ArcGIS API for JavaScript: Customizing Widgets

JC Franco – @arfncode

Matt Driscoll – @driskull

2021 ESRI
DEVELOPER SUMMIT

Agenda

- What can be customized
- Customization approaches with demos

Customization Approaches

Customization Approaches

- Authoring a theme (Sass)
  - Changing styles: colors, sizing, font, etc.

Customization Approaches

- Authoring a theme (Sass)
  - Changing styles: colors, sizing, font, etc.
- Extending a view (TypeScript)
  - Altering presentation
  - Adding functionality


Part I
Theming

Part I: Theming

Part I: Theming

Why Theme?

Part I: Theming

Why Theme?

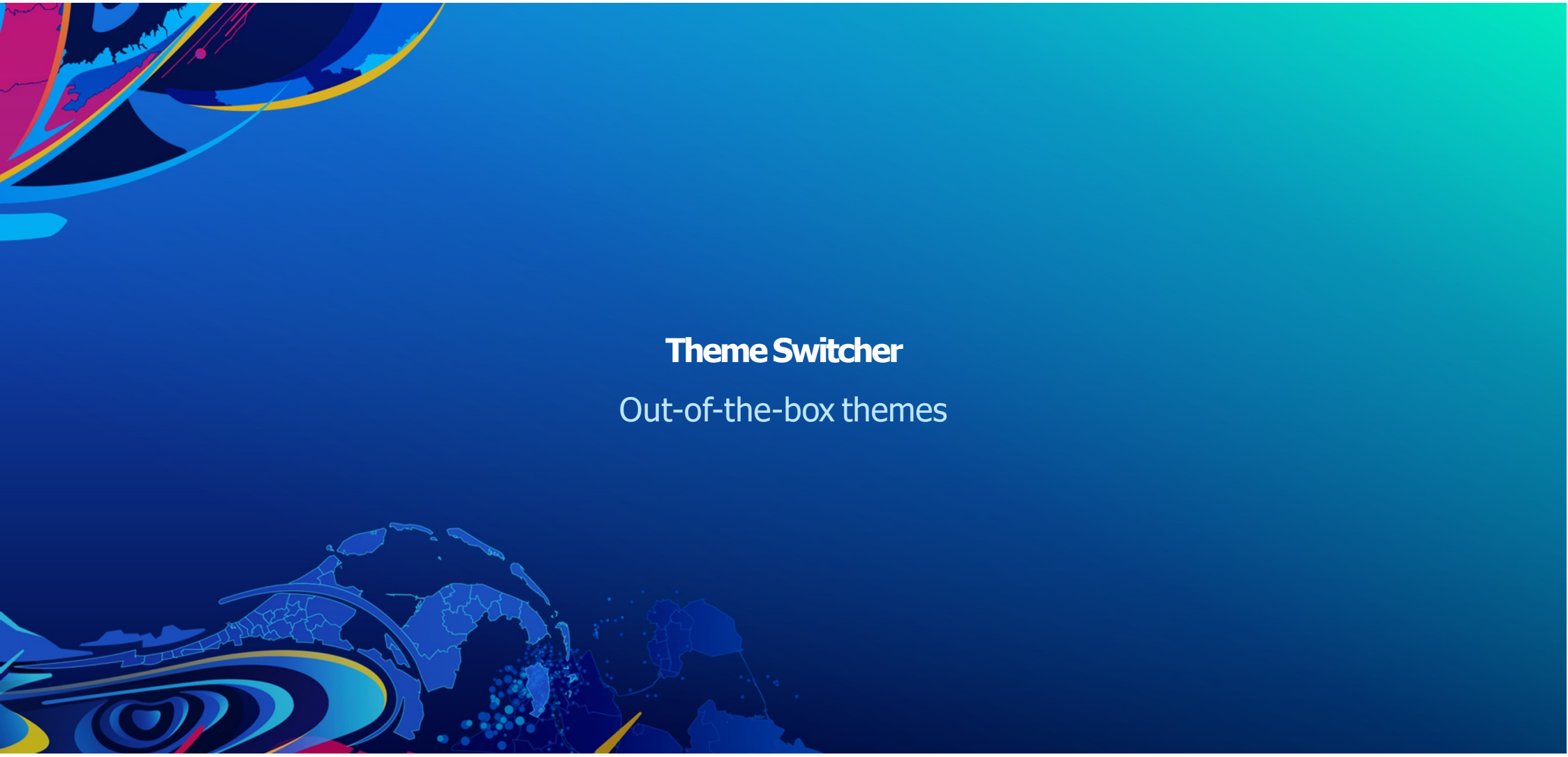
- Match branding.
- Contrast with the map.
- User-speci[illegible] (e.g. bigger buttons).

