



Lab 02B

Embedding Power BI Content

Overview

The estimated time to complete the lab is 35 minutes

*You must have completed **Lab 02A** before commencing this lab.*

In this lab, you will create an ASP.NET web application, and develop a web form for enabling the report selection of Power BI reports. This form will then be extended with logic to embed the selected report. Having completed report embedding, you will clone the web form and adapt it to embed dashboards. You will then embed a pre-developed web page that embeds the Q&A experience. Lastly, you will finalize the web application by creating a default page to navigate to one of the embed pages.

This lab focuses on development practices for embedding Power BI reports, dashboards and the Q&A experience. It does not intend to convey good web application design practices, including—but not limited to—secure storage and retrieval of sensitive data, exception handling, storage and reuse of access tokens, and web site styling.

Developing with the REST API

In this exercise, you will create a web application project, and then use the Power BI REST API to present a dropdown list of reports on a web page.

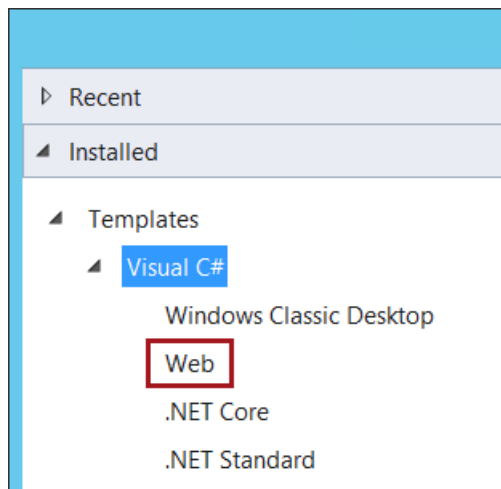
Creating an ASP.NET Web Application

In this task, you will create an ASP.NET web application.

1. Open Visual Studio.



2. To create a new project, on the **File** menu, select **File | New Project**.
3. In the **New Project** window, in the left pane, expand **Visual C#**, and then select **Web**.



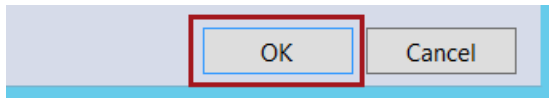
4. Ensure that the selected template is **ASP.NET Web Application**.



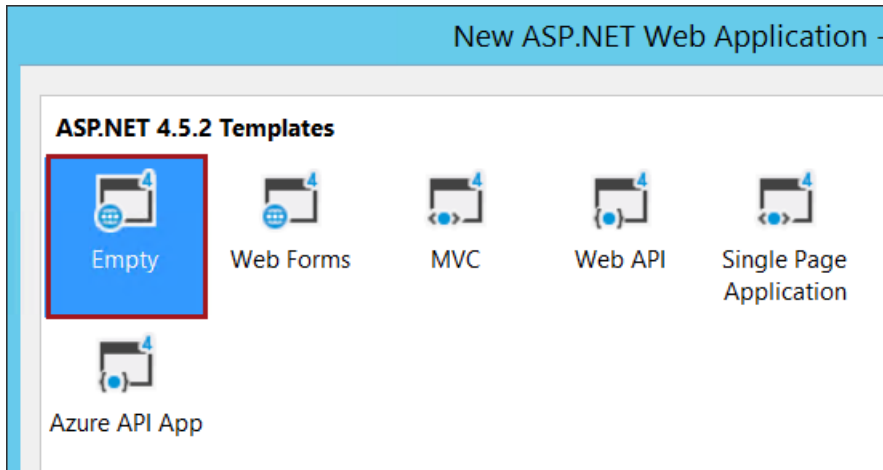
It is very important that you select the correct template.

5. In the **Name** box, replace the text with **PowerBIEmbedding**.
6. In the **Solution Name** box, if necessary, replace the text also with **PowerBIEmbedding**.

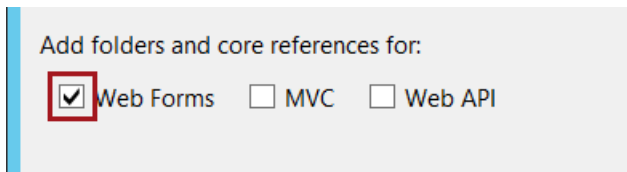
7. Click **OK**.



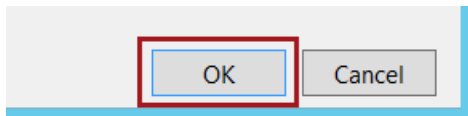
8. In the **New ASP.NET Web Application** window, ensure that the **Empty** template is selected.



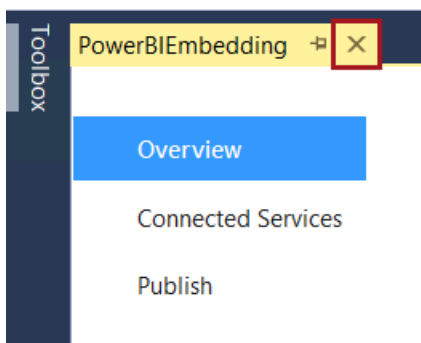
9. Check the **Web Forms** checkbox.



10. Click **OK**.



11. When the project has been created, close the project file.



Installing NuGet Packages

In this task, you will install two NuGet packages to support developing with the Power BI REST API, and Azure AD authentication.

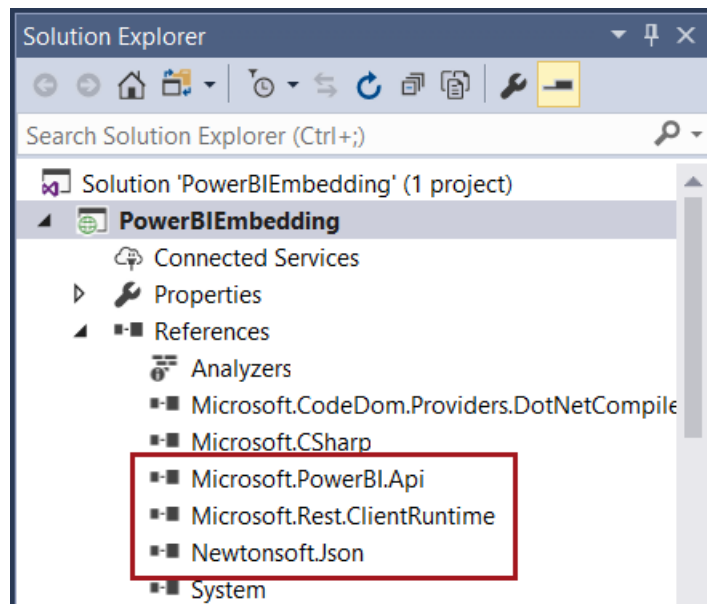
1. On the **Tools** menu, select **NuGet Package Manager | Package Manager Console**.
2. In the **Package Manager Console** pane, enter the following command:

