Microsoft Power BI

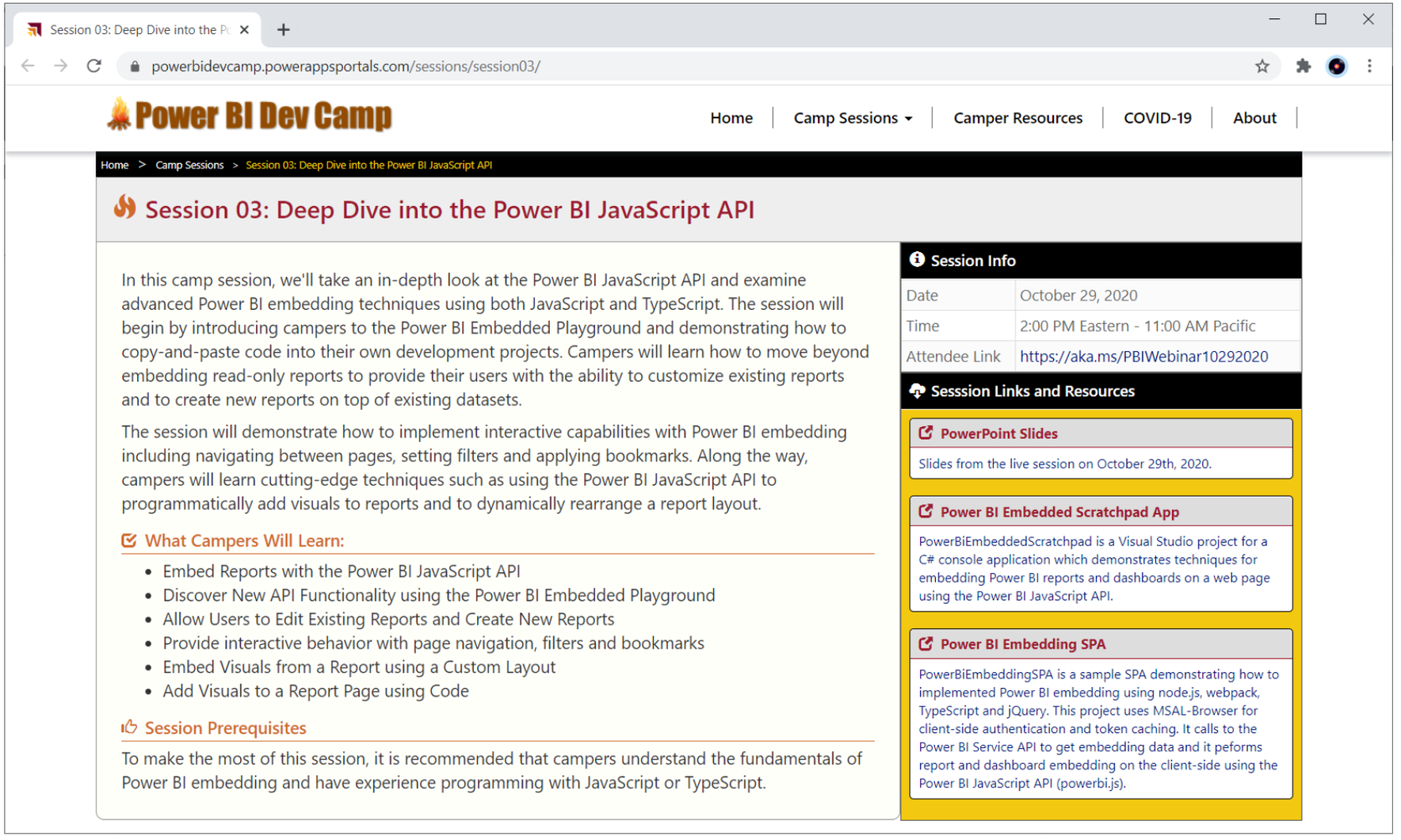
Power BI Dev Camp – Session 3

Deep Dive into the Power BI JavaScript API

Ted Pattison
Principal Program Manager
Customer Advisory Team (CAT) at Microsoft

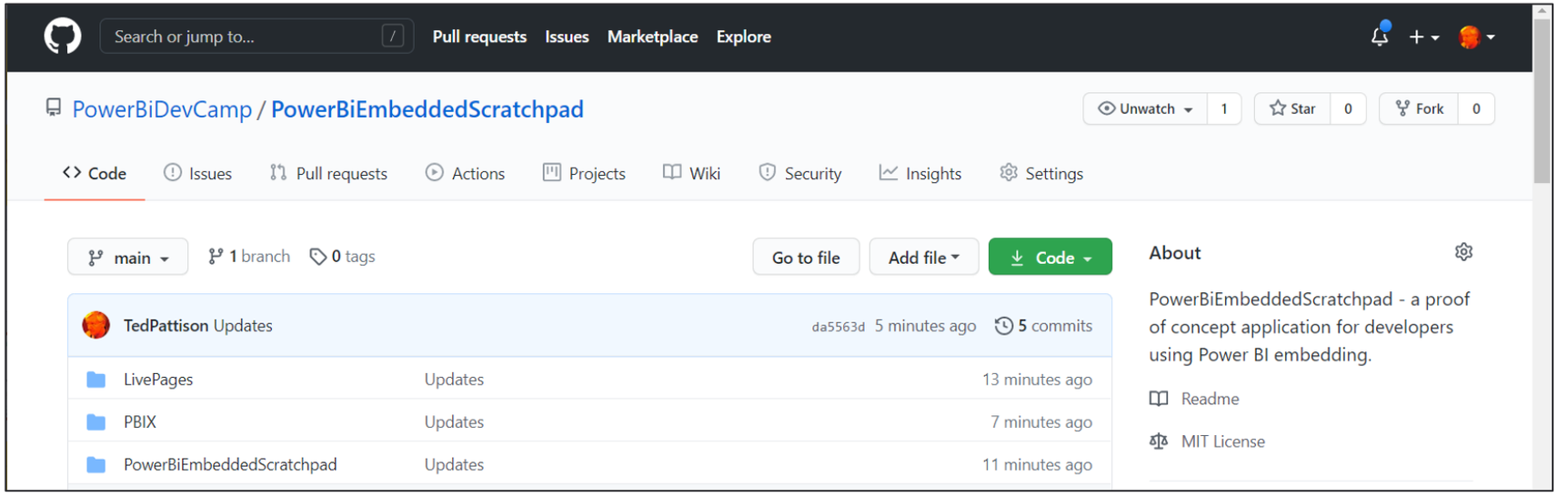
Welcome to Power BI Dev Camp

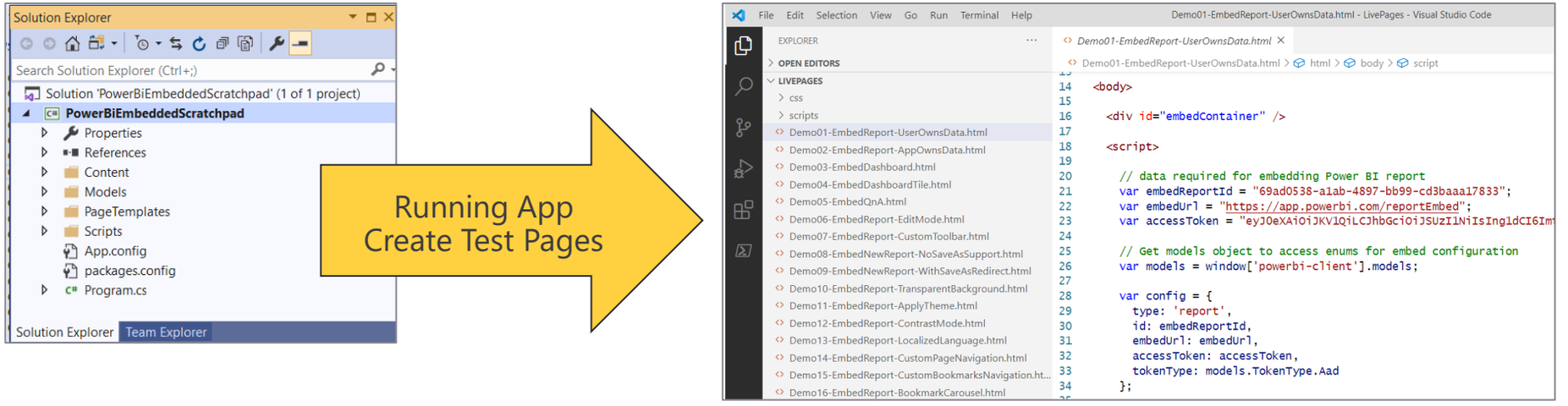
- Power BI Dev Camp Portal - https://powerbidevcamp.net



PowerBiEmbeddedScratchpad Sample Application

https://github.com/PowerBiDevCamp/PowerBiEmbeddedScratchpad



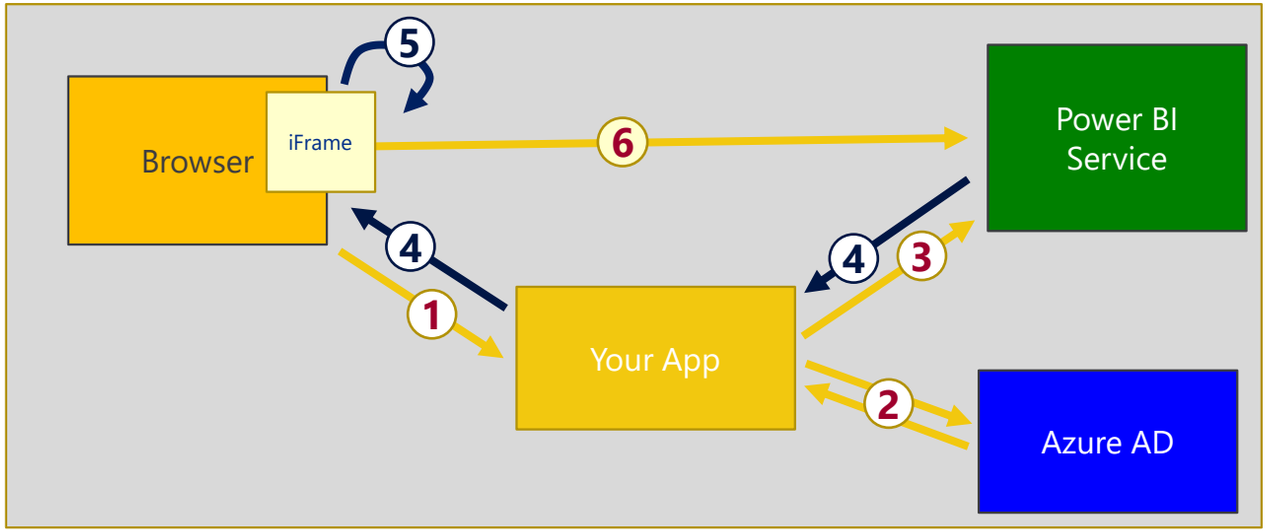


Agenda

- ➢ Power BI Embedding 101
- Building Interactive Experiences
- Programming with Report Events
- Customizing Report Layouts
- Advanced Embedding Topics
- SPA Architecture with MSAL.JS

