

# Getting Started with the Power BI Developer Tools



# Agenda

- Custom Visuals in Power BI
- Node.JS and the Cross-platform Toolchain
- Creating Projects with the PBIVIZ CLI
- Custom Visual Project Structure
- Adding Typed Definition Files
- Testing and Debugging a Custom Visual



# Install the Power BI Developer Toolchain

- Install Node.JS
  - Installs Node Package Manager (npm)
- Install Power BI visuals CLI tool (pbiviz)
  - Install using Node Package Manager (npm)
- Install Local self-signed certificate
  - Install using Power BI visuals CLI tool (pbiviz)
- Install Visual Studio Code
  - Lightweight Alternative to Visual Studio for Node.js Development



# Installing node.js

- <https://nodejs.org/en/download/>

The image shows the Node.js website's 'Downloads' page and three overlapping windows of the Node.js Setup Wizard. The website header includes the Node.js logo and navigation links: HOME, ABOUT, DOWNLOADS, DOCS, FOUNDATION, GET INVOLVED, SECURITY, and NEWS. The 'Downloads' section highlights the 'Latest LTS Version: v6.9.1 (includes npm 3.10.8)' and offers a 'Windows Installer' for 'node-v6.9.1-x64.msi'. Below this, a list of download options is provided: Windows Installer (.msi), Windows Binary (.exe), macOS Installer (.pkg), macOS Binaries (.tar.gz), Linux Binaries (x86/x64), Linux Binaries (ARM), and Source Code. The three overlapping windows of the 'Node.js Setup' wizard show the following steps: 1. 'Welcome to the Node.js Setup Wizard', 2. 'Custom Setup' where users select features, and 3. 'Ready to install Node.js' with an 'Install' button highlighted.

# Install Visual Studio Code

- <http://code.visualstudio.com/>



# Power BI Visual CLI Tool (PBIVIZ)

- What is the Power BI Custom Visual Tool?
  - Command-line utility for cross-platform dev
  - Use it with Visual Studio or Visual Studio Code
  - Requires that you first install node.js
  - Install by running command from node.js command prompt  
**npm install -g powerbi-visuals-tools**

```
C:\> Node.js command prompt

c:\DemosPBIVIZ>npm install -g powerbi-visuals-tools
C:\Users\Student\AppData\Roaming\npm\pbiviz -> C:\Users\Student\AppData\Roaming\npm\
in\pbiviz.js
C:\Users\Student\AppData\Roaming\npm
|-- powerbi-visuals-tools@1.2.1
+-- chalk@1.1.3
|   +-- ansi-styles@2.2.1
|   +-- escape-string-regexp@1.0.5
|   +-- has-ansi@2.0.0
|   |   |-- ansi-regex@2.0.0
|   |   +-- strip-ansi@3.0.1
```





# Power BI Visual Tool Dependencies

```
gulp@3.9.1 node_modules\powerbi-visuals-tools\node_modules\gulp -> node_modules\powerbi-visuals-
- ms@0.7.2 node_modules\powerbi-visuals-tools\node_modules\send\node_modules\ms
C:\Users\Student\AppData\Roaming\npm
`-- powerbi-visuals-tools@1.5.0
  +-- connect@3.6.0
    | +-- debug@2.6.1
    | | `-- ms@0.7.2
    | `-- finalhandler@1.0.0
  +-- fs-extra@0.28.0
    | `-- rimraf@2.6.0
  +-- gulp-powerbi-package-validator@1.0.0
    | +-- eslint@3.16.1
    | | +-- concat-stream@1.6.0
    | | | `-- readable-stream@2.2.3
    | | `-- ignore@3.2.4
    | `-- gulp-stylelint@0.2.0
    |   +-- postcss@5.2.15
    |   `-- stylelint@4.5.1
    |     +-- autoprefixer@6.7.5
    |     | +-- browserslist@1.7.5
    |     | | `-- electron-to-chromium@1.2.3
    |     | `-- caniuse-db@1.0.30000626
    |     `-- stylehacks@2.3.2
    |   `-- postcss-selector-parser@2.2.3
  `-- serve-static@1.11.2
    `-- send@0.14.2
      `-- debug@2.2.0
        `-- ms@0.7.1
```

