

Streaming Datasets and Real-time Dashboards



Agenda

- Power BI Embedded Overview
- Working with PBIX Project Files
- Provisioning Workspaces in Azure
- Embedding Reports in a MVC Application
- Configuring Datasets for DirectQuery Mode
- Putting It All Together



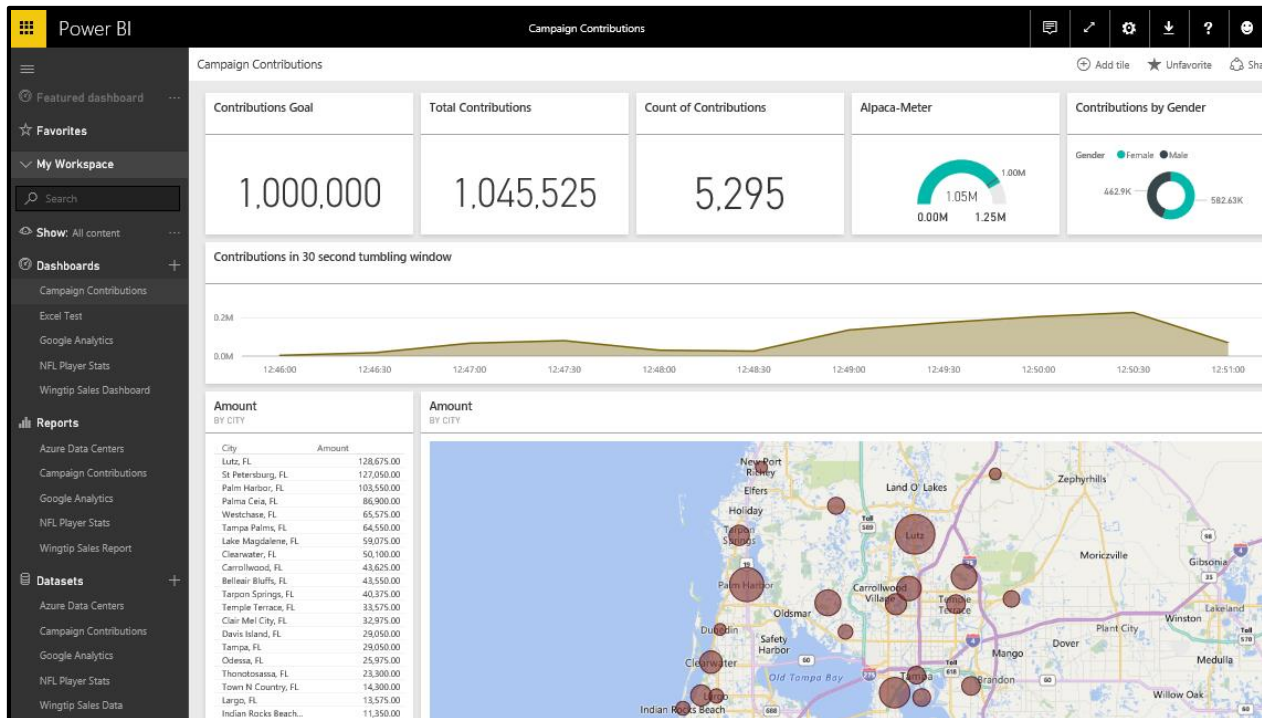
What is Power BI?

- What is Power BI?
 - Cloud-based subscription service
 - Environment which promotes self-service BI *to the end user*
 - BI Platform to assists with data import, analysis and visualization
- Power BI benefits from being a cloud-based service
 - It takes only 5 seconds to subscribe to the Power BI service
 - New users can create something significant in 5 minutes or less



The Power BI Service at **PowerBI.com**

- The Power BI service is accessible through browser
 - Provides cloud-based foundation for Power BI platform
 - Accessible through URL at <https://app.powerbi.com>
 - Users require Office 365 accounts and Power BI Licenses



What is Power BI Embedded?

- Power BI Embedded is an Azure Service
 - PBI Embedded service can be provisioned on-demand
 - Service provisioned in terms of workspace collections
 - PBI Embedded service requires an Azure subscription
- What is the core value of Power BI Embedded?
 - It eliminates need for Power BI license for each user
 - It eliminates need for Office 365 account for each user
 - It decouples user security from app security
 - It opens up PBI platform to commercial applications



PowerBI.com versus Power BI Embedded

PowerBI.com

- Accessed via <https://app.powerbi.com>
- Requires Office 365 accounts
- Requires Power BI License
- Custom development not required
- Azure subscription not required

Power BI Embedded

- Accessed via custom URL
- No Office 365 accounts required
- No Power BI user licenses required
- Requires custom development
- Requires Azure subscription



