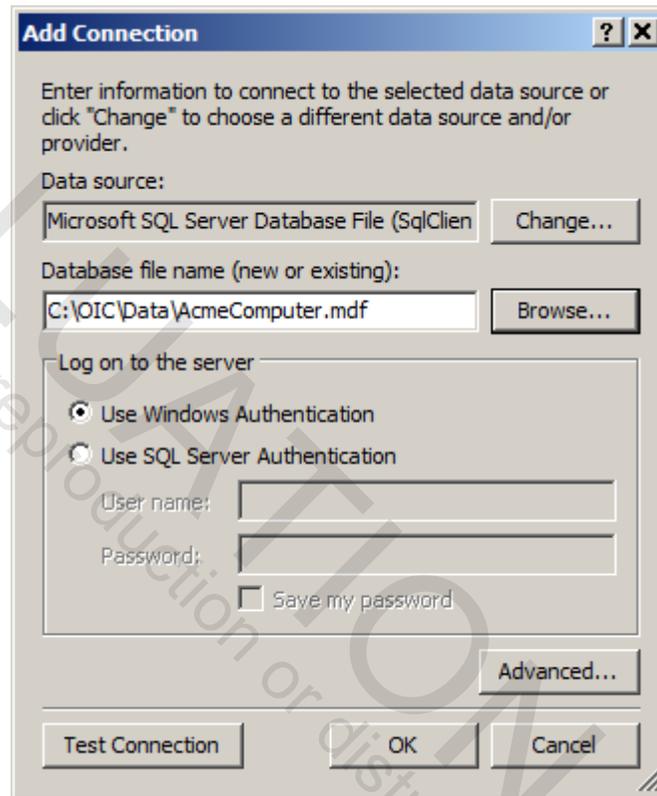
7. Drag a GridView control (from the Data group) into the UpdatePanel.



8. In the GridView tasks, select <New Data Source> from the Choose Data Source dropdown.
9. Choose Database in the Data Source Configuration Wizard. Click OK.
10. Click New Connection.
11. For Data Source select Microsoft SQL Server Database File.

ASP.NET AJAX Demo (Cont'd)

- Click Browse and navigate to **AcmeComputer.mdf** in the **C:\OIC\Data** folder.



- Test the connection if you wish, and then click OK. Back in the wizard, click Next.
- Say yes to saving the connection in the application configuration file. Click Next.

ASP.NET AJAX Demo (Cont'd)

15. Select the Part table and the first five columns from it.

Configure Data Source - SqlDataSource1

Configure the Select Statement

How would you like to retrieve data from your database?

Specify a custom SQL statement or stored procedure

Specify columns from a table or view

Name:

Part

Columns:

* RestockDate

PartId

CompId

Price

PartSize

Description

QtyOnHand

RestockQty

Return only unique rows

WHERE...

ORDER BY...

Advanced...

SELECT statement:

```
SELECT [PartId], [CompId], [Price], [PartSize], [Description] FROM [Part]
```

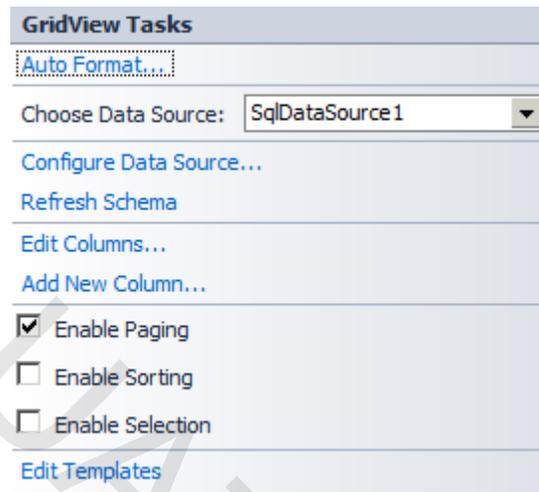
< Previous Next > Finish Cancel

16. Click Next.

17. Test the query, and click Finish.

ASP.NET AJAX Demo (Cont'd)