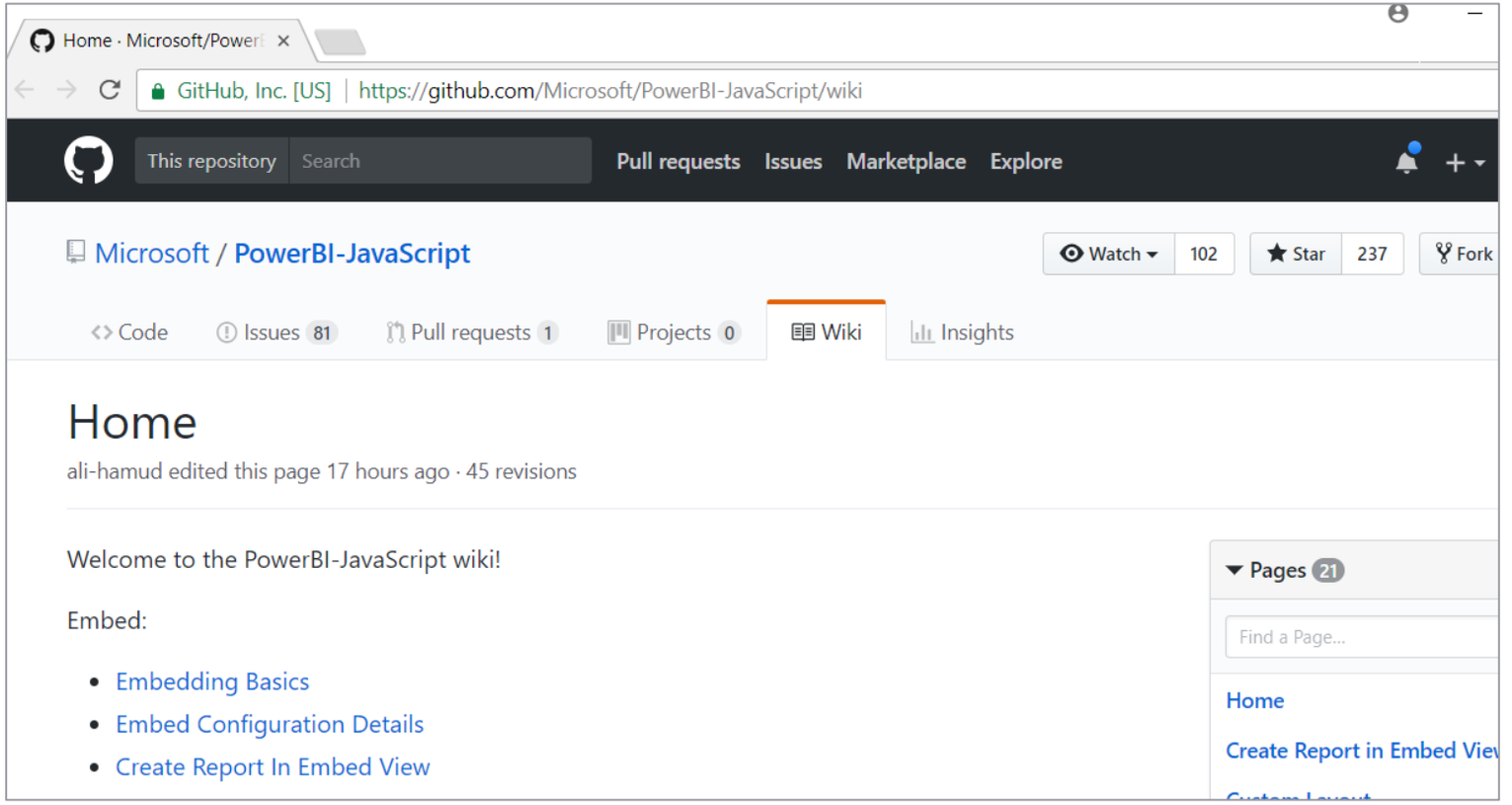
Power BI Embedding – The Big Picture

- 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



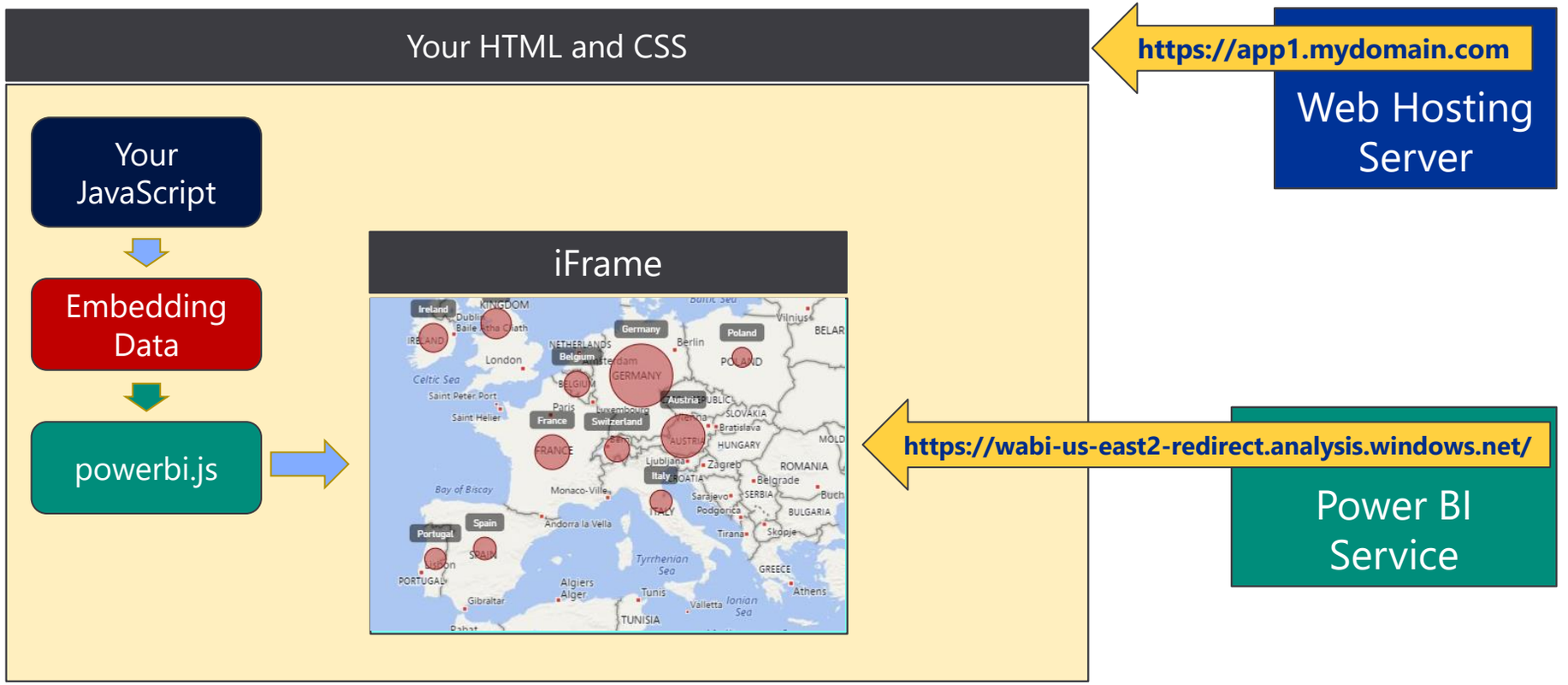
Power BI JavaScript API (powerbi.js)

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



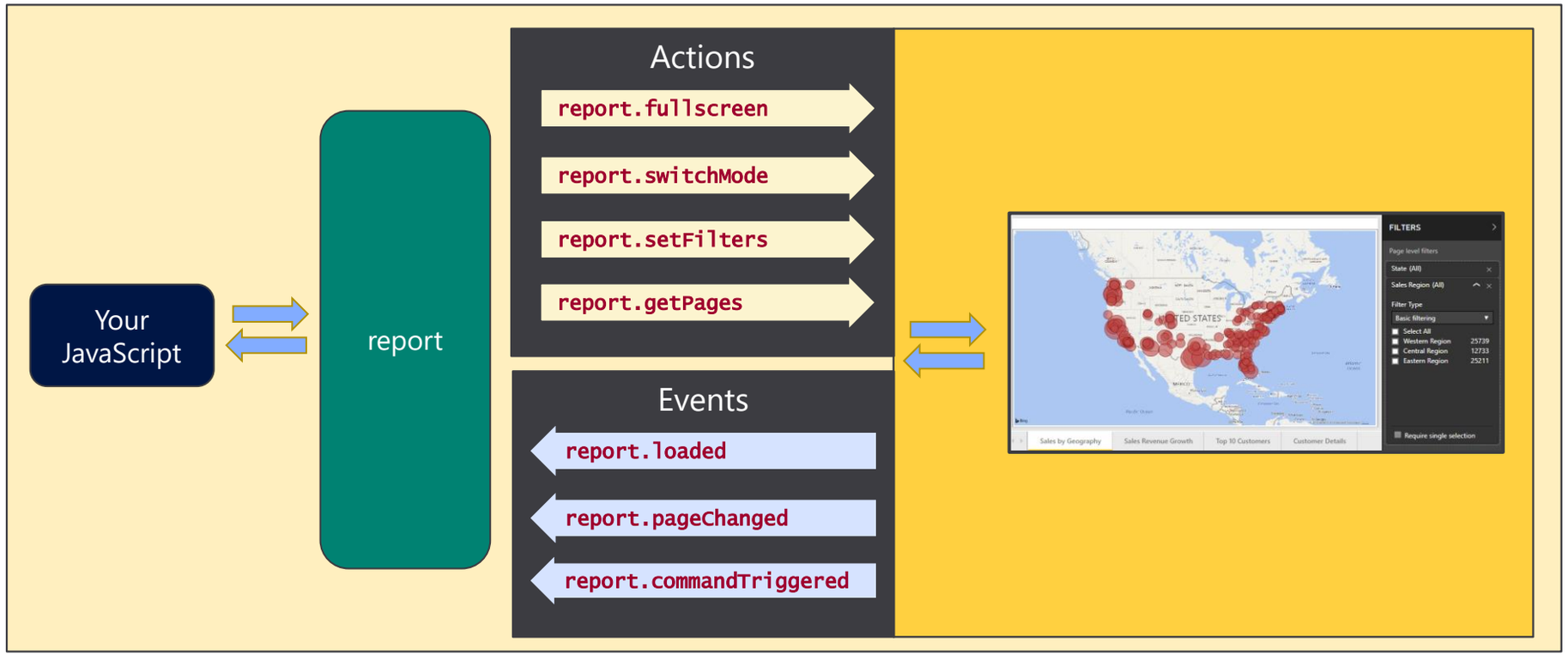
Report Embedding Architecture

- **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



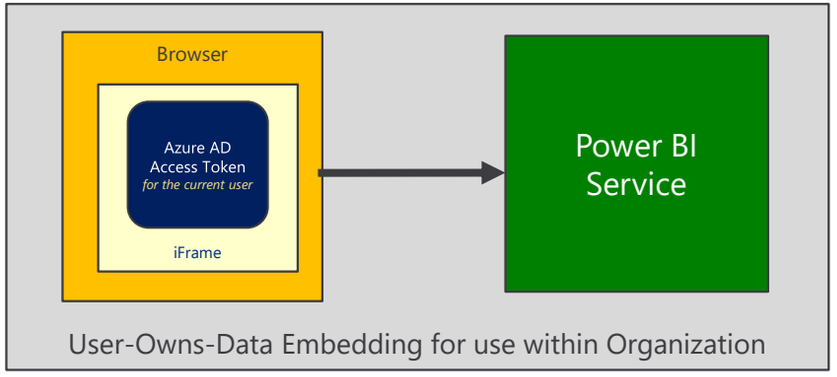
A Promise-based Programming Model

- **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



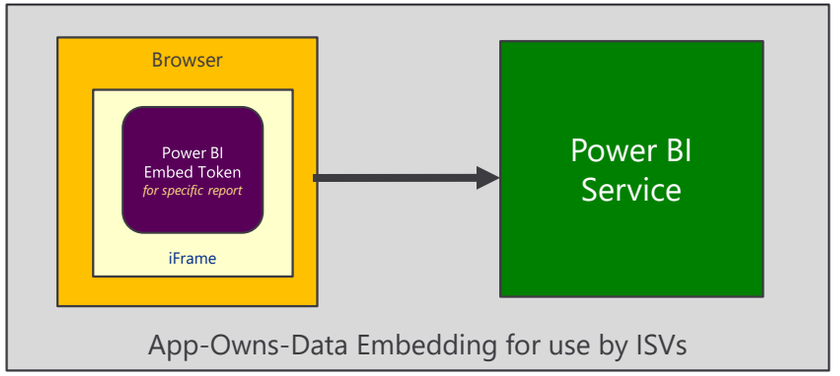
User-Owns-Data Embedding vs App-Owns-Data Embedding

- **User-Owns-Data Embedding**
  - 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



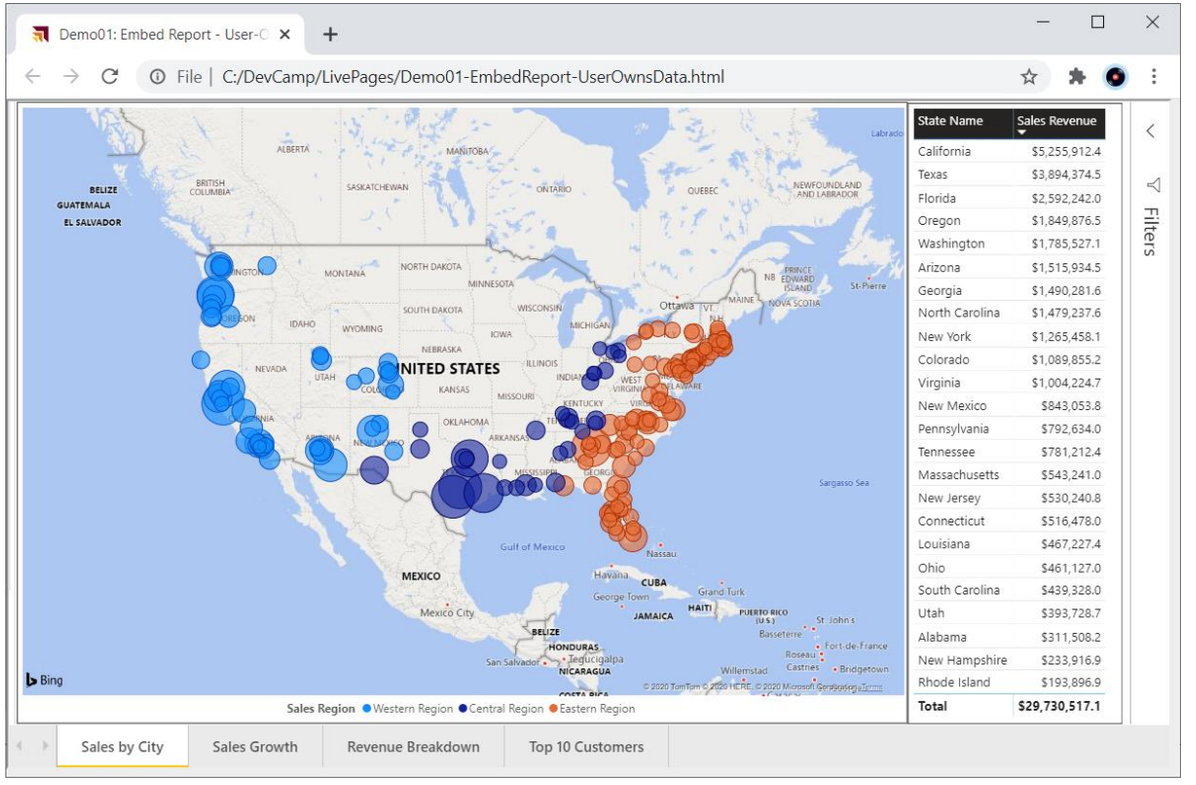
App-Owns-Data Embedding

- No consumers require Power BI license
- Useful for commercial applications
- App authenticates with app-only identity
- Your code runs with admin permissions
- Embed token passed to browser



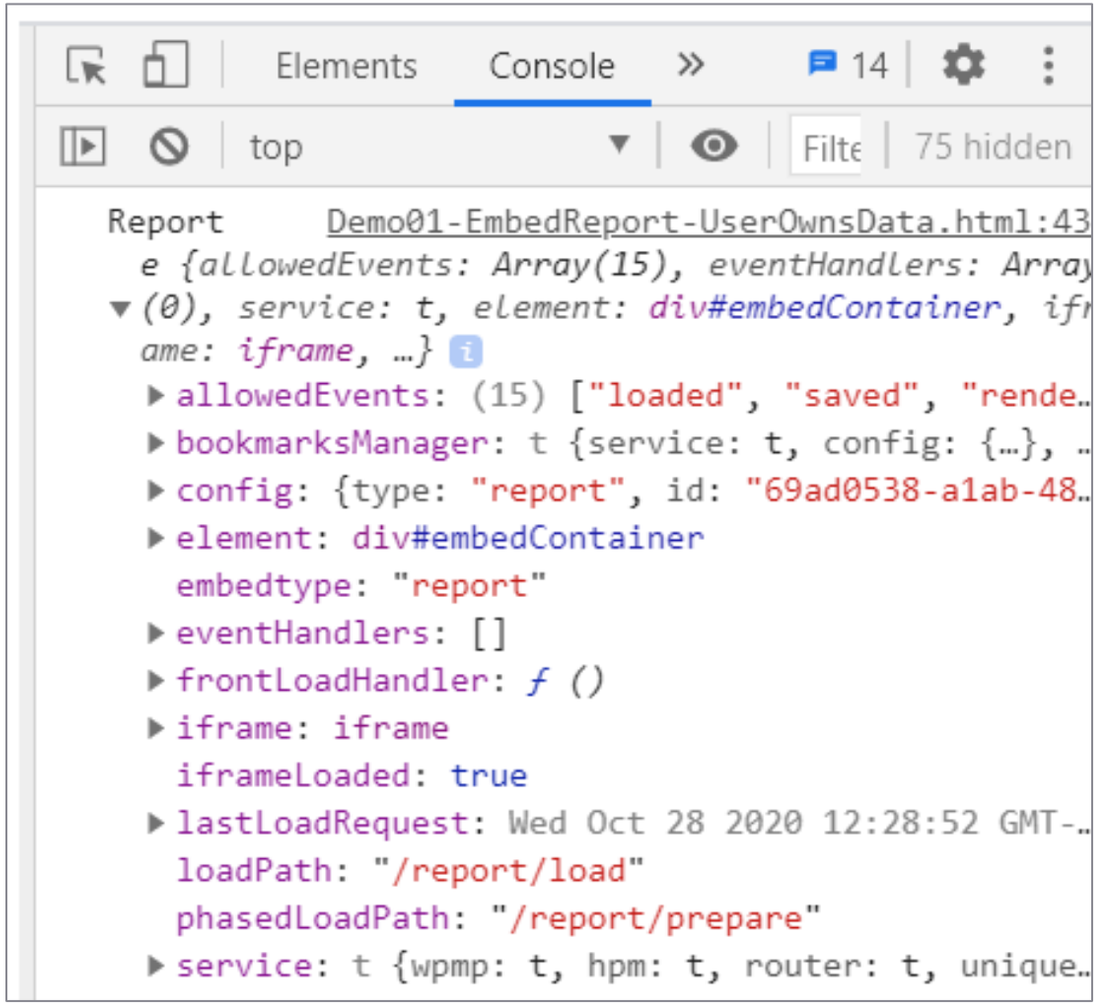
User-Owns-Data Embedding

```
<body>

  <div id="embedContainer" />

  <script>
    // data required for embedding Power BI report
    var embedReportId = "69ad0538-a1ab-4897-bb99-cd3baaa17833";
    var embedUrl = "https://app.powerbi.com/reportEmbed";
    var accessToken = "eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6ImtnMkx

    // 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.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);
```



