



Microsoft Business Application Summit

July 22–24, 2018
Seattle, WA





Embed Microsoft Power BI visual analytics into your application

Ted Pattison
Power BI MVP



Code and Slides for this Session

<https://github.com/CriticalPathTraining/PowerBiEmbedded>

The screenshot shows a GitHub repository page for the project 'CriticalPathTraining/DailyReporterPro'. The repository has 4 commits, 1 branch, 0 releases, 1 contributor (TedPattison), and is licensed under MIT. The latest commit was made a minute ago. The repository contains files like DailyReporterPro, .gitignore, DailyReporterPro.sln, LICENSE, and Power BI Embedded.pdf.

A sample app created with ASP.NET MVC to demonstrate 3rd party embedding with Power BI

4 commits 1 branch 0 releases 1 contributor MIT

Branch: master New pull request Create new file Upload files Find file Clone or download

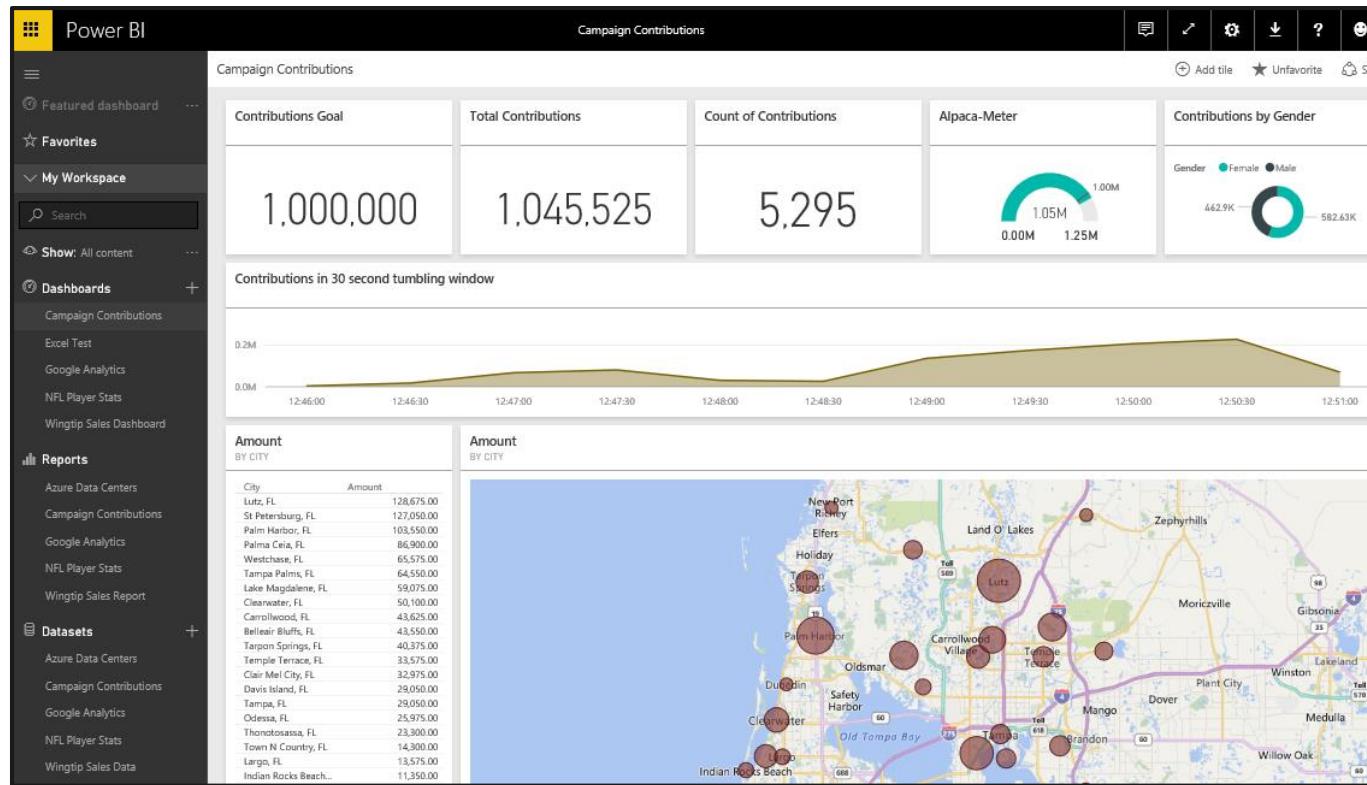
File	Description	Time
TedPattison Updates	Latest commit 3cc061d a minute ago	
DailyReporterPro	Updates for d.ts files	22 hours ago
.gitignore	Updates for d.ts files	22 hours ago
DailyReporterPro.sln	Initial Upload	a day ago
LICENSE	Initial commit	a day ago
Power BI Embedded.pdf	Updates	a minute ago

Agenda

- Power BI Embedding Fundamentals
- Managing Content using App Workspaces
- Understanding Dedicated Capacities
- Using the Onboarding Experience Tool
- Programming with Power BI Service API
- Embedding with the Power BI JavaScript API

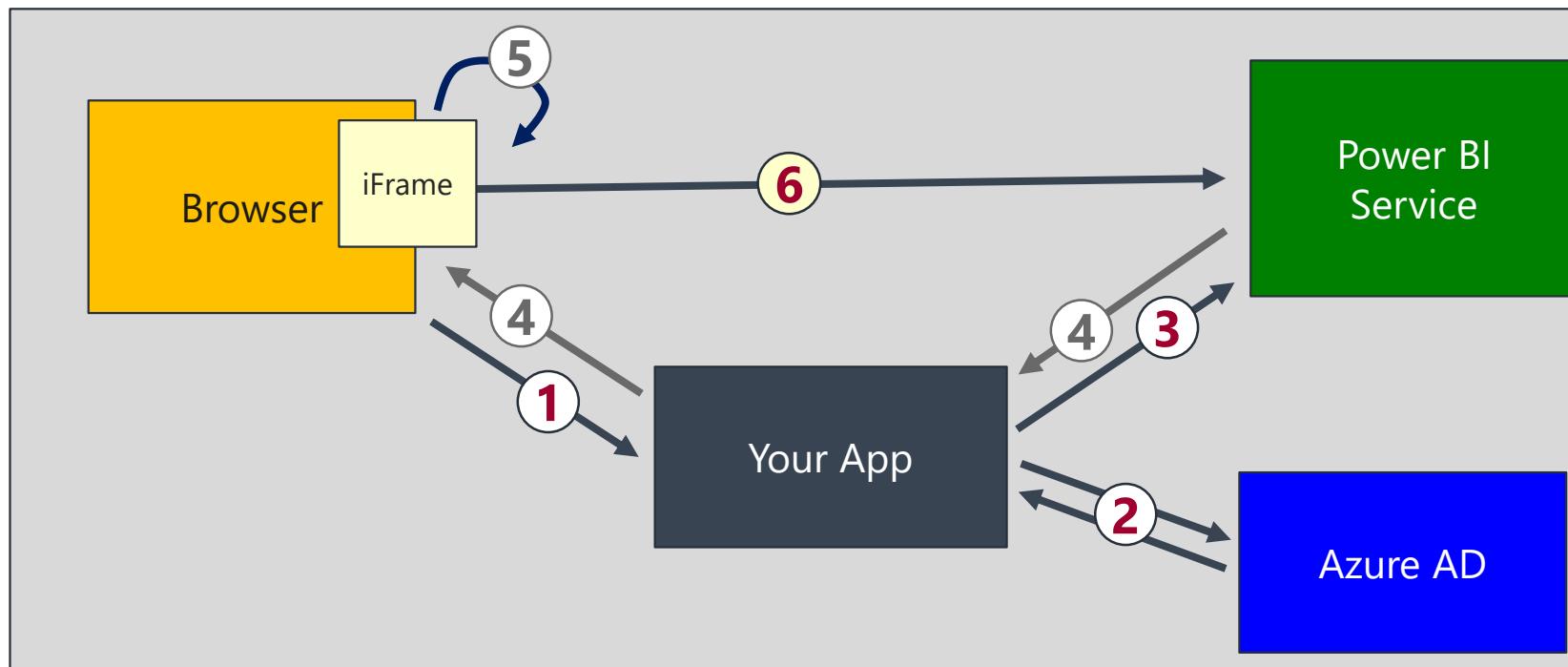
The Power BI Service – Who Is It For?

- Provides SaaS service used by web and mobile users
 - Accessible at <https://app.powerbi.com> to browser and Power BI mobile users
- Provides PaaS service used by software developers
 - Accessible at <https://api.powerbi.com> to developers using Power BI Service API



Power BI Embedding – The Big Picture

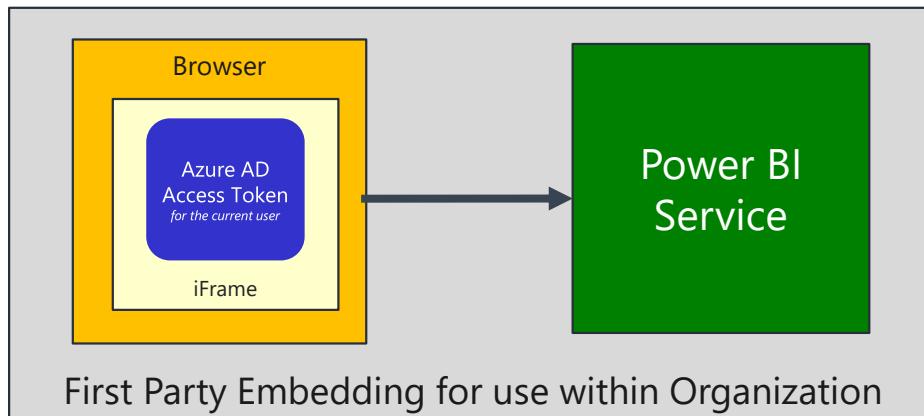
- User launches your app using a browser
- App authenticates with Azure Active Directory and obtains access token
- App uses access token to call to Power BI Service API
- App retrieves data for embedded resource and passes it to browser.
- Client-side code uses Power BI JavaScript API to create embedded resource
- Embedded resource session created between browser and Power BI service



First Party Embedding versus Third Party Embedding

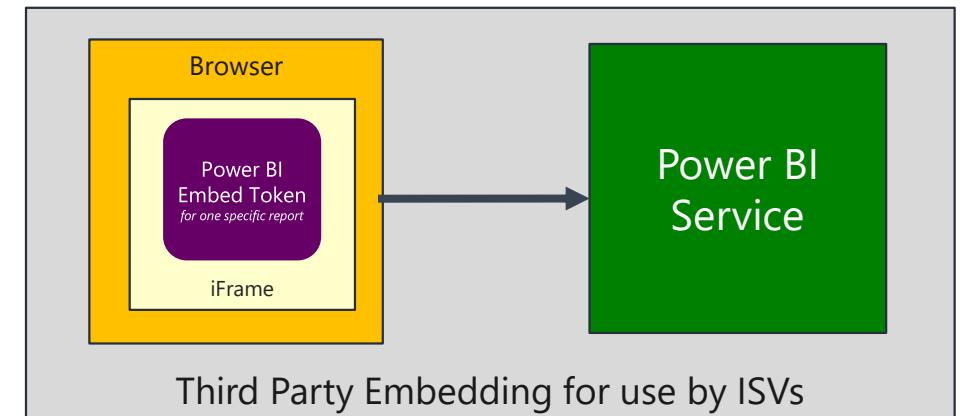
First Party Embedding

- Known as **User-Owns-Data** Model
- All users require a Power BI license
- Useful in corporate environments
- App authenticates as current user
- Your code runs with user's permissions
- User's access token passed to browser



Third Party Embedding

- Known as **App-Owns-Data** Model
- No users require Power BI license
- Useful for commercial applications
- App authenticates with master user account
- Your code runs with admin permissions
- Embed token passed to browser



Embeddable Resources

- Reports
- Dashboards
- Dashboard Tiles
- New Reports
- Q&A Experience
- Visuals in custom layout

