

An Introduction to ArcGIS Experience Builder

David Cardella,
Julian Kissling

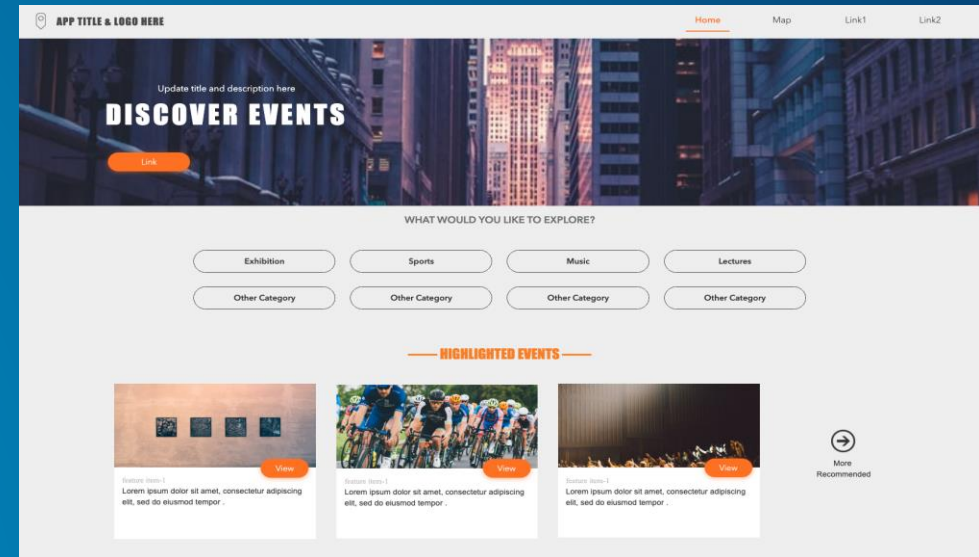
ESRI EUROPEAN DEVELOPER SUMMIT



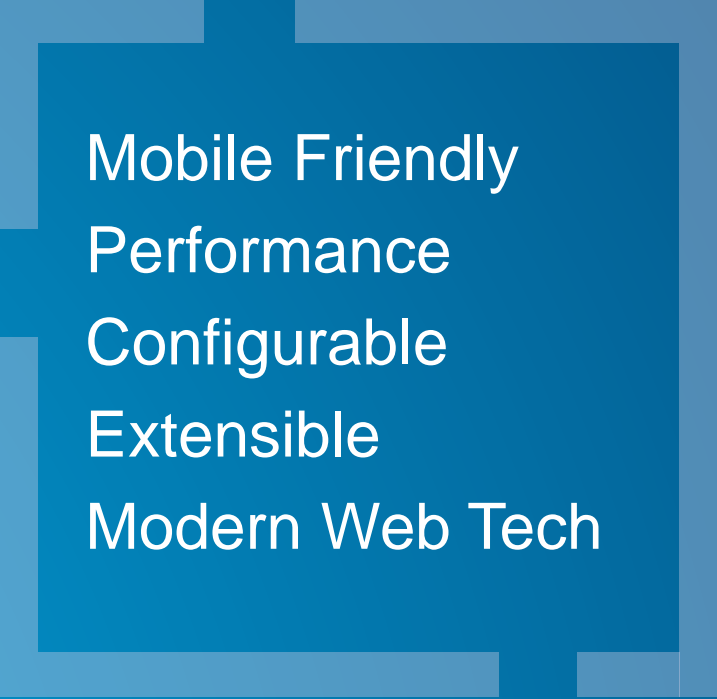
What is Experience Builder



- Enables you to create 2D and 3D web experiences
- An “experience” is a web app or web page

