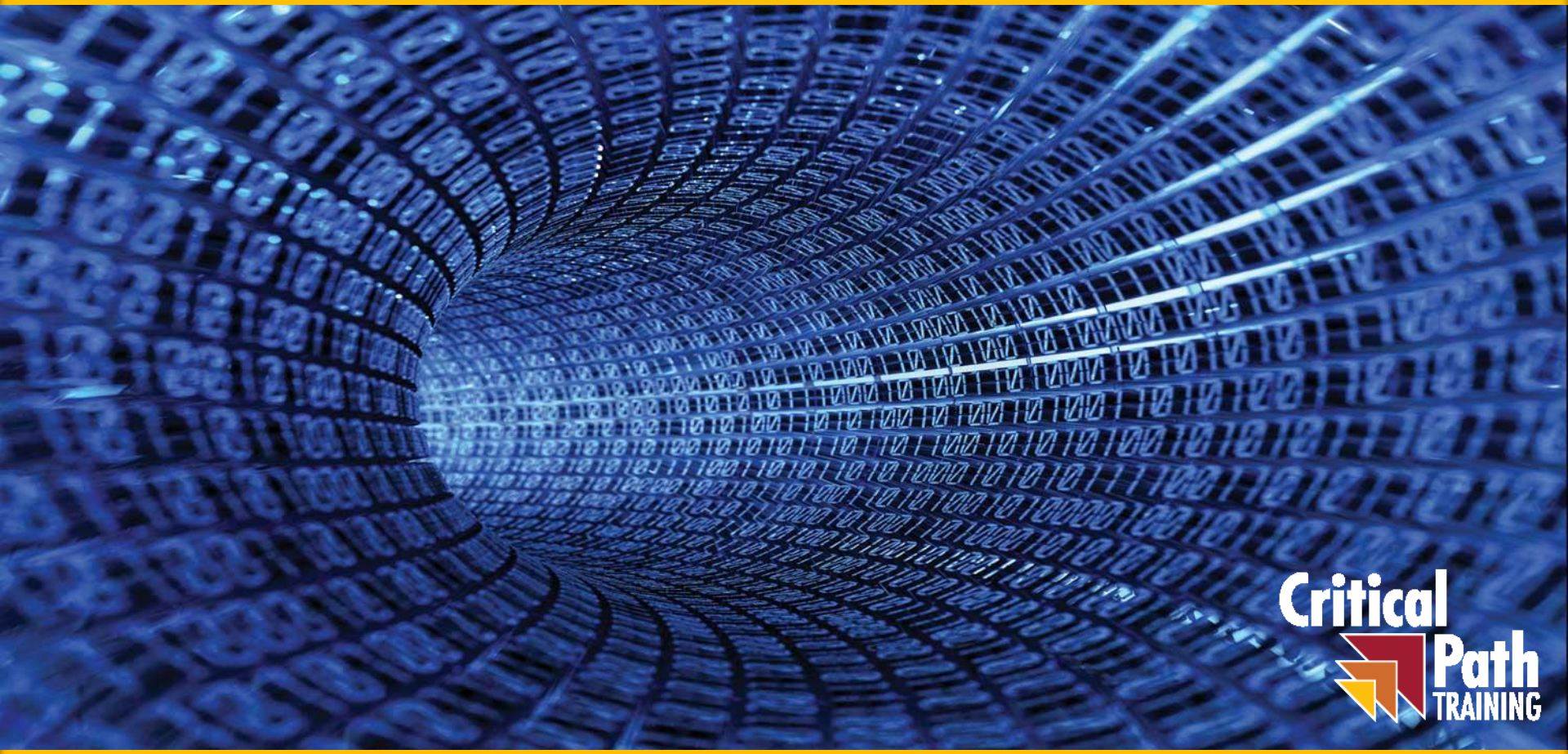
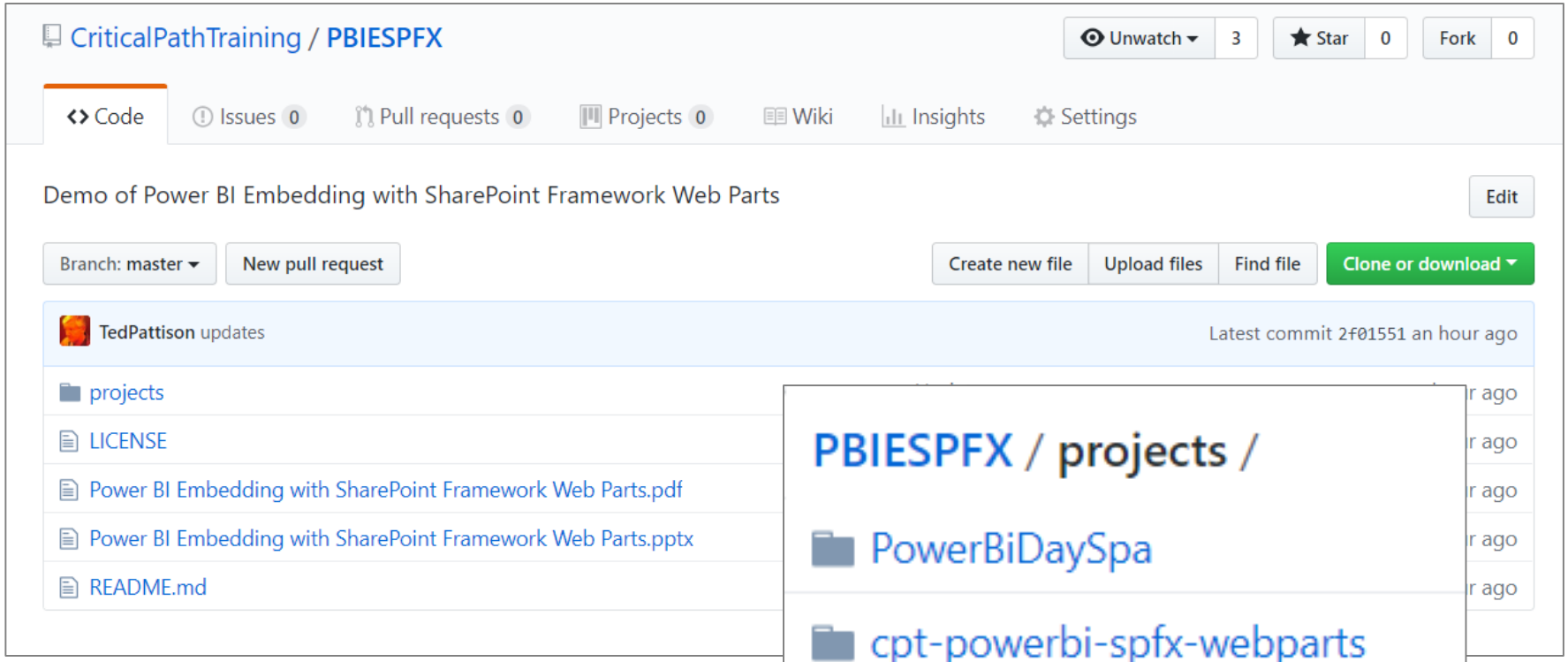
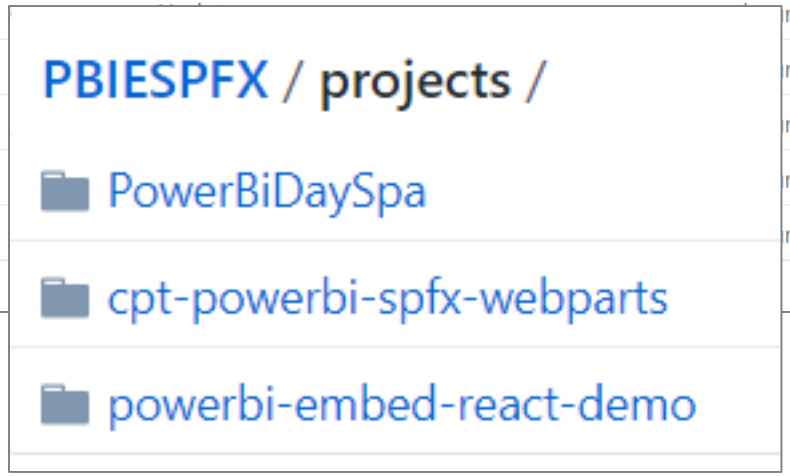
Power BI Embedding with SharePoint Framework Web Parts