Viewing Report Object in Chrome Dev Tools

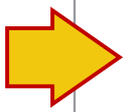
```
// Get a reference to the embedded report HTML element
var reportContainer = document.getElementById('embedContainer');

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

// Display Report object in browser console window
console.log(report);
```



Getting the Data for App-Owns-Data Report Embedding

```
public static async Task<ReportEmbeddingData> GetReportEmbeddingData() {

  PowerBIClient pbiClient = GetPowerBiClient();

  var report = await pbiClient.Reports.GetReportInGroupAsync(workspaceId, reportId);
  var embedUrl = report.EmbedUrl;
  var reportName = report.Name;

  GenerateTokenRequest generateTokenRequestParameters = new GenerateTokenRequest(accessLevel: "edit");
  string embedToken =
        (await pbiClient.Reports.GenerateTokenInGroupAsync(workspaceId,
                                                           report.Id,
                                                           generateTokenRequestParameters)).Token;

  return new ReportEmbeddingData {
    reportId = reportId,
    reportName = reportName,
    embedUrl = embedUrl,
    accessToken = embedToken
  };

}
```

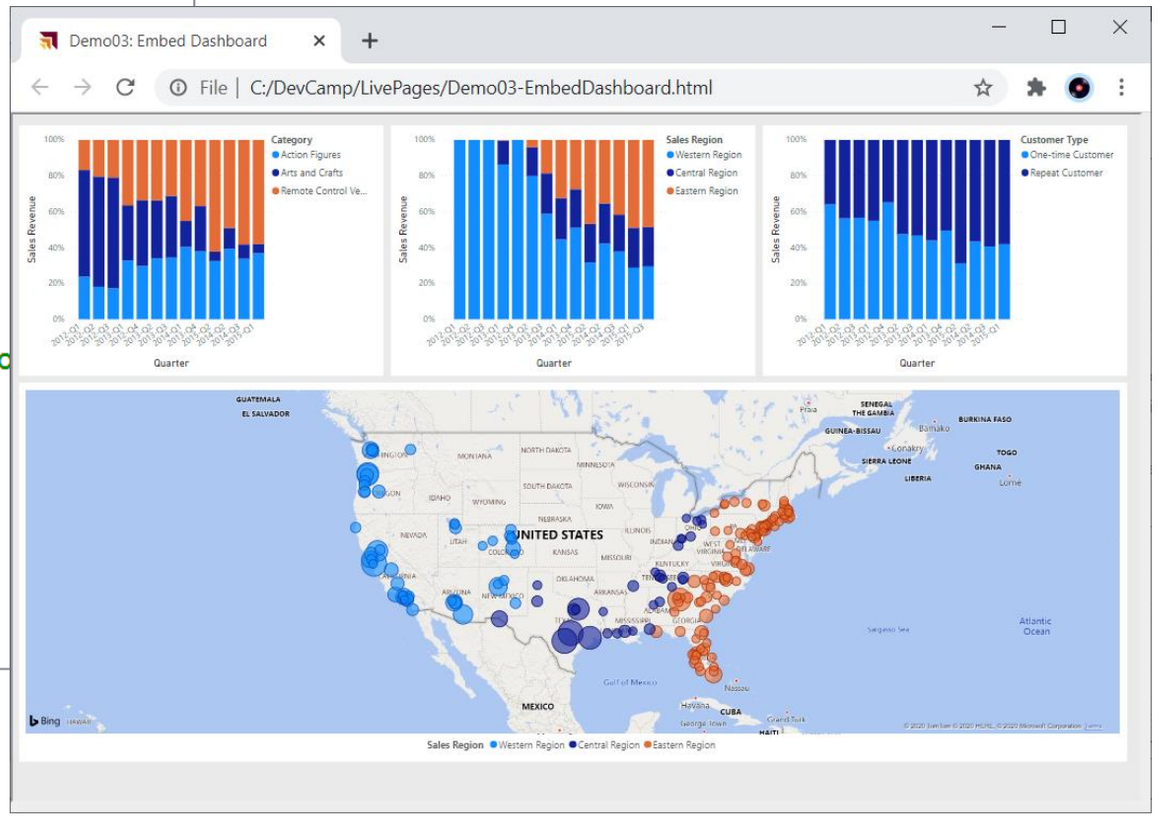
App-Owns-Data Embedding



```
// data required for embedding Power BI report
var embedReportId = "69ad0538-a1ab-4897-bb99-cd3baaa17833";
var embedUrl = "https://app.powerbi.com/reportEmbed";
var accessToken = "H4sIAAAAAAAAEAB2WxQ7OCA6E3-W_ZqQwjTSHMHcYb2Fm7tW--_as

// 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);
```

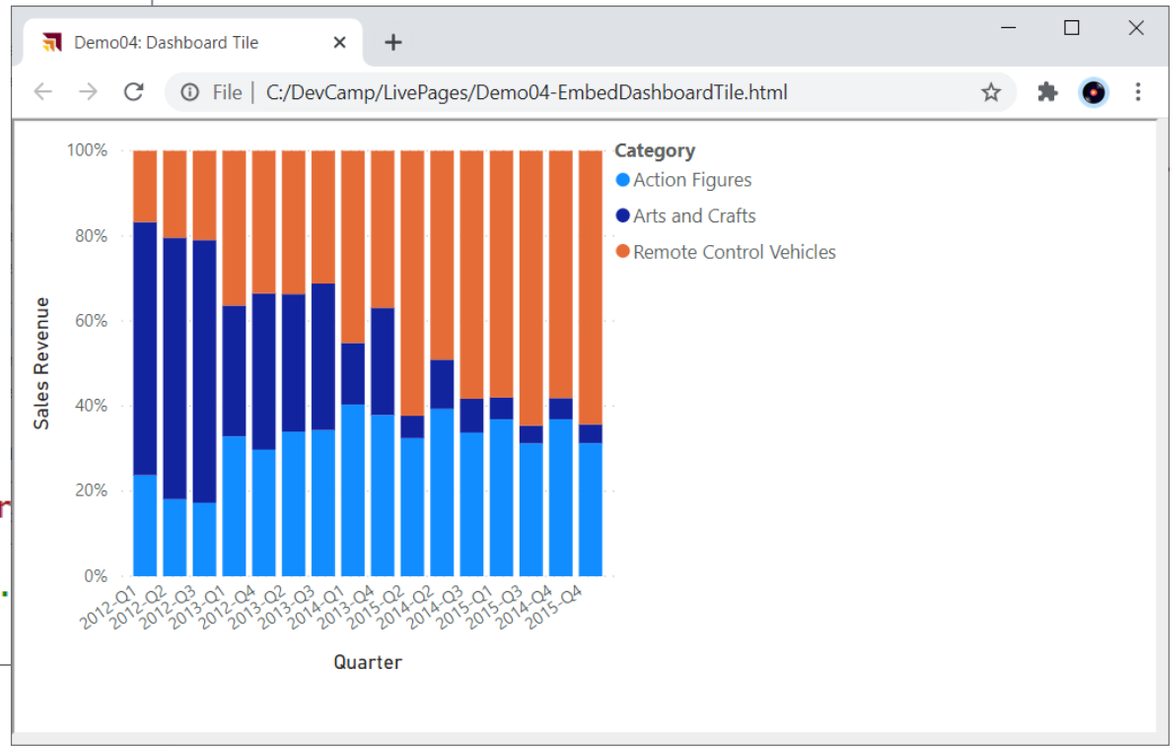
Embedding a Dashboard

```
// data required for embedding Power BI report
var embedDashboardId = "c2d667f8-1023-413c-95e4-138b12950013";
var embedUrl = "https://app.powerbi.com/dashboardEmbed";
var accessToken = "H4sIAAAAAAAAEAB2WtQ7sCBZE_-WlHslMIO3gbjMzZWZm9mr_fXs2r-jo3lF

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

var config = {
  type: 'dashboard',
  id: embedDashboardId,
  embedUrl: embedUrl,
  accessToken: accessToken,
  tokenType: models.TokenType.Embed,
  pageView: "fitToWidth" // options: "actualSize", "fitToWidth", "o
};

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

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



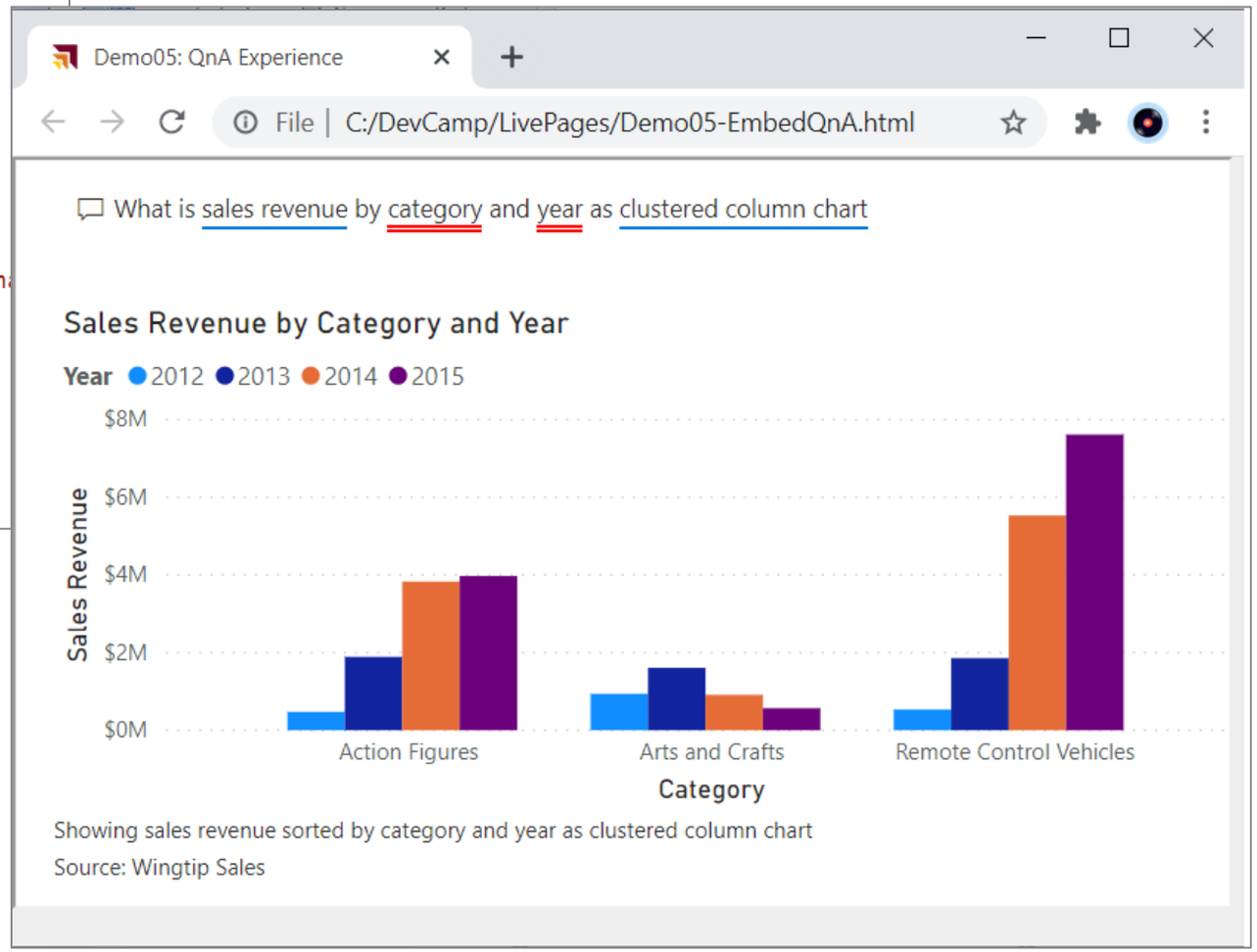
Embedding a Dashboard Tile

```
// data required for embedding Power BI report
var embedDashboardId = "c2d667f8-1023-413c-95e4-138b12950013";
var embedTileId = "68086307-2259-4a79-8b0b-d8ff5486f8bb";
var embedUrl = "https://app.powerbi.com/embed";
var accessToken = "H4sIAAAAAAAAEAB1WRQ7sSBa8y9-6JTO11AuXmcqMOzMzezR3n

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

var config = {
  type: 'tile',
  dashboardId: embedDashboardId,
  id: embedTileId,
  embedUrl: embedUrl,
  accessToken: accessToken,
  tokenType: models.TokenType.Embed,
  width: 600,
  height: 400
};

