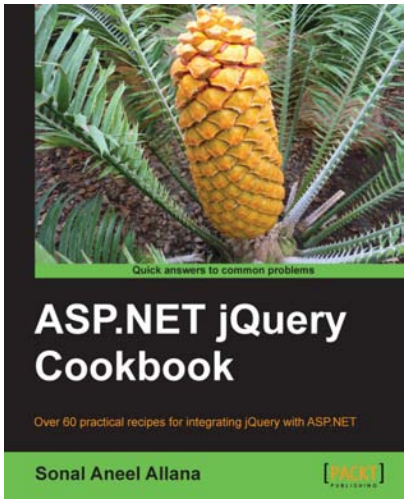
[PACKT] PUBLISHING

ASP.NET jQuery Cookbook

Sonal Aneel Allana



Chapter No.9
"Creating Rich Content in ASP.NET "

In this package, you will find:

A Biography of the author of the book

A preview chapter from the book, Chapter NO.9 "Creating Rich Content in ASP.NET "

A synopsis of the book's content

Information on where to buy this book

About the Author

Sonal Aneel Allana is a Senior Technical Consultant with Accentiv' SurfGold, Singapore. She has over a decade of experience in building web and desktop solutions in Microsoft technologies. She has worked with government agencies, credit bureaus, banks, financial institutions, and multinationals, delivering high-end customized solutions to meet their needs. She has worked on applications ranging from core banking software and B2B links to content management, workflow, and loyalty engines.

From time to time, she has also taken up the role of project lead and has carried numerous projects from conceptualization through implementation successfully. In addition to her experience in Microsoft technologies, she is well versed in J2EE and LAMP platforms.

She holds a Master's degree in Computer Science from the National University of Singapore and a Bachelor's degree in Computer Engineering from Mumbai University. She is also passionate about network and data security and has a certificate in Security Technology and Management from the Institute of Systems Science, National University of Singapore.

I would like to dedicate this book to my father for always believing in me and loving me unconditionally. Dad, I am sorry that you passed away before this book could become a reality.

Special thanks to my mother for her love, sacrifices, and encouragement.

My husband, Aneel, for his constant guidance, support, and patience. Without him this book would be impossible.

Thanks to Salim, Minaz, Rehan, and Yasmeen. There are many things I owe to you.

Thanks to the wonderful team at Packt Publishing for their hard work, dedication, and patience. Special thanks to Steven for being a constant guide throughout the process. Thanks to my reviewers for meticulously evaluating the content and for helping to make the book better.

ASP.NET jQuery Cookbook

The jQuery library has become increasingly popular with web application developers because of its simplicity and ease of use. The library is supported by an active community of developers and has grown significantly over the years after its inception in 2006 by John Resig. Using this library eases complicated tasks and adds to the interactive experience of the end user. Its extensible plugin architecture enables developers to build additional functionalities on top of the core library.

With Microsoft's contribution of Templates, DataLink, and Globalization plugins to the jQuery library and the distribution of jQuery with Visual Studio 2010 onwards, the library has gained popularity with ASP.NET developers. jQuery can be very easily interfaced with ASP.NET controls as well as custom user controls. It can be used to validate controls using client side scripts, thus giving us an alternative for server side Validation Controls. It can be used to incorporate cool animation effects as well as to create graphic-rich pages. It can be used to post AJAX requests to web services, page methods, and HTTP handlers. The possibilities with jQuery are endless.

This cookbook aims to cover some of the day-to-day tasks faced by ASP.NET developers and how jQuery can be applied to work out the same. We will be focusing on interfacing jQuery with ASP.NET applications and some of the amazing tasks that can be accomplished using this powerful library. The recipes described in this book give fast and easy solutions to some of the common encountered problems in web applications.

What This Book Covers

Chapter 1, *Working with ASP.NET Controls*, describes the interfacing of standard ASP.NET controls with the jQuery library. Controls such as TextBox, CheckBoxList, DropDownList, and Hyperlink are covered in this chapter.

Chapter 2, *Validation of ASP.NET Controls*, explains different client side validation techniques that can be used with jQuery for validation of ASP.NET controls. These validation techniques provide a substitute for server side Validation Controls.

Chapter 3, *Working with GridView Control*, explores different manipulations that can be applied to the ASP.NET GridView control. You will learn to highlight rows/cells, remove rows/ cells, use plugins, and apply animation effects using scripts.

Chapter 4, *Working with Image Control*, describes interesting applications of jQuery with the ASP.NET Image control. The recipes covered demonstrate handy utilities such as zooming, cropping, swapping, and changing the opacity of images. You will also learn to build a photo gallery using jQuery.

Chapter 5, *Animations in ASP.NET*, looks into various interesting animation effects that can be achieved using jQuery. You will learn to apply different types of animation effects to ASP. NET Label, Image, and Panel controls. Various effects such as fading, sliding, enlarging, and so on are covered in this chapter. The chapter also demonstrates animation chaining and prevention of animation queue buildup.

Chapter 6, *AJAX and ASP.NET (Part 1)*, demonstrates the making of AJAX calls using jQuery. The chapter describes three basic techniques of sending AJAX request to the server: via page methods, web services, and HTTP handlers. The chapter also demonstrates the use of the Firebug add-on in Mozilla Firefox browser to investigate the AJAX request/response.

Chapter 7, *AJAX and ASP.NET (Part 2)*, explores more advanced applications of jQuery AJAX. The recipes in this chapter show how to work with HTML, XML, script, JSON, JSONP, and complex data types such as objects using AJAX. You will also learn to build a search engine using Yahoo! Search API.

Chapter 8, *Client Templating in ASP.NET*, explains the use of Microsoft's jQuery Templates plugin in ASP.NET websites. The chapter covers both inline as well as external templates.

9

Creating Rich Content in ASP.NET

In this chapter, we will cover:

- Creating an accordion control
- Creating a tab control
- Creating draggable controls
- Creating dialog boxes
- Using the datepicker control
- Using the progress bar control
- Using the slider control
- Adding tooltips to controls

Introduction

Various jQuery plugins are available to add interactive features to ASP.NET web applications. A popular plugin is the jQuery UI, built on top of the jQuery library, and consisting of handy functionalities for creating rich content in web applications. In a nutshell, the jQuery UI library provides the following:

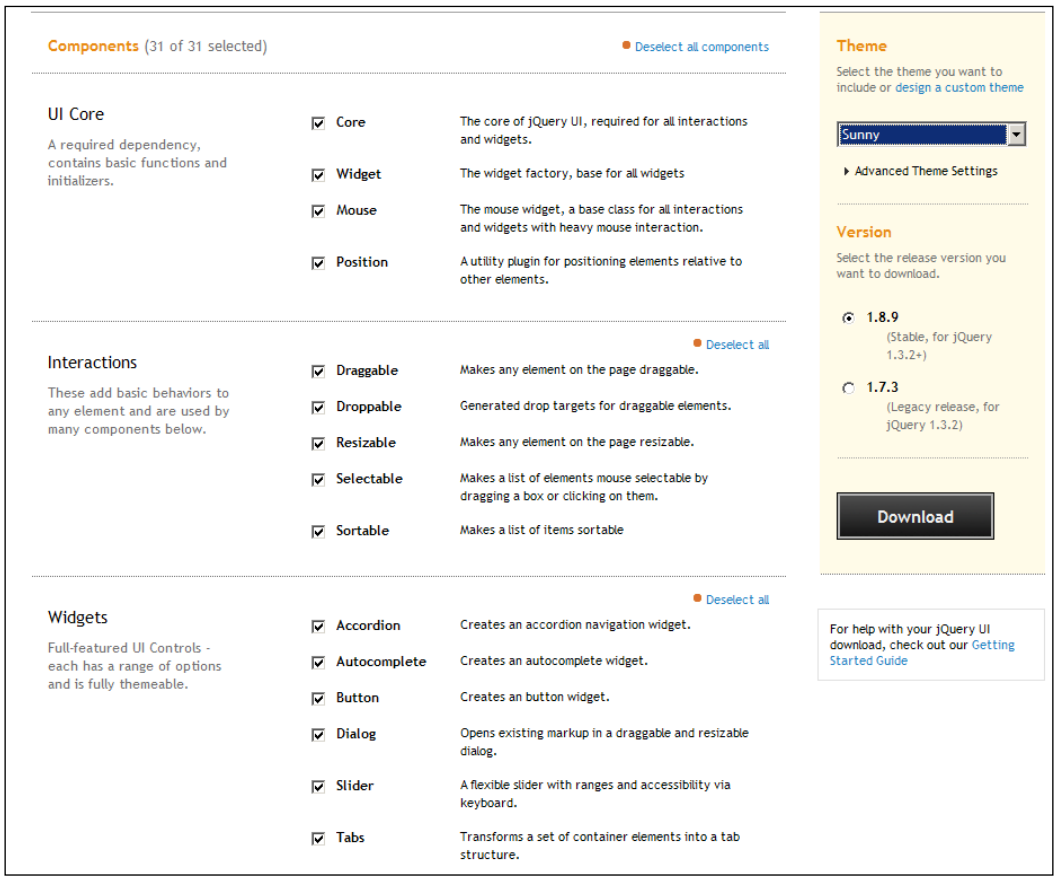
- Completely configurable widgets like accordion, tabs, progressbar, datepicker, slider, autocomplete, dialog, and button
- Interactive features like draggable, droppable, resizable, selectable, and sortable
- Advanced animation effects like show, hide, toggle, blind, bounce, explode, fade, highlight, pulsate, puff, scale, slide, etc
- Customisable themes to suit the look and feel of your website