The Big Picture for Power BI Embedded

1. Create > Design > Test a PBIX project file on local PC
 - Done using Power BI Desktop
 - Note that Power BI Desktop only runs on Windows
2. Provision Azure resources for Power BI Embedded
 - Create a Power BI workspace collection
 - Create Power BI workspaces
3. Upload PBIX file to Power BI Embedded workspace
 - Use PowerShell, Power BI CLI or Azure REST API
4. Develop Web App with Embedded Power BI Reports
 - Most easily accomplished using ASP.NET MVC



Agenda

- ✓ Power BI Embedded Overview
- Working with PBIX Project Files
 - Provisioning Workspaces in Azure
 - Embedding Reports in a MVC Application
 - Configuring Datasets for DirectQuery Mode
 - Putting It All Together



Working with Power BI Desktop

- Power BI Desktop used to design PBIX projects
 - Data Source and Query features for Data Discovery
 - Query features for ETL (extract-transform-load)
 - Data modeling features and DAX language
 - Report design features with a visual report designer

Tasks performed using Power BI Desktop

**Data
Discovery**

ELT

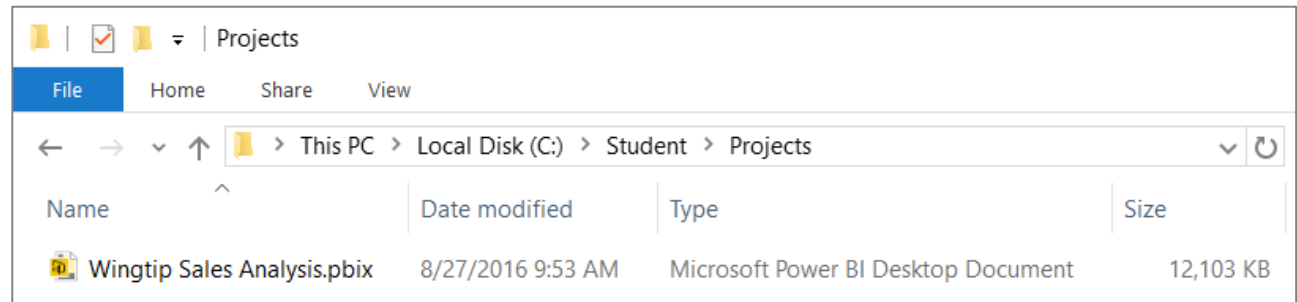
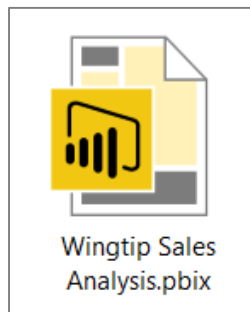
**Data
Modeling**