Esri Themes

10 themes are provided **out-of-the-box**:

Using a theme requires only a slight update to the CSS path.

```
<link rel="stylesheet" href="https://js.arcgis.com/<version>/esri/themes/<theme-name>/main.css" />
```


Theme Switcher
Out-of-the-box themes

Theming Technology

We use

to create our CSS.

We use

to create our CSS.

nodejs.org | gruntjs.com

is a powerful scripting language for compiling CSS.

is a powerful scripting language for compiling CSS.

- It's modular.
- It's DRY.
- It makes theming easy.

Theming Steps

Theming Steps

1. Install the `@arcgis/cli` NPM package
2. Use `styles` commands to
    1. Create
    2. Preview
    3. Customize
3. Host and reference your custom theme.

Step 1

Install `@arcgis/cli`

```
npm install -g @arcgis/cli
```

Step 2

Create a theme

```
arcgis-cli styles create <theme>
```

Run a preview

Step 2

Create a theme

```
arcgis-cli styles create <theme>
```

Run a preview

```
arcgis-cli styles preview <theme>
```

Step 3

Edit your theme
`<theme>/main.scss`

Step 4

Eject and host/reference your custom theme.

```
<!-- In your app: -->
<link href="path/to/your/<theme>/main.css" rel="stylesheet" />
```

Goals

Theme Smart

- Avoid adding additional CSS selectors
- Instead, use Sass to your advantage

Theme Structure

Let's look at how the core theme is structured

- Color
- Size
- Type

Theme Structure

Let's look at how the core theme is structured

- Color : `_color.scss`
- Size : `_sizes.scss`
- Type : `_type.scss`

Theme Structure

Default

```
// Inside base/_color_.scss
$background-color: #fff !default;
```

Any value assignment overrides the `!default` value.

```
// Inside my-theme/main.scss
$background-color: #116EBF;
```

Theme Structure

Default

```
// Inside base/_color_.scss
$background-color: #fff !default;
```

Any value assignment overrides the `!default` value.

```
// Inside my-theme/main.scss
$background-color: #116EBF;
```

But wait...there's more!

Theme Structure

Override the core color variables...

```
$font-color: #0442BF;
$interactive-font-color: #F20530;
$background-color: #116EBF;
$button-color: #F2B705;
```

Theme Structure

Override the core color variables…

```
$font-color: #0442BF;
$interactive-font-color: #F20530;
$background-color: #116EBF;
$button-color: #F2B705;
```

…then magic!