In this article, we will primarily take a look at integration of jQuery UI with ASP.NET to build rich content quickly and easily.

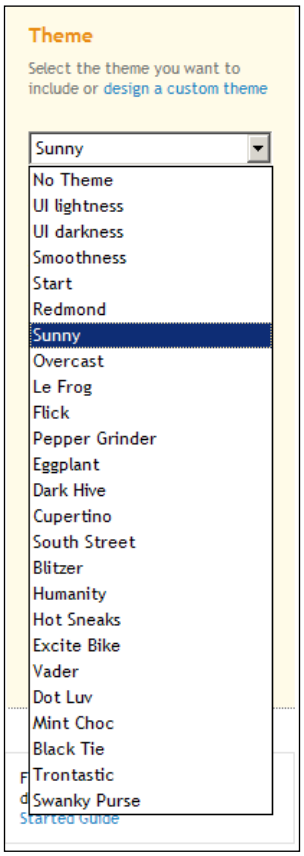
Getting started

1. Let's start by creating a new ASP.NET website `Chapter9` in Visual Studio.

   Go to the download page of jQuery UI at `http://jqueryui.com/download`, which allows customizable downloads based on the features required in the web application. For the purpose of this chapter, we will download the default build as shown next:



jQuery UI allows various custom themes. We will select the Sunny theme for our project:



2. Save the downloaded file. The download basically consists of the following:
   - `css` folder consisting of the the theme files
   - `development-bundle` folder consisting of demos, documents, raw script files, etc.
   - `js` folder consisting of the minified version of jQuery library and jQuery UI
3. Save the earlier mentioned css folder in the current project. Save the minified version of jQuery UI and jQuery library in a script folder `js` in the project.
4. In the recipes described in this chapter, in addition to including the jQuery library on ASP.NET pages, also include the UI library as shown next:

```
<script src="js/jquery-1.4.1.js" type="text/javascript"></script>
<script src="js/jquery-ui-1.8.9.custom.min.js"  type="text/
javascript"></script>
```

5. Include the downloaded theme on the aspx pages as follows:

```
<link href="css/sunny/jquery-ui-1.8.9.custom.css" rel="stylesheet"
type="text/css" />
```

Now let's move on to the recipes and explore some of the powerful functionalities of jQuery UI.

Creating an accordion control

The jQuery accordion widget allows the creation of collapsible panels on the page without the need for page refresh. Using an accordion control, a single panel is displayed at a time while the remaining panels are hidden.

Getting Ready

1. Create a new web form `Recipe1.aspx` in the current project.
2. Add a main content panel to the page. Within the main panel, add pairs of headers and subpanels as shown next:

```
<asp:Panel id="contentArea" runat="server">
<h3><a href="#">Section 1</a></h3>
<asp:Panel ID="Panel1" runat="server">
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do
eiusmod tempor incididunt ut labore et dolore magna aliqua.
</asp:Panel>
<h3><a href="#">Section 2</a></h3>
<asp:Panel ID="Panel2" runat="server">
Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris
nisi ut aliquip ex ea commodo consequat.
</asp:Panel>
<h3><a href="#">Section 3</a></h3>
<asp:Panel ID="Panel3" runat="server">
Duis aute irure dolor in reprehenderit in voluptate velit esse
cillum dolore eu fugiat nulla pariatur.
</asp:Panel>
<h3><a href="#">Section 4</a></h3>
<asp:Panel ID="Panel4" runat="server">
Excepteur sint occaecat cupidatat non proident, sunt in culpa qui
officia deserunt mollit anim id est laborum
</asp:Panel>
</asp:Panel>
```

3. Add some styling to the main content panel as required:

```
#contentArea
{
```

```
        width: 300px;
        height: 100%;
    }
```

Our accordion markup is now ready. We will now transform this markup into an accordion control using the functionalities of jQuery UI.

How to do it...

1. In the `document.ready()` function of the jQuery script block, apply the `accordion()` method to the main content panel:

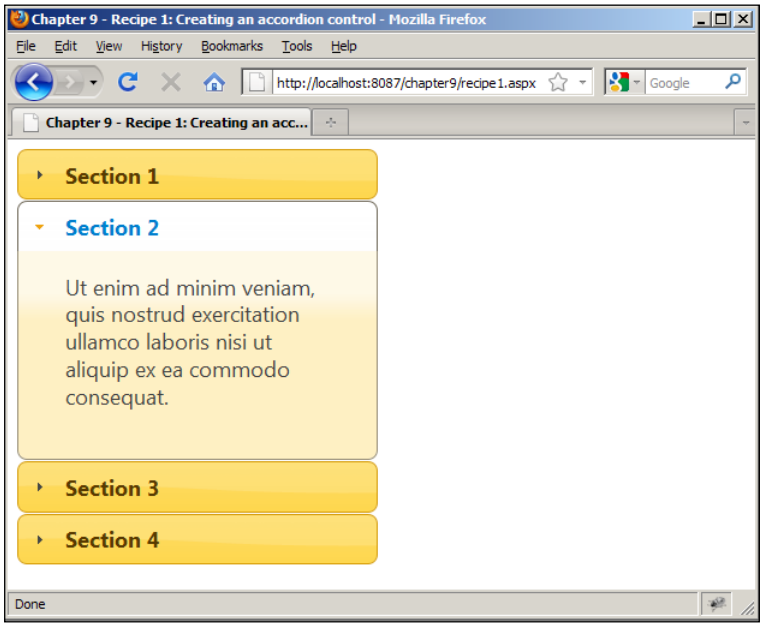
```
$("#contentArea").accordion();
```

Thus, the complete jQuery UI solution for the problem at hand is as follows:

```
<script language="javascript" type="text/javascript">
    $(document).ready(function(){
            $("#contentArea").accordion();
    });
</script>
```

How it works...

Run the web page. The accordion control is displayed as shown in the following screenshot:



Click on the respective panel headers to display the required panels. Note that the accordion control only displays the active panel at a time. The remaining panels are hidden from the user.

There's more...

For detailed documentation on the jQuery UI accordion widget, please visit `http://jqueryui.com/demos/accordion/`.

See also

Creating a tab control

Creating a tab control

The jQuery UI tab widget helps to create tab controls quickly and easily on ASP.NET web pages. The tab control helps in organizing content on a page thus improving the presentation of bulky content. With the help of jQuery UI tab widget, the content can also be retrieved dynamically using AJAX.

In this recipe, we will see a simple example of applying this powerful widget to ASP.NET forms.

Getting Ready

1. Create a new web form `Recipe2.aspx` in the current project.
2. Add an ASP.NET container panel to the page.
3. Within this container panel, add subpanels corresponding to the tab contents. Also add hyperlinks to each of the subpanels.