Code and Slides for this Webinar

https://github.com/CriticalPathTraining/PBIESPFX





Critical Path Training

https://www.CriticalPathTraining.com

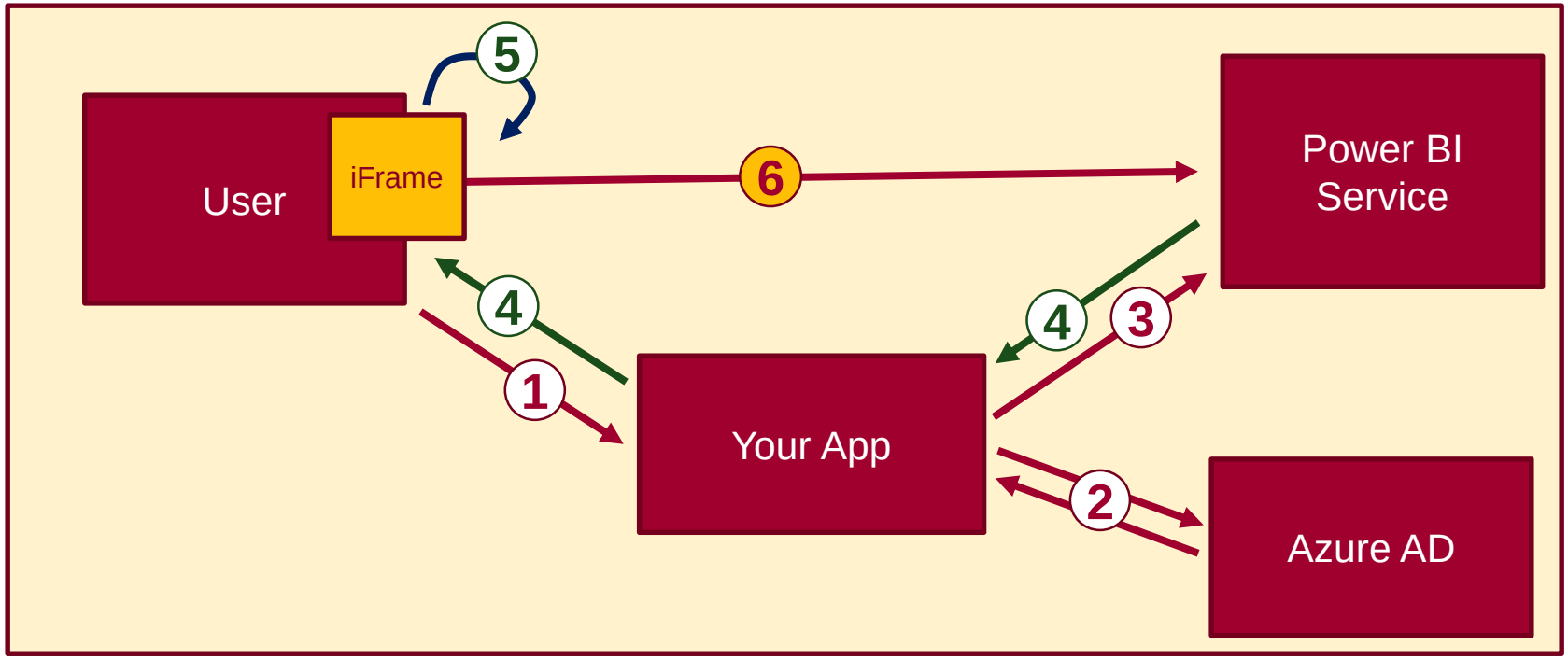
- PBI365: Power BI Certification Bootcamp – 3 Days
  - For people who have used Power BI Desktop for 6 months or more
- PBD365: Power BI Developer Bootcamp – 4 Days
  - For professional developers working with the Power BI platform
- DDPAF: Deep Dive into Power Apps and Flow – 2 Days
  - For people just getting started with Power Apps and Flow

Agenda

- Power BI Embedding Fundamentals
- Authentication with Azure Active Directory
- Programming with Power BI Service API
- Embedding with Power BI JavaScript API
- SharePoint Framework Web Part