// 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 tile = powerbi.embed(embedContainer, config);
```



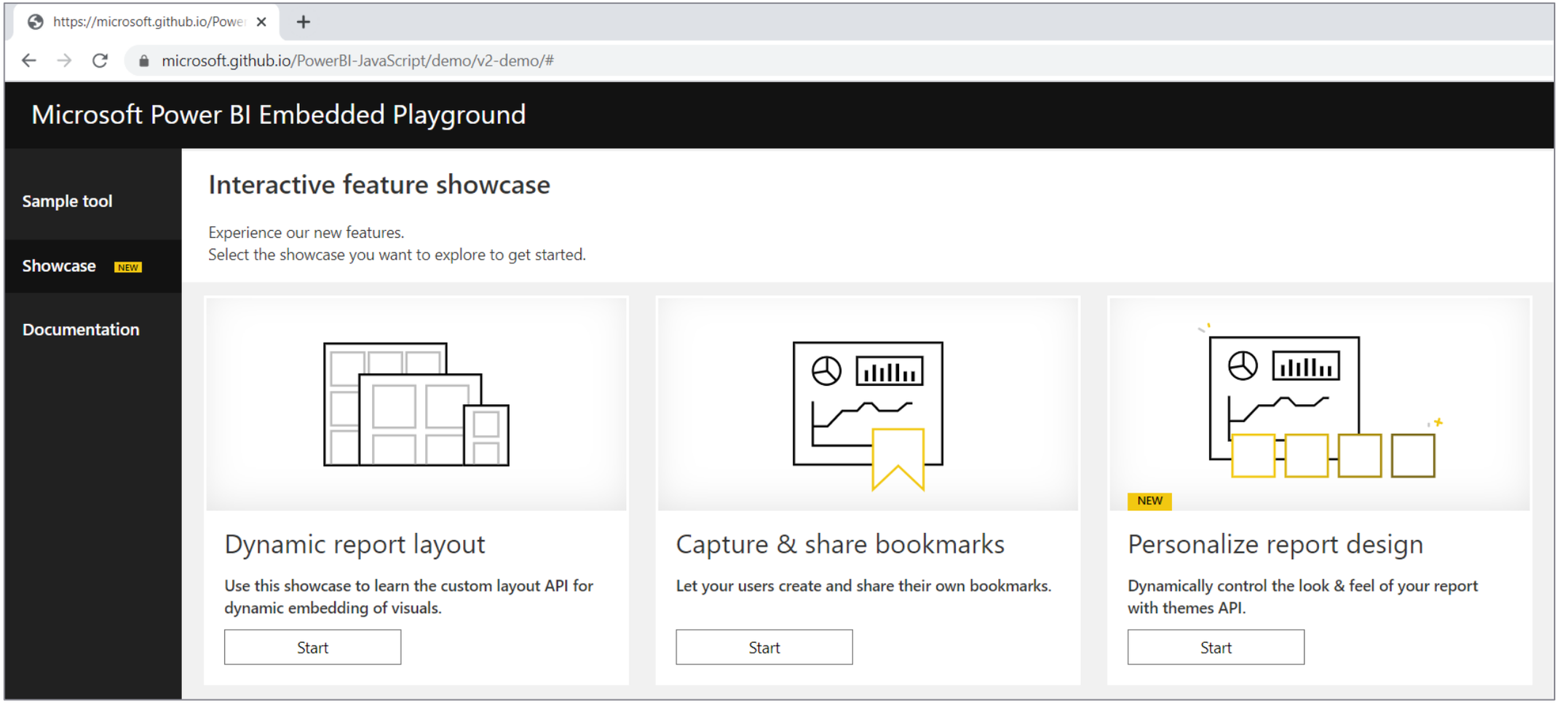
Embedding the QnA Experience

```
// Get data required for embedding
var datasetId = "ca6c991b-5bf1-4fa0-9aed-73d51680a82d";
var embedUrl = "https://app.powerbi.com/qnaEmbed";
var accessToken = "H4sIAAAAAAAAEAC1Wtc7FDHZ8l791JDNF2sLMdM3uzMzsKO-eL6vtpzkzZ-B__rHT

// 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 category and year as clustered column cha
};

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



Power BI Embedded Playground

https://microsoft.github.io/PowerBI-JavaScript/demo/v2-demo/#

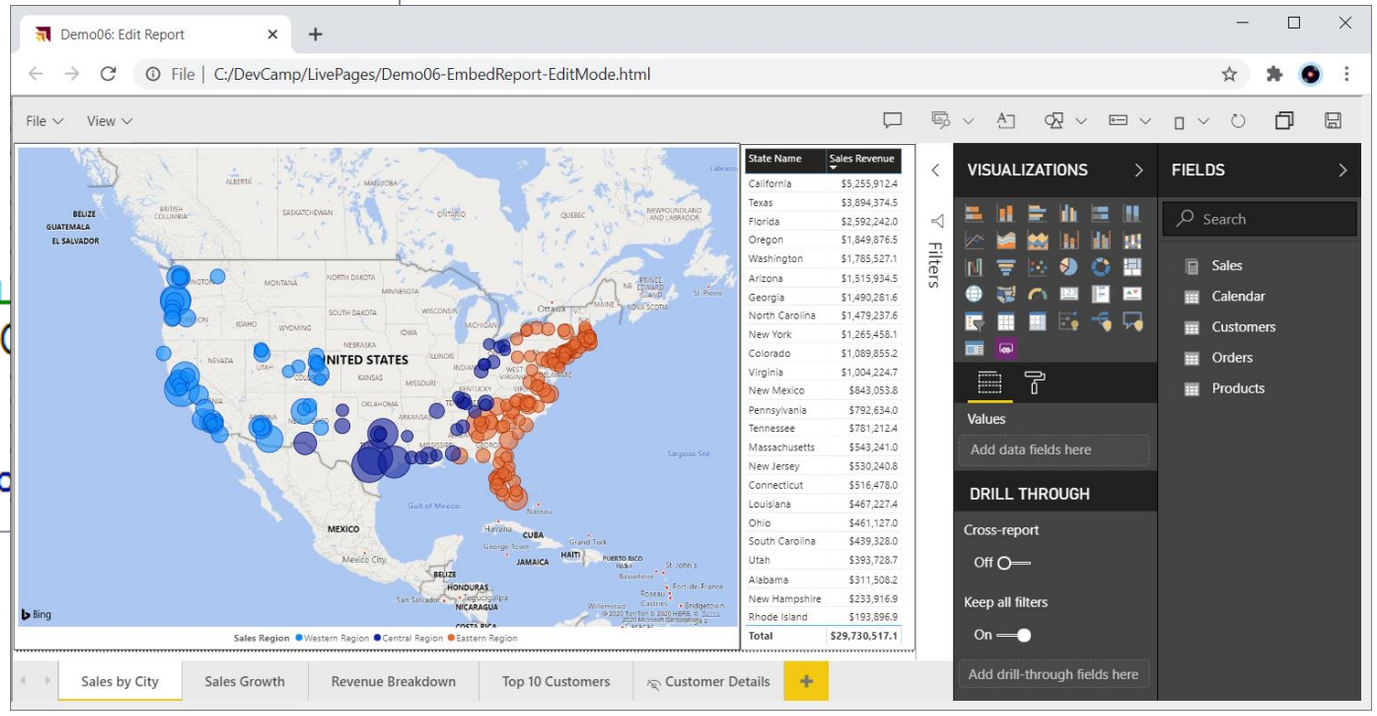


Agenda

- ✓ Power BI Embedding 101
- ➢ Building Interactive Experiences
- Programming with Report Events
- Customizing Report Layouts
- Advanced Embedding Topics
- SPA Architecture with MSAL.JS

Embedding Reports in Edit Mode

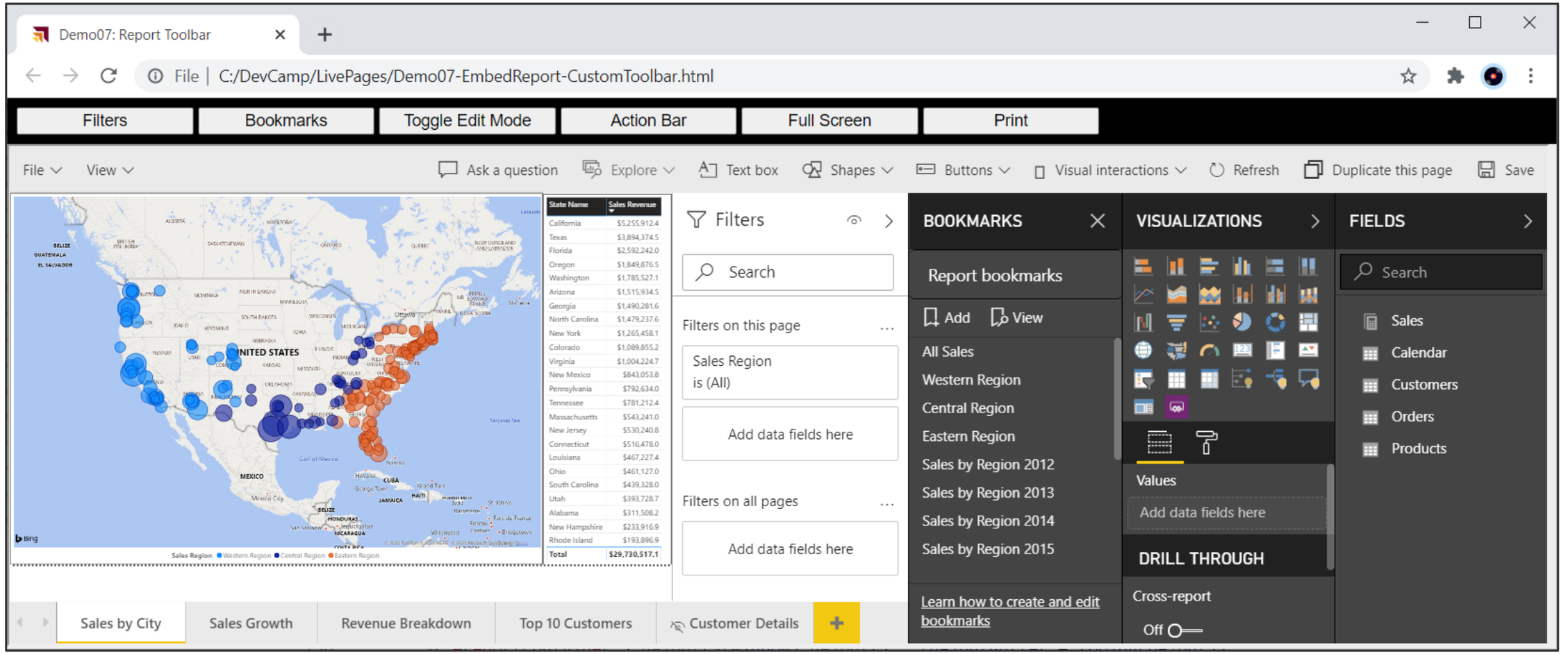
```
// Get models object to access enums for embed configuration
var models = window['powerbi-client'].models;

var config = {
  type: 'report',
  id: embedReportId,
  embedUrl: embedUrl,
  accessToken: accessToken,
  tokenType: models.TokenType.Embed,
  permissions: models.Permissions.All,
  viewMode: models.ViewMode.Edit,
};

// Get a reference to the embedded report HTML
var reportContainer = document.getElementById(

// Embed the report and display it within the
var report = powerbi.embed(reportContainer, co
```



Report Panes

```
// 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,
  permissions: models.Permissions.All,
  viewMode: models.ViewMode.Edit,
  settings: {
    bars: {
      actionBar: { visible: true }
    },
    panes: {
      filters: { visible: true, expanded: false},
      bookmarks: { visible: false },
      visualizations: { expanded: true },
      fields: { expanded: true },
      selection: { visible: false },
      syncSlicers: { visible: false }
    }
  }
};

// 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);
```

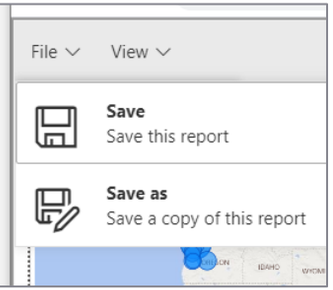
Embedding with a Custom Toolbar



Showing and Hiding the Save and Save as Menus

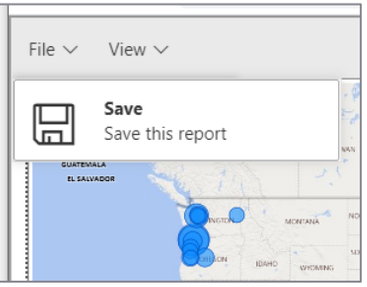
- permissions: models.Permissions.**All**

```
var config = {
  type: 'report',
  id: embedReportId,
  embedUrl: embedUrl,
  accessToken: accessToken,
  tokenType: models.TokenType.Embed,
  permissions: models.Permissions.All,
  viewMode: models.ViewMode.Edit,
};
```



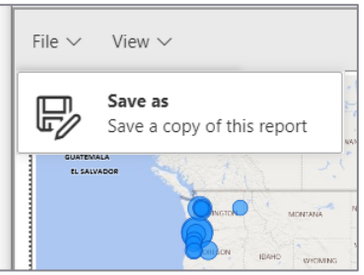
- permissions: models.Permissions.**ReadWrite**

```
var config = {
  type: 'report',
  id: embedReportId,
  embedUrl: embedUrl,
  accessToken: accessToken,
  tokenType: models.TokenType.Embed,
  permissions: models.Permissions.ReadWrite,
  viewMode: models.ViewMode.Edit,
};
```