Demo Time

Exploring the Embeddable Resources
available with Power BI embedding

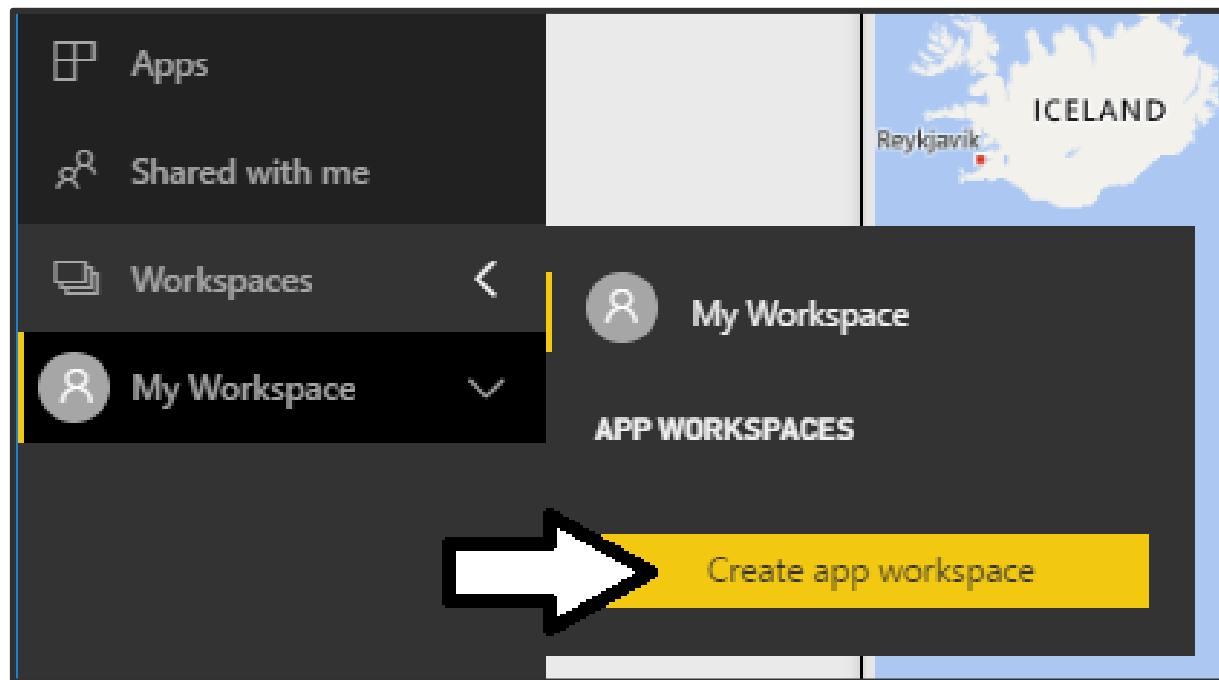


Agenda

- Power BI Embedding Fundamentals
- **Managing Content using App Workspaces**
- Understanding Dedicated Capacities
- Using the Onboarding Experience Tool
- Programming with Power BI Service API
- Embedding with the Power BI JavaScript API

Solution Deployment using App Workspaces

- App workspaces used to deploy custom solutions
 - App workspaces required for team-based development
 - App workspace can be secured using private membership
 - App workspace used to publish apps for licensed users
 - App workspace used to build Power BI embedding solutions



Create an app workspace

Name your workspace
Ted Pattison

Workspace ID
tedpattison
Available

Private - Only approved members can see what's inside

Members can edit Power BI content

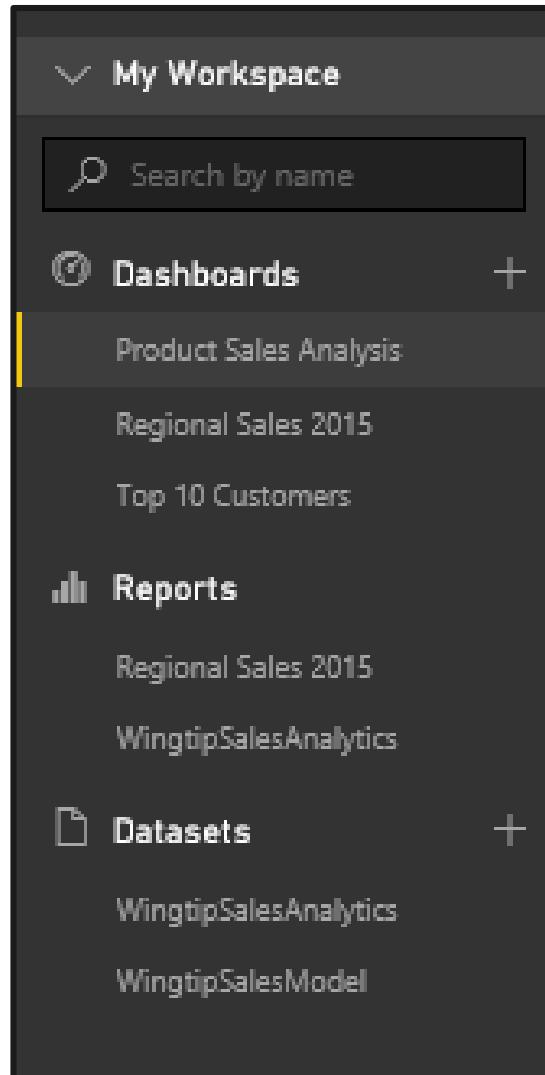
Add workspace members
Enter email addresses
Add
criticalpathuser42@powerbimvps.onmicrosoft.com Admin

Advanced

Save Cancel

This is a detailed view of the 'Create an app workspace' dialog box. It includes fields for the workspace name ('Name your workspace'), workspace ID ('Workspace ID'), security settings ('Private - Only approved members can see what's inside'), member editing permissions ('Members can edit Power BI content'), and workspace members ('Add workspace members'). The 'Add' button is highlighted with a yellow background. At the bottom, there are 'Save' and 'Cancel' buttons.

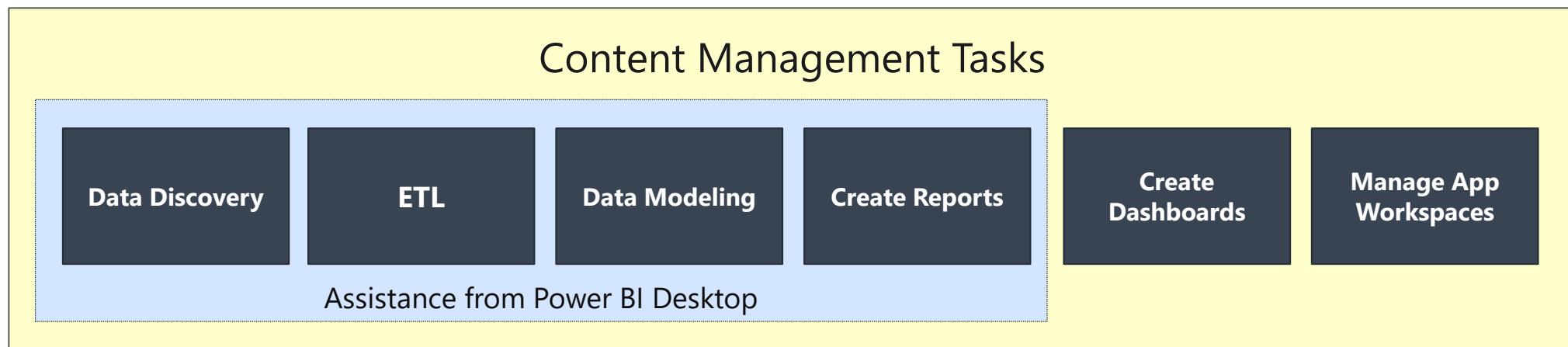
Adding Content to a Power BI App Workspace



- Dashboard
 - Consolidated view of insights
 - Often used for navigation across reports
 - Automatically updates when viewing real-time datasets
 - No support for interactive filtering
- Report
 - Collection of pages which contain visualizations
 - Provides user with interactive filtering
 - Supports bookmarks to enable rich story telling
- Dataset
 - Data model containing one or more tables
 - Usually designed using Power BI Desktop

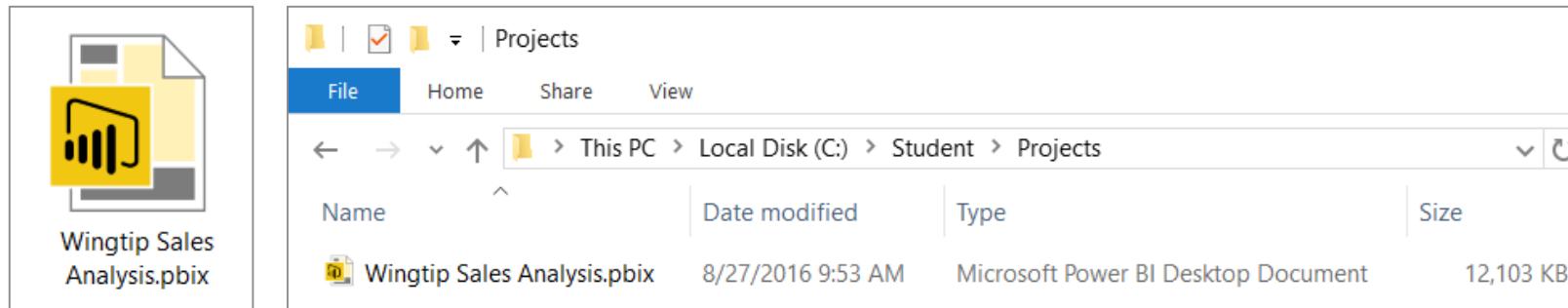
Working with Power BI Desktop

- Power BI Desktop focuses on first four phases
 - Query features for Data Discovery and ETL
 - Data modeling features and DAX language for building data model
 - Report design features for building interactive report
 - No support for building dashboards
 - No support for packaging an entire solution



Power BI Desktop 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



- **PBIX files can be tracked in source control** (e.g. github.com, TFS, etc.)
 - Some Power BI resources (e.g. Dashboard) have no backing file support