Thus the complete aspx markup of the web form is as shown next:

```
<form id="form1" runat="server">
   <asp:panel id="contentArea" runat="server">
   <ul>
      <li><a href="#tab1">Tab 1</a></li>
      <li><a href="#tab2">Tab 2</a></li>
      <li><a href="#tab3">Tab 3</a></li>
      <li><a href="#tab4">Tab 4</a></li>
   </ul>
```

```
   <asp:panel ID="tab1" runat="server">
      <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed
do eiusmod tempor incididunt ut labore et dolore magna aliqua.</p>
   </asp:panel>
   <asp:panel ID="tab2" runat="server">
      <p>Ut enim ad minim veniam, quis nostrud exercitation ullamco
laboris nisi ut aliquip ex ea commodo consequat.</p>
   </asp:panel>
   <asp:panel ID="tab3" runat="server">
      <p>Duis aute irure dolor in reprehenderit in voluptate velit
esse cillum dolore eu fugiat nulla pariatur. </p>
   </asp:panel>
   <asp:panel ID="tab4" runat="server">
      <p>Excepteur sint occaecat cupidatat non proident, sunt in culpa
qui officia deserunt mollit anim id est laborum</p>
   </asp:panel>
</asp:panel>
    </form>
```

Next, we will see how we can transform this markup into a tab control using jQuery UI.

How to do it...

1. In the `document.ready()` function of the jQuery script block, apply the `tabs()` method to the container panel as follows:

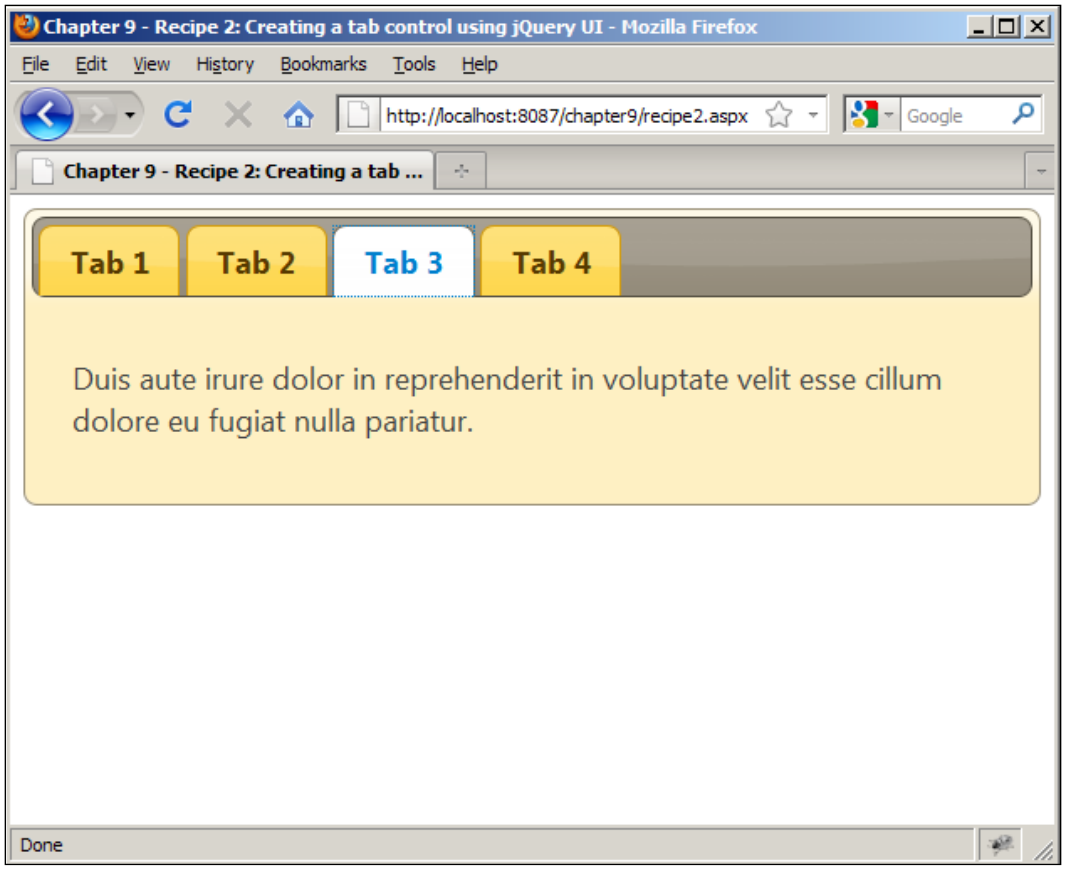
```
$("#contentArea").tabs();
```

Thus, the complete jQuery UI solution for creating the tab control is as follows:

```
<script language="javascript" type="text/javascript">
    $(document).ready(function(){
       $("#contentArea").tabs();
       });
</script>
```

How it works...

Run the web form. The page displays the tabbed content as follows:



Click on the respective tab headers to view the required content.

There's more...

For detailed documentation on the jQuery UI tabs widget, please visit

`http://jqueryui.com/demos/tabs/`.

See also

Creating an accordion control

Creating draggable controls

With the help of jQuery UI, web controls can be enhanced and interactive properties such as draggable, droppable, resizable, sortable, and selectable can be added. In this recipe, let's create some draggable ASP.NET panels on the web form.

Getting Ready

1. Create a new web form `Recipe3.aspx` in the current project.
2. Add a container div area on the web form as follows:

```
<div align="center" id="contentArea">
</div>
```

   Add the following css style to the above div area:

```
#contentArea
    {
      width: 300px;
      height: 100%;
    }
```

3. Within the container div area, add few panel controls. Create the following css class for these panel controls:

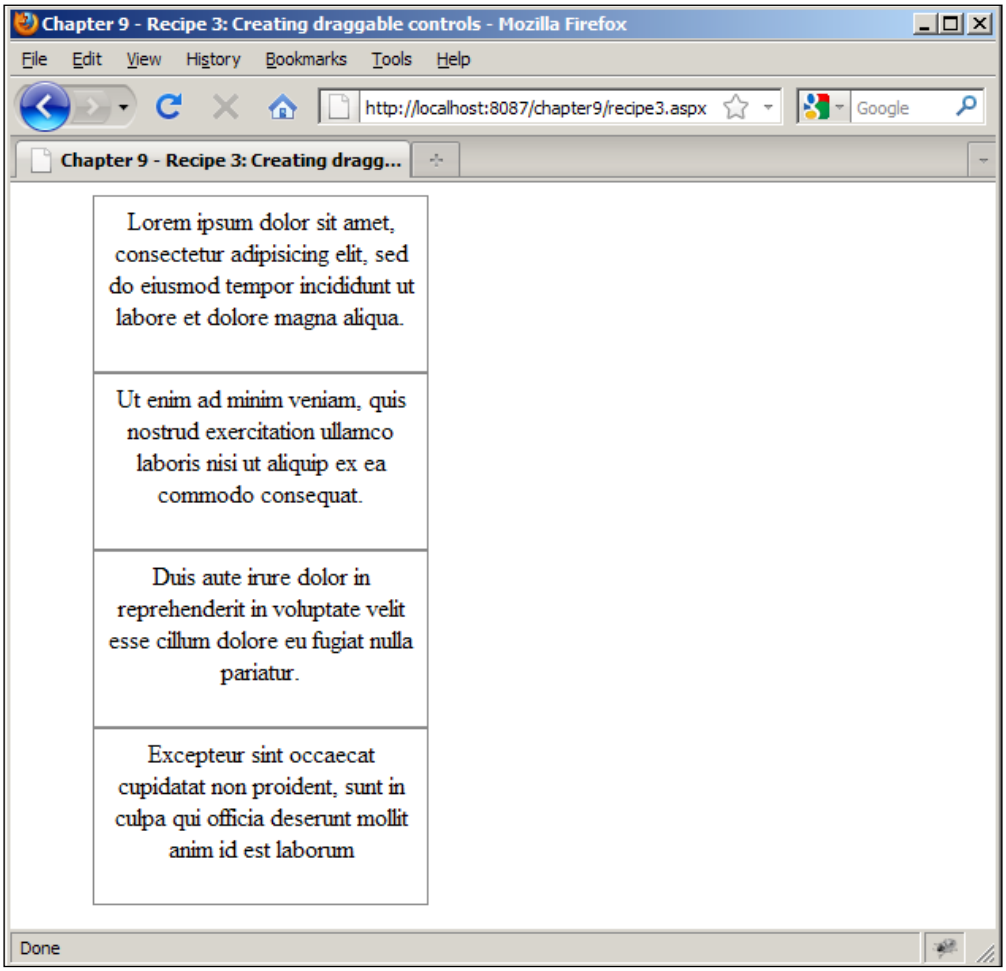
```
.panel
{
  border:1px solid;
  border-color:Gray;
  width:200px;
  height:100px;
  padding:5px;
}
```