*For convenience, the command can be copied from the
<CourseFolder>\PowerBIDev\AD\Lab02B\Assets\Snippets.txt file.*

NuGet

```
Install-Package Microsoft.PowerBI.Api
```

3. Press **Enter**.
4. When the installation has completed, in **Solution Explorer**, expand the **References** folder.
5. Notice the addition of:
 - Microsoft.PowerBI.Api
 - Microsoft.Rest.ClientRuntime
 - Newtonsoft.Json



6. Install the second package:

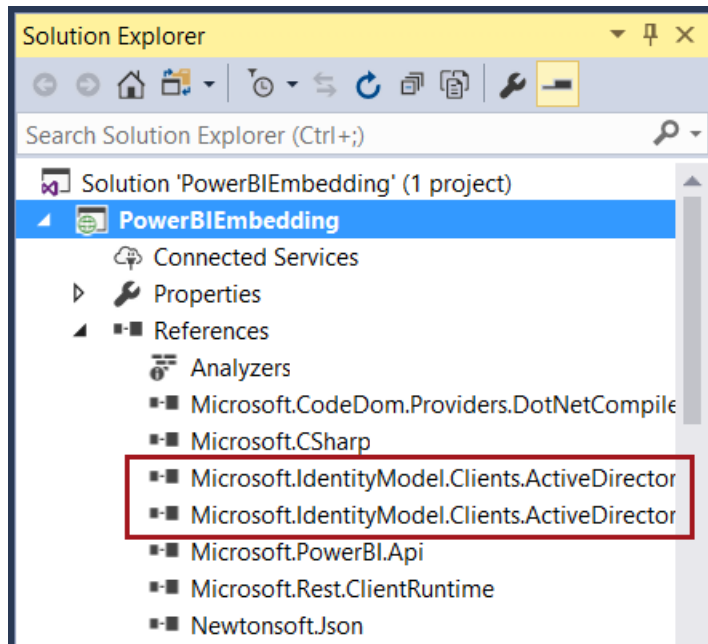
*For convenience, the command can be copied from the
<CourseFolder>\PowerBIDev\AD\Lab02B\Assets\Snippets.txt file.*

NuGet

```
Install-Package Microsoft.IdentityModel.Clients.ActiveDirectory
```

This package is required to authenticate the master app account.

7. In the **References** folder, notice the addition of:
- Microsoft.IdentityModel.Clients.ActiveDirectory
 - Microsoft.IdentityModel.Clients.ActiveDirectory.Platform



8. Collapse the **References** folder.

There is presently a requirement to update the Newtonsoft.Json package to the latest version, which at the time of publishing this course is v11.0.1.

9. Install the latest **Newtonsoft.Json** package.

For convenience, the command can be copied from the <CourseFolder>\PowerBIDevIAD\Lab02B\Assets\Snippets.txt file.

NuGet

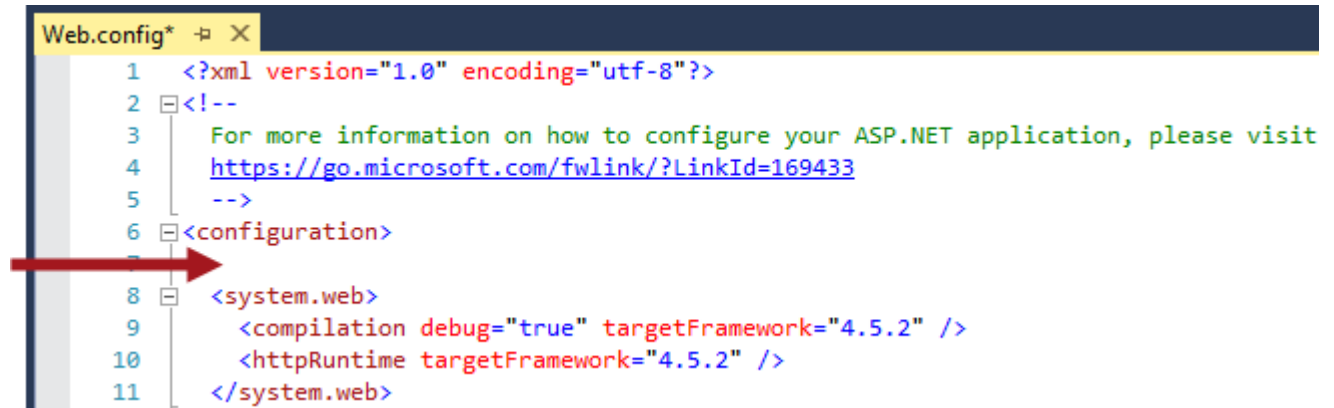
```
Install-Package Newtonsoft.Json -Version 11.0.1
```

Configuring the Web.config File

In this task, you will configure the **Web.config** file to store service endpoints, and also details of your workspace app id, client id, and master app credentials.

1. In **Solution Explorer**, to open the file, double-click the **Web.config** file.
2. Immediately beneath the **configuration** element, on a new line, paste the **appSettings** element.