**Design
Reports**

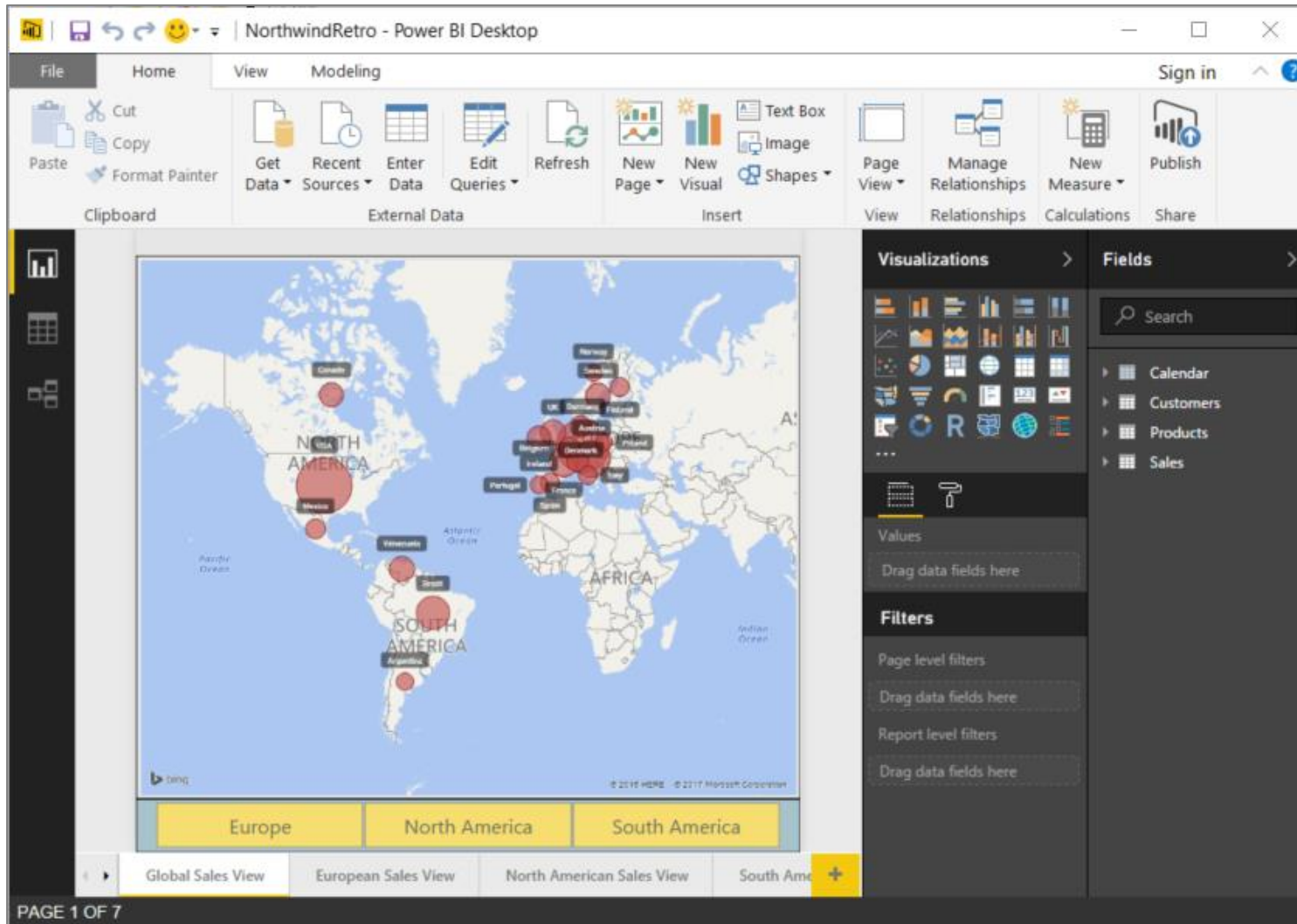


Projects and PBIX Files

- Power BI Desktop projects saved using PBIX files
 - PBIX file contains data source definitions
 - PBIX file contains query definitions
 - PBIX file contains data imported from queries
 - PBIX file contains exactly one data model definition
 - PBIX file contains exactly one report
 - PBIX file never contains data source credentials



Demo: NorthwindRetro.pbix



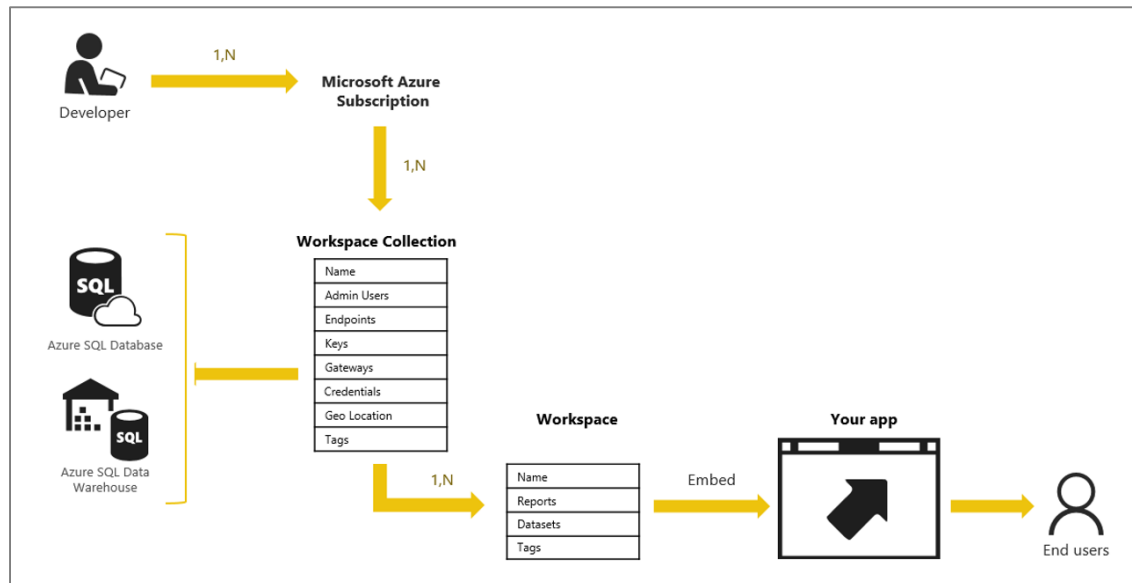
Agenda

- ✓ Power BI Embedded Overview
- ✓ Working with PBIX Project Files
- Provisioning Workspaces in Azure
 - Embedding Reports in a MVC Application
 - Configuring Datasets for DirectQuery Mode
 - Putting It All Together