# Getting Started with PBIVIZ

```
Usage: pbiviz [options] [command]
```

## Commands:

new [name]	Create a new visual
info	Display info about the current visual
start	Start the current visual
package	Package the current visual into a pbiviz file
update [version]	Updates the api definitions and schemas in the current visual. Changes the version if specified
help [cmd]	display help for [cmd]

## Options:

-h, --help	output usage information
-V, --version	output the version number
--install-cert	Install localhost certificate

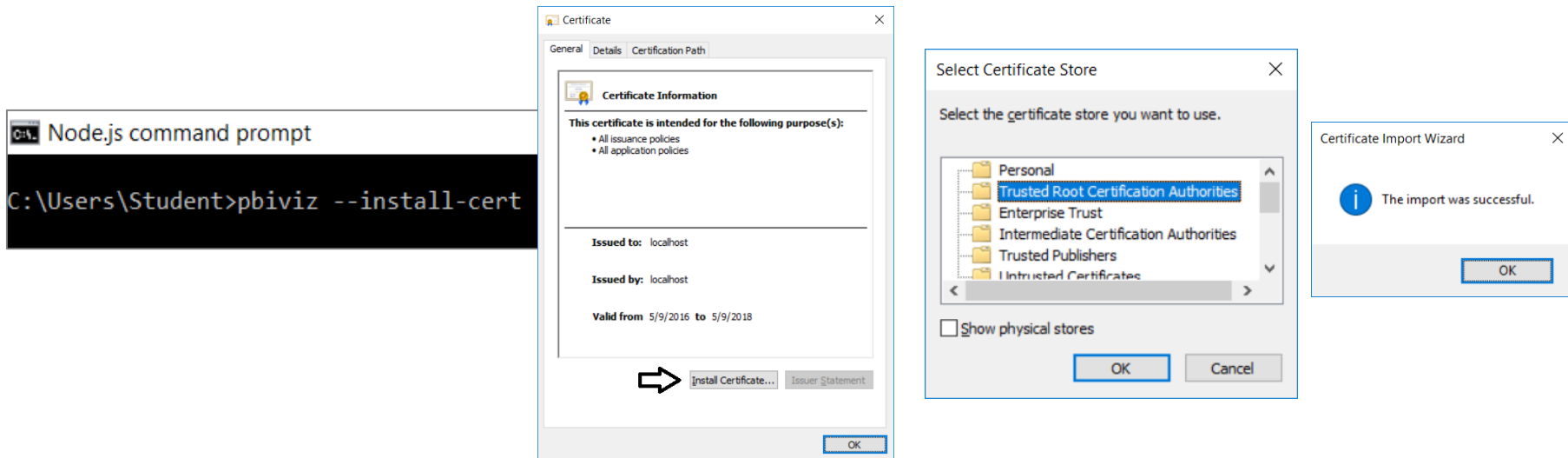
```
c:\Demo\PBI\simplebarchart>_
```