*For convenience, the element can be copied from the **<CourseFolder>\PowerBIDevIAD\Lab02B\Assets\Snippets.txt** file.*



```
Web.config* - X
1 <?xml version="1.0" encoding="utf-8"?>
2 <!--
3   For more information on how to configure your ASP.NET application, please visit
4   https://go.microsoft.com/fwlink/?LinkId=169433
5   -->
6 <configuration>
7
8   <system.web>
9     <compilation debug="true" targetFramework="4.5.2" />
10    <httpRuntime targetFramework="4.5.2" />
11  </system.web>
```

XML

```
<appSettings>
  <add key="authorityUrl" value="https://login.windows.net/common/oauth2/authorize/" />
  <add key="resourceUrl" value="https://analysis.windows.net/powerbi/api" />
  <add key="apiUrl" value="https://api.powerbi.com/" />
  <add key="embedUrlBase" value="https://app.powerbi.com/" />
  <add key="appWorkspaceId" value="" />
  <add key="applicationId" value="" />

  <!-- Note: Except in this lab, do NOT store credentials in cleartext -->
  <add key="pbiUsername" value="" />
  <add key="pbiPassword" value="" />
</appSettings>
```

3. Copy the **App Workspace ID** value from **MySettings.txt** file into the value of the **appWorkspaceId** key.



```
6 <configuration>
7   <appSettings>
8     <add key="authorityUrl" value="https://login.windows.net/common/oauth2/authorize/" />
9     <add key="resourceUrl" value="https://analysis.windows.net/powerbi/api" />
10    <add key="apiUrl" value="https://api.powerbi.com/" />
11    <add key="embedUrlBase" value="https://app.powerbi.com/" />
12    <add key="appWorkspaceId" value="" />
13    <add key="applicationId" value="" />
```

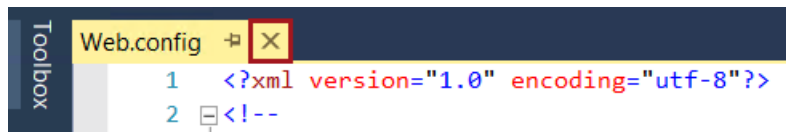
- Copy the **Application ID** value from **MySettings.txt** file into the value of the **applicationId** key.

```
6 <configuration>
7 <appSettings>
8 <add key="authorityUrl" value="https://login.windows.net/common/oauth2/authorize/" />
9 <add key="resourceUrl" value="https://analysis.windows.net/powerbi/api" />
10 <add key="apiUrl" value="https://api.powerbi.com/" />
11 <add key="embedUrlBase" value="https://app.powerbi.com/" />
12 <add key="appWorkspaceId" value=" " />
13 <add key="applicationId" value=" " />
```

- Notice the comment warning that the Master App account credentials should not ordinarily be stored in cleartext.

The Master App account password will be stored in cleartext. This is not a recommended practice, and is done to simplify the lab. If you want to encrypt the password, you will need to implement the logic yourself.

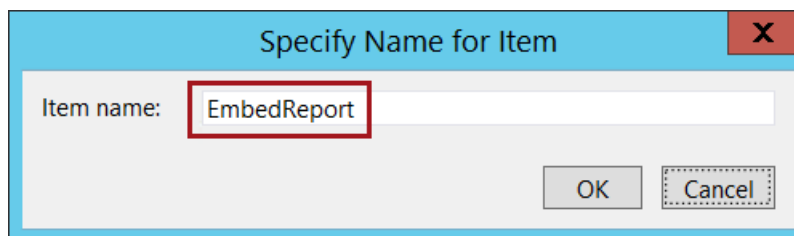
- Enter the Power BI account credentials from the **MySettings.txt** file:
 - Copy the **Power BI Account** value from **MySettings.txt** file into the value of the **pbiUsername** key
 - Copy the **Power BI Password** value from **MySettings.txt** file into the value of the **pbiPassword** key
- To save the file, on the **File** menu, select **Save Web.config**.
- To close the file, close the tab.



Adding a Web Form

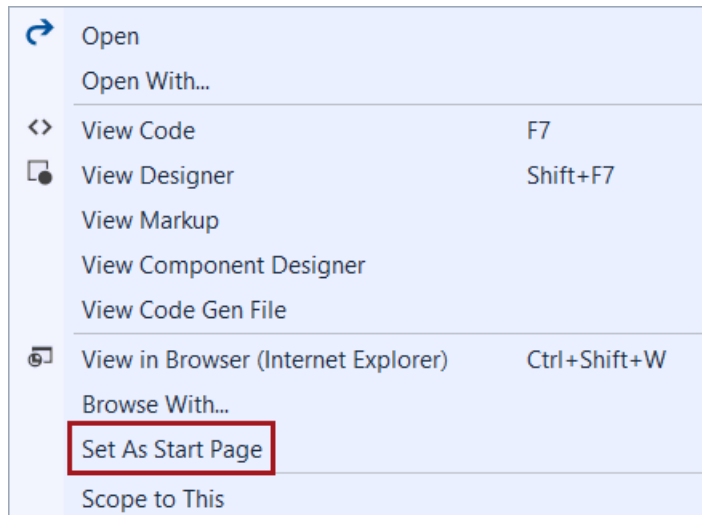
In this task, you will add a web form that will be used to embed a report, and add a server-side dropdown list.

- In **Solution Explorer**, right-click the **PowerBIEmbedding** project (not the solution), and then select **Add | Web Form**.
- In the **Specify Name for item** window, in the **Item Name** box, replace the text with **EmbedReport**.



- Click **OK**.

4. Notice that the web page file (**EmbedReport.aspx**) has automatically opened.
5. In **Solution Explorer**, right-click the **EmbedReport.aspx** item, and then select **Set as Start Page**.



6. To add a dropdown list, beneath the **form** element, add the following element:

```
9  <body>
10    <form id="form1" runat="server">
11    →
12        <div>
13            </div>
14    </form>
15 </body>
16 </html>
17
```

For convenience, the element can be copied from the <CourseFolder>\PowerBIDevIAD\Lab02B\Assets\Snippets.txt file.

ASP.NET

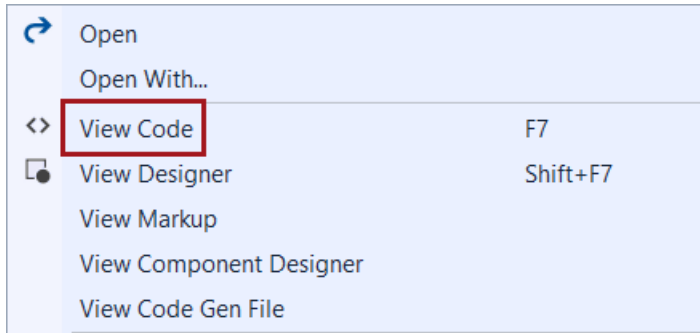
```
<asp:DropDownList ID="ddlReport" runat="server" AutoPostBack="True"
    OnSelectedIndexChanged="Page_Load" />
```

This element will add a server-side dropdown list, which will be populated with one item for each report found in the app workspace. It will raise an index changed event when the app user selected a different dropdown item.