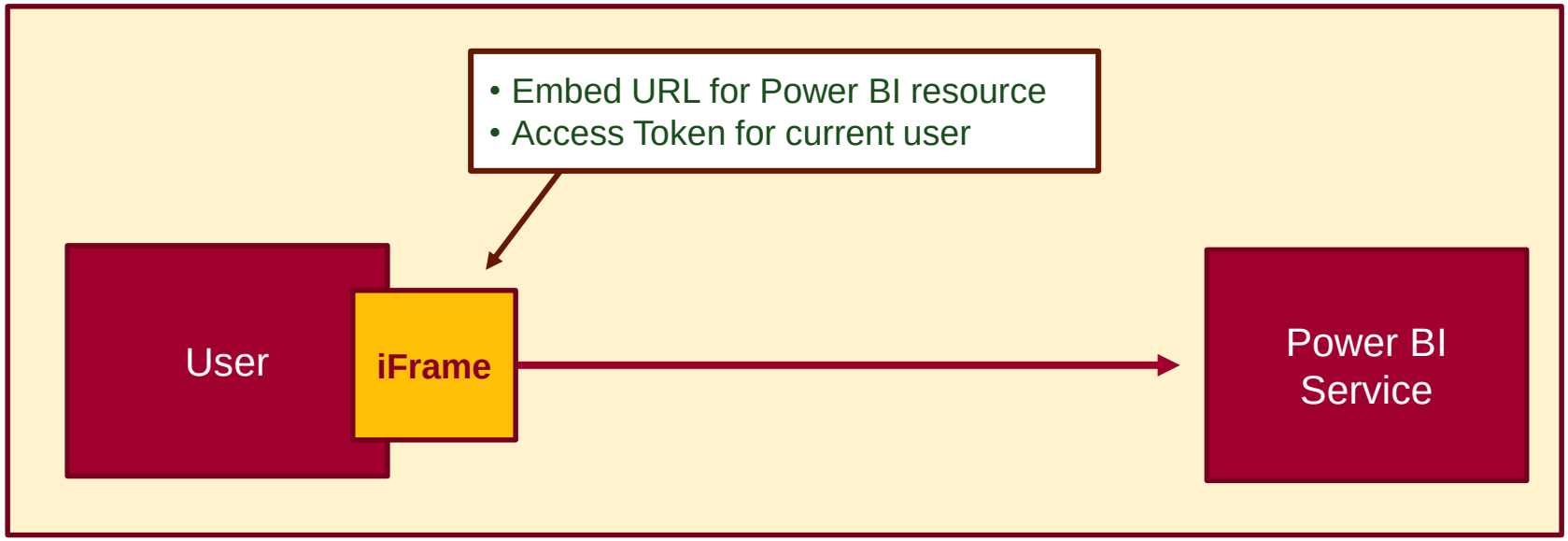
Power BI Embedding – The Big Picture

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



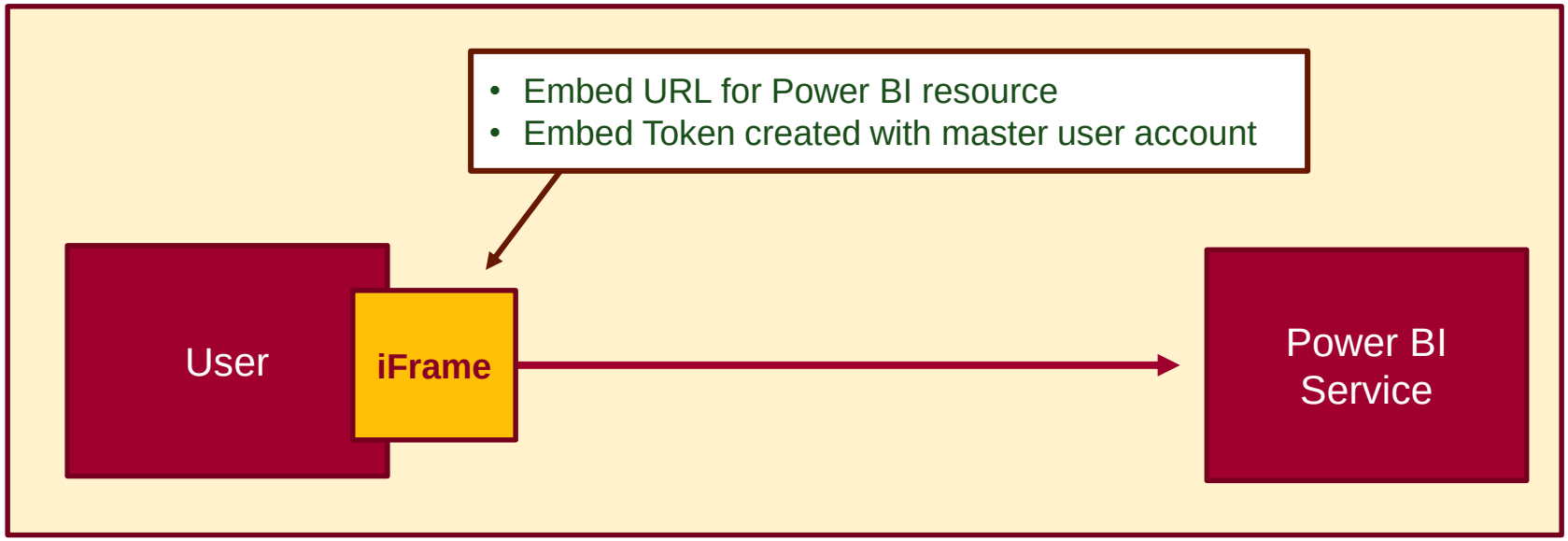
First Party Embedding

- App authenticates current user with Azure AD
  - Your code accesses Power BI Service as current user
  - Embedding requires Azure AD access token for user
  - User requires Azure AD account and Power BI license
  - Your code has access to whatever user has access to



Third Party Embedding

- App authenticates using Master User Account
  - Your code accesses Power BI Service as master user
  - Embedding uses embed token instead of access token
  - Users don’t need AAD accounts and Power BI licenses
  - Your code has access to whatever master has access to



First Party vs Third Party Embedding

- What scenarios use first party embedding?
  - Organizations where users have Power BI licenses
  - Embedding Power BI reports in SharePoint and Teams
  - Development should go beyond out-of-box experience

- What scenarios use third party embedding?
  - Scenarios where users don’t have Power BI licenses
  - Applications which have custom identity providers
  - Applications which use identity provider other than AAD

Embeddable Resources

1. Reports
2. Dashboards
3. Dashboard Tiles
4. Q & A Experience
5. Visual *

* really just a trick you do when embedding a report

DEMO

The Daily Reporter Pro Sample App

Creating an Azure AD Application

Microsoft Azure premium demo tenant - App regist

Create

* Name
My Native App

Application type
Native

* Redirect URI
https://locali

Create

Microsoft Azure premium demo tenant - App registrations > My Native App > Settings