Power BI Embedded Service in Azure

- Creating a Power BI workspace collection
 - Can be done by hand in Azure portal
 - Can be done with PowerShell and PowerBI-CLI utility
 - Can be done using C# and Azure ARM APIs
 - Workspace collection contains access keys



Create-Workspace-Collection.ps1

Create-Workspace-Collection.ps1 X

```
19 Login-AzureRmAccount -Credential $credential | Out-Null
20
21 $location = "southcentralus"
22 $resourceGroupName = "powerbi-demo"
23 $resourceGroup = Get-AzureRmResourceGroup -Name $resourceGroupName -ErrorAction Ignore
24
25 # Create resource group if it doesn't already exist
26 if(!$resourceGroup){
27     Write-Host "Resource group named" $resourceGroupName "does not exist - now creating it"
28     $resourceGroup = New-AzureRmResourceGroup -Name $resourceGroupName -Location $location
29 }
30
31 $workspaceCollectionName = "wingtip-sales"
32 $workspaceCollection = Get-AzureRmPowerBIWorkspaceCollection `
33     -ResourceGroupName $resourceGroupName `
34     -WorkspaceCollectionName $workspaceCollectionName `
35     -ErrorAction Ignore
36
37 # Create new workspace collection if it doesn't already exist
38 if(!$workspaceCollection){
39     Write-Host "Workspace collection named" $workspaceCollectionName "does not exist - now creating it"
40     $workspaceCollection = New-AzureRmPowerBIWorkspaceCollection `
41         -ResourceGroupName $resourceGroupName `
42         -WorkspaceCollectionName $workspaceCollectionName `
43         -Location $location
44 }
45
46 Write-Host
47 Write-Host "Workspace Collection Name: " + $workspaceCollection.Name
48 Write-Host "Workspace Collection Location: " + $workspaceCollection.Location
49 Write-Host "Workspace Collection ID: " + $workspaceCollection.Id
50 Write-Host
```



PowerBI-CLI

- PowerBI-CLI is command-line utility for Power BI
 - Maintained at <https://github.com/Microsoft/PowerBI-Cli>
 - Installed using Node Package Manager

```
npm install powerbi-cli -g
```
 - Supported on Windows, Mac and Linux
 - Can be accessed using any command prompt
 - Can be automated using PowerShell script

```
C:\WINDOWS\system32\cmd.exe

C:\GIT\PowerBiEmbedded\PowerShell>powerbi get-workspaces
[ powerbi ] =====
[ powerbi ] Gettings workspaces for Collection: wingtip-sales
[ powerbi ] =====
[ powerbi ] 76fc5d88-6957-4e4b-bb0e-b80ea9fa9cb4

C:\GIT\PowerBiEmbedded\PowerShell>_
```



Import-PBIX-Into-Workspace.ps1

Import-PBIX-Into-Workspace.ps1 X

```
43
44 $keys = Get-AzureRmPowerBIWorkspaceCollectionAccessKeys `
45     -ResourceGroupName $resourceGroupName `
46     -WorkspaceCollectionName $workspaceCollectionName
47
48 $accessKey = $keys[0].Value
49
50 # determine if there are any workspaces in this workspace collection
51 $workspaces = Get-AzureRmPowerBIWorkspace `
52     -ResourceGroupName $resourceGroupName `
53     -WorkspaceCollectionName $workspaceCollectionName
54
55 if(!$workspaces) {
56     Write-Host "This workspace collection has no workspaces. Creating new workspace..."
57     powerbi create-workspace -c $workspaceCollectionName -k $accessKey
58     $workspaces = Get-AzureRmPowerBIWorkspace `
59         -ResourceGroupName $resourceGroupName `
60         -WorkspaceCollectionName $workspaceCollectionName
61 }
62
63 # create variable to reference first workspace in workspace collection
64 $workspace = $workspaces[0].Name
65
66 Write-Host "Importing PBIX file into workspace with name of $workspace ..."
67
68 $pbixFilePath = "C:\git\PowerBIEmbedded\PBIX\NorthwindRetro.pbix"
69 $reportName = "Wingtip Sales"
70
71 $importResult = powerbi import -c $workspaceCollectionName -k $accessKey -w $workspace -f $pbixFilePath -n $reportName
72 Write-Host "PBIX Import information"
73 Write-Host "-----"
74 $importResult
75
76 $createEmbedTokenResult = powerbi create-embed-token -c $workspaceCollectionName -w $workspace -r $reportName -k $accessKey
77 Write-Host "Embed Token"
78 Write-Host "-----"
79 $createEmbedTokenResult
80
```