Developing the Web Form

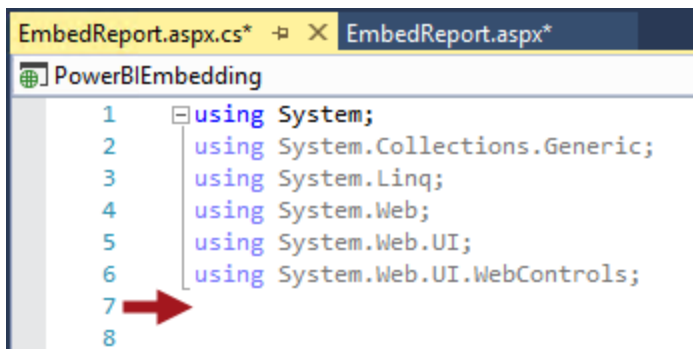
In this task, you will develop the web form to retrieve an Access token, and then use the token to retrieve all report definitions from the app workspace.

1. In **Solution Explorer**, right-click the **EmbedReport.aspx** item, and then select **View Code**.



2. To develop the app settings retrieval code, add the following namespace directive beneath the last namespace directive.

For convenience, all remaining code in this lab can be copied from the <CourseFolder>PowerBIDevIAD\Lab02B\Assets\Snippets.txt file.



Visual C#

```
using System.Configuration;
```

3. Add the following class-level declarations inside the **EmbedReport** class.

```
9 namespace PowerBIEmbedding
10 {
11     1 reference
12     public partial class EmbedReport : System.Web.UI.Page
13     {
14         0 references
15         protected void Page_Load(object sender, EventArgs e)
16         {
```

Visual C#

```
private static readonly string AuthorityUrl = ConfigurationManager.AppSettings["authorityUrl"];
private static readonly string ResourceUrl = ConfigurationManager.AppSettings["resourceUrl"];
private static readonly string ApiUrl = ConfigurationManager.AppSettings["apiUrl"];
private static readonly string AppWorkspaceId = ConfigurationManager.AppSettings["appWorkspaceId"];
private static readonly string ApplicationId = ConfigurationManager.AppSettings["applicationId"];

private static readonly string Username = ConfigurationManager.AppSettings["pbiUsername"];
private static readonly string Password = ConfigurationManager.AppSettings["pbiPassword"];
```

*Tip: When pasting snippets, they can be easily formatted (correctly indented) by first selecting the inserted lines, and then by using the **Edit** menu, and selecting **Format Selection (Ctrl+K, Ctrl+F)**.*

*This code retrieves all **Web.config** app setting values and stores them in read-only variables.*

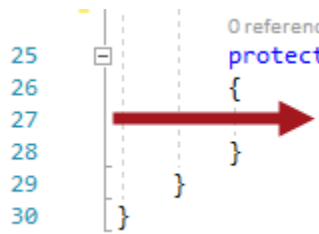
4. To develop the Access token retrieval code, add the following two namespace directive beneath the last added namespace directive.

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Web;
5 using System.Web.UI;
6 using System.Web.UI.WebControls;
7 using System.Configuration;
8
9
10
```

Visual C#

```
using Microsoft.IdentityModel.Clients.ActiveDirectory;
using Microsoft.Rest;
```

5. Add the following code inside the **Page_Load** method.



0 references

```
25 protected void Page_Load(object sender, EventArgs e)
26 {
27     //
28 }
29
30 }
```

Visual C#

```
var credential = new UserPasswordCredential(Username, Password);

// Authenticate using app settings credentials
var authenticationContext = new AuthenticationContext(AuthorityUrl);
var authenticationResult = authenticationContext.AcquireTokenAsync(ResourceUrl, ApplicationId,
    credential).Result;

var tokenCredentials = new TokenCredentials(authenticationResult.AccessToken, "Bearer");
```

6. To populate the dropdown list with the app workspace reports, add the following two namespace directive beneath the last added namespace directives.

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Web;
5 using System.Web.UI;
6 using System.Web.UI.WebControls;
7 using System.Configuration;
8 using Microsoft.IdentityModel.Clients.ActiveDirectory;
9 using Microsoft.Rest;
10
11
12
```

Visual C#

```
using Microsoft.PowerBI.Api.V2;
using Microsoft.PowerBI.Api.V2.Models;
```

Add the following code inside the **Page_Load** method, beneath the last added lines.

Visual C#

```
// Populate the dropdown list
if (!IsPostBack)
{
    using (var client = new PowerBIClient(new Uri(ApiUrl), tokenCredentials))
    {
        // Get a list of reports
        var reports = client.Reports.GetReportsInGroup(AppWorkspaceId);

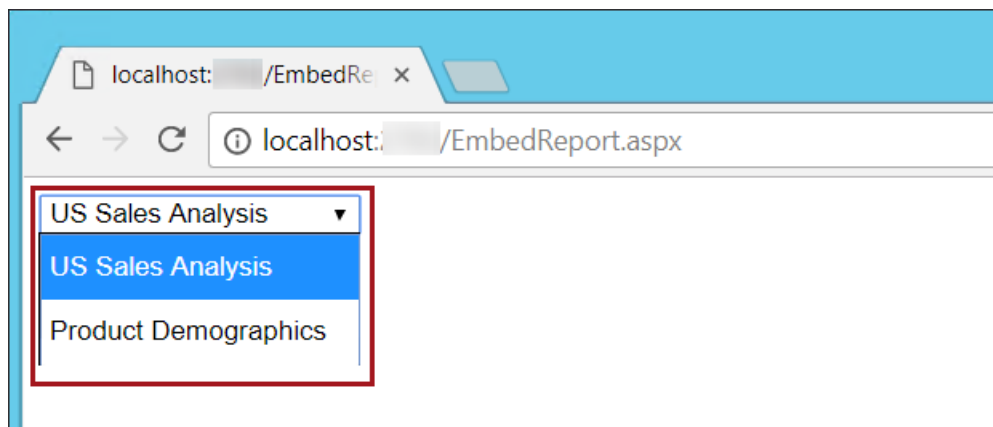
        // Populate dropdown list
        foreach (Report item in reports.Value)
        {
            ddlReport.Items.Add(new ListItem(item.Name, item.Id));
        }

        // Select first item
        ddlReport.SelectedIndex = 0;
    }
}
```

7. On the toolbar, start the web application (start debugging).



8. When the web browser opens, verify that a dropdown list appears, and that it contains two items, one for each report in the app workspace.