My Native App
Registered app

Settings Manifest Delete

Essentials

Display name
My Native App

Application type
Native

Home page

Application ID
7802d697-c0d5-480f-8f30-5039226f02a7

Object ID
e9ee95cc-0e31-4705-a6ac-b2b10053df4b

Managed application in local directory
My Native App

All settings

Settings

Filter settings

GENERAL

Properties

Redirect URIs

Owners

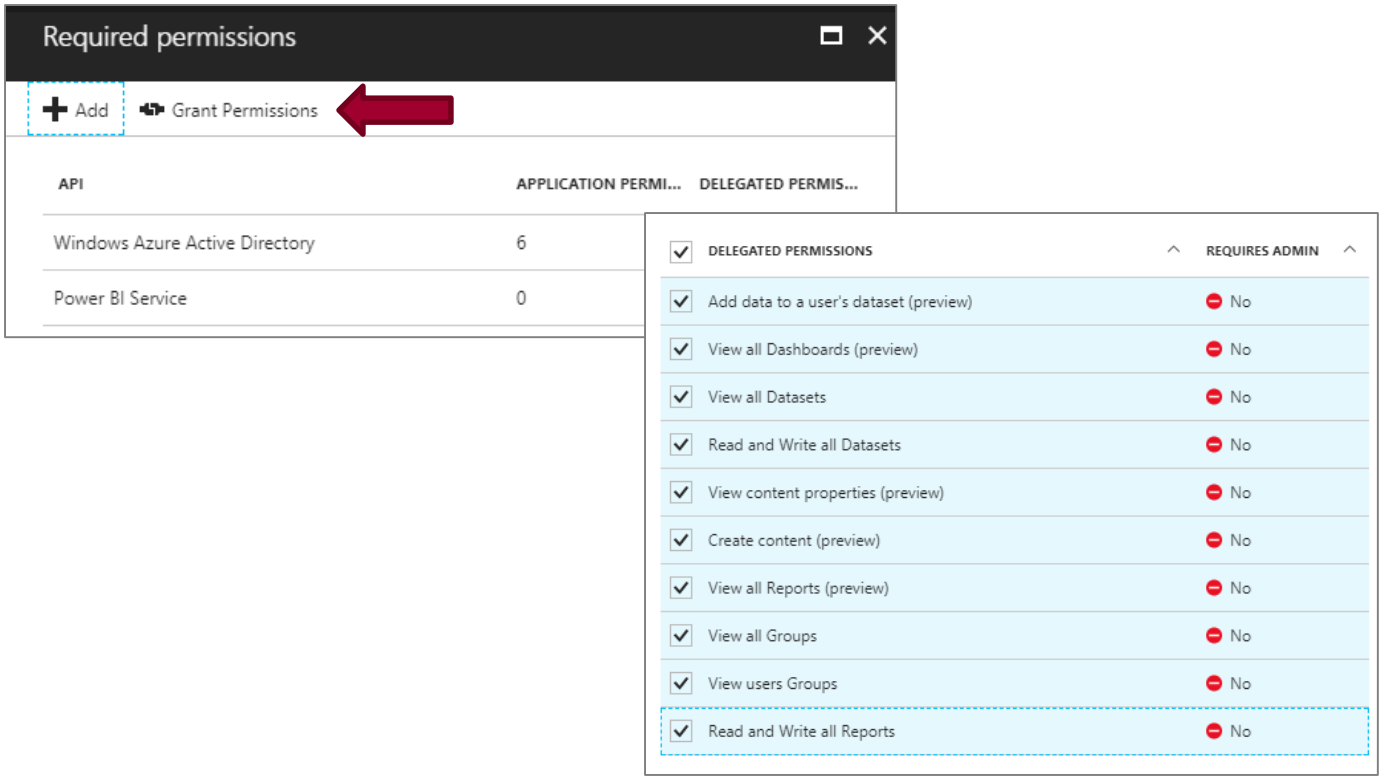
API ACCESS

Required permissions

Keys

Application Permissions

- Applications can be granted permissions to other applications
  - Application permissions are app-only permissions
  - Delegated permissions are (app + user) permissions
  - Delegated permissions requires 1-time consent from user

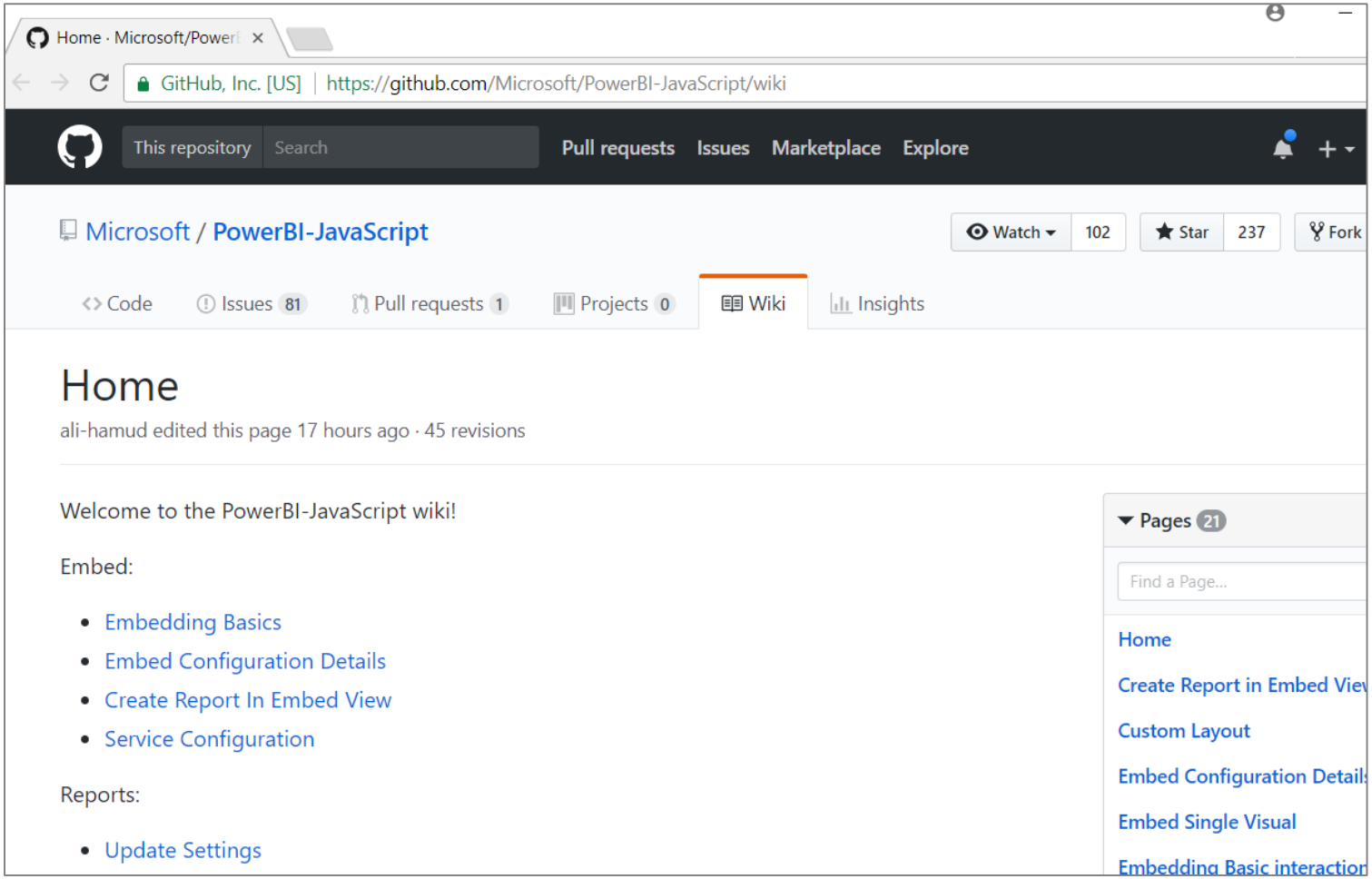


DEMO

PowerBiDaySpa Sample App

Power BI JavaScript API (PBIJS)