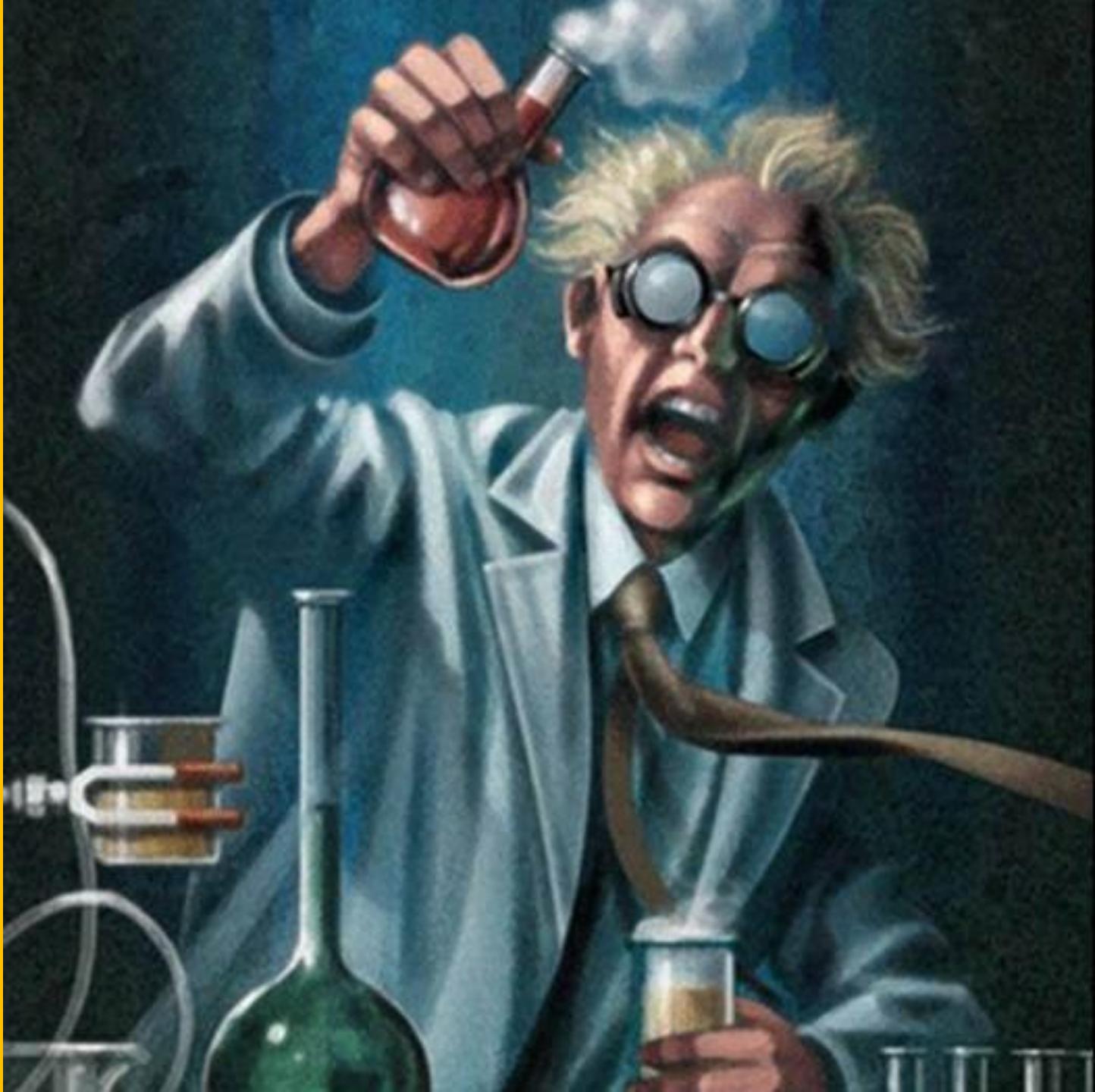
Division of Labor with Power BI Embedding

- Content Management Team
 - Build Power BI Desktop projects
 - Publish reports & datasets via PBIX files
 - Create dashboards in PBI Service
 - Publish App Workspaces as Apps
 - Monitor Power BI environment
- Application Development Team
 - Develop web apps with PBI embedding
 - Authenticate with Azure Active Directory
 - Retrieve data using Power BI Service API
 - Embed resources using Power BI JavaScript API
 - This team sees PBIX files as black boxes



Demo Time

Embedding App Workspace
Resources using DailyReporterPro



Agenda

- Power BI Embedding Fundamentals
- Managing Content using App Workspaces
- **Understanding Dedicated Capacities**
- Using the Onboarding Experience Tool
- Programming with Power BI Service API
- Embedding with the Power BI JavaScript API

Dedicated Capacities

- Power BI workspaces run in one of two possible environments
 - Shared capacity
 - Dedicated capacity
- Dedicated capacity required for third party embedding
 - You pay Microsoft capacity-based fee for processors cores and RAM
 - No need to pay Microsoft for user licensing
- Dedicated capacity can optionally be used for first party embedding
 - Allows users to run with free license instead of Power BI Pro license (\$10/month)
- Dedicated capacities come in two flavors
 - Power BI Premium capacities purchased through Office 365
 - Power BI Embedded capacities purchased through Azure SKU

Power BI Licensing Through Office 365

User-based licensing

Power BI (free)

\$0.00 user/month

A cloud-based business analytics service that enables anyone to visualize and analyze data with greater speed, efficiency, and understanding. It ...

Office 2016 desktop & mobile apps
Not included

Office 365 services



...

Power BI Pro

\$9.99 user/month

A cloud-based business analytics service that enables anyone to visualize and analyze data with greater speed, efficiency, and understanding. Power ...

Office 2016 desktop & mobile apps
Not included

Office 365 services



...

Office 365 Enterprise E5

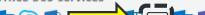
\$35.00 user/month

The Office suite, plus email, instant messaging, HD video conferencing, 1 TB personal file storage and sharing, and advanced security, analytics and PSTN ...

Office 2016 desktop & mobile apps



Office 365 services



...

Capacity-based Licensing

Power BI Premium P1

\$4,995.00 instance/month

Power BI capacity dedicated to your organization, unlocking unlimited content distribution and dependable performance. P1 offers 8 virtual core ...

Office 2016 desktop & mobile apps
Not included

Office 365 services



...

Power BI Premium P2

\$9,995.00 instance/month

Power BI capacity dedicated to your organization, unlocking unlimited content distribution and dependable performance. P1 offers 16 virtual core ...

Office 2016 desktop & mobile apps
Not included

Office 365 services



...

Power BI Premium P3

\$19,995.00 instance/month

Power BI capacity dedicated to your organization, unlocking unlimited content distribution and dependable performance. P1 offers 32 virtual core ...

Office 2016 desktop & mobile apps
Not included

Office 365 services



...

Power BI Premium P4

\$39,995.00 instance/month

Power BI capacity dedicated to your organization, unlocking unlimited content distribution and dependable performance. P4 offers 64 virtual core ...

Office 2016 desktop & mobile apps
Not included

Office 365 services



...

Power BI Premium P5

\$79,995.00 instance/month

Power BI capacity dedicated to your organization, unlocking unlimited content distribution and dependable performance. P5 offers 128 virtual ...

Office 2016 desktop & mobile apps
Not included

Office 365 services



...

Power BI Premium EM3 (Month to Month)

\$2,495.00 instance/month

Embed Power BI content in your custom application, powered by 4 virtual cores of dedicated capacity. Some Premium features are disabled ...

Office 2016 desktop & mobile apps
Not included

Office 365 services



...

Premium Capacity Nodes

- Power BI Premium Purchased using Nodes
 - Node type defines v-core and RAM capabilities
 - P nodes used for Power BI Service deployments and embedded deployments
 - EM nodes only used for embedded deployments

Capacity Node	Total cores	Backend Cores	Frontend Cores	Direct Query Limits	Page renders/hour
EM1	1 v-cores	.5 cores, 3GB RAM	.5 cores		1-300
EM2	2 v-cores	1 core, 5GB RAM	1 core		301-600
EM3	4 v-cores	2 cores, 10GB RAM	2 cores		601-1,200
P1	8 v-cores	4 cores, 25GB RAM	4 cores	30 per second	1,201-2,400
P2	16 v-cores	8 cores, 50GB RAM	8 cores	60 per second	2,401-4,800
P3	32 v-cores	16 cores, 100GB RAM	16 cores	120 per second	4,801-9600
P4	64 v-cores	32 cores, 200GB RAM	32 cores	240 per second	9601-19,200
P5	128 v-cores	64 cores 400GB	64	480 per second	19,201- 38,400

Managing Premium Capacities

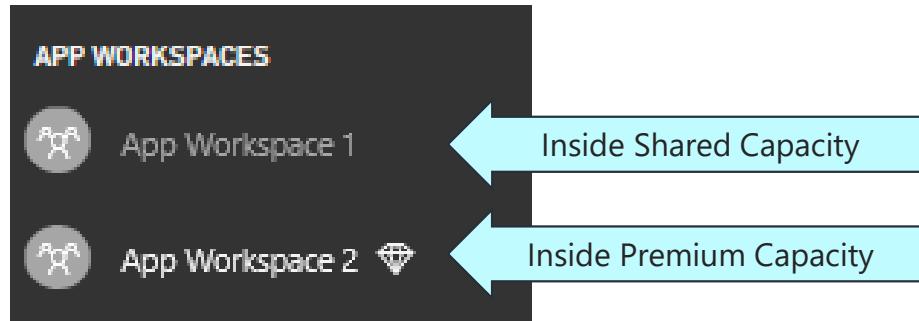
- Capacities managed in Power BI Admin Portal

The screenshot shows the Power BI Admin portal interface. The top navigation bar includes the Power BI logo, 'Admin portal', and various icons for notifications, settings, and help. The main menu on the left has sections for 'Favorites', 'Recent', 'Apps', 'Shared', 'Workspaces', and 'My Workspaces'. The 'My Workspaces' section is currently selected and shows a list of workspaces: 'Adventure Works' and 'ChucksAADGroup'. The workspace 'Adventure Works' is highlighted. The central content area displays usage metrics for the selected workspace: 'Power BI Premium > P1'. It shows 'USAGE IN THE LAST 7 DAYS' with four status boxes: CPU (Exceeded 80% utilization 0 times), Memory Thrashing (Exceeded 80% utilization 0 times), Memory Usage (Average: 2GB), and Direct Query (Exceeded 80% utilization 0 times). Below these metrics, there are sections for 'CAPACITY SIZE' (P1, 8 v-cores), 'REGION' (East US 2), 'USER PERMISSIONS' (Capacity admins), and 'WORKSPACES (53)'. A table at the bottom lists workspaces with columns for 'WORKSPACE NAME', 'WORKSPACE ADMINS', 'ACTIONS', and 'STATUS'. The 'Adventure Works' workspace is listed with 'View admins' under 'WORKSPACE ADMINS' and 'Assigned' under 'STATUS'.

WORKSPACE NAME	WORKSPACE ADMINS	ACTIONS	STATUS
Adventure Works	View admins	X	Assigned
ChucksAADGroup	View admins	X	Assigned

Associating App Workspaces with a Capacity

- App Workspace in Premium Capacity has diamond icon



- App workspace moved into Premium capacity in Advanced settings

Edit workspace

Advanced ^

Premium On

Choose an available dedicated capacity for this workspace

Capacity P1 #1 8GB model

Learn more about Premium capacity

Creating the Power BI Embedded Service

- Power BI Embedded in an Azure on-demand service
 - Must be created in same location as Power BI Service for tenant
 - Can be created manually through the Azure portal
 - Can be created in automated fashion using PowerShell
 - Requires an Azure subscription

The screenshot shows the Microsoft Azure Marketplace interface. On the left, there's a sidebar with navigation links like 'Create a resource', 'All services', 'All resources', 'Resource groups', 'App Services', 'SQL databases', and 'SQL servers'. The main area is titled 'Marketplace' and shows a search bar with the text 'power bi'. Below the search bar, the results table has columns: NAME, PUBLISHER, and CATEGORY. A single result is listed: 'Power BI Embedded' by Microsoft, categorized under 'Data + Analytics'.

This screenshot shows the configuration page for creating a new Power BI Embedded service. It includes fields for 'Resource name' (set to 'myfirstcapacity'), 'Subscription' (set to 'Pay-As-You-Go'), 'Resource group' (radio button selected for 'Create new', set to 'biz-app-summit-demos'), 'Power BI capacity administrator' (set to 'TedP@BizAppSummit.onmicrosoft.com'), 'Location' (set to 'West US 2'), and 'Pricing tier' (set to 'A1 (1 V-Cores)'). At the bottom are 'Create' and 'Automation options' buttons.