4. Apply the above class to the panels, so that the complete aspx markup of the form is as shown next:

```
<form id="form1" runat="server">
<div align="center" id="contentArea">
<asp:Panel ID="Panel1" runat="server" CssClass="panel">
          Lorem ipsum dolor sit amet, consectetur adipisicing
elit, sed do eiusmod tempor incididunt ut labore et dolore magna
aliqua.
</asp:Panel>
<asp:Panel ID="Panel2" runat="server" CssClass="panel">
       Ut enim ad minim veniam, quis nostrud exercitation ullamco
laboris nisi ut aliquip ex ea commodo consequat.
```

```
</asp:Panel>
<asp:Panel ID="Panel3" runat="server" CssClass="panel">
        Duis aute irure dolor in reprehenderit in voluptate velit
esse cillum dolore eu fugiat nulla pariatur.
</asp:Panel>
<asp:Panel ID="Panel4" runat="server" CssClass="panel">
                Excepteur sint occaecat cupidatat non proident,
sunt in culpa qui officia deserunt mollit anim id est laborum
</asp:Panel>
</div>
</form>
```

5. The page will display the panels as captured in the following screenshot:



We will now add jQuery UI's draggable feature to these panels so that they can be dragged anywhere on the page on the mousedown event.

How to do it...

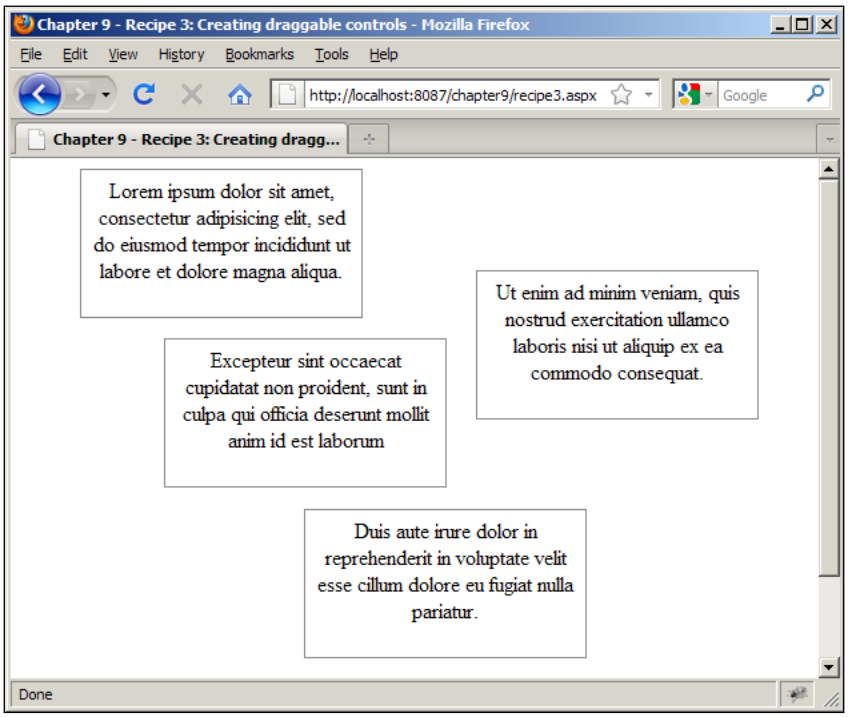
1. In the `document.ready()` function of the jQuery script block, apply the `draggable()` method to the panel controls. Use the css selector `$(".panel")` to retrieve all the panel controls on the page:

```
$(".panel").draggable();
```

```
Thus, the complete jQuery solution is as follows:
<script language="javascript" type="text/javascript">
$(document).ready(function(){
       $(".panel").draggable();
});
</script>
```

How it works...

Run the web form. Now click on any panel and drag on the screen. You will notice that the panels can be dragged and dropped as required, as shown in the following screenshot:



There's more...

For detailed documentation on the jQuery UI draggable feature, please visit

`http://jqueryui.com/demos/draggable/`.

See also

Adding tooltips to controls

Creating dialog boxes

Dialog boxes are an important mode of interaction with the end user. They are used to deliver information, error, warning, or confirmation messages to the end user.

jQuery UI provides a dialog widget as an alternative to the traditional JavaScript pop-up windows. In this recipe, let's see how to use jQuery UI dialog widget to display a basic confirmation dialog box.

Getting Ready

1. Create a new web form `Recipe4.aspx` in the current project.
2. Add an ASP.NET panel on the form with the required confirmation message as follows:

```
<asp:Panel ID="dialogbox" ToolTip="Confirmation" runat="server">
        Are you sure you want to continue?
</asp:Panel>
```

At runtime, the ASP.NET panel control is rendered as a div element and the `ToolTip` property is rendered as `title` property of the div element. Hence, we can use the `ToolTip` property to set the title of the dialog box as done in the code snippet mentioned earlier.

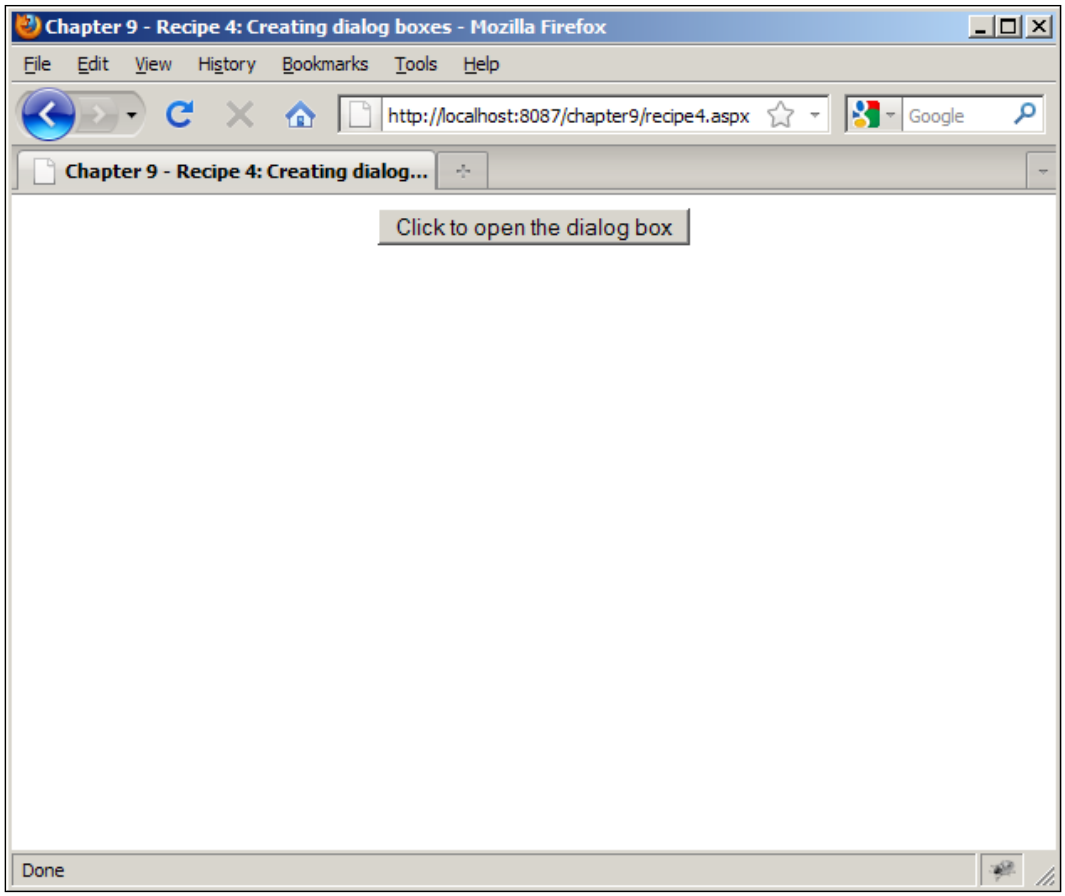
3. Add a button control to the form to trigger the dialog box on the `click` event.

Thus, the complete aspx markup of the web form is as follows:

```
<form id="form1" runat="server">
    <div align="center">
        <asp:Panel ID="dialogbox" ToolTip="Confirmation"
runat="server">
        Are you sure you want to continue?
        </asp:Panel>
```

```
        <asp:Button ID="btnOpen" runat="server" Text="Click to open the
dialog box" />
    </div>
</form>
```

Thus, on page load, the page will display as shown in the following screenshot:



Now, let's use jQuery UI to set the properties of the dialog box.