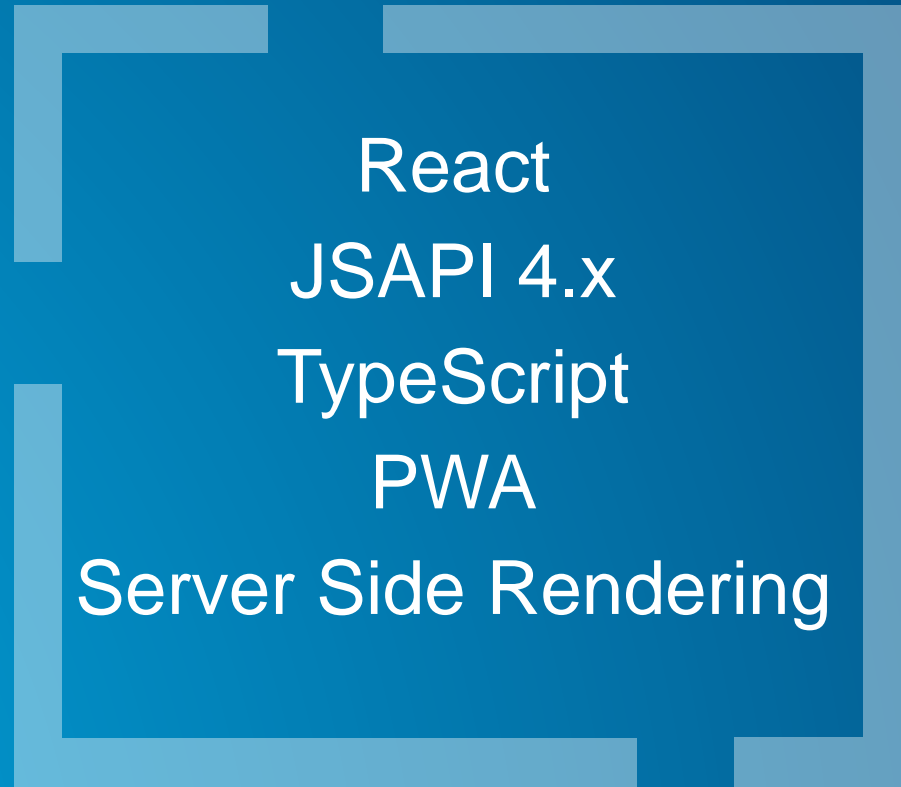


Goals



- Mobile Friendly
- Performance
- Configurable
- Extensible
- Modern Web Tech

Modern Web Technology



Key Features

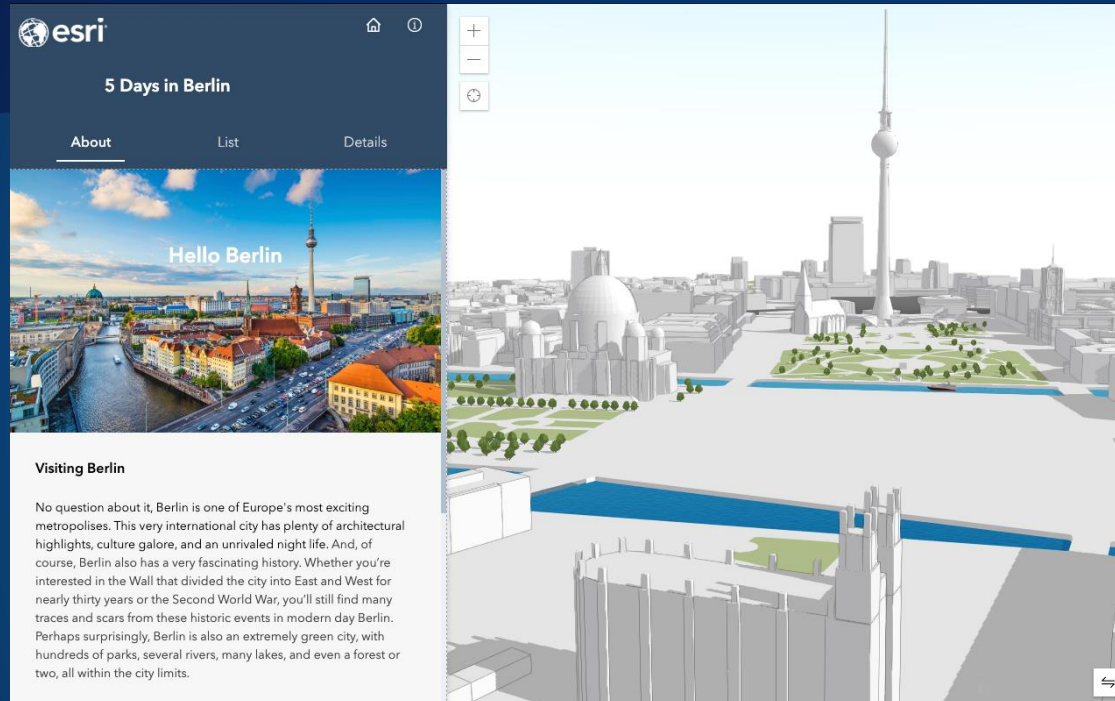
- GUI builder to configure web apps and web pages
- Extensible framework for creating widgets and templates
- Extensible framework for different development roles
 - Developers
 - Designer
 - “Builders”

Key Features (Continued)