Agenda

- ✓ Power BI Embedded Overview
- ✓ Working with PBIX Project Files
- ✓ Provisioning Workspaces in Azure
- Embedding Reports in a MVC Application
 - Configuring Datasets for DirectQuery Mode
 - Putting It All Together



Embedding Internals

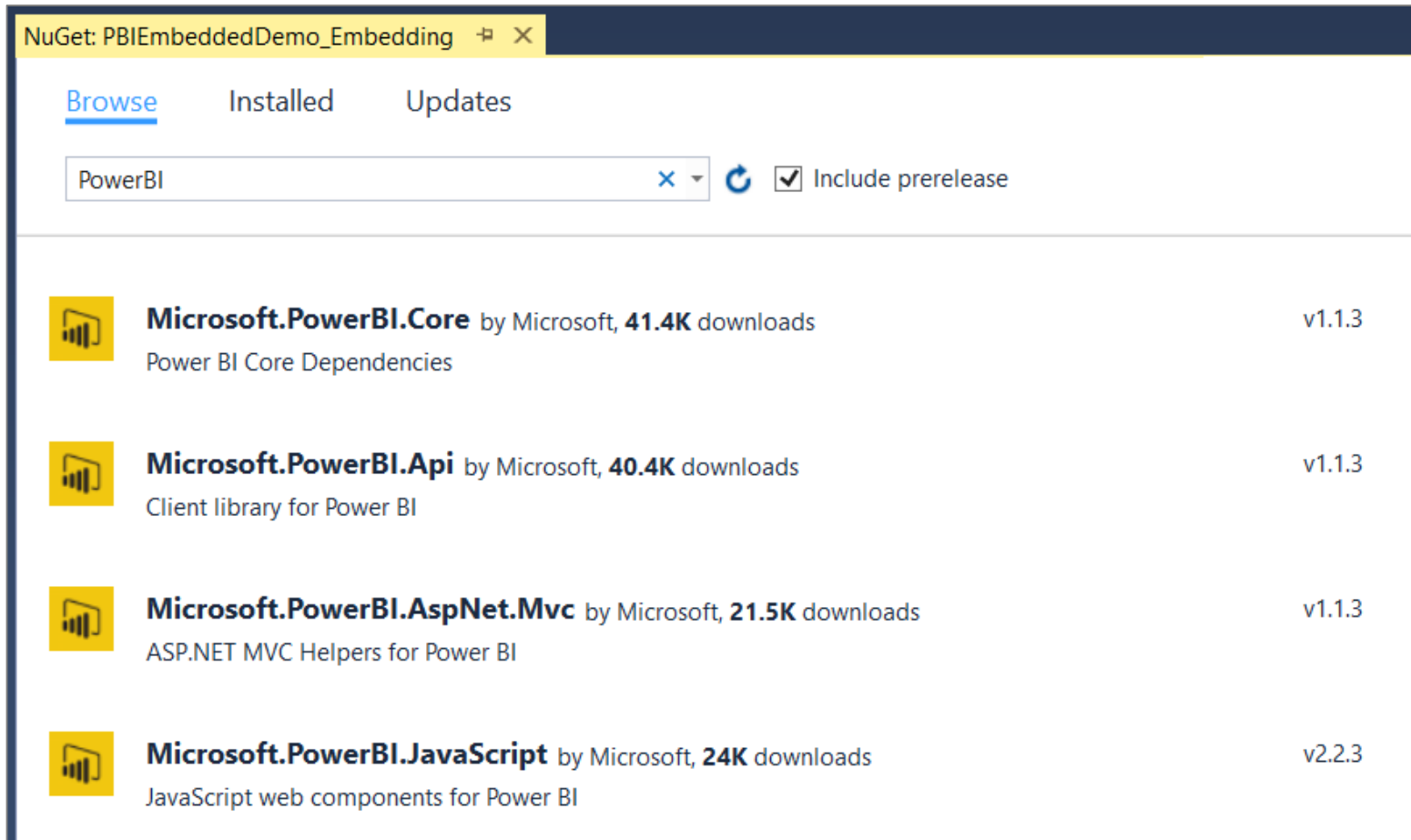
- How are Power BI Reports Embedded?
 - It's done using an iFrame
 - iFrame requires event to load embed token

```
<div id="pbi-report"
  style="height:95vh;"
  powerbi-type="report"
  powerbi-embed-url="https://embedded.powerbi.com/appTokenReportEmbed?reportId=8c6ec229-f3
  powerbi-access-token="eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ2ZXIiOiIwLjIuMCIsc
  <iframe
    src="https://embedded.powerbi.com/appTokenReportEmbed?reportId=8c6ec229-f3
    scrolling="no"
    allowfullscreen="true"
    style="width:100%;height:100%;">
  </iframe>
</div>
```