Search (Ctrl+/)

Pause Move Delete

Essentials ^

Resource group (change)

biz-app-summit-demos

Resource name

myfirstcapacity

Pricing tier

A1

Status

Active

Location

West US 2

Subscription name (change)

Pay-As-You-Go

Subscription ID

63bf3c07-a58b-4940-ab25-a6a66dd26dc

SCALE

Pricing tier

A1

1 V-Cores

Up to 3 GB Cache

Shared infrastructure

750.03

USD/MONTH (ESTIMATED)

A2

2 V-Cores

Up to 5 GB Cache

Shared infrastructure

1,494.03

USD/MONTH (ESTIMATED)

A3

4 V-Cores

Up to 10 GB Cache

Dedicated service

2,994.00

USD/MONTH (ESTIMATED)

A4

8 V-Cores

Up to 25 GB Cache

Dedicated service

5,994.04

USD/MONTH (ESTIMATED)

A5

16 V-Cores

Up to 50 GB Cache

Dedicated service

11,994.02

USD/MONTH (ESTIMATED)

A6

32 V-Cores

Up to 100 GB Cache

Dedicated service

23,994.45

USD/MONTH (ESTIMATED)

SETTINGS

Quick Start

Power BI capacity administrators

Properties

Locks

Automation script

MONITORING

Diagnostics logs

Metrics

Alerts



Managing Power BI Embedded Capacities

- Managing Power BI Embedded capacity done Azure portal
 - Use Azure portal to assign administrators and change pricing tier to scale up/down
- Managing Power BI Embedded capacity in Power BI Admin portal
 - Power BI Admin portal to assign workspaces to capacities

The screenshot shows the Power BI Admin portal interface. The top navigation bar includes the 'Power BI' logo, 'Admin portal', and various icons for export, settings, download, help, and search. On the left, a sidebar lists 'Favorites', 'Recent', 'Apps', 'Shared with me', 'Workspaces', and 'My Workspace'. The main content area is titled 'Admin portal' and contains links for 'Usage metrics', 'Users', 'Audit logs', 'Tenant settings', and 'Capacity settings'. The 'Capacity settings' link is highlighted. Below these are 'Embed Codes' and 'Organization visuals'. The right side features two tabs: 'Power BI Premium' and 'Power BI Embedded', with 'Power BI Embedded' selected. A table displays capacity details: CAPACITY NAME (myfirstcapacity), CAPACITY ADMINS (Maxwell Smart, Ted Pattison), ACTIONS (gear icon), SKU (A1), REGION (West US 2), and STATUS (Active). At the bottom of the table, there is a link to 'Set up new capacity in Azure'.

CAPACITY NAME	CAPACITY ADMINS	ACTIONS	SKU	REGION	STATUS
myfirstcapacity	Maxwell Smart, Ted Pattison		A1	West US 2	Active

PBI Capacity SKU Decoder Ring

	P SKU	EM SKU	A SKU
Purchased through...	Office 365	Office 365	Azure
Supports Power BI embedding in custom applications	Yes	Yes	Yes
Supports 3 rd Party Embedding in custom applications	Yes	Yes	Yes
Supports 1 st Party Embedding in custom applications	Yes	Yes	No
Supports free users accessing content in app.powerbi.com	Yes	No	No
Supports free users accessing content in Power BI Mobile	Yes	No	No
Billing cycle	Monthly	Monthly	Hourly
Commitment	Monthly	Monthly/Yearly	None
Turn it off when your not using it	No	No	Yes

Agenda

- Power BI Embedding Fundamentals
 - Managing Content using App Workspaces
 - Understanding Dedicated Capacities
- **Using the Onboarding Experience Tool**
- Programming with Power BI Service API
 - Embedding with the Power BI JavaScript API

Understanding Authentication with Azure AD

- Requirements of calling the Power BI Service API
 - App must be registered with Azure AD with Application ID (aka Client ID)
 - App must be created with app type of as **Web app / API or Native**
 - App registration must be configured with required permissions
 - App must implement authentication flow to authenticate user and obtain access token

The screenshot shows the Microsoft Azure portal interface. On the left, there is a dark sidebar with various service icons and links: Create a resource, All services, Favorites, Dashboard, All resources, Resource groups, App Services, Function Apps, SQL databases, Azure Active Directory (which is highlighted), and Virtual machines. The main content area has a title bar: Home > critical path training labs - App registrations. Below this is a sub-header: critical path training labs - App registrations (Azure Active Directory). The page features a search bar, a 'New application registration' button, an 'Endpoints' link, and a 'Troubleshoot' link. A note says: 'To view and manage your registrations for converged applications, please visit the Microsoft Application Console.' There is a search bar labeled 'Search by name or AppID' and a dropdown menu set to 'All apps'. A table lists six application registrations:

DISPLAY NAME	APPLICATION TYPE	APPLICATION ID
PB Power BI Day SPA	Web app / API	f892f67e-8683-4b2f-8eef-ee5221d90046
PB Power BI Embedding with React Demo	Web app / API	5f367be1-1bd4-42a4-abc0-7c994efc3a86
DR Daily Reporter Pro	Native	12b6f8ba-0af7-46bc-b672-12f02b28d194
MU My User-Owns-Data App	Web app / API	70c57777-bb2f-4079-bc4d-aca03c51dc78
PB Power BI Embedded Scatchpad	Native	01139d48-bac8-4811-bb7b-4f749cb04e1c
PB Power BI Custom Connector	Web app / API	2ff49d4f-7cf4-4a5d-a61f-4f1971cc0f09

Power BI Embedding Onboarding Experience Tool

<https://app.powerbi.com/embedsetup>

The screenshot shows a web browser window titled "Onboarding Embed Tool". The address bar indicates a secure connection to <https://app.powerbi.com/embedsetup>. The main content area has a yellow header bar with the "Power BI for Developers" logo. Below it, the title "Set up your Power BI embedding environment" is displayed in large, bold, dark font. A descriptive text follows: "Choose whether you'd like to embed Power BI in an application for your customers or in an application, website, or portal for your organization's internal users." Two main options are presented in boxes:

- Embed Power BI in your product for your customers—for ISVs and developers**
Users will not need a Power BI account to view and interact with embedded data.
- Embed Power BI for your organization's internal users—for enterprises**
Users will need a Power BI account and permission to access the underlying content to view and interact with embedded data.

At the bottom, two buttons are shown: "Embed for your customers" (under the first option) and "Embed for your organization" (under the second option). Red arrows at the bottom point to these buttons: a red arrow pointing right from the text "Third Party Embedding" points to the "Embed for your customers" button; another red arrow pointing left from the text "First Party Embedding" points to the "Embed for your organization" button.

App-Owns-Data Experience – Step 1

The screenshot shows a web browser window titled "Onboarding Embed Tool". The URL is https://app.powerbi.com/embedsetup/AppOwnsData?session_id=8a6d3ea7-a9df-46cd-a488-35311b624a97. The page has a yellow header bar with the "Power BI for Developers" logo. The main content area has a white background and features the title "Set up your Power BI embedding environment" in large, bold, dark text. Below the title, there is a brief description: "With this quick, 5-step process, you can set up everything you need to start embedding analytics in your application. If you encounter any issue, use our [troubleshooting page](#)". To the left, a vertical list of steps is shown with icons: STEP 1 (blue circle) "Sign in to Power BI", STEP 2 (white circle) "Register your application", STEP 3 (white circle) "Create an app workspace", STEP 4 (white circle) "Import content", and STEP 5 (white circle) "Grant permissions". The "Sign in to Power BI" step is highlighted with a blue circle. To the right of the steps, there is a yellow summary box containing the text "Summary: User: Ted Pattison" and a "Next" button.

Onboarding Embed Tool

Secure | https://app.powerbi.com/embedsetup/AppOwnsData?session_id=8a6d3ea7-a9df-46cd-a488-35311b624a97

Power BI for Developers

Set up your Power BI embedding environment

With this quick, 5-step process, you can set up everything you need to start embedding analytics in your application. If you encounter any issue, use our [troubleshooting page](#)

STEP 1
Sign in to Power BI

Welcome, Ted Pattison! (Wrong account? No problem, [logout](#) and try again.)

STEP 2
Register your application

STEP 3
Create an app workspace

STEP 4
Import content

STEP 5
Grant permissions

Summary:
User: Ted Pattison

Next

App-Owns-Data Experience – Step 2

STEP 2

Register your application

Register your application with Azure AD to allow your application access to the Power BI REST APIs and to set resource permissions for your app. You can change this later in the Microsoft Azure portal. [Learn more](#)

Application Name
Enter a display name to identify your application in Azure

API access
Select the APIs and the level of access your app needs. You can change these settings later in the Azure portal.

[Learn more](#)

Select All

Read APIs ⓘ	Write APIs ⓘ	Create APIs ⓘ
<input checked="" type="checkbox"/> Read All Datasets	<input type="checkbox"/> Read and Write All Datasets	<input type="checkbox"/> Create Content
<input checked="" type="checkbox"/> Read All Dashboards	<input type="checkbox"/> Read and Write All Dashboards	
<input checked="" type="checkbox"/> Read All Reports	<input type="checkbox"/> Read and Write All Reports	
<input checked="" type="checkbox"/> Read All Groups	<input type="checkbox"/> Read and Write All Workspaces	
<input checked="" type="checkbox"/> Read All Workspaces	<input type="checkbox"/> Read and Write All Capacities	
<input checked="" type="checkbox"/> Read All Capacities		

By clicking Register, you agree to the [terms of use](#)

Register

What Just Happened?

- The Onboarding Tool created a new Azure AD App
 - Configured as a **Native** app as opposed to a **Web app / API**

The screenshot shows the Microsoft Azure portal's 'App registrations' page. The left sidebar includes links for 'Create a resource', 'All services', 'Dashboard', 'All resources', 'Resource groups', 'App Services', 'Function Apps', 'SQL databases', 'Azure Cosmos DB', and 'Virtual machines'. The 'App registrations' link is highlighted with a blue background and a black arrow pointing to it. The main area displays a list of registered applications, with 'My First Power BI Embedding App' being the most recent addition, indicated by a green icon and a dashed border.

DELEGATED PERMISSIONS	REQUIRES ADMIN
Read and write all dataflows	No
View all dataflows	No
Read and write all content in tenant	Yes
Read and Write all Reports	No
<input checked="" type="checkbox"/> View users Groups	No
View all Groups	No
<input checked="" type="checkbox"/> View all Reports (preview)	No
Create content (preview)	No
View content properties (preview)	No
<input type="checkbox"/> Read and Write all Datasets	No
<input checked="" type="checkbox"/> View all Datasets	No
<input checked="" type="checkbox"/> View all Dashboards (preview)	No
Add data to a user's dataset (preview)	No
Read and Write all Dashboards	No
View all content in tenant	Yes
Read and write all workspaces	No
<input checked="" type="checkbox"/> View all workspaces	No
Read and write all capacities	No
<input checked="" type="checkbox"/> View all capacities	No

App-Owns-Data Experience – Step 3

The screenshot shows a step-by-step guide for creating an app workspace. Step 3 is highlighted with a blue circle. A summary box on the right provides details about the user and application.

STEP 1
Sign in to Power BI

STEP 2
Register your application

STEP 3
Create an app workspace

In order to start adding content to Power BI, you need to create an app workspace. You can manage your app workspace in the Power BI web service. [Learn more](#)

Name your app workspace
My New App Workspace

Create app workspace **Skip**

STEP 4
Import content

STEP 5
Grant permissions

Summary:
User: Ted Pattison
Application: My First Power BI Embedding App
Application ID:
122e858a-1b70-40f8-aa5e-de0ff4f6fcab

App-Owns-Data Experience – Step 4



STEP 1

Sign in to Power BI



STEP 2

Register your application



STEP 3

Create an app workspace



STEP 4

Import content

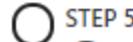
To embed Power BI content in your app, you can use a sample report to get started, or upload your own .pbix file.

Sample Power BI report

Upload .pbix file

Import

Skip



STEP 5

Grant permissions

Summary:

User: Ted Pattison

Application: My First Power BI Embedding App

Application ID:

122e858a-1b70-40f8-aa5e-de0ff4f6fc6

Workspace: My New App Workspace

Workspace ID:

ed3bd652-89c9-4262-8ac0-4757b38ebf69

What Just Happened?

- The Onboarding Tool created a new Power BI App Workspace
 - It also uploaded a PBIX file so there is already one dataset and one report
 - New app workspace is not automatically associated with a dedicated capacity

The screenshot shows the Power BI service interface for the 'My New App Workspace'. The left sidebar displays sections for DASHBOARDS, REPORTS, WORKBOOKS, and DATASETS. The REPORTS section lists 'US Sales Analysis'. The main content area shows a navigation bar with 'Power BI' and 'My New App Workspace' tabs, along with a search bar and filters for Dashboards, Reports, Workbooks, and Datasets. A list of reports is displayed, with 'US Sales Analysis' being the first item.