- Mobile friendly- The same web app or web page on mobile can be configured separately from desktop
- Supports Web Map/Web Scene and multiple maps
- Themes define the Typography and color
- Drag and drop functionality of widgets

Key Features (Continued)

- Design any template you want with easy drag and drop and share with others
 - Interact with 3D and 2D contents in one app
 - Unified experience
- Performance optimization
 - Extensible framework for developers to create custom solution more rapidly



Demo: Create an Experience

Developer tools: Under the Hood

Things you will need to know

- Jimu
- React
- TypeScript

Jimu



- What is Jimu? A JavaScript library used in Experience Builder
- Packages:
 - jimu-core – loads and parses the app config, loads the layout, and widgets.
 - Defines several classes such as WidgetManager, ConfigManager and BaseWidget etc.
 - jimu-layouts – includes common implementations for layout widgets
 - jimu-arcgis – contains components on ArcGIS JavaScript API
 - jimu-ui – includes all UI components for your experience
 - jimu-for-builder – used for developing the widget setting page

React



- What is React? A JavaScript library for creating user interfaces
- The “V” in MVC
- React is an abstraction away from the DOM
- Components === State Machines
 - UI in terms of state, rather than explicit UI manipulations
- Components === Functions
 - $UI = f(\text{state})$

TypeScript



- What is TypeScript? Typed superset of JavaScript that compiles to plain JavaScript
- TypeScript + Jimu + React =  widgets

```
import { React, IMState, jimuHistory, DataSourceComponent } from 'jimu-core';

import { BaseWidget, AllWidgetProps, DataSourceManager, IMDataSourceJson } from 'jimu-core';
import { MapViewDataSource } from 'jimu-arcgis/arcgis-data-source';
import MapView = require('esri/views/MapView');
import WebMap = require('esri/WebMap');
import Extent = require('esri/geometry/Extent');
import { MapViewDataSourceConstructorOptions } from 'jimu-arcgis/lib/data-sources/map-view-data-source';

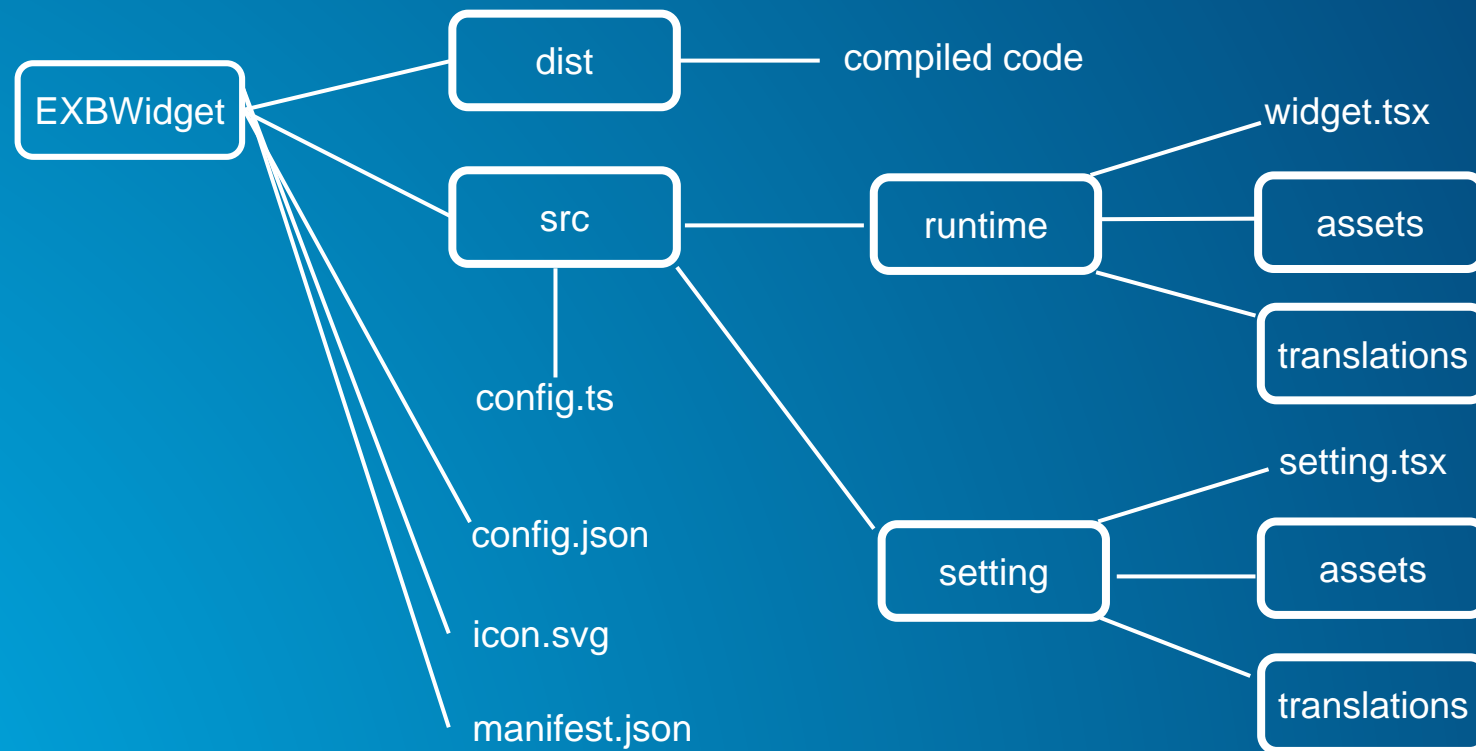
MapViewDataSource;

interface Config{
  webmapId: string;
}
interface ExtraProps{
  outputDataSourceJson: IMDataSourceJson
}

export default class Widget extends BaseWidget<AllWidgetProps<Config> & ExtraProps, {}>{
  mapContainer = React.createRef<HTMLDivElement>();
  mapView: MapView;
  webMap: WebMap;
  extentWatch: __esri.WatchHandle;
```