9. Close the web session, and stop debugging.



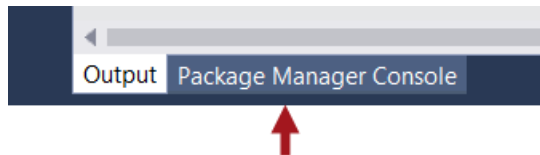
Embedding a Report

In this exercise, you will continue developing the web form to embed a report.

Installing a NuGet Package

In this task, you will install a NuGet package to embed reports client-side with the Power BI JavaScript API.

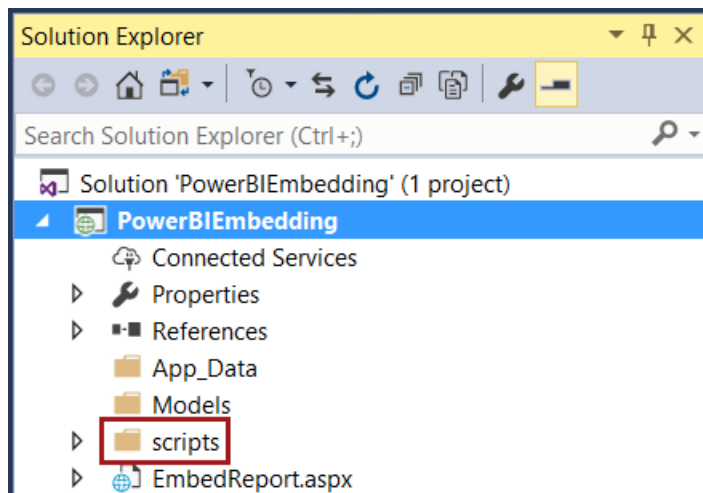
1. At the bottom-left corner, select the **Package Manager Console** pane.



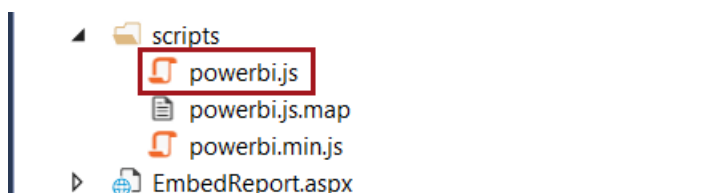
2. In the **Package Manager Console** pane, install the following package:

NuGet	
Install-Package	Microsoft.PowerBI.JavaScript

3. When the installation has completed, in **Solution Explorer**, notice the addition of the **scripts** folder.



4. Expand the **scripts** folder, and notice the **powerbi.js** file.



*The **powerbi.js** file must be referenced by web pages that will perform embedding.*

Developing the Web Form

In this task, you will continue to develop the web form to embed a report.

1. Switch to the **EmbedReport.aspx** file.
2. Immediately after the **body** element, insert the following **script** element.

```
9   <body>
10  →
11   <form id="form1" runat="server">
```

HTML

```
<script src="/scripts/powerbi.js"></script>
```

3. Immediately after the **asp:DropDownList** element, replace the **div** element (both lines) with the following **div** element.

```
11  <form id="form1" runat="server">
12    <asp:DropDownList ID="ddlReport" runat="server" AutoPostBack="True"
13    <div>
14    </div>
15  </form>
```

HTML

```
<br />
```

```
<div id="embedDiv" style="height: 600px; width: 100%; max-width: 1000px;" />
```

```
11  <form id="form1" runat="server">
12    <asp:DropDownList ID="ddlReport" runat="server" AutoPostBack="True" OnSelecte
13    <br />
14    <div id="embedDiv" style="height: 600px; width: 100%; max-width: 1000px;" />
15  </form>
```

4. Switch to the **EmbedReport.aspx.cs** file (code-behind).
5. Add the following additional class-level declarations inside the **EmbedReport** class, after the last added declarations.

```
23  private static readonly string Username = ConfigurationManager.AppSettings["pbiUsername"];
24  private static readonly string Password = ConfigurationManager.AppSettings["pbiPassword"];
25
26  →
```


Visual C#

```
public string embedToken;
public string embedUrl;
public string reportId;
```

These variables will store values required for client-side embedding.

6. Add the following additional code inside the **Page_Load** method, after the last added code.

```
54 // Select first item
55 ddlReport.SelectedIndex = 0;
56 }
57 }
58
59
```




Visual C#

```
// Generate an embed token and populate embed variables
using (var client = new PowerBIClient(new Uri(ApiUrl), tokenCredentials))
{
    // Retrieve the selected report
    var report = client.Reports.GetReportInGroup(AppWorkspaceId, ddlReport.SelectedValue);

    // Generate an embed token to view
    var generateTokenRequestParameters = new GenerateTokenRequest(accessLevel: "view");
    var tokenResponse = client.Reports.GenerateTokenInGroup(AppWorkspaceId, report.Id,
        generateTokenRequestParameters);

    // Populate embed variables (to be passed client-side)
    embedToken = tokenResponse.Token;
    embedUrl = report.EmbedUrl;
    reportId = report.Id;
}
```


7. Switch to the **EmbedReport.aspx** file.
8. Immediately after the **div** element, add the following **script** element.

```
11 <form id="form1" runat="server">
12     <asp:DropDownList ID="ddlReport" runat="server" AutoPostBack="True" OnSelecte
13     <br />
14     <div id="embedDiv" style="height: 600px; width: 100%; max-width: 1000px;" />
15     
16 </form>
```

HTML

```
<script>
    // Read embed token
    var embedToken = "<% =this.embedToken %>";

    // Read embed URL
    var embedUrl = "<% = this.embedUrl %>";

    // Read report Id
    var reportId = "<% = this.reportId %>";

    // Get models (models contains enums)
    var models = window['powerbi-client'].models;

    // Embed configuration is used to describe what and how to embed
    // This object is used when calling powerbi.embed
    // It can also includes settings and options such as filters
    var config = {
        type: 'report',
        tokenType: models.TokenType.Embed,
        accessToken: embedToken,
        embedUrl: embedUrl,
        id: reportId,
        settings: {
            filterPaneEnabled: true,
            navContentPaneEnabled: true
        }
    };

    // Embed the report within the div element
    var report = powerbi.embed(embedDiv, config);
</script>
```

*Tip: Remember, you can select the lines of code added, and then press **Ctrl+K**, **Ctrl+F** to format the selection.*

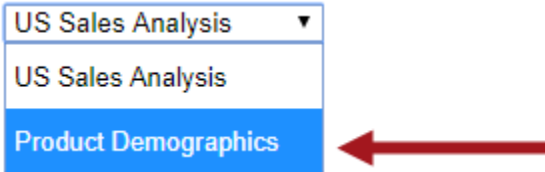
Reviewing the Web App

In this task, you will start the web app, and then review the **EmbedReport.aspx** web form design.