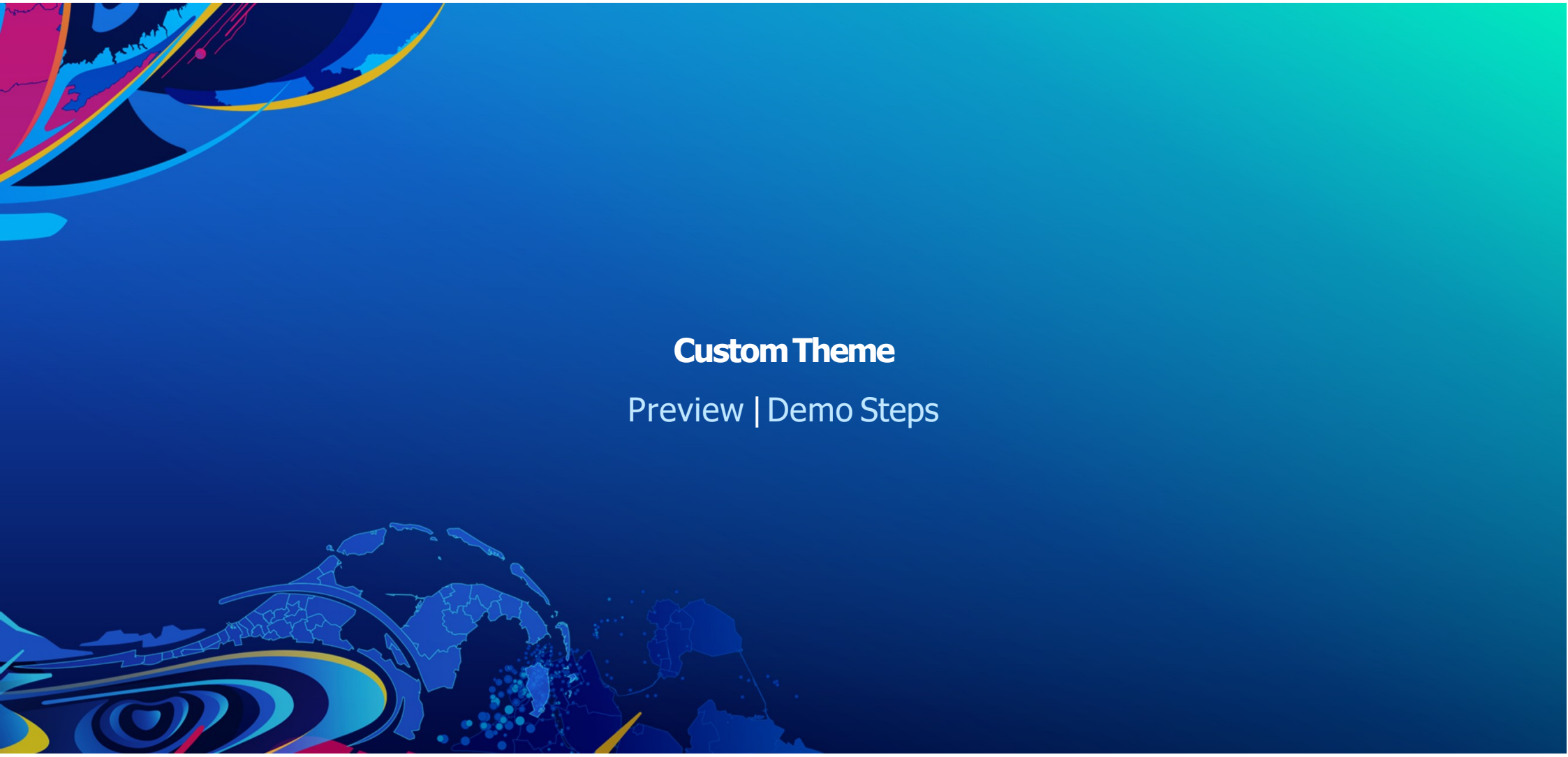
Magic

Using `$button-color` we "automagically" set the hover color.

```
$button-color--hover: darken($button-color, 10%) !default;
// ...etc
```

API Styling Guide

Part I: Let's make a theme


Custom Theme
Preview | Demo Steps

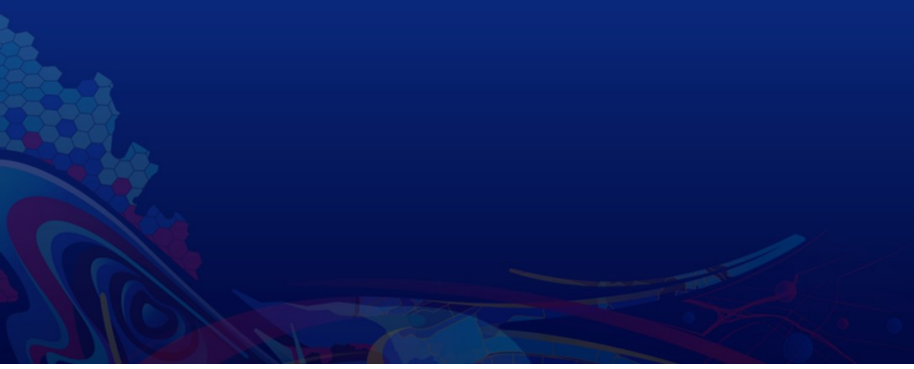
Part I: Theming Recap

- Used `@arcgis/cli` for easy theming
- Saw theme structure
  - Color
  - Size
  - Typography
- Used the core and override values.