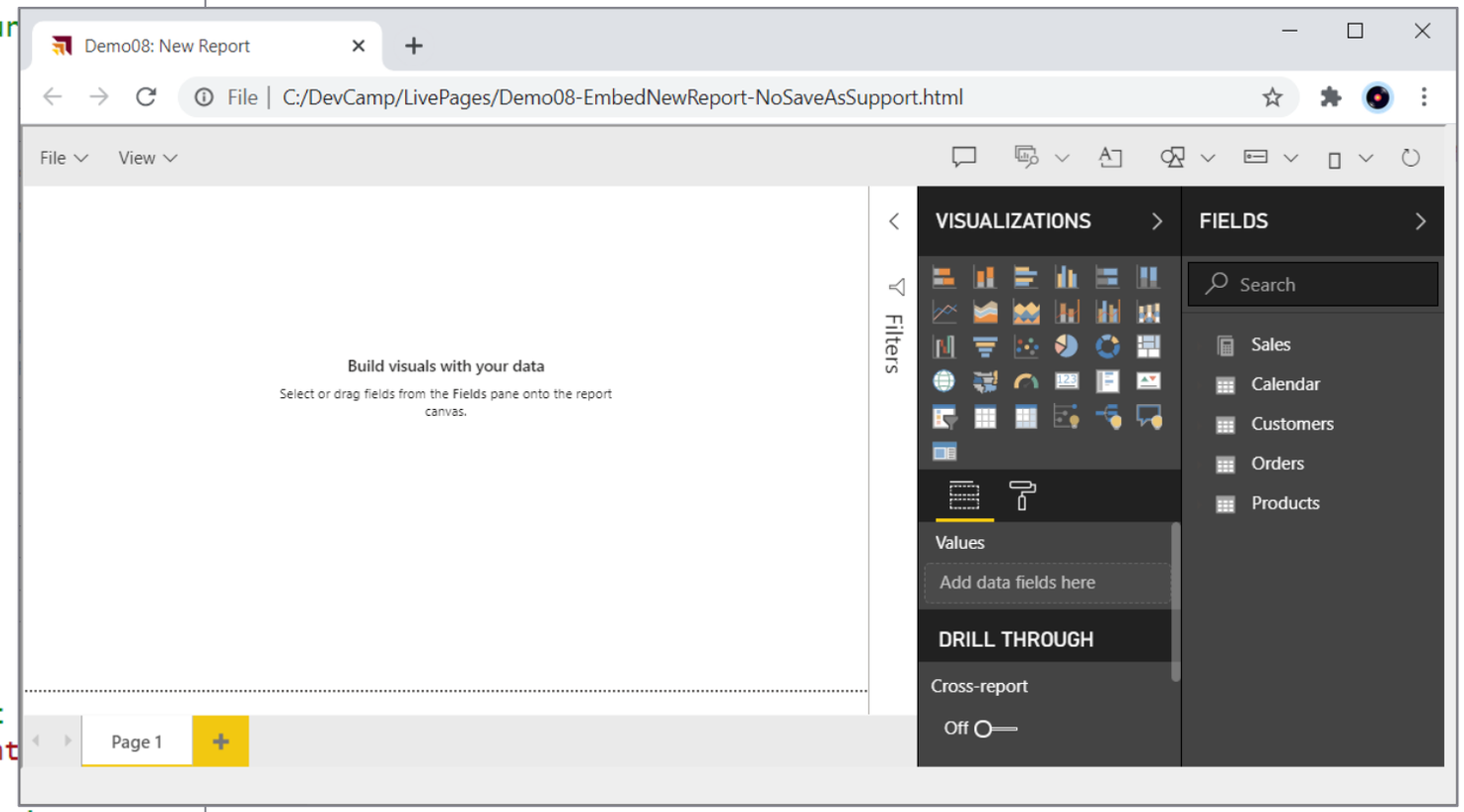
- permissions: models.Permissions.**Copy**

```
var config = {
  type: 'report',
  id: embedReportId,
  embedUrl: embedUrl,
  accessToken: accessToken,
  tokenType: models.TokenType.Embed,
  permissions: models.Permissions.Copy,
  viewMode: models.ViewMode.Edit,
};
```



Embed a New Report

```
// Get data required for embedding
var embedWorkspaceId= "912f2b34-7daa-4589-83df-35c75944d864";
var embedDatasetId = "ca6c991b-5bf1-4fa0-9aed-73d51680a82d";
var embedUrl = "https://embedded.powerbi.com/ReportEmbed";
var accessToken = "H4sIAAAAAAAAEAB2WxQ7sSBZE_-VtPZLLbI_UizIz887MzG7

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

var config = {
  datasetId: embedDatasetId,
  embedUrl: embedUrl,
  accessToken: accessToken,
  tokenType: models.TokenType.Embed,
  settings: {
    bars: {
      actionBar: { visible: true }
    },
    panes: {
      filters: { visible: true, expanded: false },
      bookmarks: { visible: false }
    }
  }
};

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

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



New Report with Save As 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 container with previously embedded object
  powerbi.reset(embedContainer);

  config = {
    type: 'report',
    id: newReportId,
    embedUrl: "https://app.powerbi.com/reportEmbed",
    accessToken: accessToken,
    tokenType: models.TokenType.Aad,
    permissions: models.Permissions.ReadWrite,
    viewMode: models.ViewMode.Edit,
  };

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

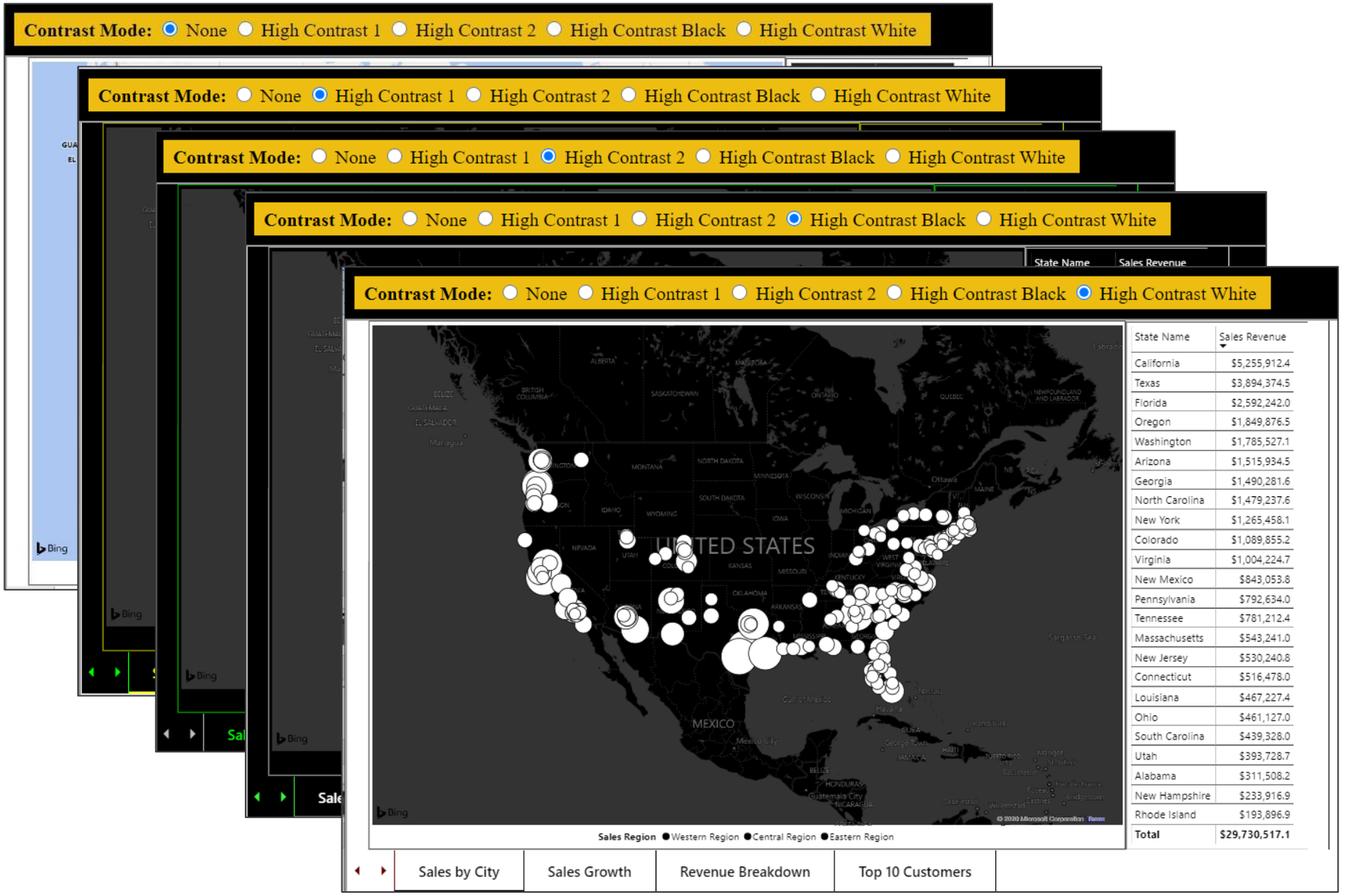
Embed Report with Transparent Background

```
var config = {
  type: 'report',
  id: embedReportId,
  embedUrl: embedUrl,
  accessToken: accessToken,
  tokenType: models.TokenType.Embed,
  permissions: models.Permissions.All,
  viewMode: models.ViewMode.View,
  settings: {
    background: models.BackgroundType.Transparent
  }
};

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

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

Applying Report Themes

```
var theme1 = {
  "name": "theme1",
  "dataColors": ["#FF0000", "#008000", "#0000FF"],
  "background": "#DDDDDD",
  "foreground": "#00008B",
  "tableAccent": "#222222"
};

var config = {
  type: 'report',
  id: embedReportId,
  embedUrl: embedUrl,
  accessToken: accessToken,
  tokenType: models.TokenType.Embed,
  permissions: models.Permissions.All,
  viewMode: models.ViewMode.View,
  theme: { themeJson: theme1 }
};

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

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

Contrast Mode



Loading Reports with Localized Languages

```
var config = {
  type: 'report',
  id: embedReportId,
  embedUrl: embedUrl,
  accessToken: accessToken,
  tokenType: models.TokenType.Embed,
  permissions: models.Permissions.All,
  viewMode: models.ViewMode.Edit,
  settings: {
    localeSettings: { language: "fr", formatLocale: "fr" }
  }
};

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

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

Agenda

- ✓ Power BI Embedding 101
- ✓ Building Interactive Experiences
- ➢ Programming with Report Events
- Customizing Report Layouts
- Advanced Embedding Topics
- SPA Architecture with MSAL.JS

Report Allowed Events

```
▼ allowedEvents: Array(15)
    0: "loaded"
    1: "saved"
    2: "rendered"
    3: "saveAsTriggered"
    4: "error"
    5: "dataSelected"
    6: "buttonClicked"
    7: "filtersApplied"
    8: "pageChanged"
    9: "commandTriggered"
    10: "swipeStart"
    11: "swipeEnd"
    12: "bookmarkApplied"
    13: "dataHyperlinkClicked"
    14: "visualRendered"
```

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

// 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.on('loaded', function (event) { console.log("loaded", event) });
report.on('saved', function (event) { console.log("saved", event) });
report.on('rendered', function (event) { console.log("rendered", event) });
report.on('saveAsTriggered', function (event) { console.log("saveAsTriggered", event) });
report.on('error', function (event) { console.log("error", event) });
report.on('dataSelected', function (event) { console.log("dataSelected", event) });
report.on('buttonClicked', function (event) { console.log("buttonClicked", event) });
report.on('filtersApplied', function (event) { console.log("filtersApplied", event) });
report.on('pageChanged', function (event) { console.log("pageChanged", event) });
report.on('commandTriggered', function (event) { console.log("commandTriggered", event) });
report.on('bookmarkApplied', function (event) { console.log("bookmarkApplied", event) });
report.on('dataHyperlinkClicked', function (event) { console.log("dataHyperlinkClicked", event) });
report.on('visualRendered', function (event) { console.log("visualRendered", event) });
```

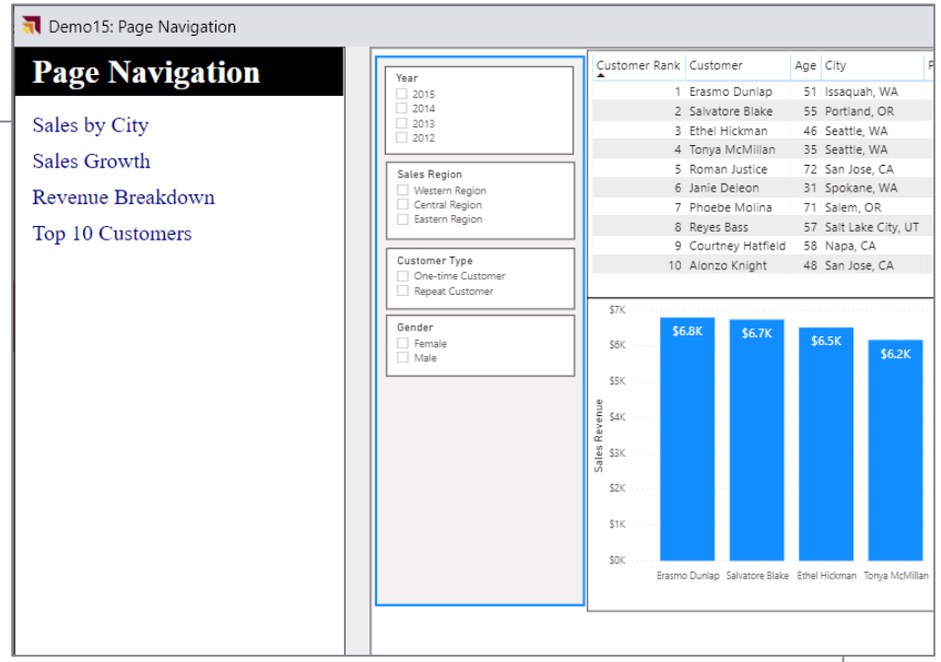
Calling report.getPages() in the loaded event

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

report.on('loaded', function () {

  report.getPages().then(function (pages) {

    // asychronous callback
    for (var index = 0; index < pages.length; index++) {
      // enumerate through pages
    }

  });
});
```

Using async and await keywords in JavaScript

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

report.on('loaded', async function () {
  $("#loading").show();

  // function call with await blocks until async method completes
  var pages = await report.getPages();

  $("#loading").hide();

  for (var index = 0; index < pages.length; index++) {
    // enumerate through pages in report
  }