1. On the toolbar, start the web application (start debugging).



2. When the web browser opens, wait until the **US Sales Analysis** report is embedded.
3. Interact with the report by:
 - Modifying the **Year** slicer value
 - Multi-selecting **Category** slicer values
 - Highlighting (selecting) month columns, or demographic bars
 - Opening the **Filters** pane, and multi-selecting specific regions
4. To embed the second report, in the dropdown list, select the **Product Demographics** report.



5. When the report has loaded, close the web session, and then stop debugging.
6. Close all open project items.

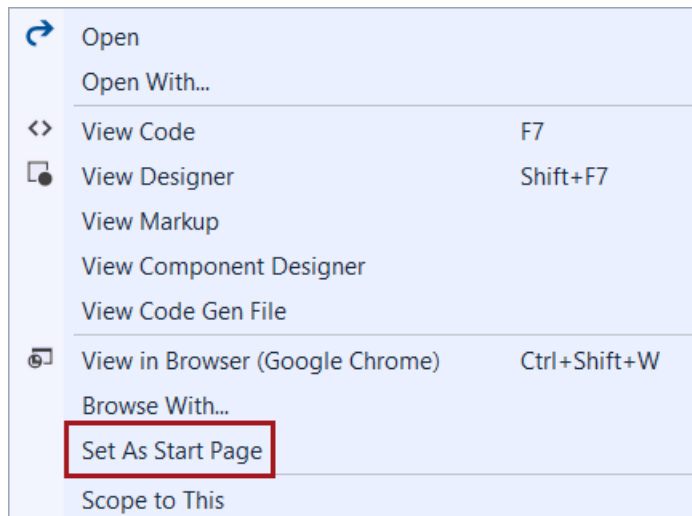
Embedding a Dashboard

In this exercise, you will develop a new web form to embed a dashboard.

Cloning the EmbedReport Form

In this task, you will clone (copy) the **EmbedReport.aspx** project item.

1. Close all open project items.
2. In **Solution Explorer**, right-click the **EmbedReport.aspx** item, and then select **Copy**.
3. In **Solution Explorer**, right-click the **PowerBIEmbedding** project, and then select **Paste**.
4. In **Solution Explorer**, right-click the **EmbedReport - Copy.aspx** item, and then select **Rename**.
5. Rename the item to **EmbedDashboard.aspx**, and then press **Enter**.
6. Set the **EmbedDashboard.aspx** item as the start page.



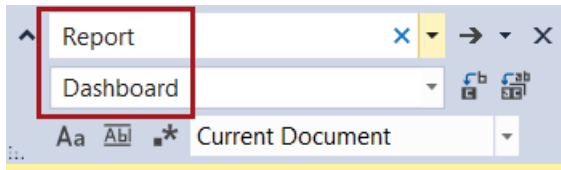
Adapting the Web Form

In this task, you will adapt the new web form by replacing all instances of the word **Report** with **Dashboard**.

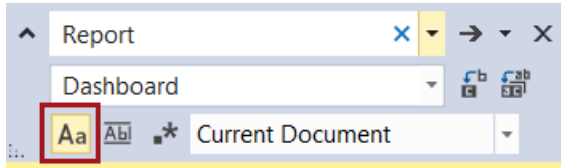
It is important that you follow these instructions very carefully.

1. Open the **EmbedDashboard.aspx** item.
2. To perform search-and-replace, press **Ctrl+H**.
3. In the **Find** box (first box), enter **Report**.

4. In the **Replace** box (second box), enter **Dashboard**.

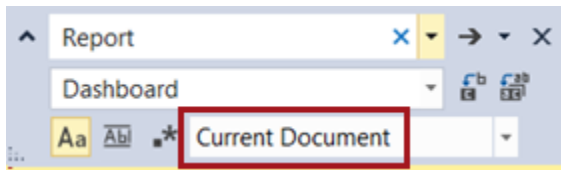


5. Ensure the **Aa** button is highlighted (click, if required) to enable **Match Case**.

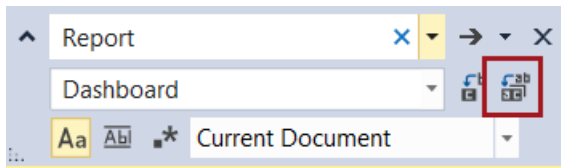


Visual C# is a case-sensitive language, and so you will replace the words twice, once with a capitalized first letter, and once in lower-case.

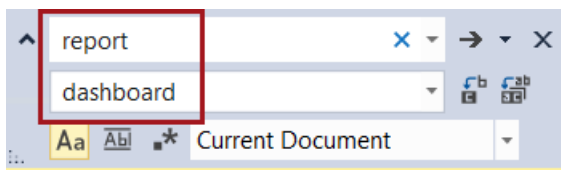
6. Ensure that the **Current Document** scope is selected.



7. Click **Replace All**.




8. When prompted with the message (2 occurrences replaced), click **OK**.
9. Replace all lower-cased instances also (7 occurrences replaced).



10. In the div element, remove the **max-width** style property.

```
13 | <div id="embedDiv" style="height: 600px; width: 100%; max-width: 1000px;" />
```



*Unlike reports which are fixed dimension, dashboards can be almost any practical dimensions. It is possible, however, to modify the **pageView** setting to determine how “overflow” content is made visible.*

11. In the **script** element, modify the **config** variable declaration by removing the **settings** assignment (leave the comma after the **id** assignment).

```
30  var config = {
31      type: 'dashboard',
32      tokenType: models.TokenType.Embed,
33      accessToken: embedToken,
34      embedUrl: embedUrl,
35      id: dashboardId,
36      settings: {
37          filterPaneEnabled: true,
38          newContentPaneEnabled: true
39      },
40  };
```

Settings are specific to report embedding.