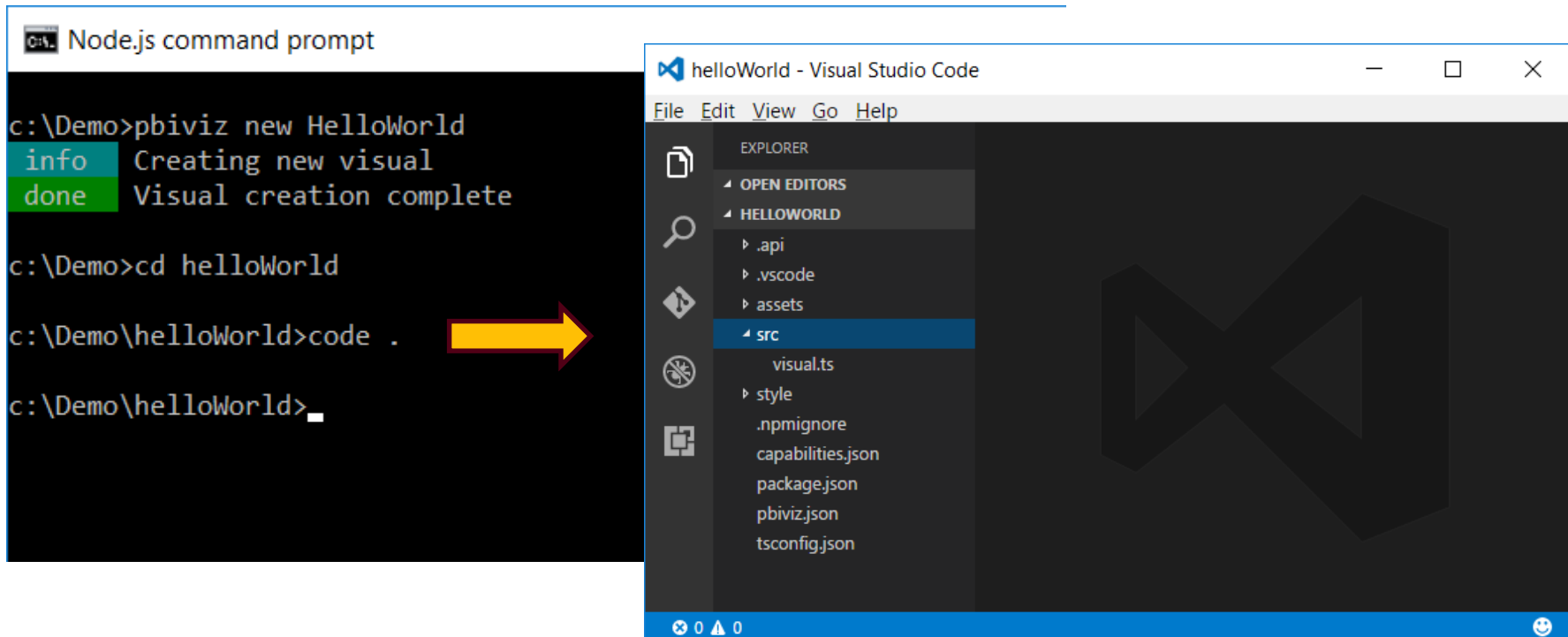
# Installing the Developer Certificate

- Debugging visuals inside PowerBI.com requires SSL
  - PBIVIZ leverages Node.js to provide debugging experience
  - Node.js acts as web service to serve project files through HTTP
  - Node.js debugging session uses <http://localhost> address
  - Installing certificate enables SSL through <https://localhost>
  - Installing certificate is a one time operation – not once per project



# Creating a New Custom Visual Project

- Creating a new project  
`pbiviz new <ProjectName>`
- Open the Project with Visual Studio Code  
`code .`