});
```

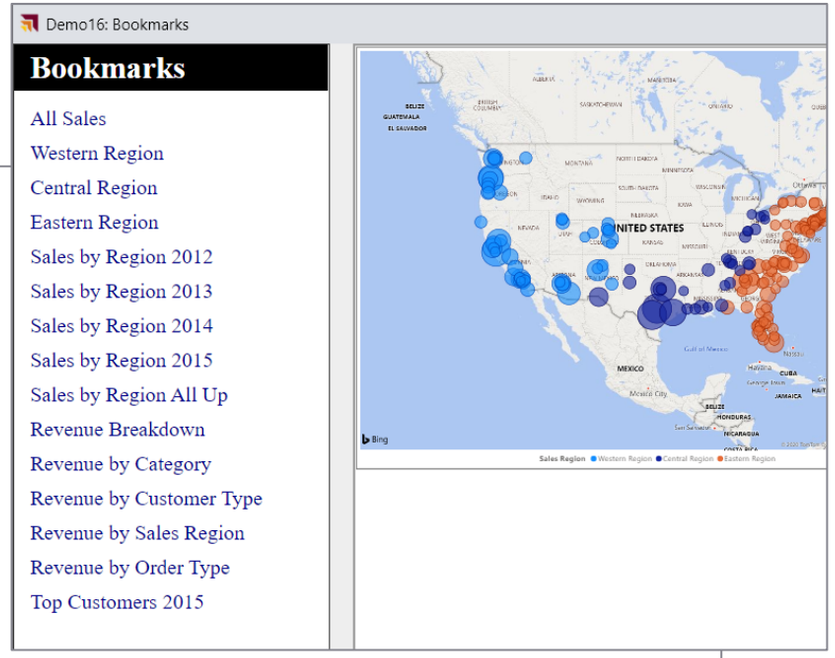
Custom Page Navigation




```
report.on('loaded', async function () {
  $("#loading").show();
  var pages = await report.getPages();
  $("#loading").hide();
  var pageNavigation = $("#page-navigation").empty();
  // enumerate through pages of the report
  for (var index = 0; index < pages.length; index++) {
    // determine which pages are visible and not hidden
    if (pages[index].visibility == 0) { // 0 means visible and 1 means hidden
      // get page display name
      var reportPageDisplayName = pages[index].displayName;
      // use jQuery create a navigation link for each page
      pageNavigation.append($("<li>")
        .append($('<a href="javascript:;" >')
        .text(pages[index].displayName))
        .click(function (domEvent) {
          // get text from link to get page name
          var targetPageName = domEvent.target.textContent ? domEvent.target.textContent : domEvent.target.firstChild.textContent;
          // get target page from pages collection
          var targetPage = pages.find(function (page) { return page.displayName === targetPageName; });
          // display page object in browser console
          console.log("Navigate to " + targetPage.displayName);
          // navigate report to target page
          targetPage.setActive();
        }));
    }
  }
});
```

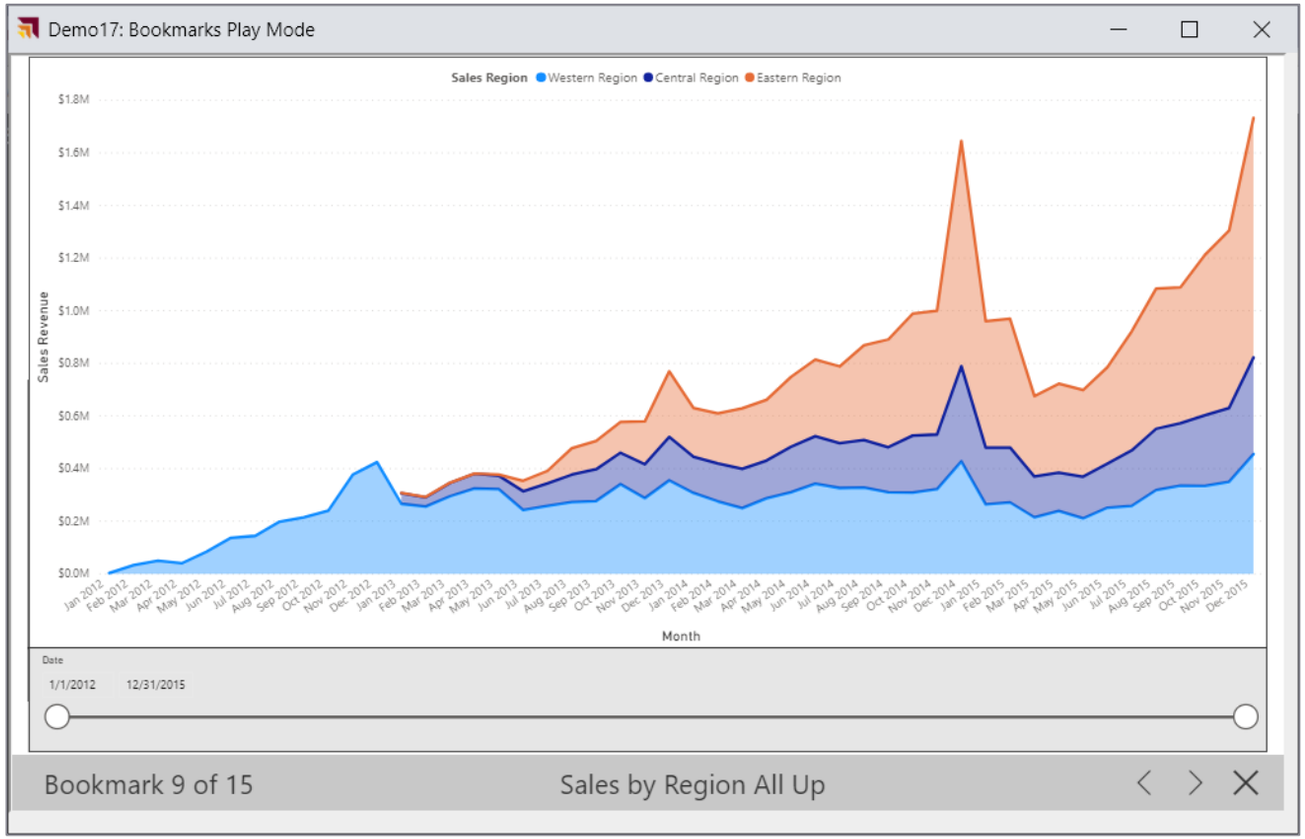
Custom Bookmark Navigation

```
report.on('loaded', async function () {

  var bookmarks = await report.bookmarksManager.getBookmarks();

  var pageNavigation = $("#page-navigation").empty();

  for (var index = 0; index < bookmarks.length; index++) {
    // get name for each book mark
    var bookmarkDisplayName = bookmarks[index].displayName;
    // use jQuery create a navigation link for each page
    pageNavigation.append($("<li>")
      .append($('<a href="javascript:void(0);" >')
        .text(bookmarks[index].displayName))
      .click(function (domEvent) {
        // get text from link to get bookmark display name
        var bookmarkDisplayName = domEvent.target.textContent ? domEvent.target.textContent : domEvent.target.firstChild.textContent;
        // get target bookmark from bookmarks collection
        var targetBookmark = bookmarks.find(function (bookmark) { return bookmark.displayName === bookmarkDisplayName; });
        // apply bookmakr to report
        report.bookmarksManager.apply(targetBookmark.name);
      }));
  }
```



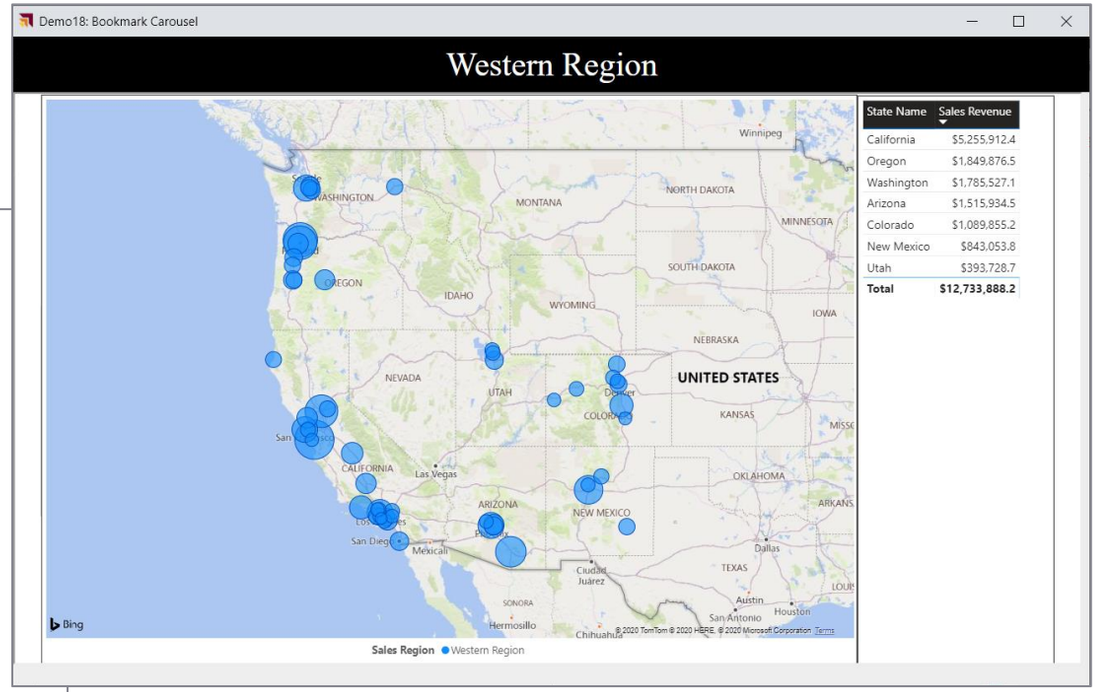
Bookmark Play Mode

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

report.on('loaded', function () {
  report.bookmarksManager.play(models.BookmarksPlayMode.Presentation);
});
```



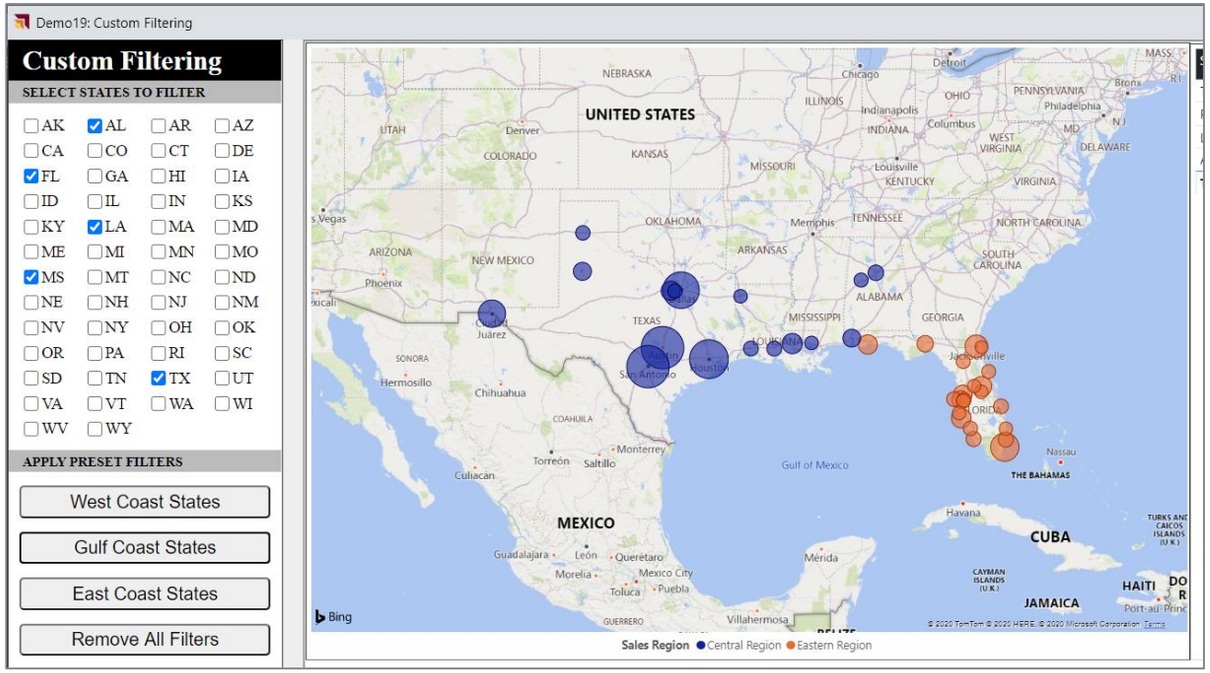
Bookmark Carousel

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

var reportBookmarks;
var nextBookmarksIndex = 1;

report.on('loaded', async function () {
  reportBookmarks = await report.bookmarksManager.getBookmarks();
  $("#toolbar").text(reportBookmarks[0].displayName);
  setInterval(applyNextBookmark, 2000)
});

var applyNextBookmark = function () {
  // get the next bookmark from the bookmarks connection
  var targetBookmark = reportBookmarks[nextBookmarksIndex];
  $("#toolbar").text(targetBookmark.displayName);
  // apply the bookmark
  report.bookmarksManager.apply(targetBookmark.name);

  // advance nextBookmarksIndex variable
  nextBookmarksIndex++;
  if (nextBookmarksIndex >= reportBookmarks.length) {
    nextBookmarksIndex = 0;
  }
}
```



Implementing a Custom Filtering Experience

```
var setStateFilter = function () {

  var states = [];
  $("#state-picker input:checked").each(function () {
    states.push($(this).attr("name"));
  });

  if (states.length > 0) {

    // create basic filter
    const basicStateFilter = {
      $schema: "http://powerbi.com/product/schema#basic",
      target: {
        table: "Customers",
        column: "State"
      },
      operator: "In",
      values: states
    }

    // apply filter to report
    report.setFilters([basicStateFilter]);\
  }
  else {
    // remove all filters
    report.removeFilters();
  }
};
```



Phased Embedding

```
// call load() instead of embed() to load the report while delaying the rendering process
var report = powerbi.load(embedContainer, config);

// when loaded event occurs, set current page then call render()
report.on("loaded", async function () {

  var pages = await report.getPages();

  // find target page using displayName
  var targetPageDisplayName = "Sales Growth";
  var targetPage = pages.find(function (page) {
    return page.displayName === targetPageDisplayName;
  });

  // create date filter
  var dateFilter = {
    $schema: "http://powerbi.com/product/schema#advanced",
    target: { table: "Calendar", column: "Date" },
    logicalOperator: "And",
    conditions: [
      { operator: "GreaterThanOrEqual", value: "2014-01-01 00:00:00" },
      { operator: "LessThanOrEqual", value: "2015-12-31 00:00:00" }
    ]
  }

  // manipulate embed configuation
  config.pageName = targetPage.name;
  config.filters = [dateFilter];

  // Call report.render() to display report
  report.render(config);
});
```

White Label Report Loading

```
<div id="loading" style="text-align:center;">
  <div style="font-size:52px;margin:32px;">Loading report...</div>
  <div>
    <img title="Loading..." src="css/img/loading.gif" style="width:25%;margin:0;" />
  </div>
</div>

<div id="embedContainer" style="display:none;" />
```

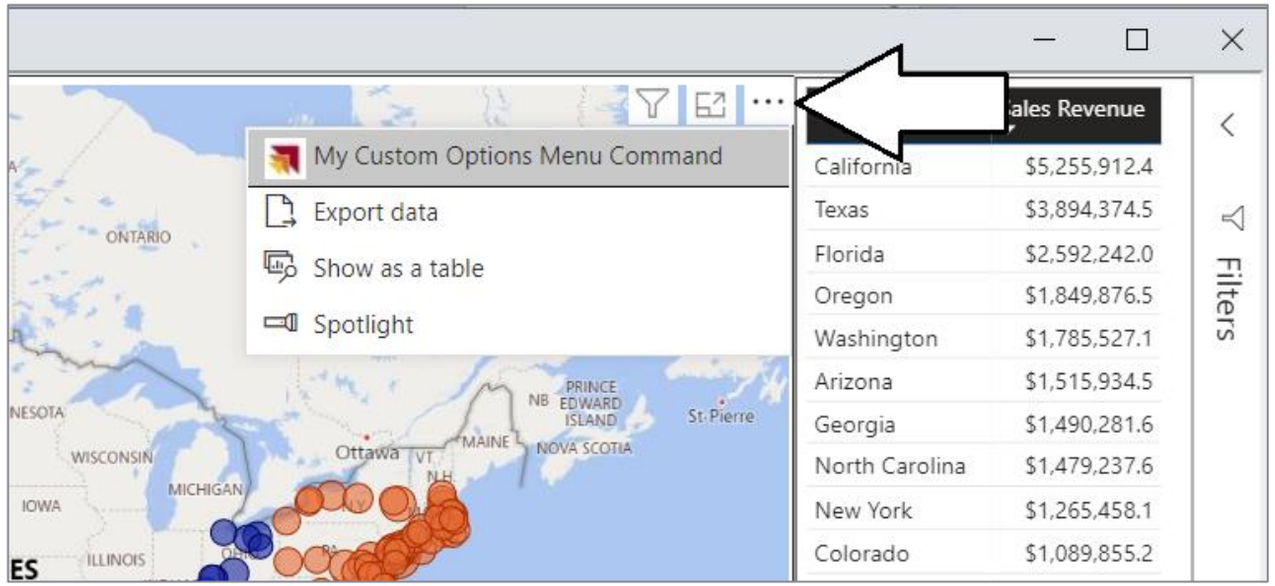
```
// call load() instead of embed() to load the report while delaying the rendering process
var report = powerbi.load(embedContainer, config);