Creating an ASP.NET MVC Application

- Add the following Nuget Packages



The screenshot shows the NuGet Package Manager window for a project named 'PBIEmbeddedDemo_Embedding'. The 'Browse' tab is selected. A search box contains the text 'PowerBI'. To the right of the search box are icons for refreshing the package list and a checkbox labeled 'Include prerelease' which is checked. Below the search bar, four packages are listed, each with a yellow icon representing a bar chart:

Package Name	Author	Downloads	Version
Microsoft.PowerBI.Core	Microsoft	41.4K	v1.1.3
Power BI Core Dependencies			
Microsoft.PowerBI.Api	Microsoft	40.4K	v1.1.3
Client library for Power BI			
Microsoft.PowerBI.AspNet.Mvc	Microsoft	21.5K	v1.1.3
ASP.NET MVC Helpers for Power BI			
Microsoft.PowerBI.JavaScript	Microsoft	24K	v2.2.3
JavaScript web components for Power BI			



Power BI .NET APIs

■ ■ Microsoft.PowerBI.Core

- ▷ {} Microsoft.PowerBI
- ▲ {} Microsoft.PowerBI.Security

{ } Microsoft.PowerBI.Security

- ▷ 🛠 PowerBIToken
- ▷ 🛠 PowerBIToken.ClaimTypes
- ▷ 🛠 TokenManager

■ ■ Microsoft.PowerBI.Api

- ▷ {} Microsoft.PowerBI.Api.V1
- ▷ {} Microsoft.PowerBI.Api.V1.Models

{ } Microsoft.PowerBI.Api.V1

- ▷ 🛠 Datasets
- ▷ 🛠 DatasetsExtensions
- ▷ 🛠 Gateways
- ▷ 🛠 GatewaysExtensions
- ▷ 🔗 IDatasets
- ▷ 🔗 IGateways
- ▷ 🔗 IImports
- ▷ 🛠 Imports
- ▷ 🛠 ImportsExtensions
- ▷ 🔗 IPowerBIClient
- ▷ 🔗 IReports
- ▷ 🔗 IWorkspaces
- ▷ 🛠 PowerBIClient
- ▷ 🛠 Reports
- ▷ 🛠 ReportsExtensions
- ▷ 🛠 Workspaces
- ▷ 🛠 WorkspacesExtensions

{ } Microsoft.PowerBI.Api.V1.Models

- ▷ 🛠 BasicCredentials
- ▷ 🛠 Column
- ▷ 🛠 Dataset
- ▷ 🛠 Datasource
- ▷ 🛠 GatewayDatasource
- ▷ 🛠 Import
- ▷ 🛠 ImportInfo
- ▷ 🛠 ODataResponseListDataset
- ▷ 🛠 ODataResponseListDatasource
- ▷ 🛠 ODataResponseListGatewayDatasource
- ▷ 🛠 ODataResponseListImport
- ▷ 🛠 ODataResponseListReport
- ▷ 🛠 ODataResponseListTable
- ▷ 🛠 ODataResponseListWorkspace
- ▷ 🛠 Report
- ▷ 🛠 Row
- ▷ 🛠 Table
- ▷ 🛠 Workspace



MVC Helper Classes

- Microsoft.PowerBI.AsNet.Mvc
 - { } Microsoft.PowerBI.AsNet.Mvc.Html
 - ReportExtensions
 - TokenExtensions

- Microsoft.PowerBI.AspNet.Mvc
 - { } Microsoft.PowerBI.AspNet.Mvc.Html
 - ReportExtensions
 - TokenExtensions

- PowerBIReport(System.Web.Mvc.HtmlHelper, string, Microsoft.Pow
- PowerBIReport(System.Web.Mvc.HtmlHelper, string, string, object)
- PowerBIReportFor<TModel, TProperty>(System.Web.Mvc.HtmlHel

- Microsoft.PowerBI.AspNet.Mvc
 - Microsoft.PowerBI.AspNet.Mvc.Html
 - ReportExtensions
 - TokenExtensions

- PowerBIAccessToken(System.Web.Mvc.HtmlHelper, string)
- PowerBIAccessTokenFor<TModel, TProperty>(System.Web.M...

```
<div>  
    @Html.PowerBIAccessToken(Model.AccessToken)  
</div>
```



```
<div>
  <script>
    window.powerbi = window.powerbi || {};
    window.powerbi.accessToken = 'eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1I3NmZjNWQ4OC02OTU3LTRlNGItYmIwZS1iODBlYTltMTYjYjQlICJyYWQlOi3dIklJJU0RLiwiYXVkJjoiaHR0cHM6Ly9hbmFseXNpcy53aW5kb3dzLm5ldCQ.XKzudTN7LzLA1RDC4gY2AJF5-1-bEQeaXj1cps0EywA';
  </script>
</div>
```