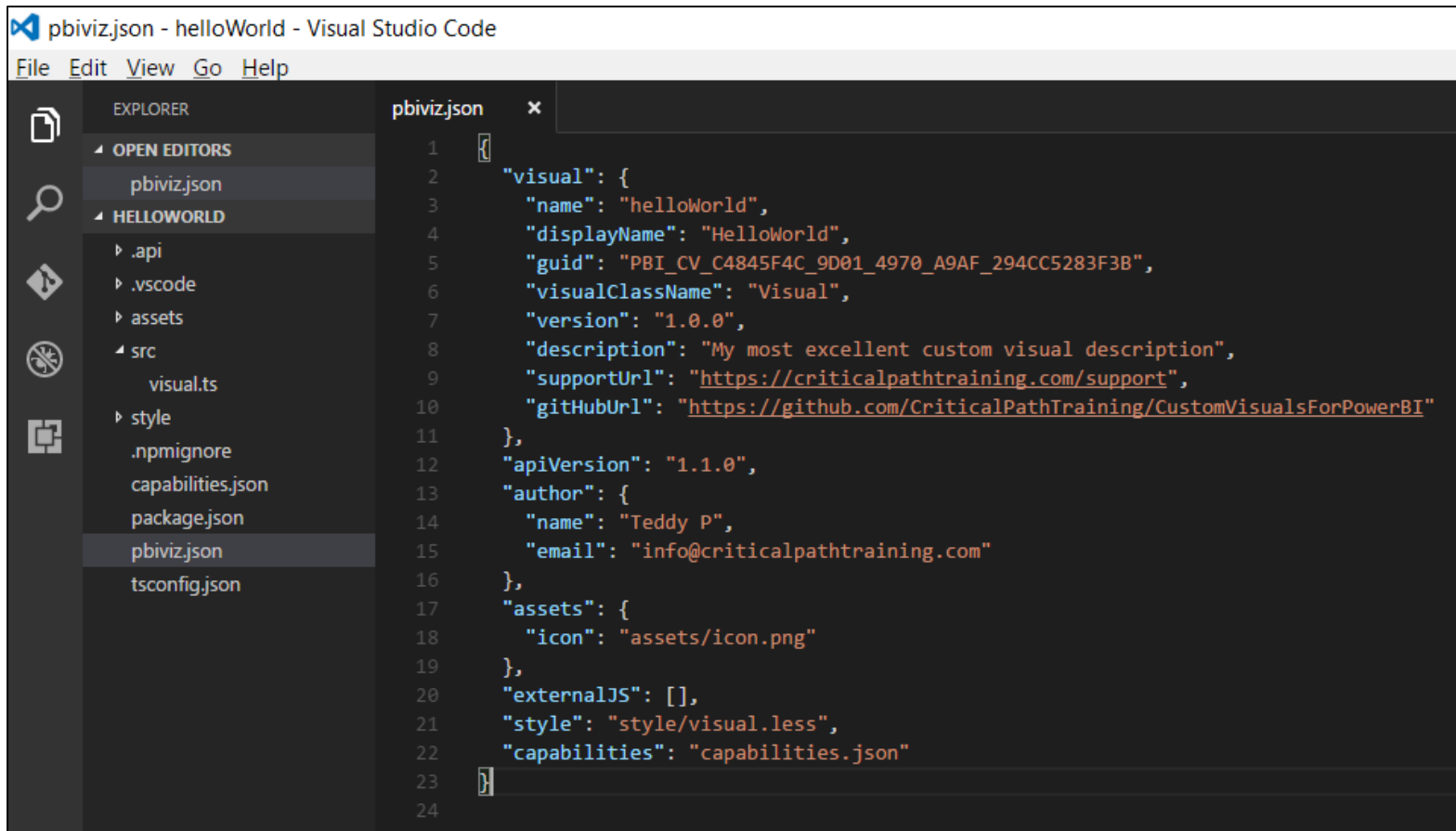


# Files in the new project

- `gitignore`
  - tells git to ignore files that shouldn't be tracked in the repository
- `capabilities.json`
  - used to define the capabilities of your visual learn more about visual capabilities
- `package.json`
  - Used by npm to manage modules learn more about npm
- `pbviz.json`
  - Main configuration file for your visual
- `tsconfig.json`
  - Typescript compiler settings learn more about tsconfig