- https://github.com/Microsoft/PowerBI-JavaScript/wiki



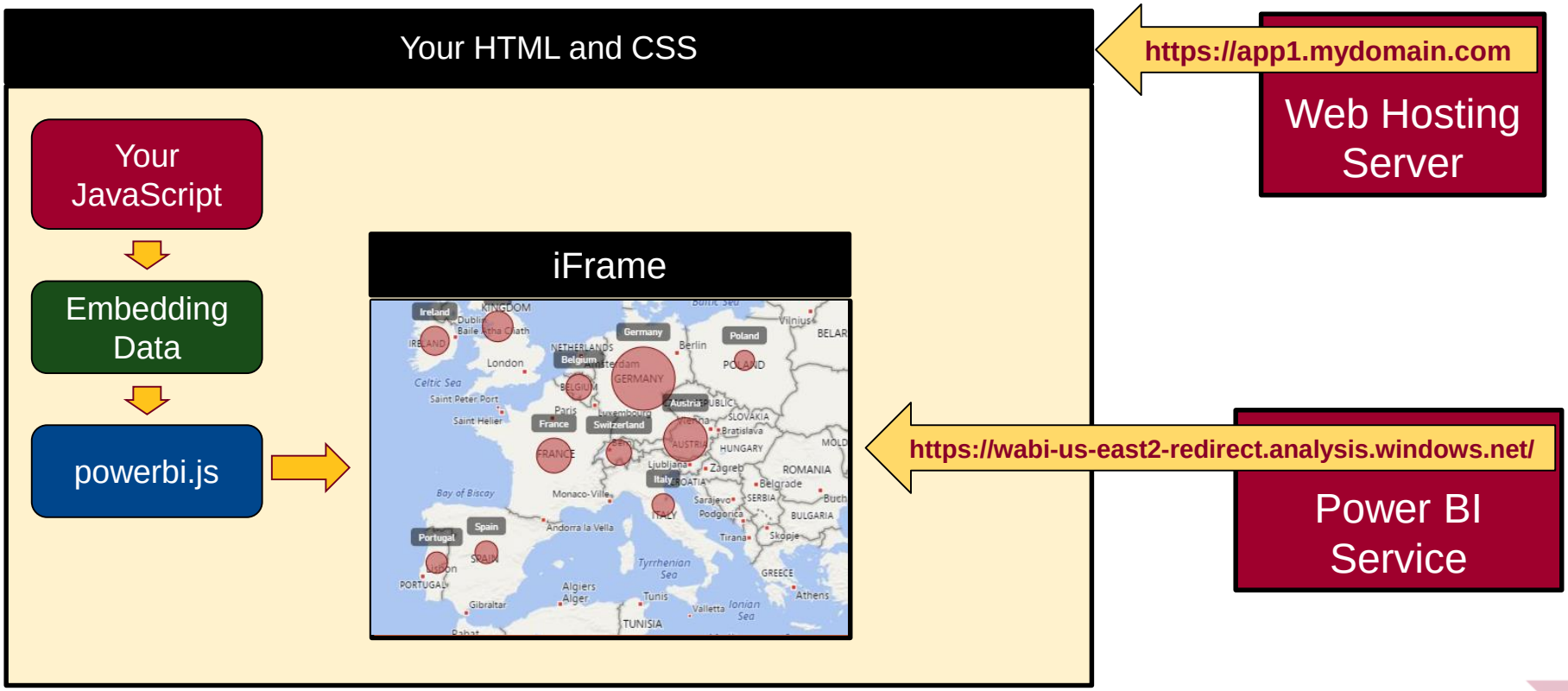
Hello World with Power BI Embedding

- PBIJS library provides **powerbi** as top-level service object
  - You create configuration and then call **powerbi.embed** to embed a report
  - You must pass access token as part of the configuration

```
// data required for embedding Power BI report
var embedReportId = "f10c9de9-a325-4a43-af9f-0cf35cca2ab7";
var embedUrl = "https://app.powerbi.com/reportEmbed?reportId=f10c9de9-a325-4a43-af9f
var accessToken = "H4sIAAAAAAAAEACWWtw6sCBZE_-WlrIR3K02A9x66gQzvvWe0_76tmbySW6pbdf7-Y

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

var config = {
  type: 'report',
  id: embedReportId,
  embedUrl: embedUrl,
  accessToken: accessToken,
  tokenType: models.TokenType.Embed,
};

// 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);
```

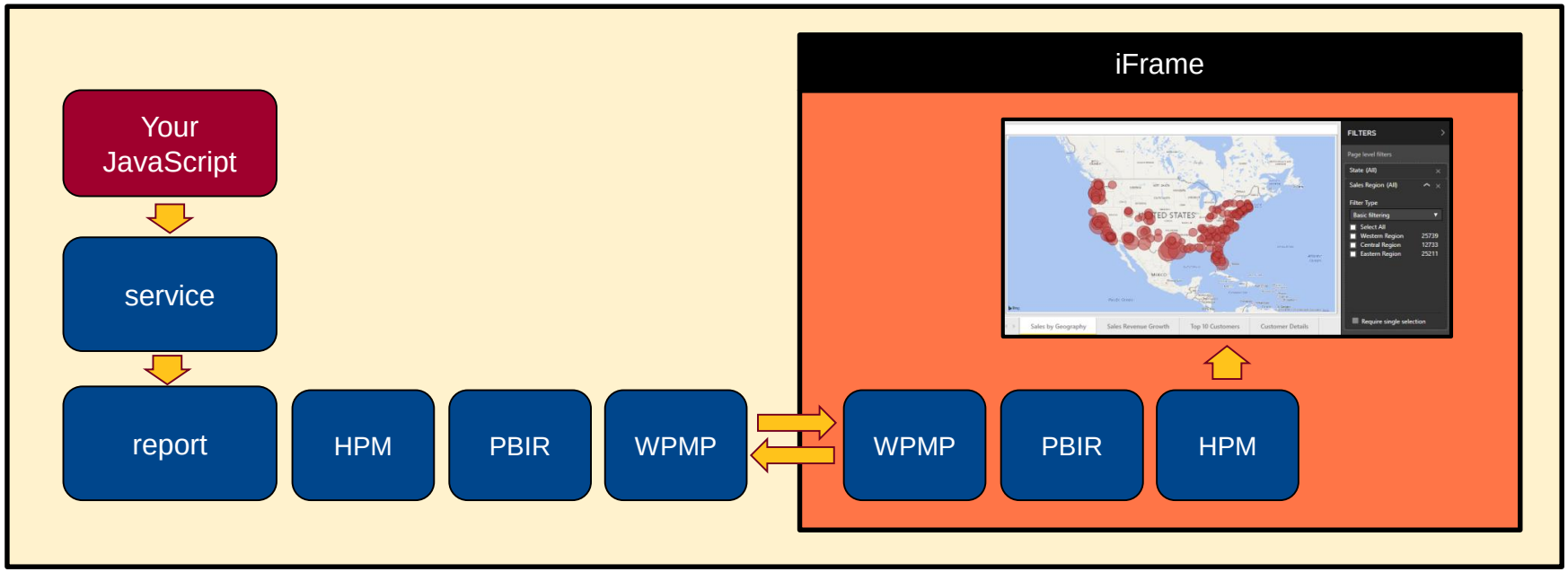
Report Embedding Architecture

- Embedding involves creating an iFrame on the page
  - PBIJS transparently creates iFrame and sets source to Power BI Service
  - ***The iFrame and hosting page originate from different DNS domains***