Power BI JavaScript API

- Used to generate iFrame required for embedding
 - Call to **powerbi.embed** creates iFrame for embedding

```
var reportConfig = {  
  settings: {  
    filterPaneEnabled: false,  
    navContentPaneEnabled: false  
  }  
};  
  
var reportContainer = document.getElementById("pbi-report");  
var report = powerbi.embed(reportContainer, reportConfig);
```

- API supports binding to report-level events

```
report.on('loaded', onReportLoaded);  
report.on('pageChanged', onReportPageChanged);  
  
function onReportLoaded()...  
  
function onReportPageChanged(e)...
```



Demo: HelloPowerBIEmbedded.sln

The image shows a development environment with a Visual Studio Solution Explorer on the left and a web browser on the right. The Solution Explorer displays the project 'HelloPBIEmbedded' with a tree view of files including Properties, References, App_Data, App_Start, Content, Controllers, fonts, Models, Scripts, Views, Global.asax, packages.config, and Web.config. The web browser, titled 'Power BI Embedded', shows a report at 'localhost:19407'. The report features a world map with red circular markers of varying sizes representing sales data for various countries. On the left side of the browser window, there are two panels: 'Report Configuration Settings' with checkboxes for 'Show Navigation' and 'Show Filter Pane', and 'Page Navigation' with radio buttons for 'Global Sales View' (selected), 'European Sales View', 'North American Sales View', 'South American Sales View', 'Sales by Month', 'Geography Drilldown', and 'Sales by Product'. Below these are 'Previous Page' and 'Next Page' buttons. The map itself is labeled with continents (NORTH AMERICA, SOUTH AMERICA, AFRICA) and oceans (Pacific Ocean, Atlantic Ocean, Indian Ocean). Specific countries labeled on the map include Canada, Mexico, Venezuela, Brazil, Argentina, Norway, Sweden, UK, Germany, Finland, Austria, Poland, Belgium, Denmark, France, Spain, Portugal, and Italy. At the bottom of the browser window, there are three buttons: 'Europe', 'North America', and 'South America'. The footer of the browser window includes the Bing logo and copyright information: '© 2017 Microsoft Corporation © 2016 HERE'.



Agenda

- ✓ Power BI Embedded Overview
- ✓ Working with PBIX Project Files
- ✓ Provisioning Workspaces in Azure
- ✓ Embedding Reports in a MVC Application
- Configuring Datasets for DirectQuery Mode
- Putting It All Together



Imported Datasets Versus DirectQuery

- Imported Dataset
 - Data is imported into storage within Power BI cloud
 - Entire dataset loaded into memory when in use
 - Reports and Datasets query data in imported dataset
 - Dataset size is maxed out at 1GB
 - Dataset must be refreshed when source data changes
- DirectQuery Dataset
 - No data is imported or cached in the Power BI cloud
 - Reports and Datasets query for data using live connection
 - No need to refresh data
 - No need to worry about 1GB dataset size limitation
 - Limitation are placed on features for querying and data modeling



Supported Data Sources

Data source	Live/DirectQuery	User configured manual or scheduled refresh
Analysis Services Tabular	Yes	Yes
Analysis Services Multidimensional	Yes	Yes
SQL Server	Yes	Yes
SAP HANA	Yes	Yes
Oracle	Yes	Yes
Teradata	Yes	Yes
File	No	Yes
Folder	No	Yes
SharePoint list (on-premises)	No	Yes
Web	No	Yes
OData	No	Yes
IBM DB2	No	Yes
MySQL	No	Yes
Sybase	No	Yes



Limitations of DirectQuery

- DirectQuery imposes the following limitations
 - All tables must come from a single database
 - Many types of query steps are not supported
 - Relationship filtering limited to single direction
 - Time intelligence capabilities are not available
 - No special treatment of date columns
 - Calculated columns not allowed
 - By default, limitations placed on DAX in measures



Demo: WingtipSalesDirectQuery.pbix

- Power BI Embedded does not support dataset refresh
 - Now way to refresh imported dataset – different than PowerBI.com
 - Requires delete and reimporting updated PBIX project
 - DirectQuery mode can be used to eliminate the refresh problem



Provisioning with C# and Azure REST APIs

- Can be programmed in simple Console app
 - Use same Nuget packages as shown earlier
- Dataset management is important
 - Credentials cannot be included in PBIX
 - Credentials must be configured after PBIX import
- Terminology confusion
 - Remember that “Import” = “Dataset” = “Report”