# The pbiviz.json File



The screenshot shows the Visual Studio Code interface with the file `pbiviz.json` open. The Explorer sidebar on the left shows the project structure, including the `pbiviz.json` file. The main editor area displays the JSON content of the file, which defines a custom visual for Power BI. The JSON includes fields for name, displayName, guid, className, version, description, supportUrl, githubUrl, apiVersion, author, assets, externalJS, style, and capabilities.

```
1 {  
2   "visual": {  
3     "name": "helloWorld",  
4     "displayName": "HelloWorld",  
5     "guid": "PBI_CV_C4845F4C_9D01_4970_A9AF_294CC5283F3B",  
6     "visualClassName": "Visual",  
7     "version": "1.0.0",  
8     "description": "My most excellent custom visual description",  
9     "supportUrl": "https://criticalpathtraining.com/support",  
10    "githubUrl": "https://github.com/CriticalPathTraining/CustomVisualsForPowerBI"  
11  },  
12  "apiVersion": "1.1.0",  
13  "author": {  
14    "name": "Teddy P",  
15    "email": "info@criticalpathtraining.com"  
16  },  
17  "assets": {  
18    "icon": "assets/icon.png"  
19  },  
20  "externalJS": [],  
21  "style": "style/visual.less",  
22  "capabilities": "capabilities.json"  
23 }  
24
```



# Folders in the new project

- assets/
  - Used to store visual assets (icon, screenshots, etc)
- dist/
  - when you run pbiviz package the pbiviz file will be generated here
- src/
  - Typescript code for your visual goes here
- style/
  - Less styles for your visual go here



# Installing Typed Definitions for D3

- Two choices for installing typed definitions files
  - Using the **typings** utility *(the original way)*
  - Using the **npm** utility *(the new and better way)*
- Installing typed definitions files using typings utility  
**npm install typings -g**  
**typings install --save --global dt~d3#0.0.0+20160907005744**
- Installing typed definitions files using typings utility  
**npm install @types/d3@3 --save-dev**



# The tsconfig.json File

- Used to add references to typings files
  - This is what enables Intellisense

```
tsconfig.json
1 {
2   "compilerOptions": {
3     "allowJs": true,
4     "emitDecoratorMetadata": true,
5     "experimentalDecorators": true,
6     "target": "ES5",
7     "sourceMap": true,
8     "out": "./.tmp/build/visual.js"
9   },
10  "files": [
11    ".api/v1.1.0/PowerBI-visuals.d.ts",
12    "typings/index.d.ts",
13    "src/visual.ts"
14  ]
15 }
```

Typings file reference





# Developing a Custom Visual?

- Create a class that implements IVisual
  - Class wrapped in module with namespace to APIs
  - You code can program again PBI APIs types

```
module powerbi.extensibility.visual {  
  
  export class Visual implements IVisual {  
  
    constructor(options: VisualConstructorOptions) {  
      // one-time initializaion code  
    }  
  
    public update(options: VisualUpdateOptions) {  
      // called when viewport or data changes  
    }  
  
    public destroy(): void {  
      // add cleanup code here  
    }  
  }  
}
```