Part II
Views

Part II: Widget Composition

Widgets are composed of Views & ViewModels



Part II: Widget Composition

Widgets are composed of Views & ViewModels

- Logic is separate from presentation
- Reusable
- UI replacement
- Framework integration

Part II: TypeScript

- Widgets written in TypeScript (Typed JavaScript)

Part II: TypeScript

- Widgets written in TypeScript (Typed JavaScript)
- JS of the future, now

Part II: TypeScript

- Widgets written in TypeScript (Typed JavaScript)
- JS of the future, now
- IDE support
  - Visual Studio
  - WebStorm
  - Sublime
  - and more!

Part II: Views

Part II: Views

- Presentation of the Widget
  - DOM structure
- Use ViewModel APIs to render the UI
- View-speci[?]c logic resides here
- Extend `esri/widgets/Widget`

Part II: Widget Class

`esri/widgets/Widget`

Part II: Widget Class

`esri/widgets/Widget`

- Provides lifecycle
- API consistency

Part II: Widget Lifecycle

Part II: Widget Lifecycle

- `constructor()`

Part II: Widget Lifecycle

- `constructor()`
- `postInitialize()`

Part II: Widget Lifecycle

- `constructor()`
- `postInitialize()`
- `render()`

Part II: Widget Lifecycle

- `constructor()`
- `postInitialize()`
- `render()`
  - `when()` - after first render

Part II: Widget Lifecycle

- `constructor()`
- `postInitialize()`
- `render()`
  - `when()` - after first render
- `destroy()`

Part II: `render()`

Part II: `render()`

- De  nes UI
- Reacts to state changes
- Uses JSX (VDOM)
- Re-renders on prop updates

Part II: Working with Views

API Exploration

BasemapToggle Doc

Part II: Views Recap

What have we learned about Widget Views?

Part II: Views Recap

What have we learned about Widget Views?

- Face of the widget
- Present ViewModel logic
- ViewModel separation
  - Allows framework integration/custom views
- Downloadable on API docs

Part III: Extending a View

Part III: Extending a View

- Why?

Part III: Extending a View

- Why?
  - Reusable

Part III: Extending a View

- Why?
  - Reusable
  - Same ecosystem

Part III: Extending a View

- Why?
  - Reusable
  - Same ecosystem
- How?

Part III: Extending a View

- Why?
  - Reusable
  - Same ecosystem
- How?
  - Leveraging `esri/widgets/Widget`

Part III: Extending a View

- Why?
  - Reusable
  - Same ecosystem
- How?
  - Leveraging `esri/widgets/Widget`
  - API widget views

Part III: Lets extend a View (presentation)

- View custom LayerFX widget
- Extend the LayerFX widget view
  - Use Esri's design system web components
  - Alter the presentation of the widget

Esri's design system*

*Evolved from Esri's calcite style and currently in beta

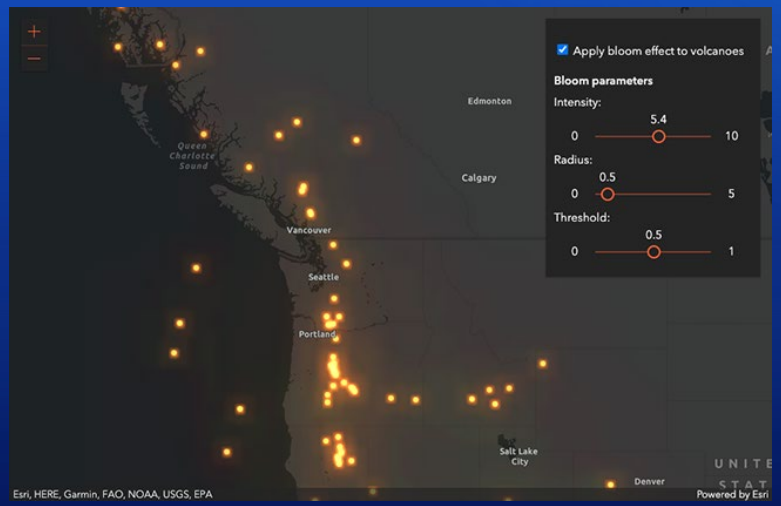
- Create beautiful, consistent apps faster
- Web components
  - Custom, reusable, encapsulated HTML tags
  - Quickly build Esri branded, lightweight, and accessible web apps