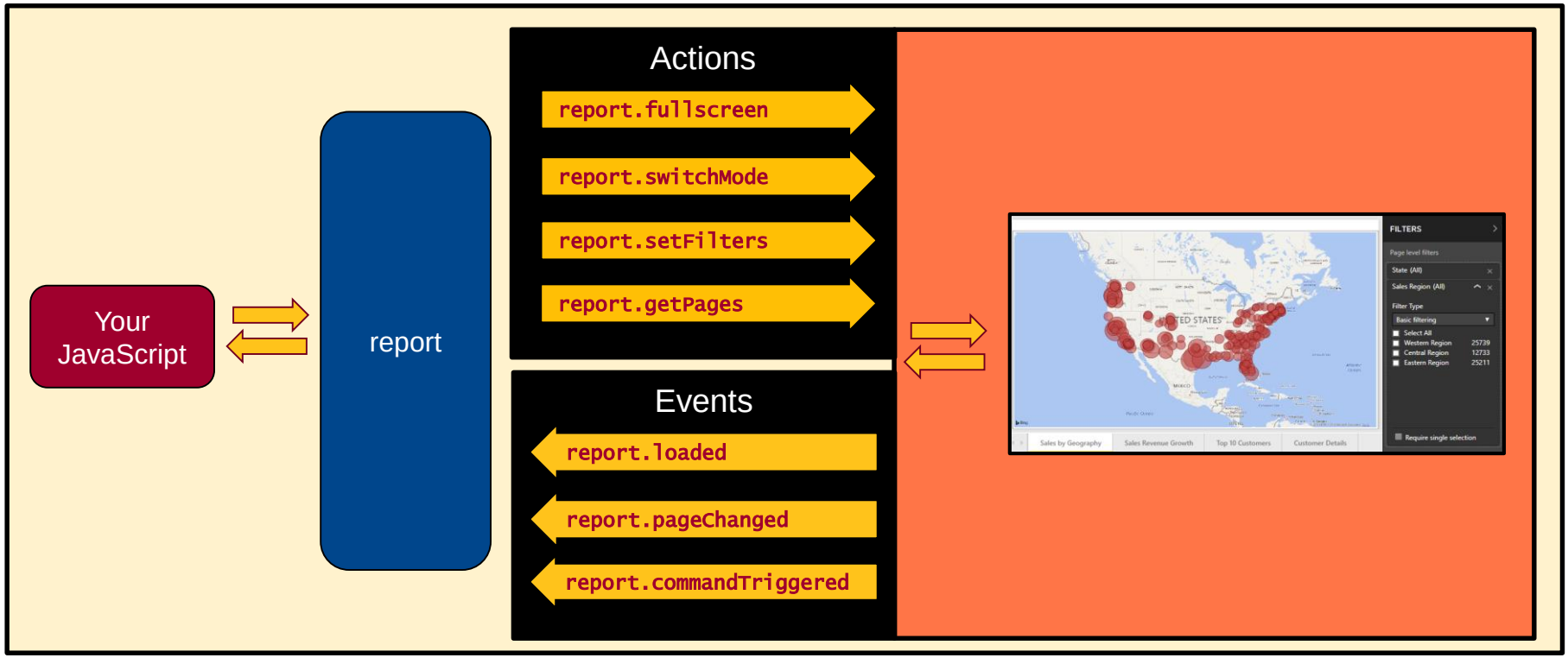
Post Message Communications Flow

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



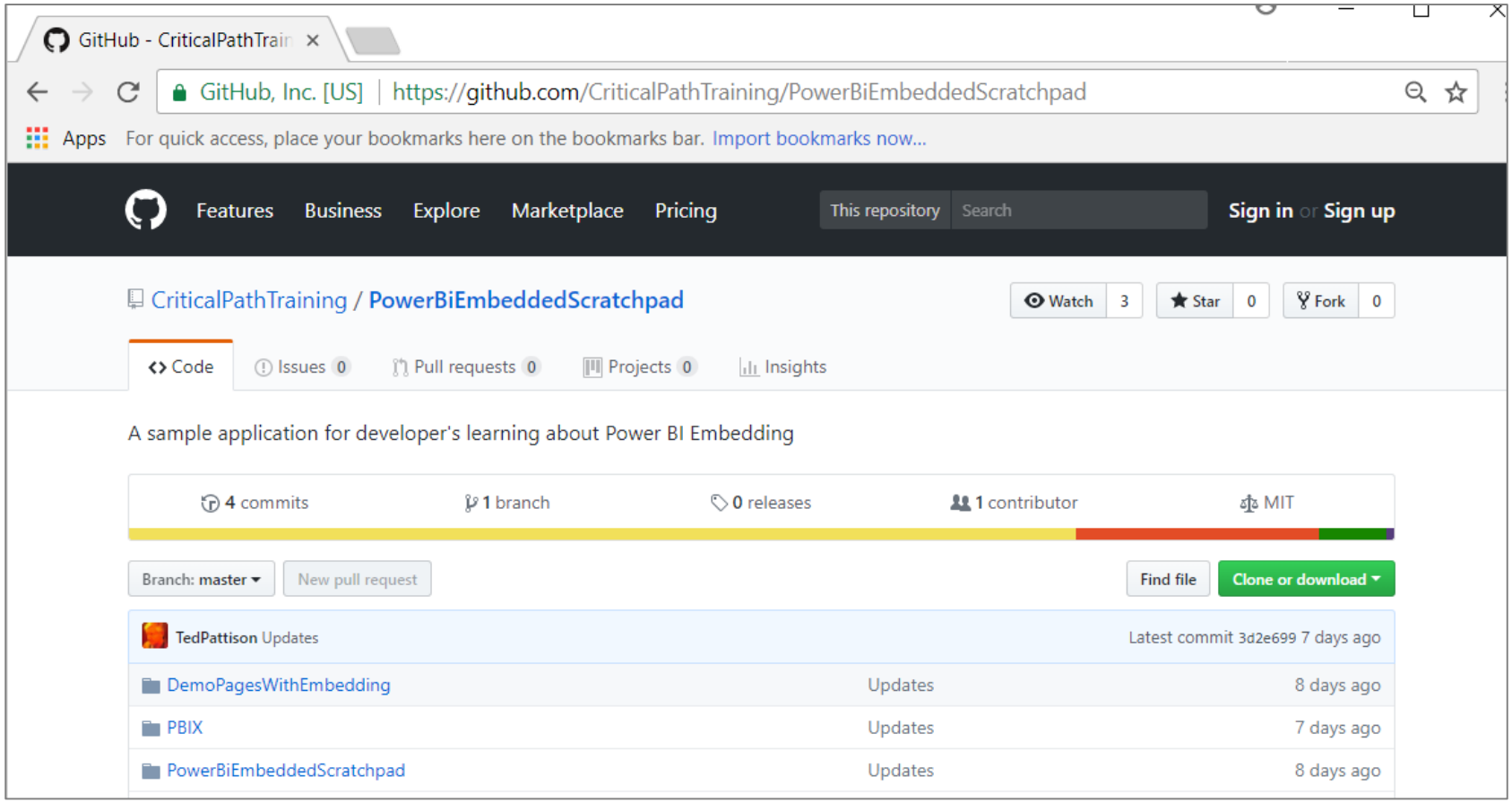
A Promise-based Programming Model

- Design of PBIJS simulates HTTP protocol
  - Creates more intuitive programming model for developers
  - Programming based on asynchronous requests and promises
  - Embedded objects programmed using actions and events



PowerBiEmbeddedScratchpad Sample

https://github.com/CriticalPathTraining/PowerBiEmbeddedScratchpad



DEMO

The Power BI Embedded Scratchpad App

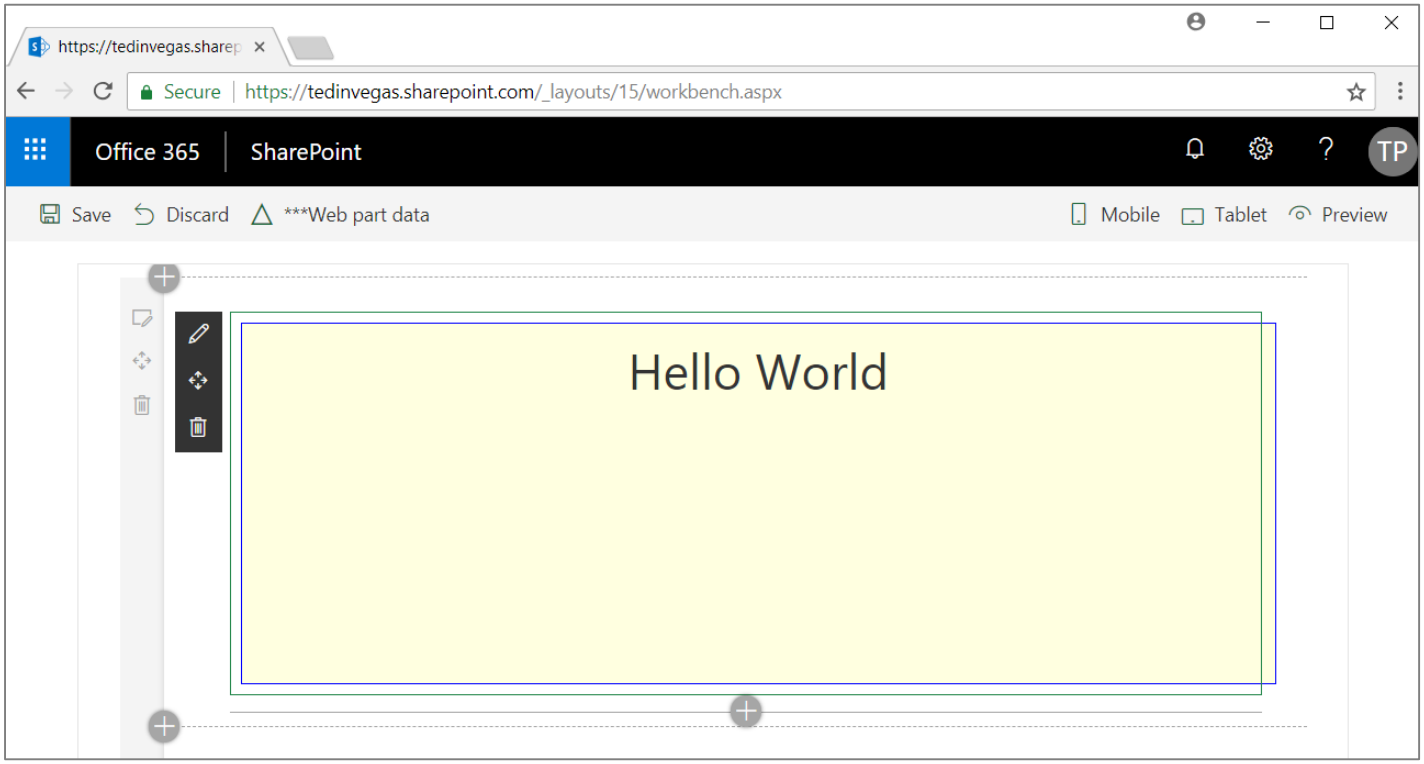
Create a New SPFX Web Part Project

npm

```
c:\Vegas\PBIESPFX\powerbi-api-demo>yo @microsoft/sharepoint

     _-----_
    |       |    .--------------------------.
    |--(o)--|    |      Welcome to the      |
   `---------´   |  SharePoint Client-side  |
    ( _´U`_ )    |    Solution Generator    |
    /___A___\    '--------------------------'
     |  ~  |
     '`___`'
  __ .___. __
 ´   `  |°  ´ Y `

Let's create a new SharePoint solution.
? What is your solution name? powerbi-api-demo
? Which baseline packages do you want to target for your component(s)? SharePoint Online only (latest)
? Where do you want to place the files? Use the current folder
? Do you want to allow the tenant admin the choice of being able to deploy the solution to all sites immediately without
 running any feature deployment or adding apps in sites? Yes
? Which type of client-side component to create? WebPart
? What is your Web part name? reportlist
? What is your Web part description? A sample web part which calls the Power BI Service API
? Which framework would you like to use? (Use arrow keys)
> No JavaScript framework
  React
  Knockout
```