Category	Item
DASHBOARDS	(None)
REPORTS	US Sales Analysis
WORKBOOKS	(None)
DATASETS	US Sales Analysis

Category	Item
Favorites	(None)
Recent	(None)
Apps	(None)
Shared with me	(None)
Workspaces	My New App Workspace

Category	Item
Dashboards	(None)
Reports	US Sales Analysis
Workbooks	(None)
Datasets	(None)

App-Owns-Data Experience – Step 5

STEP 1
Sign in to Power BI

STEP 2
Register your application

STEP 3
Create an app workspace

STEP 4
Import content

STEP 5
Grant permissions

Grant permissions for your application to access the selected APIs with the signed in account.
[Learn how to grant permissions directly in Azure Portal.](#)

Grant permissions

Summary:

User: Ted Pattison

Application: My First Power BI Embedding App

Application ID:

122e858a-1b70-40f8-aa5e-de0ff4f6fcfa6

Workspace: My New App Workspace

Workspace ID:

ed3bd652-89c9-4262-8ac0-4757b38ebf69

Report ID:

a5c37d6c-f26d-4565-a5e7-19fe1a3fcb79

Granting Permissions to the App

- Users must consent to the permissions requested by an app

STEP 5
Grant permissions

Grant permissions for your application to access the selected APIs with the signed in account.
Learn how to grant permissions directly in Azure Portal.

Grant permissions

 Microsoft
tedp@bizappsummit.onmicrosoft.com

Permissions requested

My First Power BI Embedding App

This app would like to:

- ✓ View users Groups
- ✓ View all Reports (preview)
- ✓ View all Datasets
- ✓ View all Dashboards (preview)
- ✓ View all workspaces
- ✓ View all capacities
- ✓ Sign you in and read your profile

Accepting these permissions means that you allow this app to use your data as specified in their terms of service and privacy statement. **The publisher has not provided links to their terms for you to review.** You can change these permissions at <https://myapps.microsoft.com>. [Show details](#)

Cancel **Accept**

All Great Experiences Must Come to an End

 Power BI for Developers

Set up your Power BI embedding environment

Success! Your Power BI embedding environment is ready to use.

Download the sample app to explore your embedded content with all the relevant information from the environment you've set up, pre-configured and ready to test.

If you encounter any issue, use our [troubleshooting page](#)

[Download Sample App](#)



Power BI Developer Center

Find all the resources you need in one place, including documentation, APIs, tutorials, and more!

»

Summary:

User: Ted Pattison

Application: My First Power BI Embedding App

Application ID:
122e858a-1b70-40f8-aa5e-de0ff4f6fcfa6

Workspace: My New App Workspace

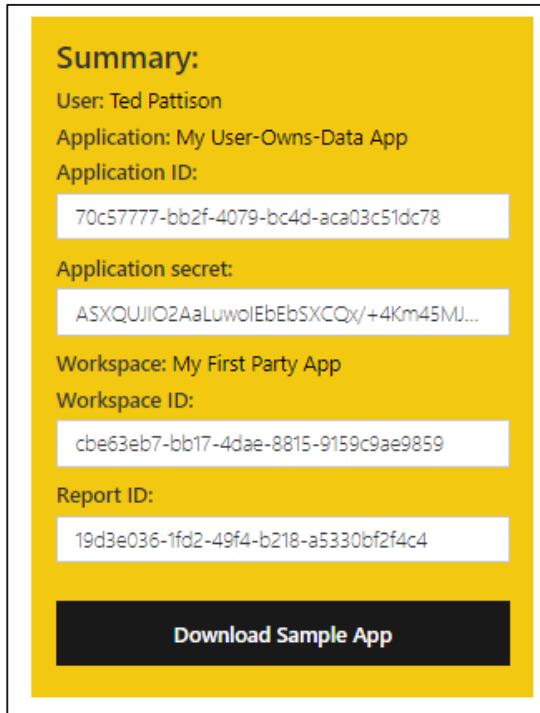
Workspace ID:
ed3bd652-89c9-4262-8ac0-4757b38ebf69

Report ID:
a5c37d6c-f26d-4565-a5e7-19fe1a3fcfb79

[Download Sample App](#)

User-Owns-Data Onboarding Experience

- Users-Owns-Data Experience is similar to App-Owns-Data experience
 - Users-Owns-Data Experience allows you to create app a **Native** or **Web app / API**
 - App create as Web app / API must be created with valid Reply URL
 - Process configures Azure app with a secret key known as Application secret
 - Application secret allows app to authenticate with AAD using authorization code grant flow



Demo Time

**Getting Started using the Power BI
Embedding Onboarding Experience**



Agenda

- Power BI Embedding Fundamentals
 - Managing Content using App Workspaces
 - Understanding Dedicated Capacities
 - Using the Onboarding Experience Tool
- **Programming with Power BI Service API**
- Embedding with the Power BI JavaScript API

The Power BI Service API

- ▲ Microsoft.PowerBI.Api
 - ▷ {} Microsoft.PowerBI.Api.V1
 - ▷ {} Microsoft.PowerBI.Api.V1.Models
 - ▲ {} Microsoft.PowerBI.Api.V2
 - ▷ 📊 Dashboards
 - ▷ 📊 DashboardsExtensions
 - ▷ 📊 Datasets
 - ▷ 📊 DatasetsExtensions
 - ▷ 📊 Gateways
 - ▷ 📊 GatewaysExtensions
 - ▷ 📊 Groups
 - ▷ 📊 GroupsExtensions
 - IDashboards
 - IDatasets
 - IGateways
 - IGroups
 - IImports
 - ▷ 📊 Imports
 - ▷ 📊 ImportsExtensions
 - IPowerBIClient
 - IReports
 - ITiles
 - ▷ 📊 PowerBIClient
 - ▷ 📊 Reports
 - ▷ 📊 ReportsExtensions
 - ▷ 📊 Tiles
 - ▷ 📊 TilesExtensions

- ▲ Microsoft.PowerBI.Api
 - ▷ {} Microsoft.PowerBI.Api.V1
 - ▷ {} Microsoft.PowerBI.Api.V1.Models
 - ▷ {} Microsoft.PowerBI.Api.V2
 - ▲ {} Microsoft.PowerBI.Api.V2.Models
 - ▷ 📊 BasicCredentials
 - ▷ 📊 BindToGatewayRequest
 - ▷ 📊 CloneReportRequest
 - ▷ 📊 Column
 - ▷ 📊 ConnectionDetails
 - ▷ 📊 CredentialDetails
 - ▷ 📊 Dashboard
 - ▷ 📊 Dataset
 - ▷ 📊 DatasetMode
 - ▷ 📊 Datasource
 - ▷ 📊 EmbedToken
 - ▷ 📊 Gateway
 - ▷ 📊 GatewayDatasource
 - ▷ 📊 GatewayPublicKey
 - ▷ 📊 GenerateTokenRequest
 - ▷ 📊 Group
 - ▷ 📊 GroupCreationRequest
 - ▷ 📊 GroupUser
 - ▷ 📊 GroupUserAccessRight
 - ▷ 📊 Import
 - ▷ 📊 ImportConflictHandlerMode
 - ▷ 📊 ImportInfo
 - ▷ 📊 MemberAdminAccessRight
 - ▷ 📊 ODataResponseListDashboard
 - ▷ 📊 ODataResponseListDataset
 - ▷ 📊 ODataResponseListDatasource
 - ▷ 📊 ODataResponseListGateway
 - ▷ 📊 ODataResponseListGatewayDatasource
 - ▷ 📊 ODataResponseListGroup
 - ▷ 📊 ODataResponseListGroupUserAccessRight
 - ▷ 📊 ODataResponseListImport
 - ▷ 📊 ODataResponseListRefresh
 - ▷ 📊 ODataResponseListReport
 - ▷ 📊 ODataResponseListTable
 - ▷ 📊 ODataResponseListTile
 - ▷ 📊 ODataResponseListUserAccessRight
 - ▷ 📊 PublishDatasourceToGatewayRequest
 - ▷ 📊 RebindReportRequest
 - ▷ 📊 Refresh
 - ▷ 📊 Report
 - ▷ 📊 Row
 - ▷ 📊 Table
 - ▷ 📊 Tile
 - ▷ 📊 TokenAccessLevel
 - ▷ 📊 UpdateDatasourceRequest
 - ▷ 📊 UserAccessRight
 - ▷ 📊 UserAccessRightEnum

App Configuration Data for AAD Authentication

```
<configuration>
  <appSettings>

    <add key="clientId" value="23f6d66f-9a9a-4dba-9b7c-ff8aedadb831" />
    <add key="pbUserName" value="pbimasteruser@powerbimvps.onMicrosoft.com" />
    <add key="pbUserPassword" value="Pa$$word!" />
    <add key="appworkspaceId" value="4baab6c0-87c5-4a2a-a73e-1f97adcc6bdb" />
```

```
public class PbiEmbeddingManager {

  #region "AAD Authentication Constants"

  static string aadAuthorizationEndpoint = "https://login.windows.net/common/oauth2/authorize";
  static string resourceUriPowerBi = "https://analysis.windows.net/powerbi/api";
  static string urlPowerBiRestApiRoot = "https://api.powerbi.com/";

  static string clientId = ConfigurationManager.AppSettings["clientId"];
  static string appWorkspaceId = ConfigurationManager.AppSettings["appworkspaceId"];
  static string pbUserName = ConfigurationManager.AppSettings["pbUserName"];
  static string pbUserPassword = ConfigurationManager.AppSettings["pbUserPassword"];

  #endregion
}
```

Getting an Access Token for the Master User

```
static string GetAccessToken() {
    AuthenticationContext authContext = new AuthenticationContext(aadAuthorizationEndpoint);
    var userCredentials = new UserPasswordCredential(pbiUserName, pbiUserPassword);

    // this call will fail if permission consent has not be granted to master user account
    string aadAccessToken =
        authContext.AcquireTokenAsync(resourceUriPowerBi, clientId, userCredentials).Result.AccessToken;

    // return Azure AD access token for master user account
    return aadAccessToken;
}
```

Initializing an Instance of PowerBIClient

- PowerBIClient object serves as top-level object
 - Used to execute calls against Power BI Service
 - Initialized with function to retrieve AAD access token

```
static string GetAccessToken() { ... }

static PowerBIClient GetPowerBiClient() {
    var tokenCredentials = new TokenCredentials(GetAccessToken(), "Bearer");
    return new PowerBIClient(new Uri(urlPowerBiRestApiRoot), tokenCredentials);
}

static void Main() {
    PowerBIClient pbiClient = GetPowerBiClient();
    var reports = pbiClient.Reports.GetReports().Value;
    foreach (var report in reports) {
        Console.WriteLine(report.Name);
    }
}
```

Report and Dataset Info

```
// data required for embedding a report
class ReportEmbeddingData {
    public string reportId;
    public string reportName;
    public string embedUrl;
    public string accessToken;
}

// data required for embedding a dashboard
class DashboardEmbeddingData {
    public string dashboardId;
    public string dashboardName;
    public string embedUrl;
    public string accessToken;
}

// data required for embedding a dashboard
class DashboardTileEmbeddingData {
    public string dashboardId;
    public string TileId;
    public string TileTitle;
    public string embedUrl;
    public string accessToken;
}
```

```
// data required for embedding a new report
class NewReportEmbeddingData {
    public string workspaceId;
    public string datasetId;
    public string embedUrl;
    public string accessToken;
}

// data required for embedding QnA experience
class QnaEmbeddingData {
    public string datasetId;
    public string embedUrl;
    public string accessToken;
}
```

Getting the Data for First Party Report Embedding

```
public static ReportEmbeddingData GetReportEmbeddingDataFirstParty() {  
    PowerBIClient pbiclient = GetPowerBIClient();  
  
    var report = pbiclient.Reports.GetReportInGroup(workspaceId, reportId);  
    var embedUrl = report.EmbedUrl;  
    var reportName = report.Name;  
    var accessToken = GetAccessToken();  
  
    return new ReportEmbeddingData {  
        reportId = reportId,  
        reportName = reportName,  
        embedUrl = embedUrl,  
        accessToken = accessToken  
    };  
}
```