12. Add the **pageView** assignment in place of the **settings** assignment.

```
30  var config = {
31      type: 'dashboard',
32      tokenType: models.TokenType.Embed,
33      accessToken: embedToken,
34      embedUrl: embedUrl,
35      id: dashboardId,
36      pageView: "actualSize"
37  };
```

Adapting the Web Form Code

In this task, you will adapt the new web form code by replacing all instances of the word **Report** with **Dashboard**, and addressing several inconsistencies.

1. Open the **EmbedDashboard.aspx.cs** item.
2. Perform the same two search-and-replace operations applied to the **EmbedDashboard.aspx** item.
 - Replace **Report** with **Dashboard** (10 occurrences)
 - Replace **report** with **dashboard** (10 occurrences)

This approach almost works... however there is one inconsistency in the API that will need to be addressed.

3. In the **Page_Load** method, when populating the dropdown list, modify the **Name** property to **DisplayName**.

```
48  // Populate dropdown list
49  foreach (Dashboard item in dashboards.Value)
50  {
51      ddlDashboard.Items.Add(new ListItem(item.DisplayName, item.Id));
52  }
```

Reviewing the Web App

In this task, you will start the web application, and then review the **EmbedDashboard.aspx** web form design.

1. On the toolbar, start the web application (start debugging).



2. When the web browser opens, wait until the **US Sales Monitoring** dashboard is embedded.

Unlike embedded reports, embedded dashboards do not support interactivity.

*Depending on your screen size, it is possible that the dashboard does not display all tiles, and will require scrolling to see them. You will explore the **pageView** setting in the following steps which provide control over how the embedded dashboard will display its tiles.*

3. Close the web session, and then stop debugging.
4. In the **EmbedDashboard.aspx** web form, modify the **pageView** assignment to **fitToWidth**.

```
30  var config = {  
31      type: 'dashboard',  
32      tokenType: models.TokenType.Embed,  
33      accessToken: embedToken,  
34      embedUrl: embedUrl,  
35      id: dashboardId,  
36      pageView: "fitToWidth"  
37  };
```

5. Start the application again, and review the embedded dashboard.

Tip: You can scroll within the iFrame.

6. Close the web session, and then stop debugging.
7. In the **EmbedDashboard.aspx** web form, modify the **pageView** assignment to **oneColumn**.

```
30  var config = {  
31      type: 'dashboard',  
32      tokenType: models.TokenType.Embed,  
33      accessToken: embedToken,  
34      embedUrl: embedUrl,  
35      id: dashboardId,  
36      pageView: "oneColumn"  
37  };
```

8. Start the application again, and review the embedded dashboard.

*The **oneColumn** option is suitable for narrow form factors, like a mobile phone.*

9. Close the web session, and then stop debugging.
10. Close all open project items.

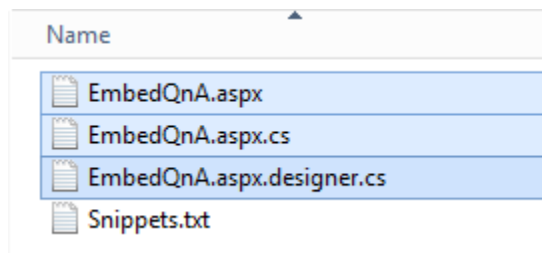
Embedding Q&A

In this exercise, you will add a pre-developed web form to embed Q&A.

Adding an Existing Web Form

In this task, you will add and then explore a pre-developed web form.

1. In Visual Studio, in **Solution Explorer**, right-click the **PowerBIEmbedding** project, and then select **Add | Existing Item**.
2. In the **Add Existing Item** window, navigate to the **<CourseFolder>\PowerBIDevIAD\Lab02B\Assets** folder.
3. Multi-select the three files where the name commences with **EmbedQnA**.



4. Click **Add**.
5. Set the **EmbedQnA.aspx** item as the start page.
6. Open the code-behind file for the **EmbedQnA.aspx** item.
7. Review the code in the **Page_Load** method, noting the following:
 - A dropdown list is populated with datasets available in the app workspace
 - An embed token is generated for the selected dataset, with view access level
 - The EmbedURL requires a specific format
8. Close the code-behind file for the **EmbedQnA.aspx** item.
9. Open the **EmbedQnA.aspx** item.
10. Review the **config** variable, noting the following:
 - **type** is set to **qna**
 - **datasetIds** is an array, which is set to the selected **datasetId** (presently only a single item can be passed to this parameter)
 - **viewMode** is set to **Interactive**, which enables entering and modifying a question
11. Start the application again, and review the embedded Q&A experience.

12. Enter the following question: **Sales by state where region is Southern as filled map.**

Dataset: **US Sales Analysis**

☐ Sales by state where region is Southern as filled map



13. Modify the region to a different one.

14. Close the web session, and then stop debugging.

15. Close all open project items.

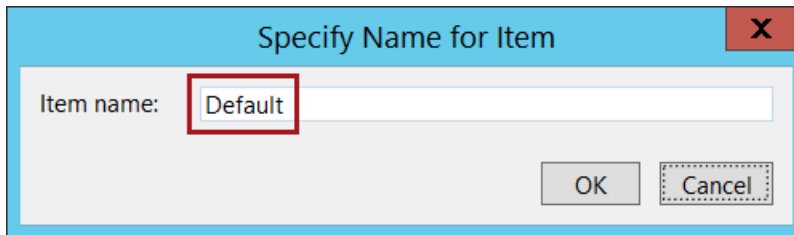
Finalizing the App Design

In this exercise, you will finalize the web application design by adding a default page to enable navigation to one of the embed pages.

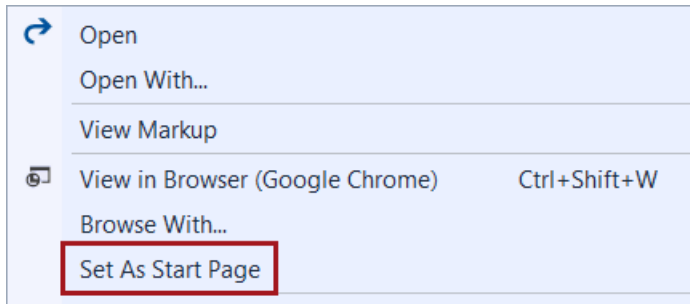
Finalizing the App Design

In this task, you will finalize the web application design by adding a default page to enable navigation to either of the embed pages.