How to do it...

1. In the `document.ready()` function of the jQuery script block, apply the `dialog()` method on the panel control:

   ```
   $( "#dialogbox" ).dialog({
   ```

2. Set the `autoOpen` property to `false`, so that the dialog box is hidden until the `dialog("open")` method is called:

```
autoOpen: false,
```

3. Set the `modal` property to `true`, so that the web form is disabled as long as the dialog box is open:

```
modal:true,
```

4. Set the `resizable` property to `false`, so that the dialog window cannot be resized at runtime:

```
resizable:false,
```

5. Set the `buttons` property to an array containing `OK` and `Cancel` buttons. Also define the desired action on the click event of the respective buttons. For the purpose of this demo, we have defined `.dialog("Close")` as the custom action for both the buttons:

```
buttons: {
                "OK" : function(){
                    $(this).dialog("close");
                },
                "Cancel" : function(){
                    $(this).dialog("close");
                }
      }
    });
```

6. Define an event handler for the `click` event of the `btnOpen` button control:

```
$( "#btnOpen" ).click(function() {
```

7. Open the dialog box:

```
$( "#dialogbox" ).dialog( "open" );
```

8. Return `false` so that the dialog box remains active on the page until further user action:

```
return false;
    });
```

Thus, the complete jQuery solution is as follows:

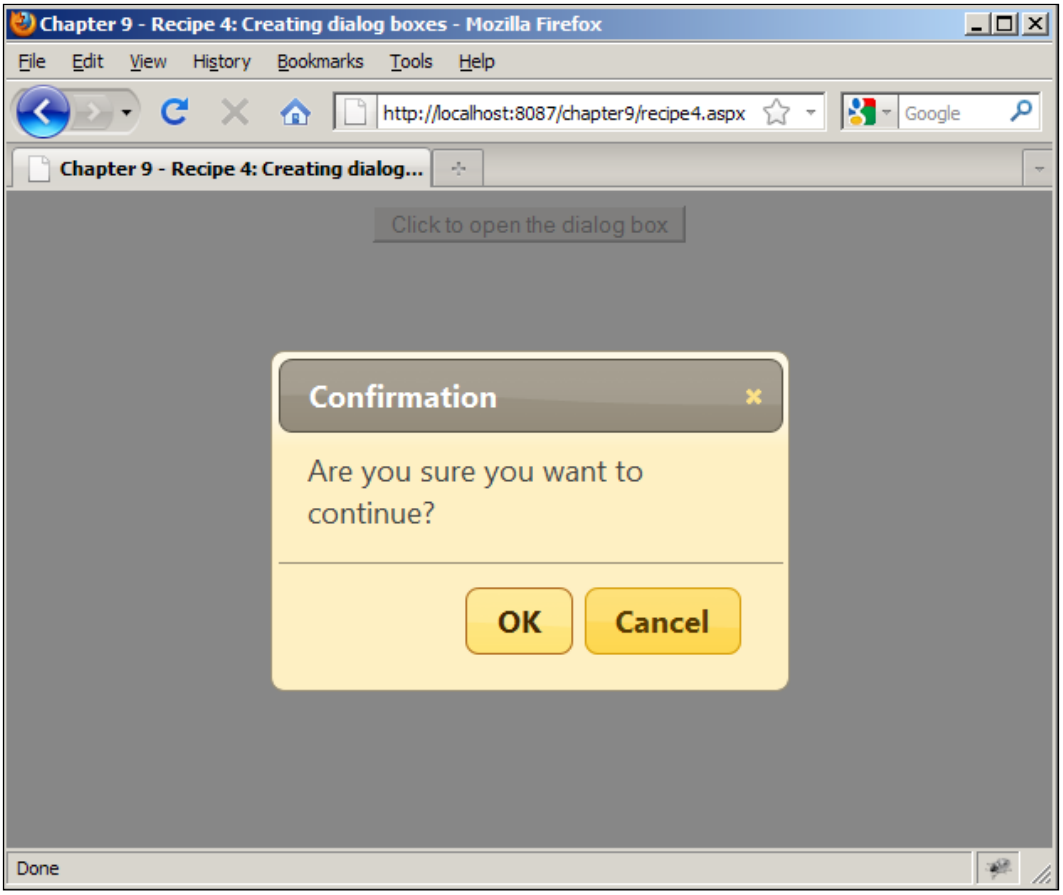
```
<script language="javascript" type="text/javascript">
$(document).ready(function(){
        $( "#dialogbox" ).dialog({
      autoOpen: false,
      modal:true,
      resizable:false,
      buttons: {
                "OK" : function(){
                    $(this).dialog("close");
                },
                "Cancel" : function(){
                    $(this).dialog("close");
                }
      }
    });

    $( "#btnOpen" ).click(function() {
      $( "#dialogbox" ).dialog( "open" );
      return false;
    });
});
</script>
```

Note that, at runtime, the dialog box is rendered as the following div element:

```
<div id="dialogbox" title="Confirmation">
        Are you sure you want to continue?
</div>
```

How it works...

Run the web form. Click on the button to launch the confirmation box as follows:



On clicking either of the **OK** or **Cancel** buttons, the dialog box disappears and the page is enabled again. Note that the dialog box can also be dragged on the page as required.

There's more...

For detailed documentation on the jQuery UI dialog widget, please visit

`http://jqueryui.com/demos/dialog/`.

See also

Using the progress bar control

Using the datepicker control

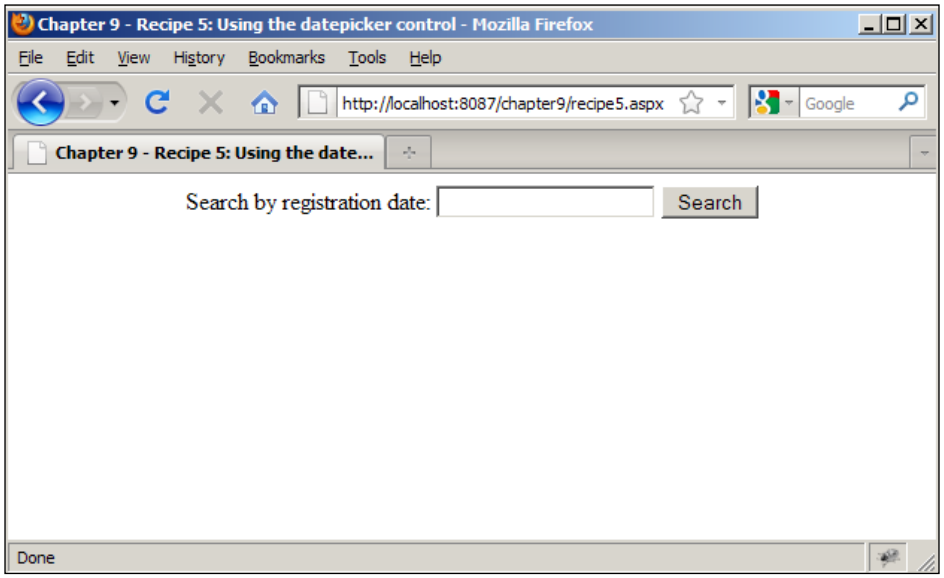
The datepicker is a popular control for date fields in online submission forms. In this recipe, let's see how to use the jQuery UI to attach a datepicker to an ASP.NET TextBox control.

Getting Ready

1. Create a new web form `Recipe5.aspx` in the current project.
2. Add controls to create a simple search form that accepts an input date field as follows:

```
<form id="form1" runat="server">
    <div align="center">
        <asp:Label ID="lblDate" runat="server">Search by
registration date: </asp:Label>
        <asp:TextBox ID="txtDate" runat="server"></asp:TextBox>
        <asp:Button ID="btnSubmit" Text="Search" runat="server" />
    </div>
</form>
```

Thus, on page load, the web form appears as shown in the following screenshot:



We will now use jQuery UI to attach a datepicker to the TextBox control.

How to do it...