Embed Tokens

- You can embed reports using master user AAD token, but...
 - You might want embed resource using more restricted tokens
 - You might want stay within the bounds of Power BI licensing terms
- You generate embed tokens with the Power BI Service API
 - Each embed token created for one specific resource
 - Embed token provides restrictions on whether user can view or edit
 - Embed token can only be generated inside dedicated capacity (*semi-enforced*)
 - Embed token can be generated to support row-level security (RLS)

```
Report report = reports.Where(r => r.Id == reportId).First();
var generateTokenRequestParameters = new GenerateTokenRequest(accessLevel: "edit");
var token = client.Reports.GenerateTokenInGroupAsync(appWorkspaceId,
                                                       report.Id,
                                                       generateTokenRequestParameters).Result;
```

Getting the Data for Third Party Report Embedding

```
public static ReportEmbeddingData GetReportEmbeddingData() {  
    PowerBIClient pbiclient = GetPowerBIClient();  
  
    var report = pbiclient.Reports.GetReportInGroup(workspaceId, reportId);  
    var embedUrl = report.EmbedUrl;  
    var reportName = report.Name;  
  
    // create token request object  
    GenerateTokenRequest generateTokenRequestParameters = new GenerateTokenRequest(accessLevel: "view");  
  
    // call to Power BI Service API and pass GenerateTokenRequest object to generate embed token  
    string embedToken = pbiclient.Reports.GenerateTokenInGroup(workspaceId,  
                                                               report.Id,  
                                                               generateTokenRequestParameters).Token;  
  
    return new ReportEmbeddingData {  
        reportId = reportId,  
        reportName = reportName,  
        embedUrl = embedUrl,  
        accessToken = embedToken  
    };  
}
```

Getting the Data for Dashboard Embedding

```
public static DashboardEmbeddingData GetDashboardEmbeddingData() {  
    PowerBIClient pbIClient = GetPowerBIClient();  
  
    var dashboard = pbIClient.Dashboards.GetDashboardInGroup(workspaceId, dashboardId);  
    var embedUrl = dashboard.EmbedUrl;  
    var dashboardDisplayName = dashboard.DisplayName;  
  
    GenerateTokenRequest generateTokenRequestParameters = new GenerateTokenRequest(accessLevel: "view");  
    string embedToken = pbIClient.Dashboards.GenerateTokenInGroup(workspaceId,  
                                                                  dashboardId,  
                                                                  generateTokenRequestParameters).Token;  
  
    return new DashboardEmbeddingData {  
        dashboardId = dashboardId,  
        dashboardName = dashboardDisplayName,  
        embedUrl = embedUrl,  
        accessToken = embedToken  
    };  
}
```

Getting the Data for Dashboard Tile Embedding

```
public static DashboardTileEmbeddingData GetDashboardTileEmbeddingData() {  
    PowerBIClient pbiclient = GetPowerBIClient();  
  
    var tiles = pbiclient.Dashboards.GetTilesInGroup(workspaceId, dashboardId).Value;  
  
    // retrieve first tile in tiles connection  
    var tile = tiles[0];  
    var tileId = tile.Id;  
    var tileTitle = tile.Title;  
    var embedUrl = tile.EmbedUrl;  
  
    GenerateTokenRequest generateTokenRequestParameters = new GenerateTokenRequest(accessLevel: "view");  
    string embedToken = pbiclient.Tiles.GenerateTokenInGroup(workspaceId,  
                                                dashboardId,  
                                                tileId,  
                                                generateTokenRequestParameters).Token;  
  
    return new DashboardTileEmbeddingData {  
        dashboardId = dashboardId,  
        TileId = tileId,  
        TileTitle = tileTitle,  
        embedUrl = embedUrl,  
        accessToken = embedToken  
    };  
}
```

Getting the Data for New Report Embedding

```
public static NewReportEmbeddingData GetNewReportEmbeddingData() {  
    string embedUrl = "https://app.powerbi.com/reportEmbed?groupId=" + workspaceId;  
  
    PowerBIClient pbiclient = GetPowerBIClient();  
  
    GenerateTokenRequest generateTokenRequestParameters = new GenerateTokenRequest(accessLevel: "create",  
                                                                 datasetId: datasetId);  
  
    string embedToken = pbiclient.Reports.GenerateTokenForCreateInGroup(workspaceId,  
                                                                      generateTokenRequestParameters).Token;  
  
    return new NewReportEmbeddingData {  
        workspaceId = workspaceId,  
        datasetId = datasetId,  
        embedUrl = embedUrl,  
        accessToken = embedToken  
    };  
}
```

Getting the Data for Q&A Experience Embedding

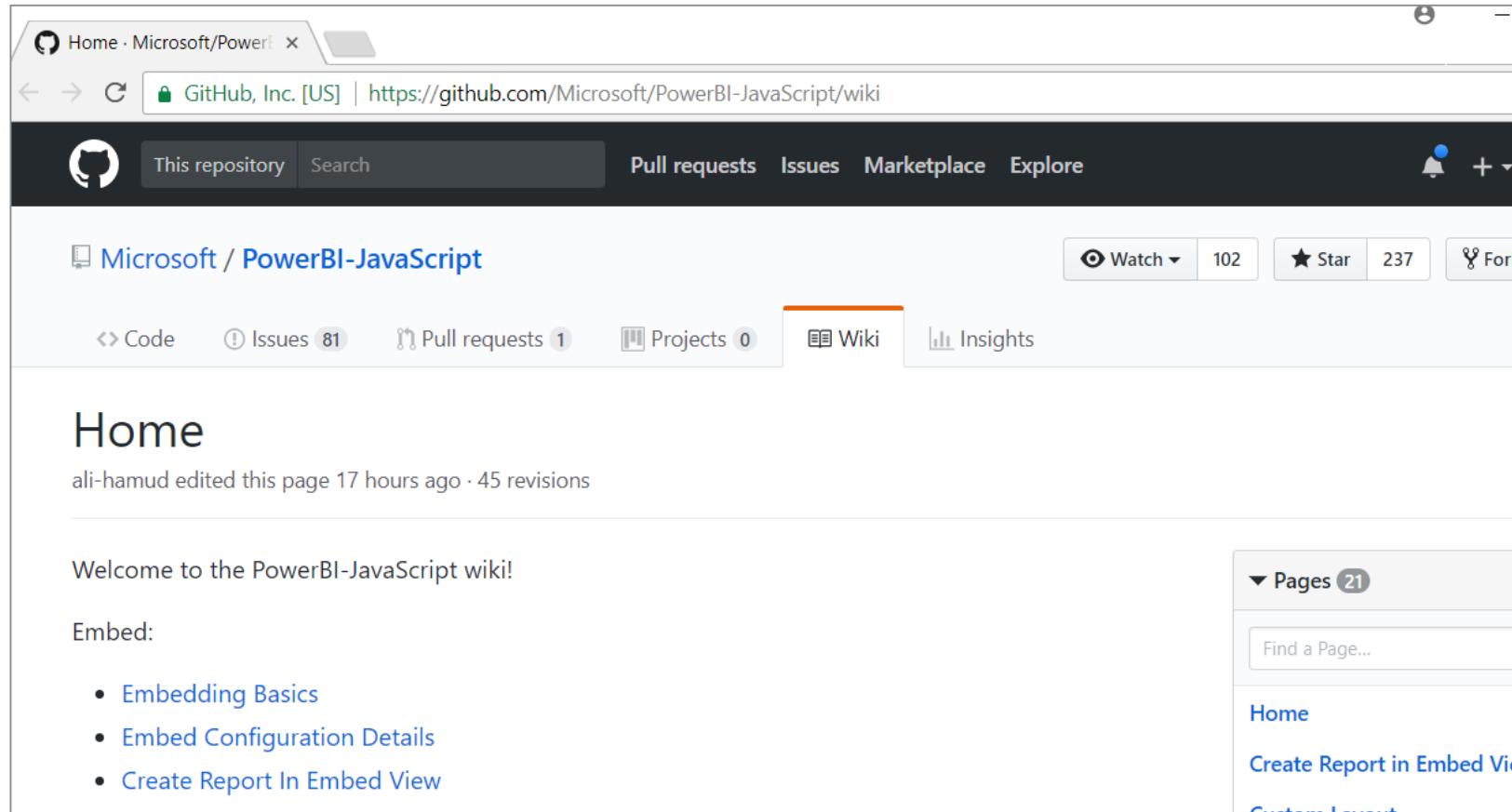
```
public static QnaEmbeddingData GetQnaEmbeddingData() {  
    PowerBIClient pbIClient = GetPowerBIClient();  
  
    var dataset = pbIClient.Datasets.GetDatasetByIdInGroup(workspaceId, datasetId);  
  
    string embedUrl = "https://app.powerbi.com/qnaEmbed?groupId=" + workspaceId;  
    string datasetID = dataset.Id;  
  
    GenerateTokenRequest generateTokenRequestParameters = new GenerateTokenRequest(accessLevel: "view");  
  
    string embedToken = pbIClient.Datasets.GenerateTokenInGroup(workspaceId,  
                                                                dataset.Id,  
                                                                generateTokenRequestParameters).Token;  
  
    return new QnaEmbeddingData {  
        datasetId = datasetId,  
        embedUrl = embedUrl,  
        accessToken = embedToken  
    };  
}
```

Agenda

- Power BI Embedding Fundamentals
- Managing Content using App Workspaces
- Understanding Dedicated Capacities
- Using the Onboarding Experience Tool
- Programming with Power BI Service API
- **Embedding with the Power BI JavaScript API**

Power BI JavaScript API (`powerbi.js`)

- Power BI JavaScript API used to embed resources in browser
 - Maintained in GitHub at <https://github.com/Microsoft/PowerBI-JavaScript/wiki>
 - GitHub repository contains code, docs, wiki and issues list