Widgets: Building Blocks to Create an Experience

Widget Structure



Define a Widget from BaseWidget Class

```
import {BaseWidget, classNames, FormattedMessage, defaultMessage as jimuCoreDefaultMessage} from 'jimu-core';

export default class Widget extends BaseWidget<AllWidgetProps<IMConfig>, any>{
  constructor(props){
    super(props);

    this.state = {
      activeTab: 'properties'
    };
  }
}
```

A widget component must extend the BaseWidget class

Datasource

- Datasource defines how your widget accesses data.
- jim-core has a DataSource interface with common methods such as
 - getRecords ()
 - getSelectedRecords()
- Custom datasource can be created through DataSource interface or by extending the AbstractDataSource class
- Datasource is managed by DataSourceManager

Datasource Features

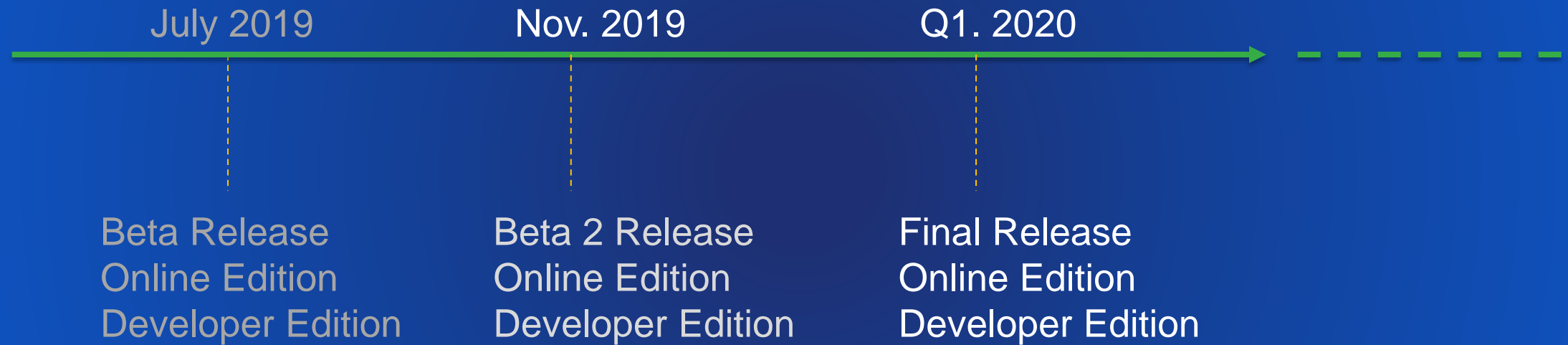
- Share data between widgets
- The ability to change a datasource and not break the app
- Put the selected data in URL to create durable URL
- Provide a continuous pipeline of data from one widget to another

Getting Started

1. Download and install Node.js (v10.x)
2. Download developer edition from the [early adopter Esri site](#)
3. Sign into ArcGIS for Developer Site
 - [Register](#) your Experience Builder client to obtain Client ID
 - Your machine name [https://\[machine_name\]:3001/](https://[machine_name]:3001/) needs to be one of the Redirect URIs
4. Update the value of ClientID in client/dist/builder/setting.json
5. Run npm ci from the server directory of Experience
6. Type node src/server to start the server
7. Open Experience Builder and login with your ArcGIS Online credentials