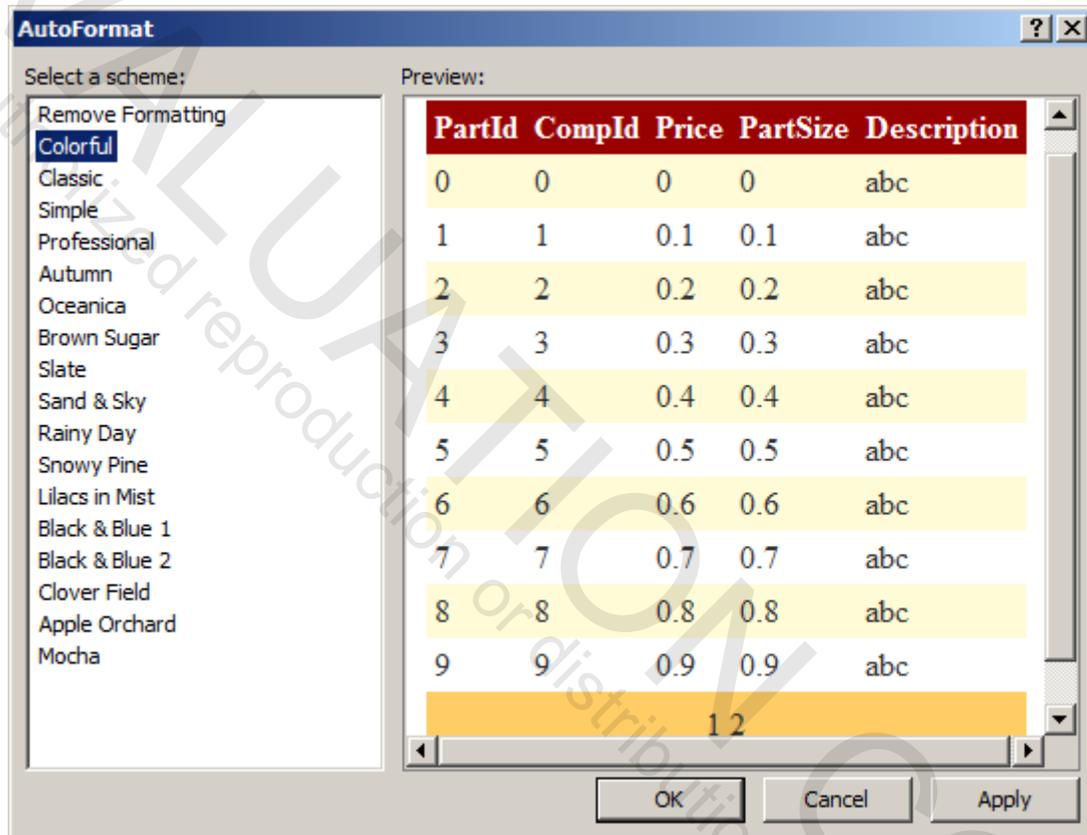
18. Back in GridView tasks select Enable Paging.



19. Build and run your new website. Try out paging. You should smoothly go from page to page without flicker.

ASP.NET AJAX Demo (Cont'd)

20. The GridView is a little drab. Spruce it up by opening the “smart tag”  and choose Auto Format.
21. Select Colorful as the scheme. Click OK.



22. In Source view specify a title of “Acme Computer With AJAX”.
23. Build and run. You should now have an application with the same UI as the previous **AcmeNoAjax**, but now we are doing partial page update, and there is no flicker.
24. Congratulate yourself on having built an AJAX-enabled website!

Summary

- **Rich Internet applications (RIAs) can provide some of the best features of desktop and Web applications.**
- **Plug-ins such as Flash and Microsoft's Silverlight are one approach to implementing RIAs, but they require running substantial code on the client.**
- **JavaScript, implemented in all modern browsers, is a lightweight approach to providing client-side functionality.**
- **Asynchronous communication with the server can provide a more responsive user interface.**
- **AJAX, or Asynchronous JavaScript and XML, is a term for technologies enabling highly interactive Web applications that give the user a better experience.**
- **Microsoft's ASP.NET AJAX provides a highly productive platform for implementing AJAX applications using Visual Studio.**