# Running a Custom Visual Project

```
Node.js command prompt - pbiviz start

c:\Demo\PBI>pbiviz new simplebarchart
info  Creating new visual
done  Visual creation complete

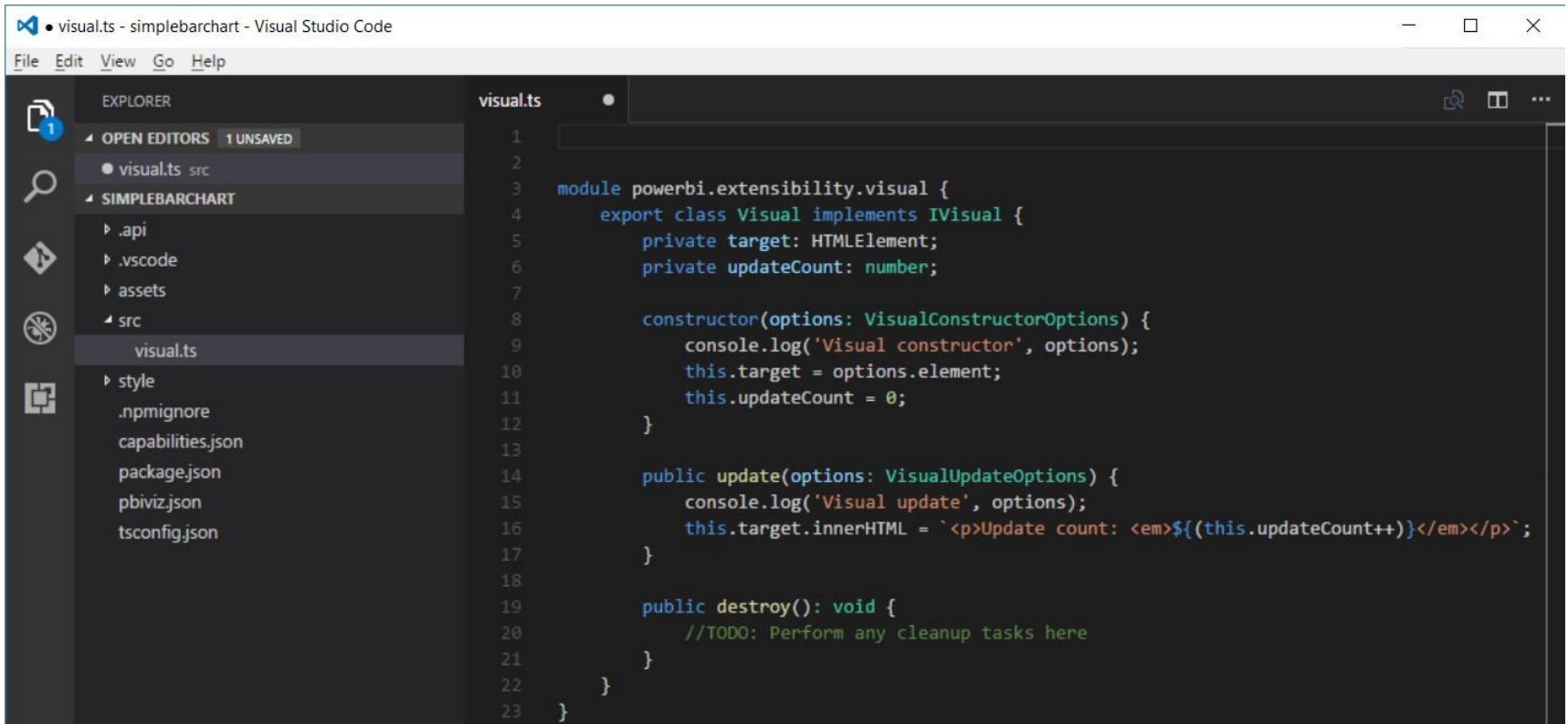
c:\Demo\PBI>cd simplebarchart

c:\Demo\PBI\simplebarchart>pbiviz start
info  Building visual...
done  build complete

info  Starting server...
info  Server listening on port 8080.
```



# Developing with Visual Studio Code



```
1  
2  
3 module powerbi.extensibility.visual {  
4   export class Visual implements IVisual {  
5     private target: HTMLElement;  
6     private updateCount: number;  
7  
8     constructor(options: VisualConstructorOptions) {  
9       console.log('Visual constructor', options);  
10      this.target = options.element;  
11      this.updateCount = 0;  
12    }  
13  
14    public update(options: VisualUpdateOptions) {  
15      console.log('Visual update', options);  
16      this.target.innerHTML = `

Update count: <em>${(this.updateCount++)}</em></p>`;  
17    }  
18  
19    public destroy(): void {  
20      //TODO: Perform any cleanup tasks here  
21    }  
22  }  
23 }


```



# Summary

- ✓ Custom Visuals in Power BI
- ✓ Node.JS and the Cross-platform Toolchain
- ✓ Creating Projects with the PBIVIZ CLI
- ✓ Custom Visual Project Structure
- ✓ Adding Typed Definition Files
- ✓ Testing and Debugging a Custom Visual