CalciteButton

```
<calcite-button>My Button!</calcite-button>
```

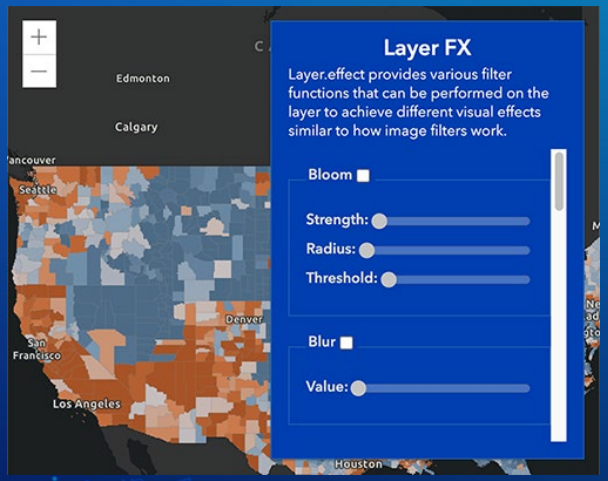
Part III: LayerFX Widget

Inspired by Intro to layer effect sample



View LayerFX widget

Demo Start



Part III: LayerFX Interface

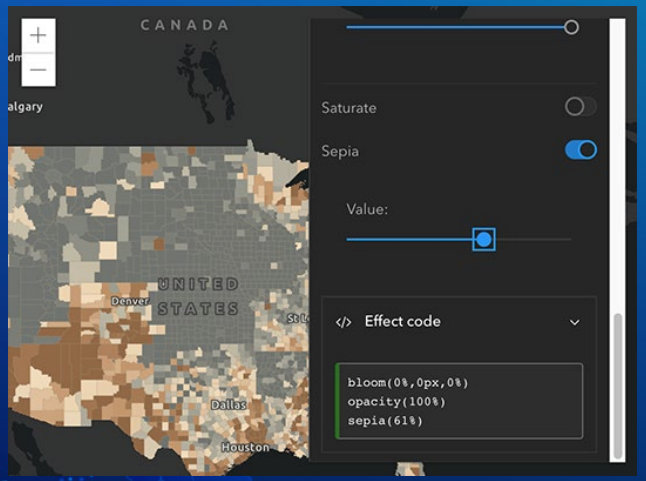
```
interface LayerFX extends Accessor {
  layer: Layer;
  readonly effects: Collection<LayerEffect>;
  readonly statements: string;
}

interface LayerEffect {
  enabled: boolean;
  id: "bloom" | "blur" | ... | "sepia";
  values: number[];
  readonly valueTypes: { unit: string; ... }[];
  readonly statement: string;
}
```

ArcGIS API for JavaScript: Building Your Own Widget

Thursday 8 April 2021 @ 10:15 a.m.