- In the `document.ready()` function of the jQuery script block, apply the `datepicker()` method to the TextBox control:

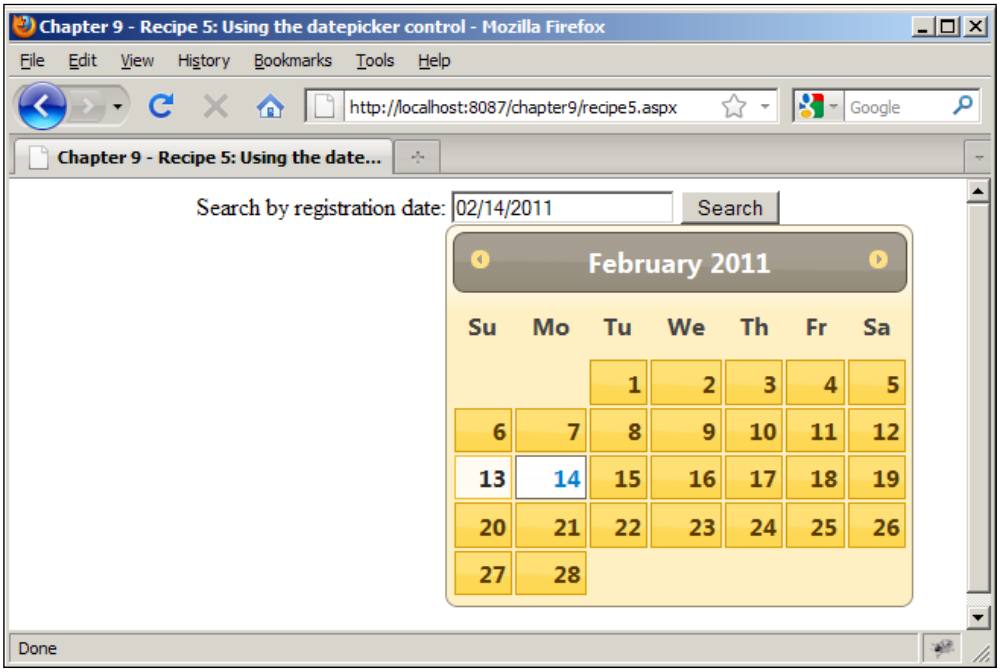
```
$("#txtDate").datepicker();
```

Thus, the complete jQuery solution for the given problem is as follows:

```
<script language="javascript" type="text/javascript">
    $(document).ready(function(){
       $("#txtDate").datepicker();
    });
</script>
```

How it works...

Run the web form. On mouseclick on the TextBox control, the datepicker is displayed as shown in the following screenshot:



The desired date can be picked from the displayed calendar as required.

There's more...

For detailed documentation on the jQuery UI datepicker widget, please visit

`http://jqueryui.com/demos/datepicker/`.

See also

Creating dialog boxes

Using the progress bar control

jQuery UI provides a Progressbar widget to show the processing status during a wait time in an application. In this recipe, we will learn to create a Progressbar in ASP.NET.

Getting Ready

1. Include an animated gif file `pbar-ani.gif` in the images folder in the project.
2. Add a new web form `Recipe6.aspx` to the current project.
3. Add an ASP.NET panel control for the progressbar as follows:

```
<asp:Panel id="progressbar" runat="server"></asp:Panel>
```

4. Define some basic css style for the above as follows:

```
#progressbar
  {
    width:300px;
    height:22px;
  }
```

5. The jQuery UI progressbar uses the jQuery UI CSS Framework for styling. Hence, to set the background of the progressbar to the animated `gif` file, add the following css style:

```
.ui-progressbar-value { background-image: url(images/pbar-ani.
gif); }
```

6. Create another content panel that is initially hidden and displayed only after the progressbar loads completely.

```
<asp:Panel id="contentArea" runat="server">Page successfully
loaded</asp:Panel>
```

7. We will use the following css class to hide this panel,

```
.hide
  {
    display:none;
  }
```

Thus, the complete aspx markup of the form is as follows:

```
<form id="form1" runat="server">
<div align="center">
   <asp:Panel id="progressbar" runat="server"></asp:Panel>
   <asp:Panel id="contentArea" runat="server">Page successfully
loaded</asp:Panel>
</div>
</form>
```

Now, we will look at the jQuery solution for applying the Progressbar widget to the ASP.NET panel.

How to do it...

1. In the `document.ready()` function of the jQuery script block, hide the display message:

```
$("#contentArea").addClass("hide");
```

2. Initialise a counter:

```
var cnt = 0;
```

3. Define the maximum value of the counter:

```
var maxCnt = 100;
```

4. Use the JavaScript timer function `setInterval()` to define the timeout interval and the callback function after each interval:

```
var id = setInterval(showprogress, 10);
```

5. Now, define the previous callback function:

```
function showprogress() {
```

6. Check if the current value of the counter is less than or equal to the maximum allowable value:

```
if (cnt <= maxCnt) {
```

7. If yes, then apply the `progressbar()` function with the current counter value:

```
$("#progressbar").progressbar({
                        value: cnt
                    });
```

8. Increment the counter:

```
cnt++;
                }
```

9. If the current value of the counter is greater than the maximum value, clear the timer using the respective ID:

```
else {
    clearInterval(id);
```

10. Show the display message:

```
$("#contentArea").removeClass("hide");
```

11. Hide the progress bar:

```
$("#progressbar").addClass("hide");
                }
            }
```

Thus, the complete jQuery solution is a follows:

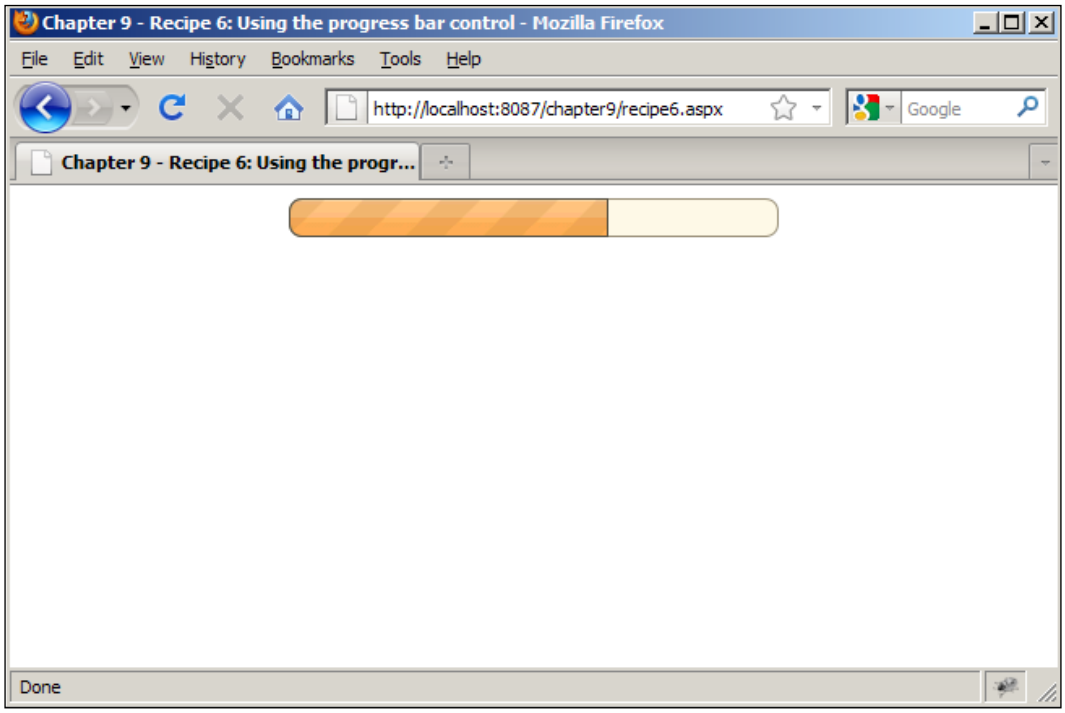
```
<script language="javascript" type="text/javascript">
        $(document).ready(function() {
            $("#contentArea").addClass("hide");
            var cnt = 0;
            var maxCnt = 100;

            var id = setInterval(showprogress, 10);

            function showprogress() {
                if (cnt <= maxCnt) {
                    $("#progressbar").progressbar({
                        value: cnt
                    });
                    cnt++;
                }
                else {
                    clearInterval(id);
                    $("#contentArea").removeClass("hide");
                    $("#progressbar").addClass("hide");
                }
            }
        });
    </script>
```

In this solution, we have used the JavaScript timer `setInterval(customFunction, timeout)` to call a custom function after the timeout (in milliseconds). Important points to note are:

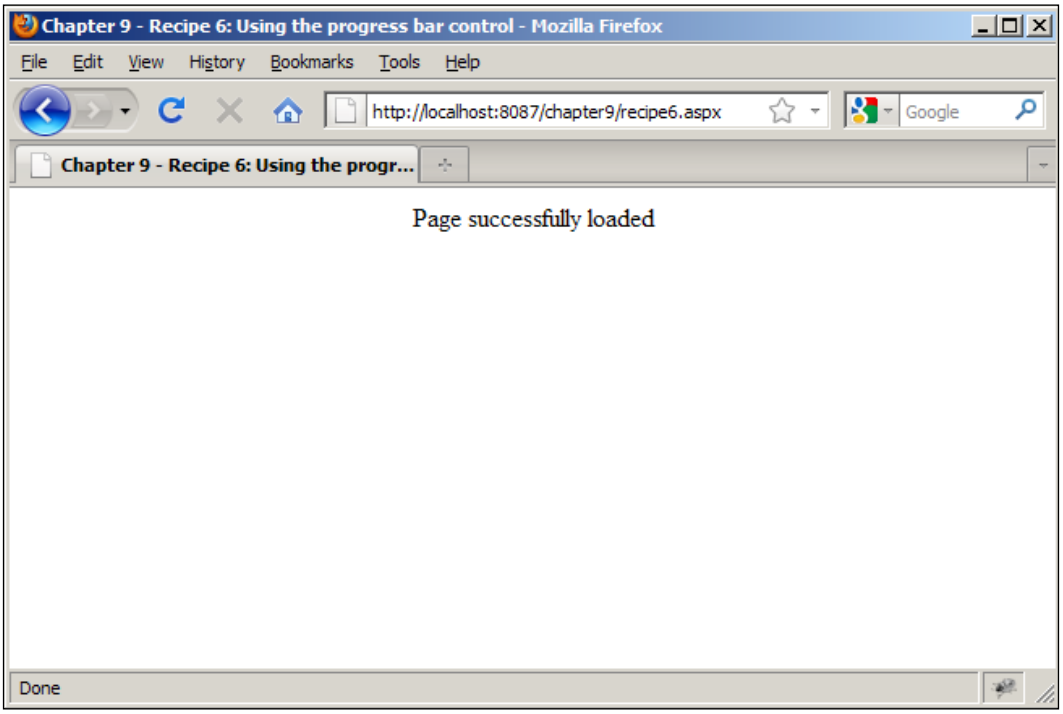
- The setInterval method returns a numeric number, `id` to track the timeout. This ID can be later used to clear the timer.
- The timer calls the custom function `showprogress()` repeatedly after every timeout interval until the `clearInterval(id)` is called.
- After every timeout, we will increment the variable `cnt` by 1 and apply it to the progressbar.
- When `cnt` reaches `maxCnt`, the progressbar loads completely.

How it works...

Run the web form. You will see that the progressbar loads in steps, as shown in the following screenshot:



After the load is complete, the progressbar is hidden and the content panel is displayed instead, as follows:



There's more...

For detailed documentation on the jQuery UI progressbar widget, please visit

`http://jqueryui.com/demos/progressbar/`.

See also

Using the slider control

Using the slider control

The slider control is useful in various applications requiring user interaction such as volume control, color pickers, etc. In this recipe, we will apply the jQuery UI slider widget to create a scrollable div.

Getting Ready

1. Create a new web form `Recipe7.aspx` in the current project.
2. Create nested div areas on the page as follows:

```
<div id="outerContent">
          <div id="innerContent">
          </div>
</div>
```

3. Since we want to scroll the inner content area, we will set the width of the `innerContent` to be longer than that of the `outerContent`. Hence, define the following styles on the respective div areas:

```
#outerContent
    {
        width:300px;
        height:100px;
        overflow:hidden;
        border: 1px solid;
    }
#innerContent
    {
        width:500px;
        position:relative;
        left: 5px;
        overflow:auto;
    }
```

4. Now create a div element to which we can apply the slider as follows:

```
<div id ="slider"></div>
```

Thus, the complete aspx markup of the web form is as follows:

```
<form id="form1" runat="server">
    <div align="center" >
       <div id ="slider"></div>
       <div id="outerContent">
          <div id="innerContent">
          Lorem ipsum dolor sit amet, consectetur adipisicing elit,
sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris
nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in
reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla
pariatur. Excepteur sint occaecat cupidatat non proident, sunt in
culpa qui officia deserunt mollit anim id est laborum
          </div>
       </div>
    </div>
</form>
```

Let's take a look at the adopted approach for scrolling the inner div based on the position of the slider:

1. First determine the amount of scrollable area available:

   ```
   Scrollable Area = (Width of Inner Div) – (Width of Outer Div)
   ```

2. Now, apply the Slider value to this Scrollable area to get the left scroll position:

   ```
   Left Scroll Position= (Slider Value * Scrollable Area)/100
   ```

   Since, the slider value is a percentage, in the above formula we divide by 100.

3. Now, set the scrollLeft property of the outer div to the above value to scroll the inner div as per the slider.

Next, let's use jQuery UI to implement the slider using the earlier mentioned approach.

How to do it...

1. In the `document.ready()` function of the jQuery script block, apply the `slider()` function on the slider panel:

   ```
   $("#slider").slider({
   ```

2. Define the minimum value of the slider:

   ```
   min: 0,
   ```

3. Define the maximum value of the slider:

   ```
   max: 100,
   ```

4. Set `animate` to `true` to slide smoothly on mouse click:

   ```
   animate: true,
   ```

5. Define the callback function on the slide event. This event is triggered on every mouse move during the slide:

   ```
   slide: function(e, ui) {
   ```

6. As explained earlier, get the scrollable width as follows:

   ```
   var iScrollWidth = $("#innerContent").width() -
   $("#outerContent").width();
   ```

7. Set the scrollLeft property by computing the left scroll position:

   ```
   $("#outerContent").attr({ scrollLeft: (ui.value * 0.01 *
   iScrollWidth) });
                  },
   ```

8. Define the callback function on the change event. This event is triggered on slide stop or if the value is changed programmatically:

```
change: function(e, ui) {
```

9. Similar to the slide event, calculate the scrollable width:

```
var iScrollWidth = $("#innerContent").width() -
$("#outerContent").width();
```

10. Animate the scrollLeft property of the outer panel:

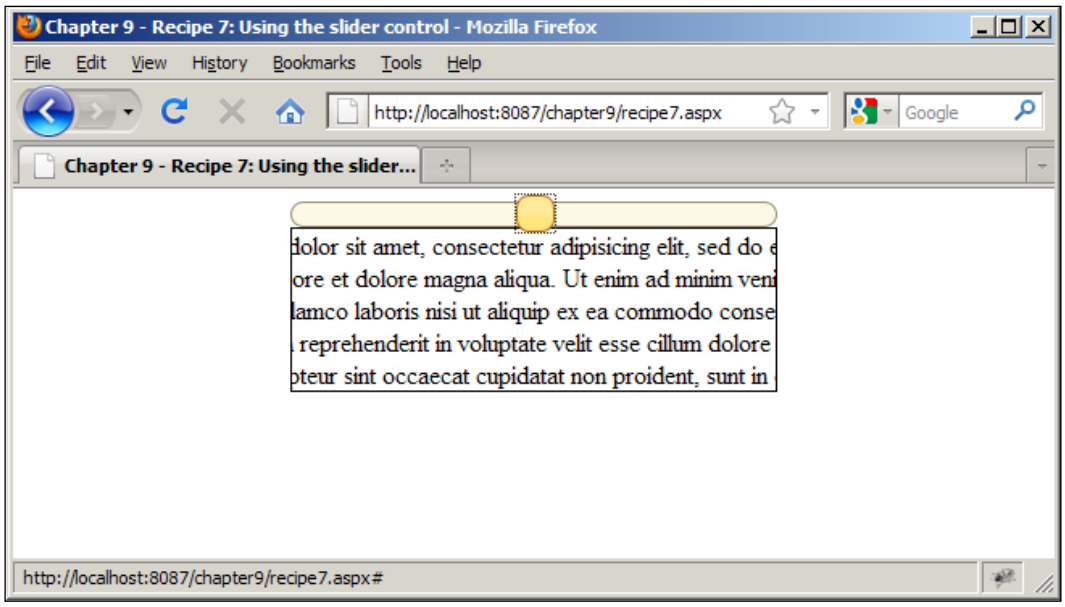
```
$("#outerContent").animate({ scrollLeft: (ui.value * 0.01 *
iScrollWidth) }, "slow");
                }
            });
```

Thus, the complete jQuery UI solution for the given problem is as follows:

```
<script language="javascript" type="text/javascript">
        $(document).ready(function() {
            $("#slider").slider({
                min: 0,
                max: 100,
                animate: true,
                slide: function(e, ui) {
                var iScrollWidth = $("#innerContent").width() -
$("#outerContent").width();
                    $("#outerContent").attr({ scrollLeft: (ui.value *
0.01 * iScrollWidth) });
                },
                change: function(e, ui) {
                var iScrollWidth = $("#innerContent").width() -
$("#outerContent").width();
                    $("#outerContent").animate({ scrollLeft: (ui.value
* 0.01 * iScrollWidth) }, "slow");
                }
            });
        });
    </script>
```

How it works...