// when loaded event occurs, set current page then call render()
report.on("loaded", async function () {

  $("#loading").hide();
  $("#embedContainer").show();

  // Call report.render() to display report
  report.render(config);
});
```

Bootstrap Optimization in Report Loading

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

var configBootstrap = {
  type: 'report',
  hostname: "https://app.powerbi.com"
};

// call bootstrap on target div to initialize embedding iFrame
powerbi.bootstrap(reportContainer, configBootstrap);

// assume there is now a call across the network and a delay to get this data
var embedReportId = "69ad0538-a1ab-4897-bb99-cd3baaa17833";
var embedUrl = "https://app.powerbi.com/reportEmbed";
var accessToken = "H4sIAAAAAAAAEAB2Wtc7GDJaD7-VvM1KSNzzSFGFmThdm5qz23veb7V35HD_2_

// 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
};

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

Adding Visual Menu Commands

```
// custom menu icon converts to base64 string
var menuIcon = "

var config = {
  type: 'report',
  id: embedReportId,
  embedUrl: embedUrl,
  accessToken: accessToken,
  tokenType: models.TokenType.Embed,
  settings: {
    extensions: [{
      command: {
        name: "MyCustomCommand",
        title: "My Custom Command",
        icon: menuIcon,
        extend: {
          visualContextMenu: {
            title: "My Custom Context Menu Command",
            menuLocation: models.MenuLocation.Top
          },
          visualOptionsMenu: {
            title: "My Custom Options Menu Command",
            menuLocation: models.MenuLocation.Top
          }
        }
      }
    }
    ]
  }
};
```



Hiding Built-in Visual Menu Commands

```
settings: {
  commands: [{ // hiding built-in visual commands
    spotlight: { displayOption: models.CommandDisplayOption.Hidden },
    exportData: { displayOption: models.CommandDisplayOption.Hidden },
    seeData: { displayOption: models.CommandDisplayOption.Hidden }
  }],
  extensions: [{ // adding a custom menu command
    command: {
      name: "CustomExportCommand",
      title: "Custom Export Command",
      icon: menuIcon,
      extend: {
        visualContextMenu: {
          title: "Custom Export Command",
          menuLocation: models.MenuLocation.Top
        },
        visualOptionsMenu: {
          title: "Custom Export Command",
          menuLocation: models.MenuLocation.Top
        }
      }
    }
  }]
}
```

Create a Custom Visual Export Command

```
report.on("commandTriggered", function (command) {
  var pageName = command.detail.page.name;
  var visualName = command.detail.visual.name;
  DownloadExportedVisualDataAsCsv(pageName, visualName);
});

async function DownloadExportedVisualDataAsCsv(pageName, visualName) {
  var reportPages = await report.getPages()
  for (var index = 0; index < reportPages.length; index++) {
    if (reportPages[index].name === pageName) {
      var targetPage = reportPages[index];
      // get visuals for that page
      var pageVisuals = await targetPage.getVisuals();
      // enumerate through visual to find target visual
      for (var indexVisuals = 0; indexVisuals < pageVisuals.length; indexVisuals++) {
        if (pageVisuals[indexVisuals].name === visualName) {
          // retrieve reference to target visual
          var targetVisual = pageVisuals[indexVisuals];
          // call exportData on visual
          var exportedDataResult = await targetVisual.exportData(models.ExportDataType.Summarized, 100);
          var exportedDataRaw = exportedDataResult.data;
          console.log(exportedDataRaw);
          // convert exported data in encoded URI format
          var exportedData = encodeURIComponent(exportedDataRaw);
          // trigger download of exported file as CSV file in browser
          var element = document.createElement('a');
          element.setAttribute('href', 'data:text/csv,' + exportedData);
          element.setAttribute('download', "data.csv");
          element.style.display = 'none';
          document.body.appendChild(element);
          element.click();
          document.body.removeChild(element);
        }
      }
    }
  }
}
```

Agenda

- ✓ Power BI Embedding 101
- ✓ Building Interactive Experiences
- ✓ Programming with Report Events
- ➢ Customizing Report Layouts
- • Advanced Embedding Topics
- • SPA Architecture with MSAL.JS

Determining Visual Names with Visual Inspector

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

report.on('loaded', async () => {

  var visualTable = $("#visual-table")
    .append($("<thead>")
      .append($("<td>").text("Page"))
      .append($("<td>").text("Visual Name"))
      .append($("<td>").text("Type")));

  var pages = await report.getPages()
  for (var index = 0; index < pages.length; index++) {
    var pageVisuals = await pages[index].getVisuals()
    for (var vindex = 0; vindex < pageVisuals.length; vindex++) {
      // add new row to HTML table
      $("#visual-table").append($("<tr>")
        .append($("<td>").text(pageVisuals[vindex].page.displayName))
        .append($("<td>").text(pageVisuals[vindex].name))
        .append($("<td>").text(pageVisuals[vindex].type)));
    }
  }

  $("#exportVisualTable").show();
  $("#exportVisualTable").on("click", ExportTableToCsv);

});
```

Visuals in this report

Page	Visual Name	Type
Sales by City	d0d0c86e46e75dc745b3	tableEx
Sales by City	a846946d789b47a1480c	map
Sales Growth	45ce56ebb00cc645a065	stackedAreaChart
Sales Growth	00babc8baa0cbba0e14e	slicer
Revenue Breakdown	6d96e7cce8ec2abfcea6	hundredPercentStackedColumnChart
Revenue Breakdown	3041835012a21c32a527	hundredPercentStackedColumnChart
Revenue Breakdown	7c8857b43206e939df9e	hundredPercentStackedColumnChart
Revenue Breakdown	82882adf321081db86ae	hundredPercentStackedColumnChart
Top 10 Customers	002fb783e7f00cb02019	tableEx
Top 10 Customers	07ccc665d57a75acb427	basicShape
Top 10 Customers	4554e0bbf1283fe061ae	slicer
Top 10 Customers	82467968608e325d7733	slicer
Top 10 Customers	6694c120e79260f46625	slicer
Top 10 Customers	a01327de3c0253df0c8f	slicer
Top 10 Customers	320bebd728720e8d03da	columnChart
Customer Details	95a315274a6d1020459b	actionButton
Customer Details	2337408638d07ca66776	card
Customer Details	74819e9c559df9a0bf96	card
Customer Details	61fc0964d28ac938a381	card
Customer Details	64be7db5ca9d8d02ddc3	card
Customer Details	6e8342ad3e46caad0f7b	card
Customer Details	f03a1735c63f53395bd0	tableEx
Customer Details	a846a0e8c67e9e54b583	map

Export Visual Table

Embedding a Single Visual

```
var pageName = "ReportSection432ffded90e56d5553a1";
var mapVisualName = "a846946d789b47a1480c";

var reportHeight = $(window).height() - 24;
var reportWidth = $(window).width() - 12;

var config = {
  type: 'visual',
  id: embedReportId,
  pageName: pageName,
  visualName: mapVisualName,
  embedUrl: embedUrl,
  accessToken: accessToken,
  tokenType: models.TokenType.Embed,
  height: reportHeight,
  width: reportWidth
};

// 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);
```

- Limit this to one visual embed per page – it really embeds the whole report then shows one visual

Custom Report Page Layouts

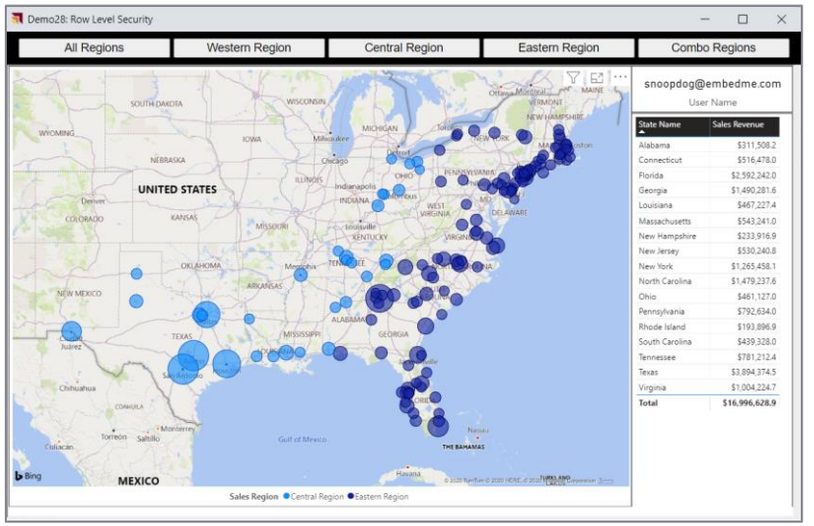
```
var defaultSetting = {
  layoutType: models.LayoutType.Master,
  panes: { filters: { visible: false } }
};

var mobileSetting = {
  layoutType: models.LayoutType.MobilePortrait,
  panes: { filters: { visible: false } }
};

var customLayout1Settings = {
  layoutType: models.LayoutType.Custom,
  panes: { filters: { visible: false } },
  customLayout: {
    pageSize: {
      type: models.PageSizeType.Custom,
      width: reportWidth,
      height: reportHeight
    },
    displayOption: models.DisplayOption.FitToPage,
    pagesLayout: {
      [pageName]: {
        visualsLayout: {
          [mapVisualName]: { x: 0, y: 0, width: reportWidth, height: reportHeight },
          [tableVisualName]: { displayState: { mode: models.VisualContainerDisplayMode.Hidden } }
        }
      }
    }
  }
};
```

Agenda

- ✓ Power BI Embedding 101
- ✓ Building Interactive Experiences
- ✓ Programming with Report Events
- ✓ Customizing Report Layouts
- ➢ Advanced Embedding Topics
- • SPA Architecture with MSAL.JS

Generating Embed Tokens with EffectiveIdentity

```
var userName = "snoopdog@embedme.com";

PowerBIClient pbiClient = TokenManager.GetPowerBiAppOnlyClient();

var report = pbiClient.Reports.GetReportInGroup(workspaceId, reportWithRlsId);
var datasetId = report.DatasetId;
var embedUrl = report.EmbedUrl;
var reportName = report.Name;

GenerateTokenRequest tokenRequest =
  new GenerateTokenRequest(accessLevel: "view",
                           identities: new List<EffectiveIdentity> {
                             new EffectiveIdentity(username: userName,
                                                   datasets: new List<string> { datasetWithRlsId.ToString() },
                                                   roles: new List<string> { "Western Region" })
                           });

string embedTokenAllData = pbiClient.Reports.GenerateTokenInGroup(workspaceId, reportWithRlsId, tokenRequest).Token;
```



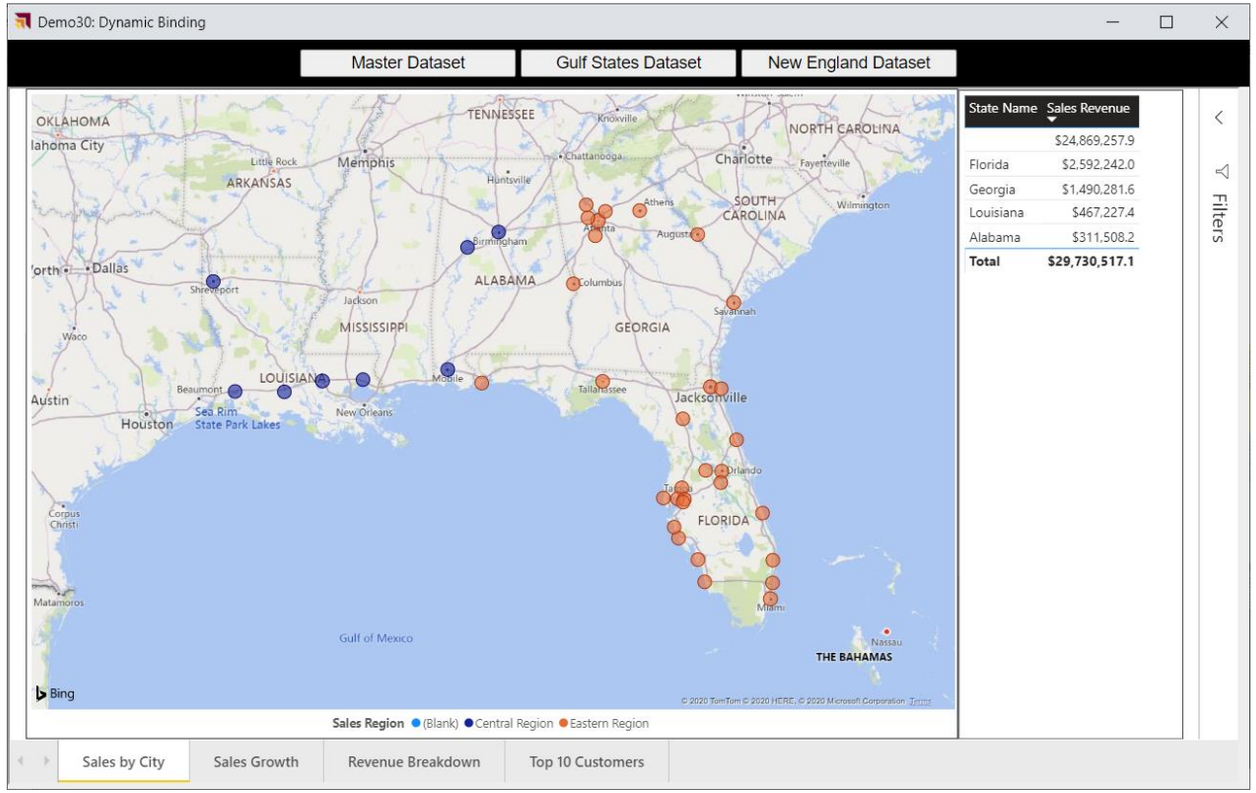
Auditing App-Owns-Data User with Correlation ID