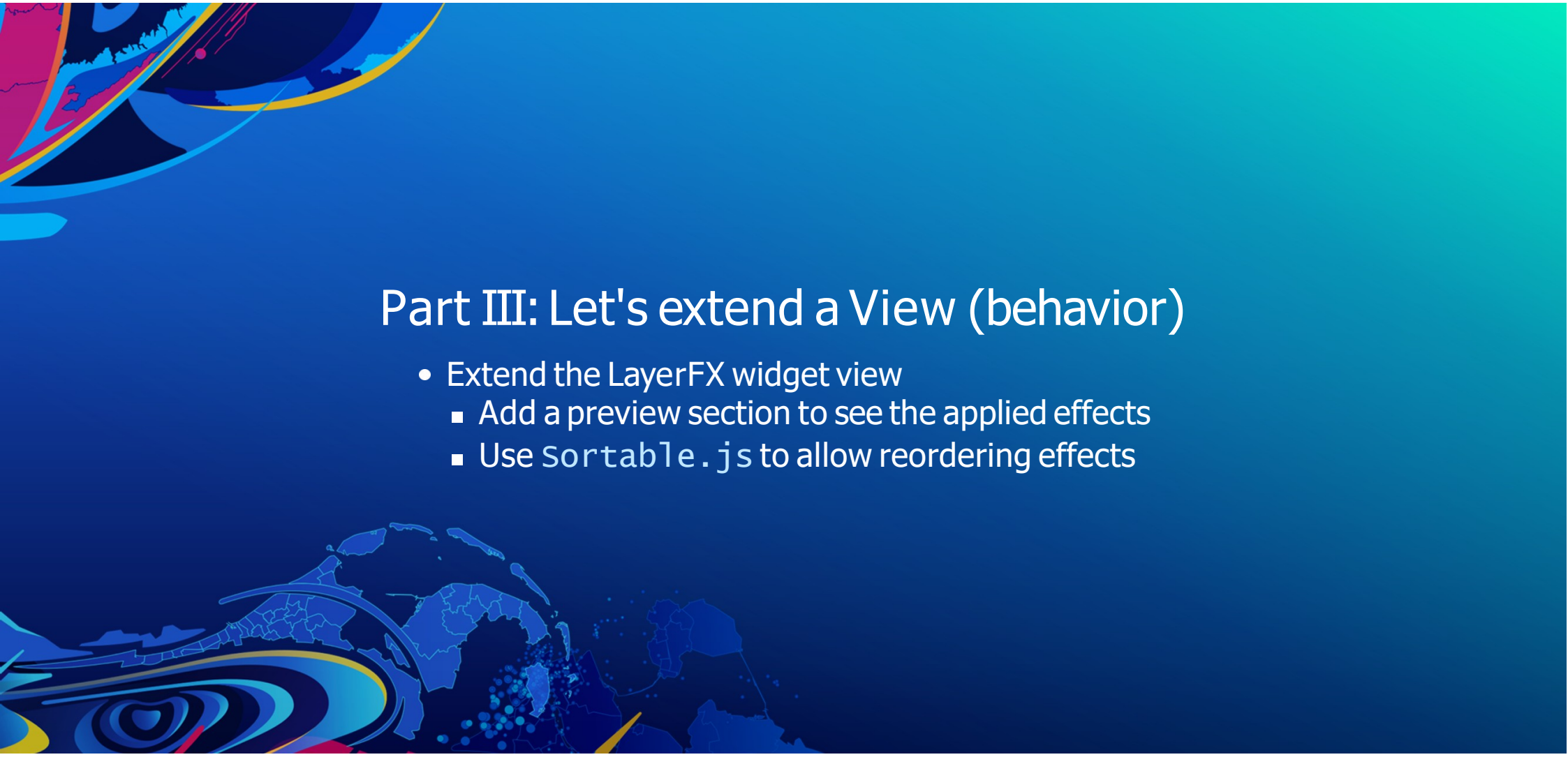
Extend LayerFX widget

- Demo Start
- Demo Steps



Part III: Extending View (presentation) Recap

- Extended an existing widget view
- Replaced render methods
- Altered presentation using Esri's design system: components
- Updated private methods where necessary



Part III: Let's extend a View (behavior)

- Extend the LayerFX widget view
  - Add a preview section to see the applied effects
  - Use `Sortable.js` to allow reordering effects

Part III: Extending View (behavior) Recap

- Extended an existing widget view
- Updated render methods
- Added a code preview block
- Used `Sortable.js` to allow custom drag behavior

Conclusion

Conclusion

- Authored a theme

Conclusion

- Authored a theme
- Extended a view

Conclusion

- Authored a theme
- Extended a view
  - Customized presentation

Conclusion

- Authored a theme
- Extended a view
  - Customized presentation
  - Added functionality

Additional Resources

- Implementing Accessor
- Setting up TypeScript
- Widget Development
- Styling
- ArcGIS API for JavaScript - next

You might also be interested in...

- ArcGIS API for JavaScript: Building Your Own Widget
- Esri Design System: Build compelling web apps faster using the new web component library
- ArcGIS API for JavaScript: Getting Started with Web Development
- ArcGIS API for JavaScript: Programming Patterns and API Fundamentals
- Accessible Web Mapping Apps: ARIA, WCAG and 508 Compliance

Questions?

Where can I find the slides/source?

bit.ly/customwidgetsds21

Where can I submit a question?

bit.ly/askjsapi


esri
THE SCIENCE OF WHERE