Create a New Widget

Product Release Timeline



Helpful Resources

Try out Experience Builder Beta

<https://experience.arcgis.com/>

Download Developer Edition Beta

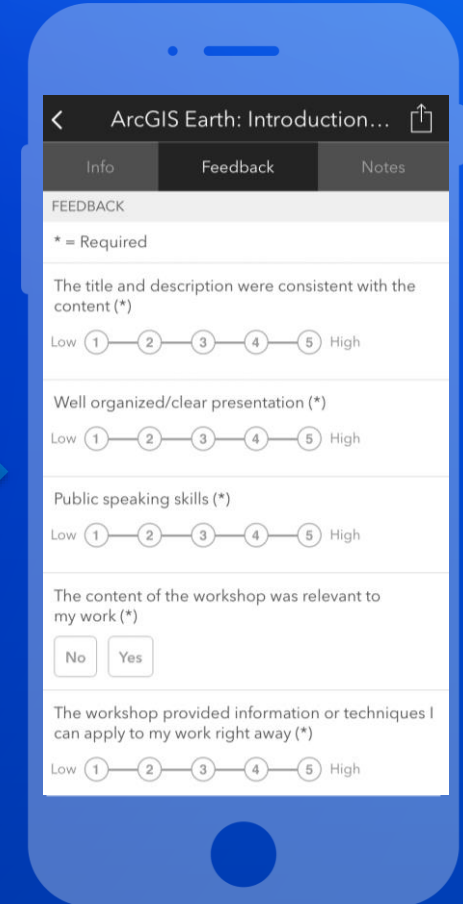
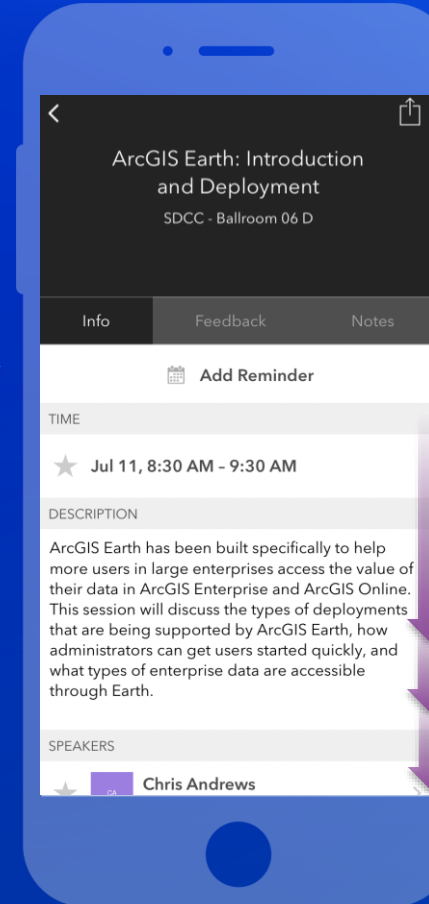
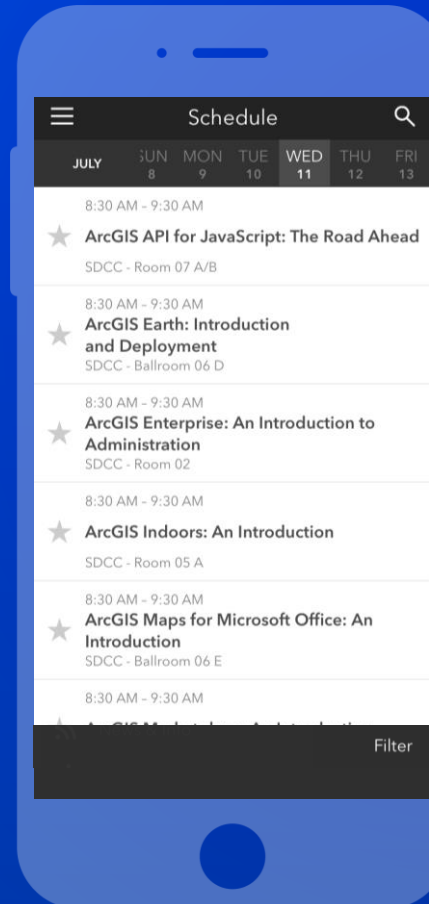
<https://earlyadopter.esri.com/>

Experience Builder on Geonet

<https://community.esri.com/community/arcgis-experience-builder>



Please Take Survey on the App





esri

THE
SCIENCE
OF
WHERE