SharePoint Workbench

- You must test/debug in hosted workbench



Configuring Web API Permissions

• package-solution.json - powerbi-api-demo - Visual Studio Code

File Edit Selection View Go Debug Tasks Help

EXPLORER

OPEN EDITORS 1 UNSAVED

• package-solution.json config

POWERBI-API-DEMO

- .vscode
- config
  - config.json
  - copy-assets.json
  - deploy-azure-storage.json
  - package-solution.json
  - serve.json
  - tslint.json
  - write-manifests.json
- dist
- lib
- node_modules
- src

package-solution.json

```
1  {
2    "$schema": "https://dev.office.com/json-schemas/spfx-build/package-solution.schema.json",
3    "solution": {
4      "name": "powerbi-api-demo",
5      "id": "3f4aaaad-387a-405d-8026-e49d653d7a7a",
6      "version": "1.0.0.0",
7      "includeClientSideAssets": true,
8      "skipFeatureDeployment": true,
9      "webApiPermissionRequests": [
10       { "resource": "Power BI Service", "scope": "Group.Read.All" },
11       { "resource": "Power BI Service", "scope": "Dataset.Read.All" },
12       { "resource": "Power BI Service", "scope": "Report.Read.All" },
13       { "resource": "Power BI Service", "scope": "Dashboard.Read.All" }
14     ]
15   },
16   "paths": {
17     "zippedPackage": "solution/powerbi-api-demo.sppkg"
18   }
19 }
```

Packaging Your SPFX Solution

powerbi-api-demo - Visual Studio Code

File Edit Selection View Go Debug Tasks Help

EXPLORER

- OPEN EDITORS
- POWERBI-API-DEMO
  - .vscode
  - config
  - dist
  - lib
  - node_modules
  - sharepoint
    - solution
      - debug
      - powerbi-api-demo.sppkg
  - src

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
PS C:\Vegas\PBIESPFX\powerbi-api-demo> gulp package-solution --ship
Build target: SHIP
[12:37:55] Using gulpfile C:\Vegas\PBIESPFX\powerbi-api-demo\gulpfile.js
[12:37:55] Starting gulp
[12:37:55] Starting 'package-solution'...
[12:37:55] Starting subtask 'configure-sp-build-rig'...
[12:37:55] Finished subtask 'configure-sp-build-rig' after 6.58 ms
[12:37:55] Starting subtask 'package-solution'...
[12:37:55] [package-solution] Found manifest: C:\Vegas\PBIESPFX\powerbi-api-
[12:37:55] [package-solution] Found client-side build resource: reportlist-w
[12:37:55] Verifying configuration...
```

Deploy the Web Part to the App Gallery

Home

Apps for SharePoint

New Upload Sync Share More

All Apps Featured Apps Unavailable Apps ...

Find a file

Title Name App Version ge Deployed

Do you trust powerbi-api-demo?

The client-side solution you are about to deploy contains full trust client side code. The components in the solution can, and usually do, run in full trust, and no resource usage restrictions are placed on them.

This client side solution will get content from the following domains:

SharePoint Online

☑ Make this solution available to all sites in the organization

Please go to the Service Principal Permissions Management Page to approve pending permissions.

powerbi-api-demo

Deploy Cancel

Granting Web API Permissions

API management

Control the third-party APIs that can be called by apps and custom script.

API name	Permission	Access
Pending approval (0)		
Approved (4)		
Power BI Service	Group.Read.All	Approved
Power BI Service	Group.Read.All Dataset.Read.All	Approved
Power BI Service	Group.Read.All Dataset.Read.All Report.	Approved
Power BI Service	Group.Read.All Dataset.Read.All Report.	Approved

admin center

Feedback

Calling the Power BI Service API

```
import { AadHttpClient, HttpClientResponse } from '@microsoft/sp-http';

export default class ReportlistWebPart extends BaseClientSideWebPart<any> {

  private powerbiApiResourceId = "https://analysis.windows.net/powerbi/api";
  private pbiClient: AadHttpClient;

  protected onInit(): Promise<void> {
    this.pbiClient = new AadHttpClient(this.context.serviceScope, this.powerbiApiResourceId);
    return Promise.resolve();
  }

  public render(): void {
    var urlReports: string = "https://api.powerbi.com/v1.0/myorg/reports/";
    this.pbiClient.get(urlReports, AadHttpClient.configurations.v1)
      .then((res: HttpClientResponse): Promise<any> => {
        return res.json();
      })
      .then((reports: any): void => {
        this.domElement.innerHTML =
        `<h2>Power BI Reports</h2>
        <ul>
            ${ reports.value.map(r => `<li><a href='${r.webUrl}' target='_blank' >${r.name}</a></li>`).join("") }
        </ul>`;
    });
  }
}
```

npm install powerbi-client --save

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

PS C:\Vegas\PBIESPFX\embed-report-demo> npm install powerbi-client --save
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.4 (node_modules\fsevents
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.4
)

+ powerbi-client@2.5.1
added 9 packages in 22.762s
```

- postcss-value-parser
- postcss-zindex
- powerbi-client
- powerbi-models
- powerbi-router
- prelude-ls

tsconfig.json

```
{
  "compilerOptions": {
    "target": "es5",
    "forceConsistentCasingInFileNames": true,
    "module": "commonjs",
    "jsx": "react",
    "declaration": true,
    "sourceMap": true,
    "experimentalDecorators": true,
    "skipLibCheck": true,
    "typeRoots": [
      "./node_modules/@types",
      "./node_modules/@microsoft",
      "./node_modules/powerbi-client",
      "./node_modules/powerbi-models"
    ],
    "types": [
      "es6-promise",
      "webpack-env"
    ],
    "lib": [
      "es5",
      "dom",
      "es2015.collection"
    ]
  }
}
```

Embedding Code

```
public render(): void {
  let hostDiv: JQuery = $(this.domElement);
  let height: string = this.properties.reportHeight + "px";
  hostDiv.empty().css({"margin": "0", "padding": "0", "height": height });

  var reqHeaders: HeadersInit = new Headers();
  reqHeaders.append("Accept", "*");
  this.pbiClient.get(this.reportUrl, AadHttpClient.configurations.v1, { headers: reqHeaders })
    .then((res: HttpClientResponse): Promise<any> => {
      return res.json();
    })
    .then((report: any): void => {
      console.log("begin embed...");
      var embedReportId: string = report.id;
      var embedUrl: string = report.embedUrl;
      var accessToken: string = window.sessionStorage["adal.access.token.keyhttps://analysis.windows.net/powerbi/api"];
      // Get models object to access enums for embed configuration
      var models = pbimodels;

      var config: any = {
        type: 'report',
        id: embedReportId,
        embedUrl: embedUrl,
        accessToken: accessToken,
        tokenType: models.TokenType.Aad,
        permissions: models.Permissions.All,
        viewMode: models.ViewMode.View,
        settings: {
          filterPaneEnabled: false,
          navContentPaneEnabled: this.properties.showPageTabs,
        }
      };
      window.powerbi.reset(this.domElement);
      window.powerbi.embed(this.domElement, config);
    });
}
```

Summary

- ✓ Power BI Embedding Fundamentals
- ✓ Authentication with Azure Active Directory
- ✓ Programming with Power BI Service API
- ✓ Embedding with Power BI JavaScript API
- ✓ SharePoint Framework Web Part