Hello World with Power BI Embedding

- powerbi.js library provides **powerbi** as top-level service object
 - You call **powerbi.embed** and pass **configuration** object with access token t
 - **models** object available to supply configuration settings
 - **configuration** object sets **tokenType** to either **models.TokenType.Embed** or **models.TokenType.Aad**

```
// data required for embedding Power BI report
var embedReportId = "ba274ba0-93be-4e53-af65-fdc8a559c557";
var embedUrl = "https://app.powerbi.com/reportEmbed?reportId=ba274ba0-93be-4e53-af65-fdc8a559c557&groupId=7f4...&accessToken = "eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6I1Rpb0d5d3dsahZkRmJYwJgxM1dwUGF5OUFsVSIsImtpZC...;

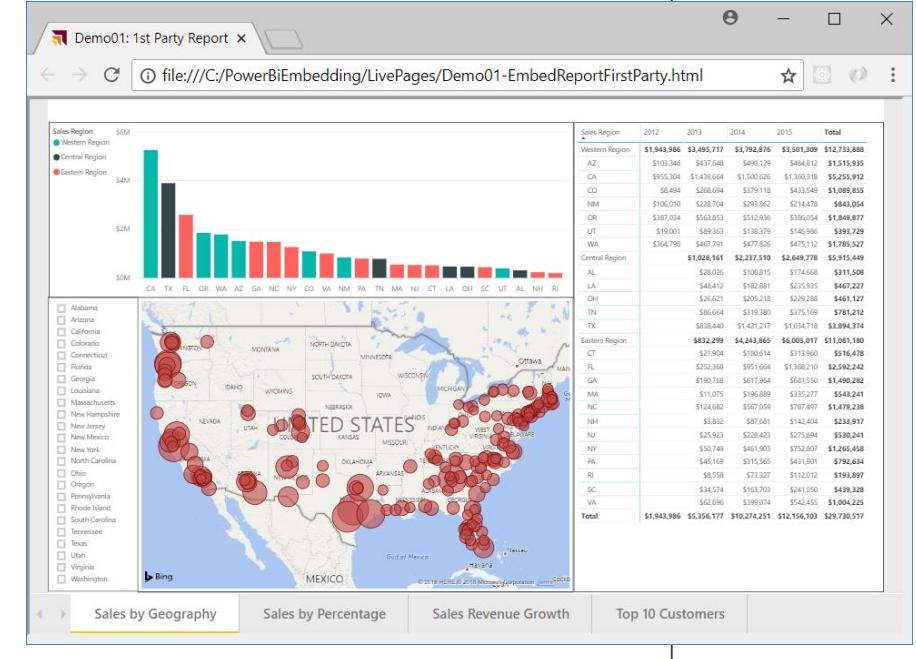
// Get models object to access enums for embed configuration
var models = window['powerbi-client'].models;

// create embed configuration object
var config = {
    type: 'report',
    id: embedReportId,
    embedUrl: embedUrl,
    accessToken: accessToken,
    tokenType: models.TokenType.Aad
};

// Get a reference to the embedded report HTML element
var reportContainer = document.getElementById('embedContainer');

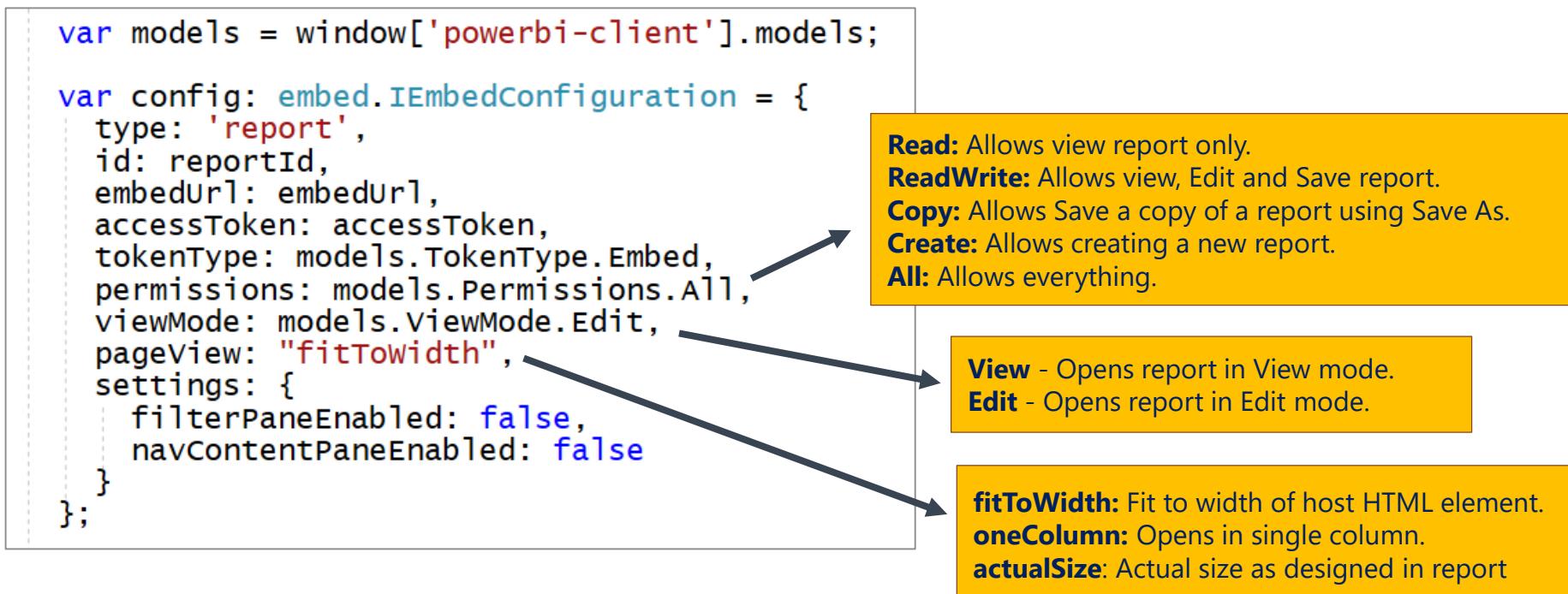
// Embed the report and display it within the div container.
var report = powerbi.embed(reportContainer, config);
```

First party embedding



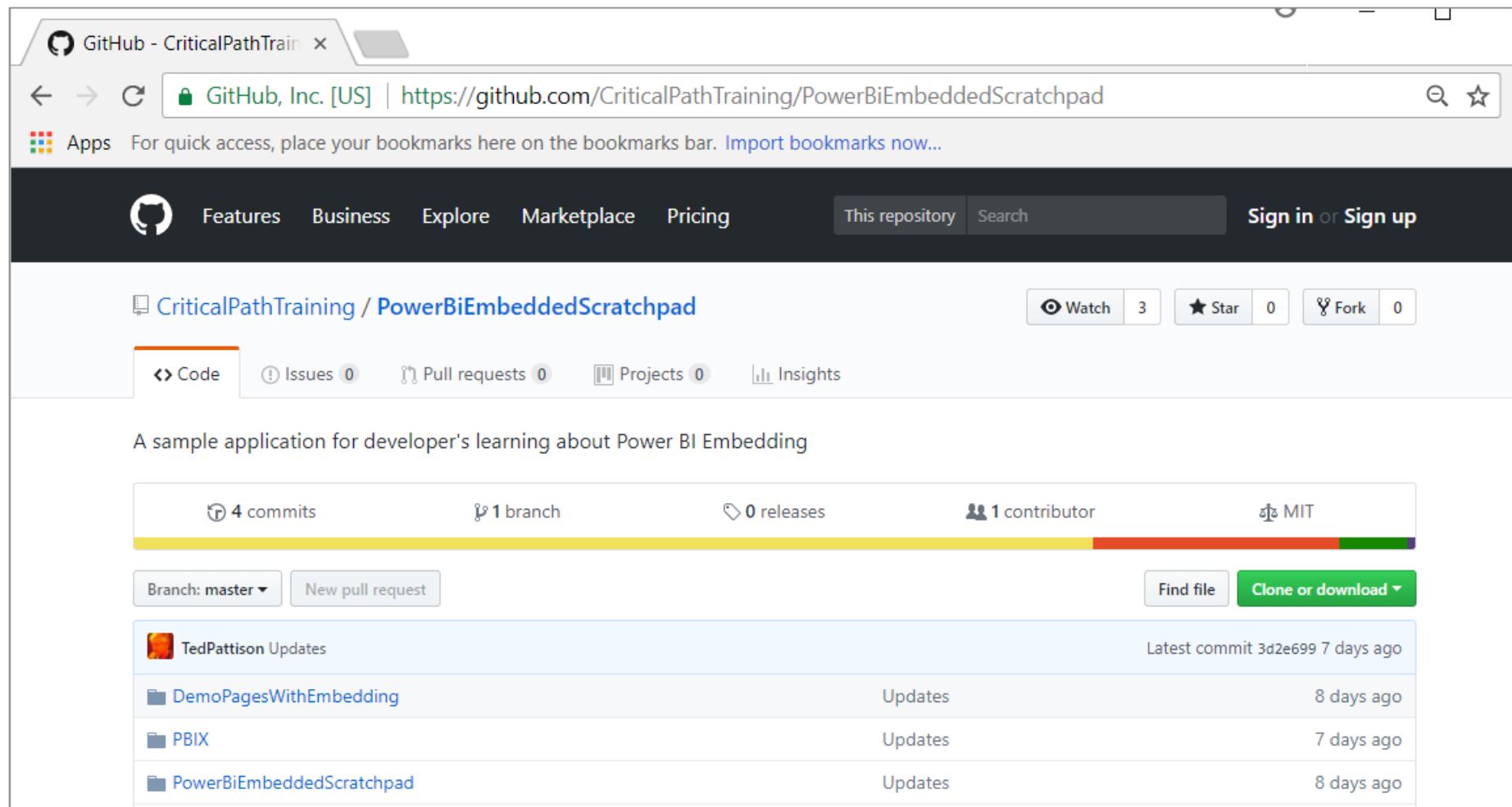
Embedded Report Configuration Options

- **permissions** determines what permissions are given to user on resource
- **viewMode** determines when report opens in read-only view or edit view
- **pageView** determines how reports scales to fit embed container element