Configuring a Dataset for DirectConnect

```
static void UpdateAzureSqlDataSource(string workspaceCollectionName, string workspaceId, string datasetId) {  
    using (var client = CreatePowerBIClient()) {  
        IList<Dataset> datasets = client.Datasets.GetDatasetsAsync(workspaceCollectionName, workspaceId).Result.Value;  
        foreach (Dataset dataset in datasets) {  
            if (dataset.Name == datasetId) {  
                var datasources = client.Datasets.GetGatewayDatasourcesAsync(workspaceCollectionName, workspaceId, dataset.Id).Result;  
                // Reset your connection credentials  
                var delta = new GatewayDatasource {  
                    CredentialType = "Basic",  
                    BasicCredentials = new BasicCredentials {  
                        Username = azureSqlUser,  
                        Password = azureSqlPassword  
                    }  
                };  
                // Update the datasource with the specified credentials  
                client.Gateways.PatchDatasourceAsync(workspaceCollectionName,  
                                                    workspaceId,  
                                                    datasources.Value[0].GatewayId,  
                                                    datasources.Value[0].Id,  
                                                    delta).Wait();  
            }  
        }  
    }  
}
```



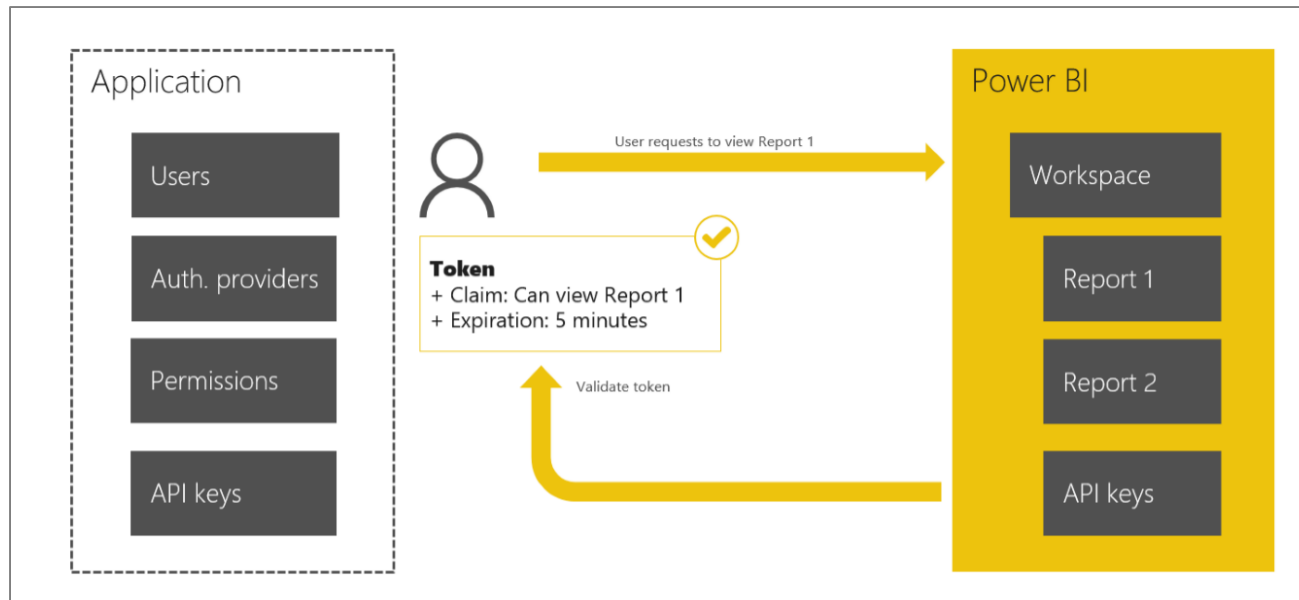
Agenda

- ✓ Power BI Embedded Overview
- ✓ Working with PBIX Project Files
- ✓ Provisioning Workspaces in Azure
- ✓ Embedding Reports in a MVC Application
- ✓ Configuring Datasets for DirectQuery Mode
- Putting It All Together



Security with Power BI Embedded

- Power BI Embedded defers to your app...
 - to perform all the necessary user authentication
 - to provide an authorization scheme



- PBI Embedded supports row-level security (RLS)



Summary

- ✓ Power BI Embedded Overview
- ✓ Working with PBIX Project Files
- ✓ Provisioning Workspaces in Azure
- ✓ Embedding Reports in a MVC Application
- ✓ Configuring Datasets for DirectQuery Mode
- ✓ Putting It All Together