Run the page. Drag the slider handle to display the scrolling content as shown next:



There's more...

For detailed documentation on the jQuery UI slider widget, please visit

`http://jqueryui.com/demos/slider/`.

See also

Using the progressbar control

Adding tooltips to controls

jQuery UI version 1.8 does not have inbuilt support for tooltips. The widget will only be released as part of jQuery UI version 1.9.

In this recipe, we will explore an alternative tooltip plugin for creating tooltips in ASP.NET pages.

Getting Ready

1. Create a new web form `Recipe8.aspx` in the current project.
2. Download the jQuery tooltip plugin from `http://bassistance.de/jquery-plugins/jquery-plugin-tooltip/`.
3. Save the above plugin file in the script folder `js` in the project.
4. On the web form, include the stylesheet as well as the plugin as follows:

```
<link href="css/jquery.tooltip.css" rel="stylesheet" type="text/
css" />
    <script src="js/jquery-1.4.1.js" type="text/javascript"></
script>
    <script src="js/jquery.tooltip.js" type="text/javascript"></
script>
```

5. Add a simple login form to the page.
6. Add `span` areas containing the tooltip text as follows:

```
<span class="tooltip" title="Please use your registered email
address">?</span>

<span class="tooltip" title="Please enter your pin number
here">?</span>
```

   where, the `tooltip` css class is defined as follows:

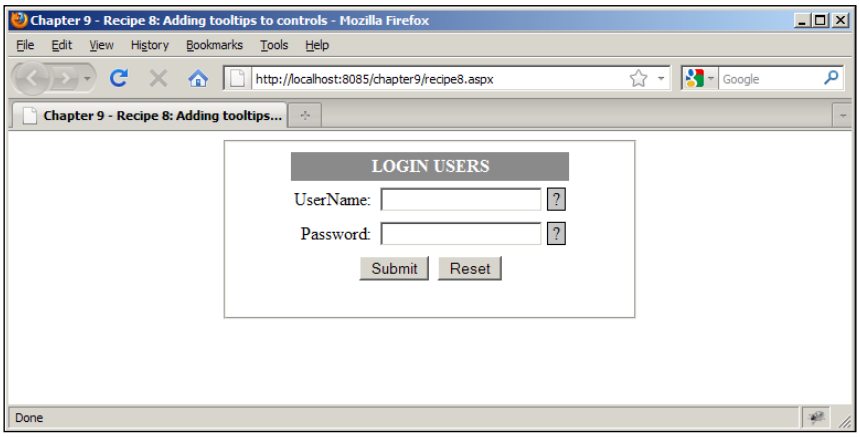
```
.tooltip
    {
      border: 1px solid;
      background-color:Silver;
      cursor:pointer;
      width:15px;
    }
```

Thus, the complete aspx markup of the page is as follows:

```
<form id="form1" runat="server">
    <div align="center" id="contentArea">
    <fieldset style="width:350px;height:140px;">
    <table border="0" cellpadding="3" cellspacing="3">
       <tr><td colspan="2" class="header">LOGIN USERS</td></tr>
        <tr>
            <td align="right">
                <asp:Label ID="lblUserName" runat="server"
Text="UserName: "></asp:Label>
            </td>
            <td align="left">
```

```
                <asp:TextBox ID="txtUserName" runat="server"></
asp:TextBox>
                <span class="tooltip" title="Please use your
registered email address"> ? </span>
            </td>
        </tr>
        <tr>
            <td align="right">
                <asp:Label ID="lblPassword" runat="server"
Text="Password: "></asp:Label>
            </td>
            <td align="left">
                <asp:TextBox ID="txtPassword" runat="server"
TextMode="Password"></asp:TextBox>
                <span class="tooltip" title="Please enter your pin
number here"> ? </span>
            </td>
        </tr>
        <tr>
            <td align="center" colspan="2">
                <asp:Button ID="btnSubmit" runat="server"
Text="Submit"/> 
                <asp:Button ID="btnReset" runat="server"
Text="Reset"/>
            </td>
        </tr>
    </table>
    </fieldset>
    </div>
    </form>
```

On page load, the web form appears as displayed in the following screenshot:



Let's see how to use the tooltip plugin for displaying the tooltip text on hover.

How to do it...

1. In the `document.ready()` function of the jQuery script block, apply the `tooltip()` method to all the span elements:

```
$("span").tooltip();
```

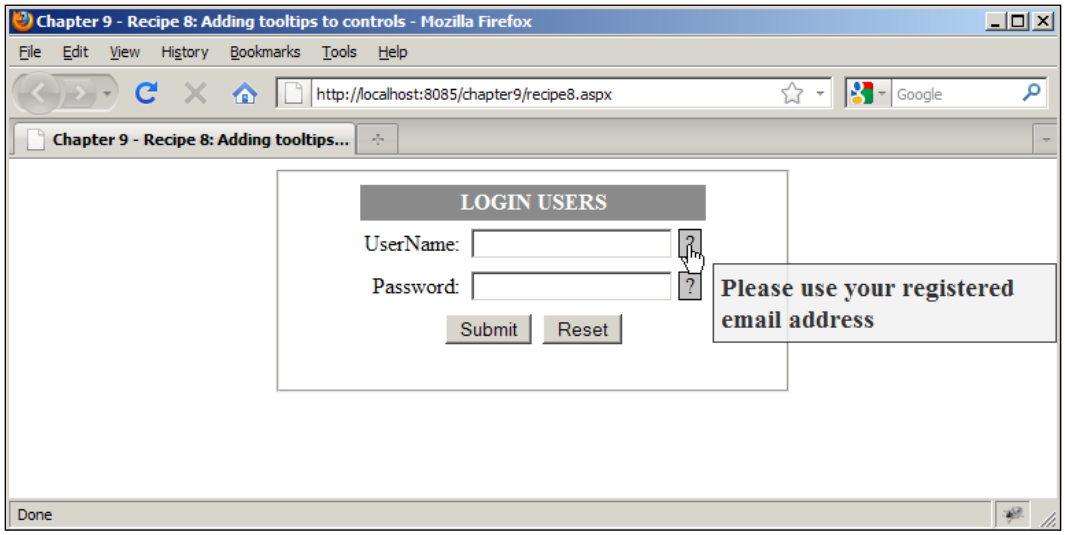
Thus, the complete jQuery solution is as follows:

```
<script language="javascript" type="text/javascript">
    $(document).ready(function(){
      $("span").tooltip();
    });
    </script>
```

Note that be applying the `.tooltip()` method on the span areas, we can create the required tooltips. The `title` text is displayed as the tooltip text.

How it works...

Run the web page. Mouseover on the **?** icon. You will notice that the tooltip is displayed as shown in the following screenshot:



There's more...

For detailed documentation on the jQuery Tooltip plugin, please visit

`http://docs.jquery.com/Plugins/Tooltip`.

See also

Creating dialog boxes

Where to buy this book

You can buy ASP.NET jQuery Cookbookfrom the Packt Publishing website:
`http://www.packtpub.com/asp-net-jquery-cookbook/book`

Free shipping to the US, UK, Europe and selected Asian countries. For more information, please read our shipping policy.

Alternatively, you can buy the book from Amazon, BN.com, Computer Manuals and most internet book retailers.

www.PacktPub.com

For More Information:
www.PacktPub.com/asp-net-jquery-cookbook/book