PowerBiEmbeddedScratchpad Sample

<https://github.com/CriticalPathTraining/PowerBiEmbeddedScratchpad>



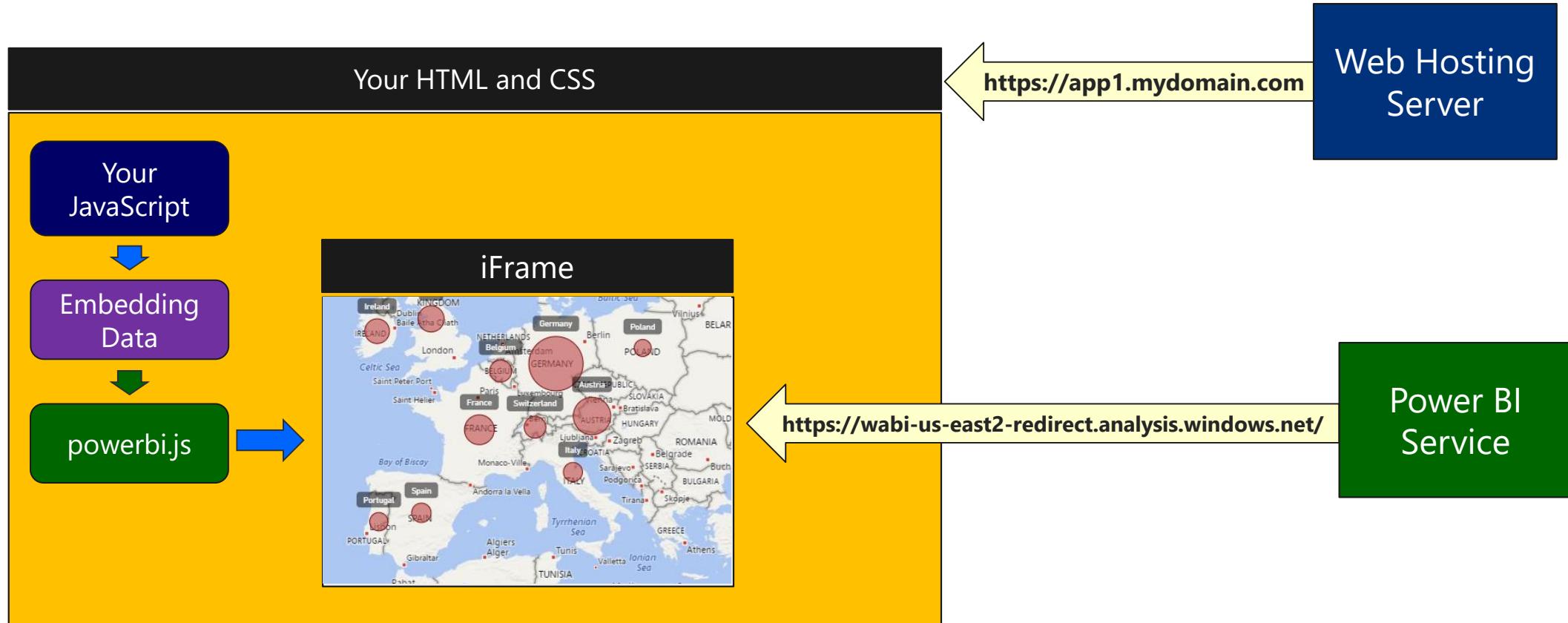
Demo Time

Cooking up something tasty using
the Power BI Embedded Scratchpad



Report Embedding Architecture

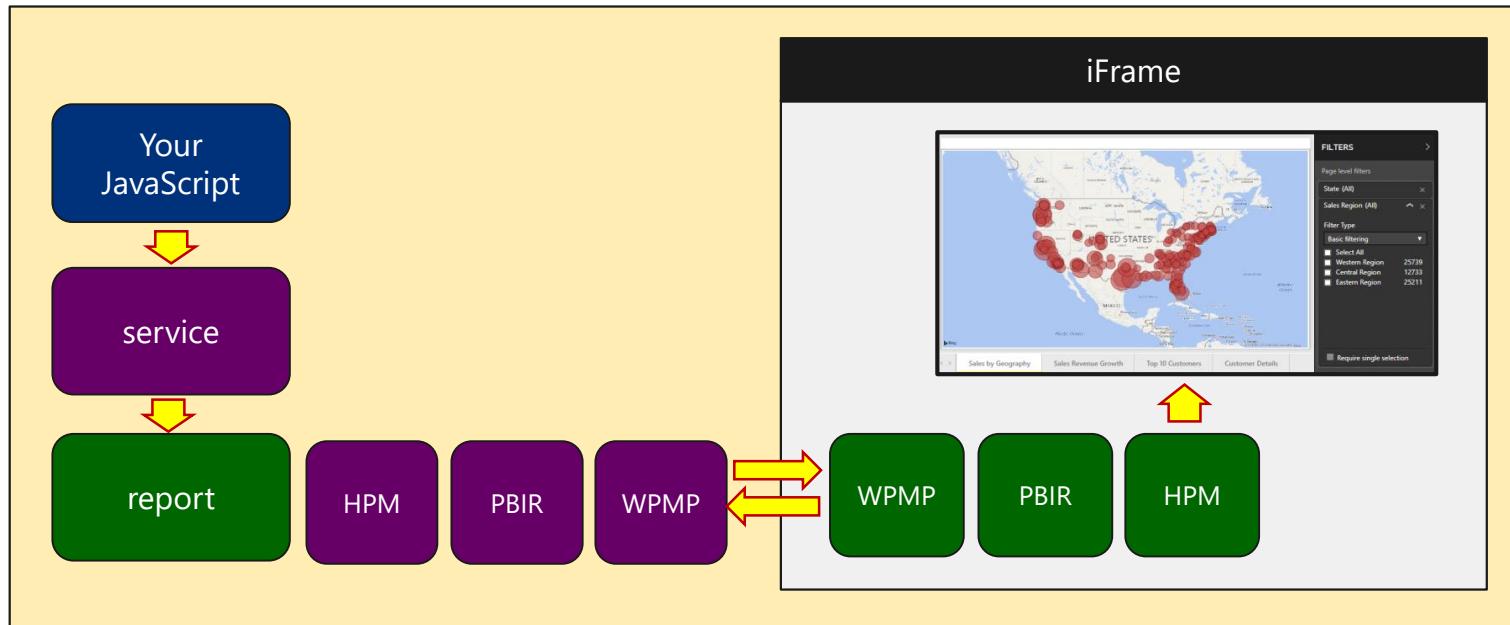
- Embedding involves creating an iFrame on the page
 - **powerbi.js** library transparently creates iFrame and sets source to Power BI Service



The iFrame and hosting page originate from different DNS domains

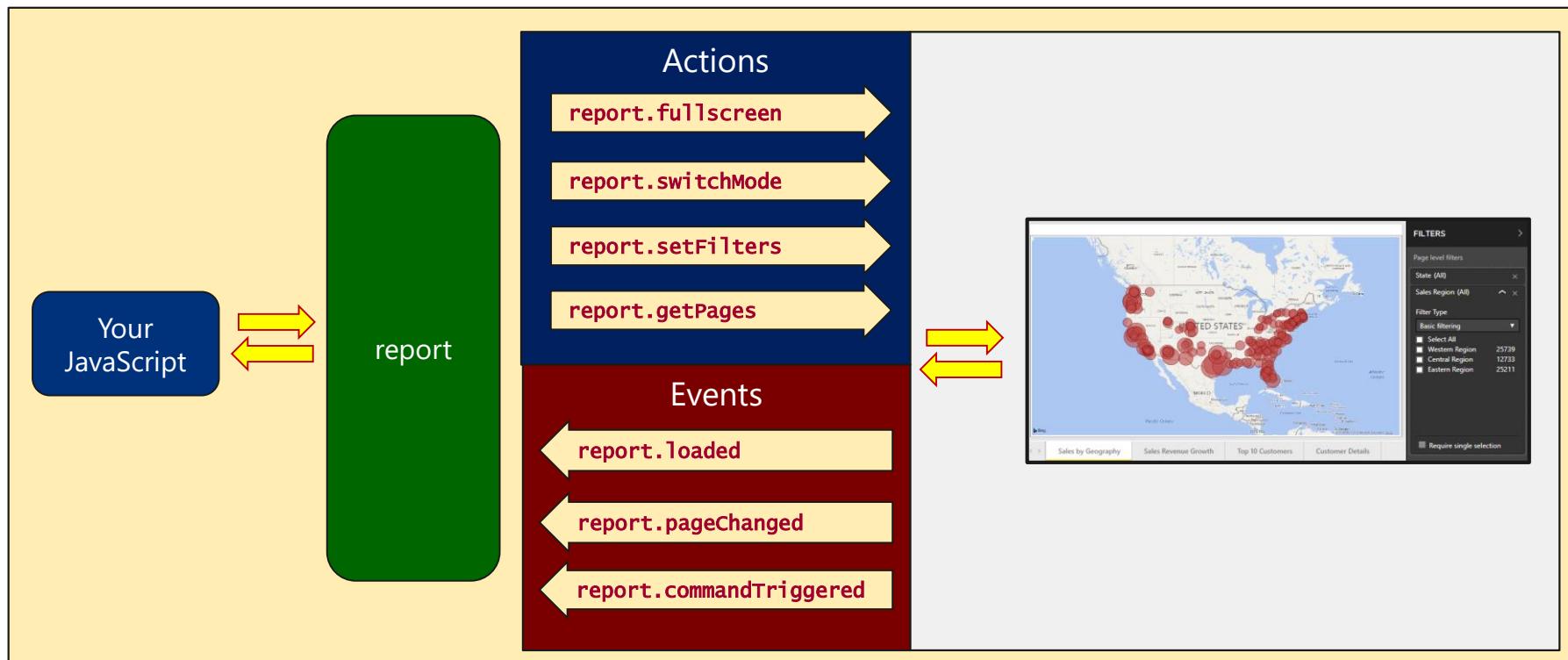
Post Message Communications Flow

- 4 extra libraries used communicate with report in iFrame
 - window-post-message-proxy (WPMP)
 - http-post-message (HPM)
 - powerbi-router (PBIR)
 - powerbi-models (PBIM)



A Promise-based Programming Model

- Design of powerbi.js library simulates HTTP protocol
 - Creates more intuitive programming model for developers
 - Programming based on asynchronous requests and promises
 - Embedded objects programmed in familiar fashion using **actions** and **events**



Executing Commands on a Report Object

- Report object exposes action methods
 - **report.switchMode** used to switch report between view mode and edit mode
 - **report.updateSettings** used to hide/show filter pane and page tabs
 - **report.fullscreen** used to bring report into full screen mode
 - **report.print** used to print the report

```
// Embed the report
var report = powerbi.embed(reportContainer, config);

// add variable to track current display mode
var viewMode = "view";

// add command handler to toggle between display mode and edit mode
$("#toggleEdit").click(function () {
    viewMode = (viewMode == "view") ? "edit" : "view";
    // call report.switchMode to change report edit mode
    report.switchMode(viewMode);
    var showFilterPane = (viewMode == "edit");
    // report.updateSettings hide or show filter pane
    report.updateSettings({ "filterPaneEnabled": showFilterPane });
});

// add command handler to enter full screen mode
$("#fullScreen").click(function () { report.fullscreen(); });

// add command handler to print report
$("#print").click(function () { report.print(); });
```

Handling Report Events

```
var report = powerbi.embed(embedContainer, config);

var pages;

report.on('loaded', function () {
    // call getPages with callback
    report.getPages().then(
        function (reportPages) {
            pages = reportPages;
            // call method to load pages into nav menu
            loadReportPages(pages);
        });
});

var loadReportPages = function (pages) {
    for (var index = 0; index < pages.length; index++) {
        // determine which pages are visible and not hidden
        if (pages[index].visibility == 0) { // 0 means visible and 1 means hidden
            var reportPageDisplayName = pages[index].displayName;
            pageNavigation.append($("<li>")
                .append($('a href="javascript:;' >')
                    .text(pages[index].displayName))
                .click(function (domEvent) {
                    var targetPageName = domEvent.target.textContent;
                    // get target page from pages collection
                    var targetPage = pages.find(function (page) { return page.displayName === targetPageName; });
                    // navigate report to target page
                    targetPage.setActive();
                }));
        }
    }
}
```

▼ Report [i](#)

▼ allowedEvents: Array(12)

- 0: "loaded"
- 1: "saved"
- 2: "rendered"
- 3: "saveAsTriggered"
- 4: "error"
- 5: "dataSelected"
- 6: "filtersApplied"
- 7: "pageChanged"
- 8: "commandTriggered"
- 9: "swipeStart"
- 10: "swipeEnd"
- 11: "bookmarkApplied"

Embedding a New Report

```
// Get data required for embedding
var embedWorkspaceId= "7f4576c7-039a-472f-b998-546a572d5da2";
var embedDatasetId = "b4a48602-71da-42b2-8cf5-44d35b2ac70b";
var embedUrl = "https://app.powerbi.com/reportEmbed?groupId=7f4576c7-039a-472f-b998-546a5
var accessToken = "H4sIAAAAAAAEAB2Wxw60CA6E3-W_shIZmpXmQE5NztzIOwdG--7bM3dbsj67qvz3HzN5-i

// Get models object to access enums for embed configuration
var models = window['powerbi-client'].models;

var config = {
    datasetId: embedDatasetId,
    embedUrl: embedUrl,
    accessToken: accessToken,
    tokenType: models.TokenType.Embed,
};

// Get a reference to the embedded report HTML element
var embedContainer = document.getElementById('embedContainer');

// Embed the report and display it within the div container.
var report = powerbi.createReport(embedContainer, config);
```

New Report with SaveAs Redirect

```
// Embed the report and display it within the div container.  
var newReport = powerbi.createReport(embedContainer, config);  
  
// this event fires whenever user runs save or SaveAs command on a new report  
newReport.on("saved", function (event) {  
  
    // get ID and name of new report  
    var newReportId = event.detail.reportObjectId;  
    var newReportName = event.detail.reportName;  
  
    // set new title for browser window  
    document.title = newReportName;  
  
    // reset report container element  
    powerbi.reset(embedContainer);  
  
    config = {  
        type: 'report',  
        id: newReportId,  
        embedUrl: "https://app.powerbi.com/reportEmbed?reportId=" + newReportId + "&groupId=" + embedworkspaceId,  
        accessToken: accessToken,  
        tokenType: models.TokenType.Aad,  
        permissions: models.Permissions.All,  
        viewMode: models.ViewMode.Edit,  
    };  
  
    // Embed the report and display it within the div container.  
    var savedReport = powerbi.embed(embedContainer, config);
```

Embedding the Q&A Experience

```
// Get data required for embedding
var datasetId = "b4a48602-71da-42b2-8cf5-44d35b2ac70b";
var embedUrl = "https://app.powerbi.com/qnaEmbed?groupId=7f4576c7-039a-472f-b998-546a57";
var accessToken = "H4sIAAAAAAAEAC2Wx6rFDI6E3-XfeuA4h4FeO0ecvXPOObuzd58L3XuBpK8K1f79j5W-"

// Get models object to access enums for embed configuration
var models = window['powerbi-client'].models;

var config = {
    type: 'qna',
    tokenType: models.TokenType.Embed,
    accessToken: accessToken,
    embedUrl: embedUrl ,
    datasetIds: [ datasetId ],
    viewMode: models.QnaMode.Interactive,
    question: "What is sales revenue by quarter and sales region as stacked area chart"
};

var embedContainer = document.getElementById('embedContainer');

var embeddedObject = powerbi.embed(embedContainer, config);
```

Summary

- ✓ Power BI Embedding Fundamentals
- ✓ Managing Content using App Workspaces
- ✓ Understanding Dedicated Capacities
- ✓ Using the Onboarding Experience Tool
- ✓ Programming with Power BI Service API
- ✓ Embedding with the Power BI JavaScript API