- Problem: apps-owns-data does add username to PBI audit logs
- Solution: develop custom logging scheme with correlation ID

```
report.on("loaded", async () => {
  var correlationId = await report.getCorrelationId();
  // add teletry log enter for current user and correlation ID
  console.log("Audit user in app-owns-data embedding");
  console.log("- User: " + userName);
  console.log("- Correlation Id: " + correlationId);
});
```

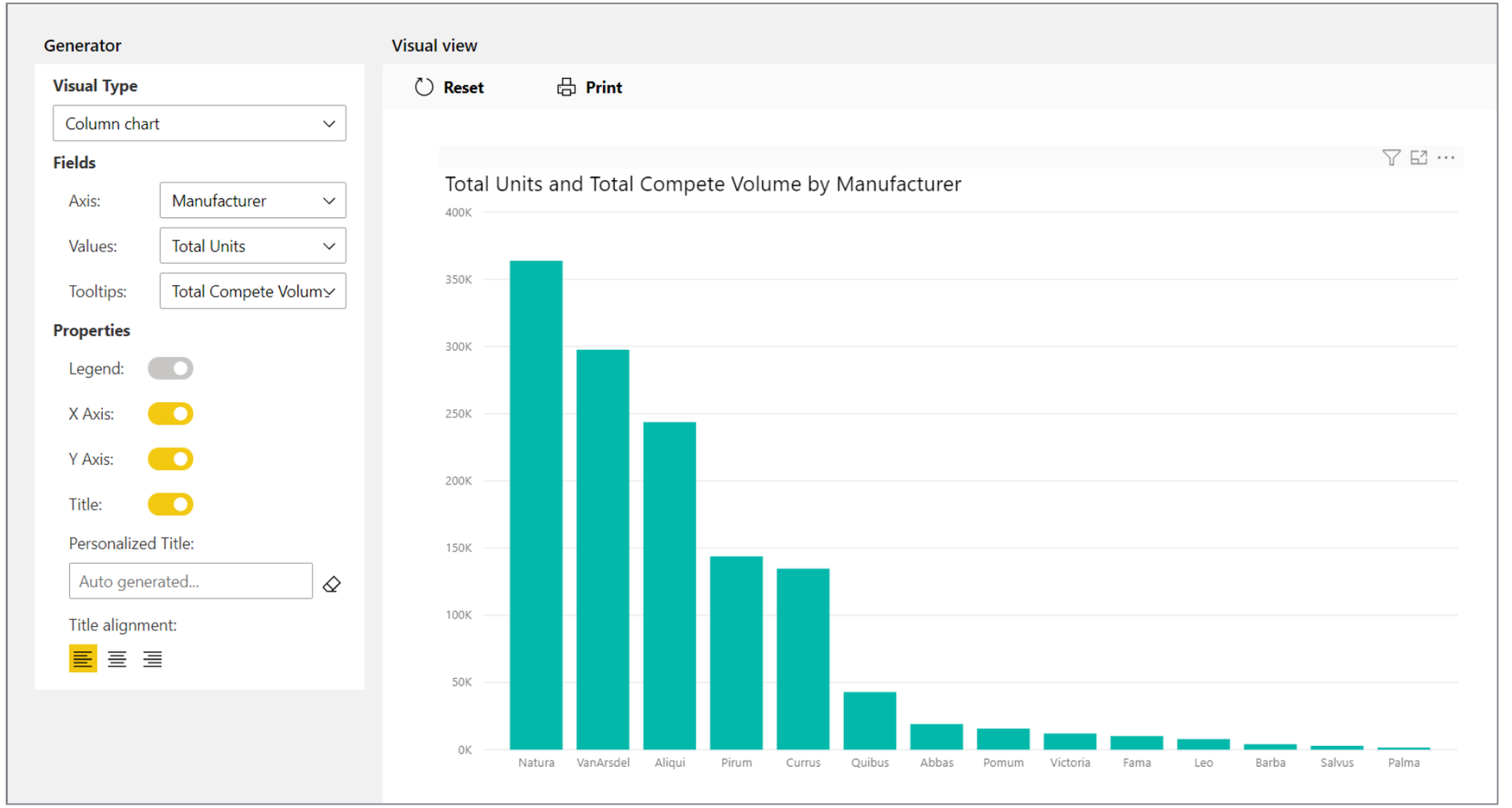
Dynamic Dataset Binding

```
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('reportContainer

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

$("#bindDataset1").click(function () {
  config.datasetBinding = { datasetId: datasetId };
  powerbi.reset(reportContainer);
  powerbi.embed(reportContainer, config);
});

$("#bindDataset2").click(function () {
  config.datasetBinding = { datasetId: dataset2Id };
  powerbi.reset(reportContainer);
  powerbi.embed(reportContainer, config);
});

$("#bindDataset3").click(function () {
  config.datasetBinding = { datasetId: dataset3Id };
  powerbi.reset(reportContainer);
  powerbi.embed(reportContainer, config);
});
```



Authoring Pages and Visuals using Code

https://github.com/microsoft/powerbi-report-authoring/wiki

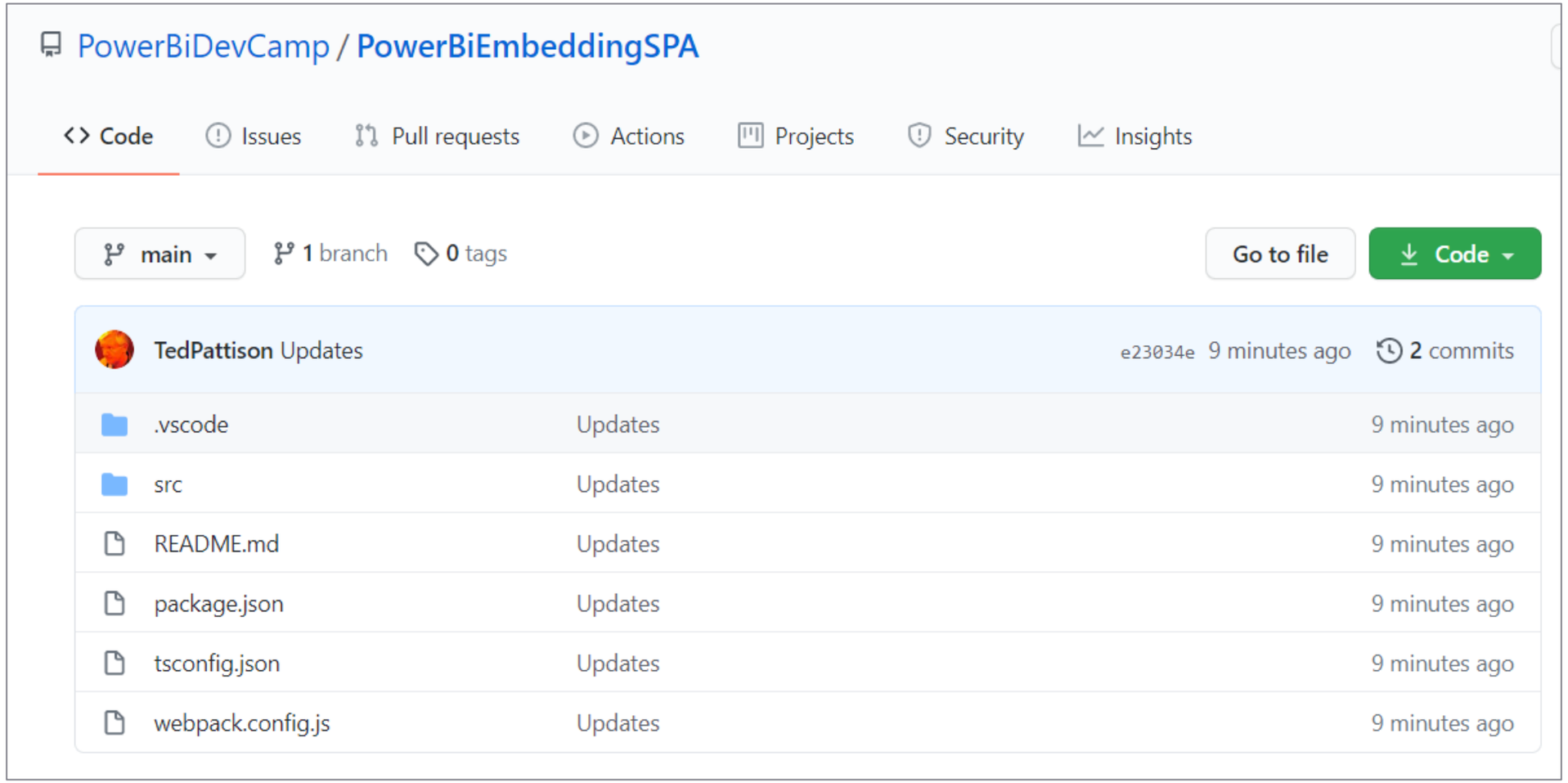


Agenda

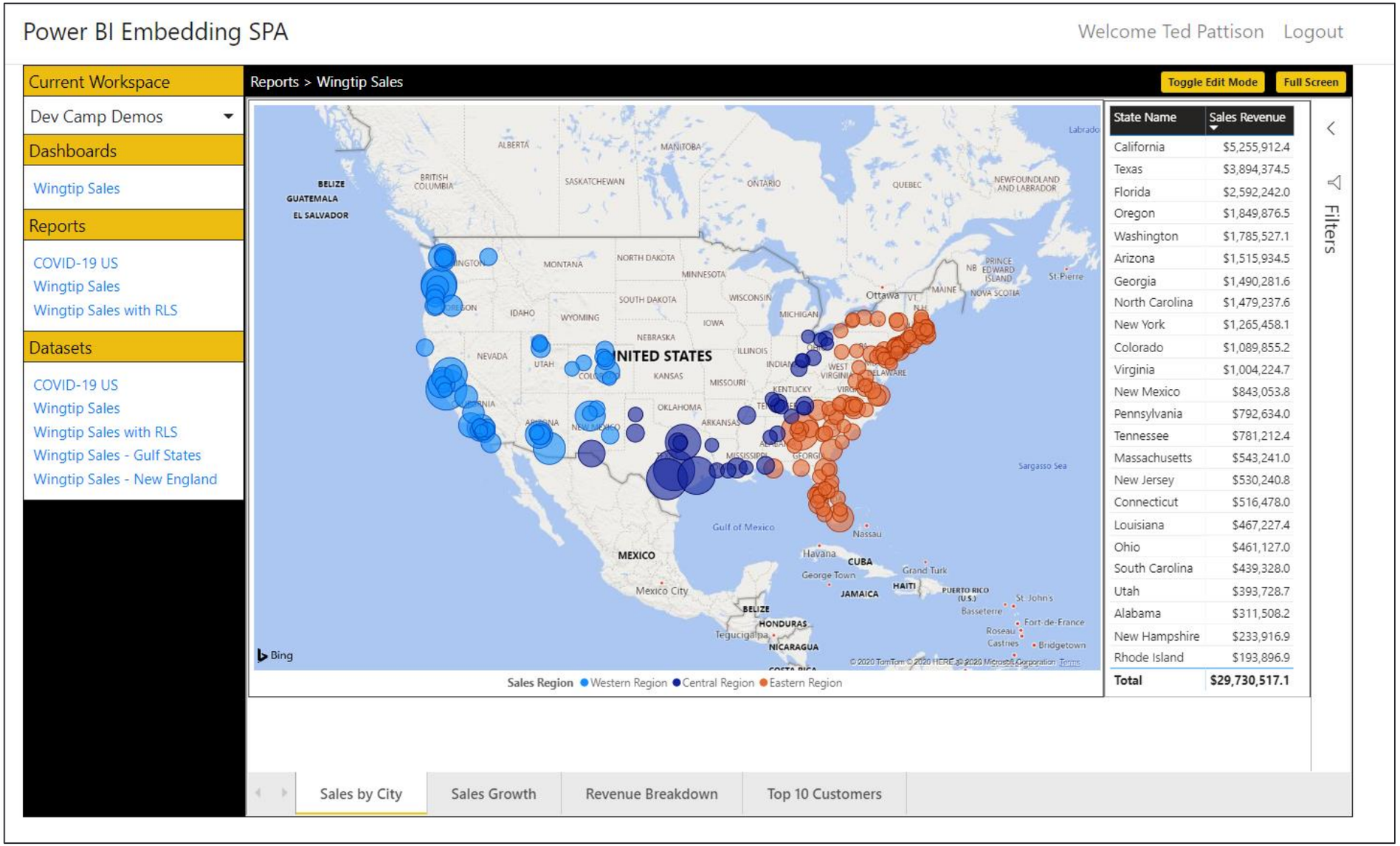
- ✓ Power BI Embedding 101
- ✓ Building Interactive Experiences
- ✓ Programming with Report Events
- ✓ Customizing Report Layouts
- ✓ Advanced Embedding Topics
- ➢ SPA Architecture with MSAL.JS

Sample Application - PowerBiEmbeddingSPA

https://github.com/PowerBiDevCamp/PowerBiEmbeddingSPA



Demo Time


Power BI Embedding SPA
Welcome Ted Pattison
Logout
Current Workspace
Dev Camp Demos
Dashboards
Wingtip Sales
Reports
COVID-19 US
Wingtip Sales
Wingtip Sales with RLS
Datasets
COVID-19 US
Wingtip Sales
Wingtip Sales with RLS
Wingtip Sales - Gulf States
Wingtip Sales - New England
Reports > Wingtip Sales
Toggle Edit Mode
Full Screen
Filters
State Name
Sales Revenue
California $5,255,912.4
Texas $3,894,374.5
Florida $2,592,242.0
Oregon $1,849,876.5
Washington $1,785,527.1
Arizona $1,515,934.5
Georgia $1,490,281.6
North Carolina $1,479,237.6
New York $1,265,458.1
Colorado $1,089,855.2
Virginia $1,004,224.7
New Mexico $843,053.8
Pennsylvania $792,634.0
Tennessee $781,212.4
Massachusetts $543,241.0
New Jersey $530,240.8
Connecticut $516,478.0
Louisiana $467,227.4
Ohio $461,127.0
South Carolina $439,328.0
Utah $393,728.7
Alabama $311,508.2
New Hampshire $233,916.9
Rhode Island $193,896.9
Total $29,730,517.1
Sales Region Western Region Central Region Eastern Region
Sales by City
Sales Growth
Revenue Breakdown
Top 10 Customers

Summary

- ✓ Power BI Embedding 101
- ✓ Building Interactive Experiences
- ✓ Programming with Report Events
- ✓ Customizing Report Layouts
- ✓ Advanced Embedding Topics
- ✓ SPA Architecture with MSAL.JS

Questions