1. In **Solution Explorer**, right-click the **PowerBIEmbedding** project, and then select **Add | HTML Page**.
2. In the **Specify Name for Item** window, replace the text with **Default**.



3. Click **OK**.
4. Set the **Default.html** item as the start page.



5. Within the **body** element, insert the following HTML to add hyperlinks to each embed page.

```
1      <!DOCTYPE html>
2      <html>
3      <head>
4          <meta charset="utf-8" />
5          <title></title>
6      </head>
7      <body>
8          
9      </body>
10     </html>
```

HTML

```
<a href="EmbedReport.aspx">> Embed Reports</a><br />
<a href="EmbedDashboard.aspx">> Embed Dashboards</a><br />
<a href="EmbedQnA.aspx">> Embed Q&A</a>
```

6. Start the application, and optionally test each link.
7. Close the web session, and then stop debugging.
8. Close all open project items.
9. On the **File** menu, select **Save All**.
10. Leave the solution open.

*You will enhance the application with client-side logic in **Lab 02C**.*

Summary

In this lab, you created an ASP.NET web application, and developed a web form for enabling the selection of Power BI reports. This form then was extended with logic to embed the selected report. Having completed report embedding, you cloned the web form and adapted it to embed dashboards. You then embedded a pre-developed web page that embeds the Q&A experience. Lastly, you finalized the web application by creating a default page to navigate to one of the embed pages.

Terms of Use

© 2017-2018 Microsoft. All rights reserved.

By using this hands-on lab, you agree to the following terms:

The technology/functionality described in this hands-on lab is provided by Microsoft Corporation in a “sandbox” testing environment for purposes of obtaining your feedback and to provide you with a learning experience. You may only use the hands-on lab to evaluate such technology features and functionality and provide feedback to Microsoft. You may not use it for any other purpose. Without written permission, you may not modify, copy, distribute, transmit, display, perform, reproduce, publish, license, create derivative works from, transfer, or sell this hands-on lab or any portion thereof.

COPYING OR REPRODUCTION OF THE HANDS-ON LAB (OR ANY PORTION OF IT) TO ANY OTHER SERVER OR LOCATION FOR FURTHER REPRODUCTION OR REDISTRIBUTION WITHOUT WRITTEN PERMISSION IS EXPRESSLY PROHIBITED.

THIS HANDS-ON LAB PROVIDES CERTAIN SOFTWARE TECHNOLOGY/PRODUCT FEATURES AND FUNCTIONALITY, INCLUDING POTENTIAL NEW FEATURES AND CONCEPTS, IN A SIMULATED ENVIRONMENT WITHOUT COMPLEX SET-UP OR INSTALLATION FOR THE PURPOSE DESCRIBED ABOVE. THE TECHNOLOGY/CONCEPTS REPRESENTED IN THIS HANDS-ON LAB MAY NOT REPRESENT FULL FEATURE FUNCTIONALITY AND MAY NOT WORK THE WAY A FINAL VERSION MAY WORK. WE ALSO MAY NOT RELEASE A FINAL VERSION OF SUCH FEATURES OR CONCEPTS. YOUR EXPERIENCE WITH USING SUCH FEATURES AND FUNCTIONALITY IN A PHYSICAL ENVIRONMENT MAY ALSO BE DIFFERENT.

FEEDBACK If you give feedback about the technology features, functionality and/or concepts described in this hands-on lab to Microsoft, you give to Microsoft, without charge, the right to use, share and commercialize your feedback in any way and for any purpose. You also give to third parties, without charge, any patent rights needed for their products, technologies and services to use or interface with any specific parts of a Microsoft software or service that includes the feedback. You will not give feedback that is subject to a license that requires Microsoft to license its software or documentation to third parties because we include your feedback in them. These rights survive this agreement.

MICROSOFT CORPORATION HEREBY DISCLAIMS ALL WARRANTIES AND CONDITIONS WITH REGARD TO THE HANDS-ON LAB, INCLUDING ALL WARRANTIES AND CONDITIONS OF MERCHANTABILITY, WHETHER EXPRESS, IMPLIED OR STATUTORY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. MICROSOFT DOES NOT MAKE ANY ASSURANCES OR REPRESENTATIONS WITH REGARD TO THE ACCURACY OF THE RESULTS, OUTPUT THAT DERIVES FROM USE OF THE VIRTUAL LAB, OR SUITABILITY OF THE INFORMATION CONTAINED IN THE VIRTUAL LAB FOR ANY PURPOSE.

DISCLAIMER This lab contains only a portion of new features and enhancements in Microsoft Power BI. Some of the features might change in future releases of the product.

Document Version

#	Date	Author	Comments
1	22-JUL-2017	Peter Myers	Visual Studio 2017 v15.2 (26430.14) Power BI service v13.0.1930.145 Microsoft.IdentityModel.Clients.ActiveDirectory v3.14.2 Microsoft.PowerBI.Api v2.0.1 MicrosoftPowerBI.JavaScript v2.3.3 Microsoft.Rest.ClientRuntime v2.0.1 Newtonsoft.Json v7.01
2	31-DEC-2017	Peter Myers	Visual Studio 2017 v15.2 (26430.14) Power BI service v13.0.3495.220 Microsoft.IdentityModel.Clients.ActiveDirectory v3.17.3 Microsoft.PowerBI.Api v2.0.8 MicrosoftPowerBI.JavaScript v2.4.2 Microsoft.Rest.ClientRuntime v2.0.1 Newtonsoft.Json v7.01
3	21-FEB-2018	Peter Myers	Visual Studio 2017 v15.2 (26430.14) Power BI service v13.0.4385.126 Microsoft.IdentityModel.Clients.ActiveDirectory v3.19.1 Microsoft.PowerBI.Api v2.0.10 MicrosoftPowerBI.JavaScript v2.4.7 Microsoft.Rest.ClientRuntime v2.3.10 Newtonsoft.Json v11.0.1
4	30-JUN-2018	Peter Myers	(Resequenced from Lab 02A to Lab 02B) Power BI service v13.0.5702.156 Microsoft.IdentityModel.Clients.ActiveDirectory v3.19.8 Microsoft.PowerBI.Api v2.0.12 MicrosoftPowerBI.JavaScript v2.5.1 Microsoft.Rest.ClientRuntime v2.3.11 Newtonsoft.Json v11.0.2