



ArcGIS API for JavaScript

ESRI DEVELOPER WORKSHOP 2022



Content

- **Basic concepts**
- ArcGIS REST API
- Securing your data
- Widgets
- Integration with other frameworks
- Plenty to Get You Going



ArcGIS API for Javascript





Documentation

The screenshot shows the top navigation bar of the ArcGIS Developer website. The main navigation includes 'ArcGIS Developer', 'Documentation', 'Features', 'Pricing', and 'Support'. A search bar and a user profile for 'Mantas Bukauskas manbuk' are also visible. Below the main navigation, there is a sub-navigation bar for 'ArcGIS API for JavaScript' with options for 'Overview', 'Sample Code', 'API Reference', 'Showcase', and 'Blog'. The 'Overview' page is currently selected. On the left side of the page, there is a search box labeled 'Find page...' and a list of search results, with 'Overview' being the first result. The main content area displays the title 'Overview' with a horizontal line underneath it.

Available at <https://developers.arcgis.com/javascript/latest/>

Basic concepts



Installation

Options:

1. Use API from CDN or host in on premise?
2. Use AMD or ESM module structure?
3. Use plain JS or use TypeScript?
4. Build (minify and pack) you application for best performance or leave it as is?

```
</title>
```

```
<link rel="stylesheet" href="https://js.arcgis.com/4.22/esri/themes/dark/main.css"/>  
<script src="https://js.arcgis.com/4.22/"></script>
```

```
require([  
  "esri/Map",  
  "esri/views/MapView"  
], function (  
  Map,  
  MapView  
) {
```

npm install @arcgis/core

```
import config from '@arcgis/core/config';  
import Map from '@arcgis/core/Map';  
import MapView from '@arcgis/core/views/MapView';  
import FeatureLayer from '@arcgis/core/layers/FeatureLayer';  
import { initWidgets } from './widgets';
```



Demo.

Creating you first application



Core concepts

- Maps and Views
- Layers and data
- Programming patterns



Basic concepts

Maps and View

```
const map = new WebMap({  
  portalItem: {  
    id: "<itemId>"  
  }  
});
```

WebMap
class

MapView class

```
const view = new MapView({  
  container: "viewDiv",  
  map: map,  
  zoom: 4,  
  center: [15, 65]  
});
```

HTML element
placing map on web
page

Properties



Layers and Data

FeatureLayer	Displaying, querying, filtering and editing large amounts of geographic features.
GraphicsLayer	Displaying individual geographic features as graphics, visual aids or text on the map.
CSVLayer / KMLLayer / GeoJSONLayer	Displaying data stored in an external file format as a layer.
TileLayer / VectorTileLayer	Displaying basemaps and other tiled datasets for geographic context.
MapImageLayer	Displaying layers dynamically rendered by an ArcGIS Server service.
WMS / WFS / WMTS Layer	Displaying map data using OGC specification.
SceneLayer	Displaying 3D data.



Basic concepts



Programming patterns (1)

- Loading Classes

```
import MapView from "@arcgis/core/views/MapView";  
import Map from "@arcgis/core/Map";  
import esriRequest from "@arcgis/core/request";  
import MapImageLayer from "@arcgis/core/layers/MapImageLayer";
```

- Constructors

```
const view = new MapView({  
  map: new Map(),  
  container: 'app-view',  
  center: [122, 38]  
});  
view.scale = 5;
```

Programming patterns (2)

- Properties

```
const basemapTitle = map.get("basemap.title");
```

Safe
getter

```
view.zoom = 6;  
map.basemap = "oceans";
```

- Watching for Property Changes

Property
name

Callback
function

```
const handle = view.watch('center', (newValue, oldValue, property, object) => {  
  console.debug(`New value: ${newValue}`,  
    `Old value: ${oldValue}`,  
    `Watched property: ${property}`,  
    `Watched object: ${object}`)  
});
```

Programming patterns (3)

- Promises

Promise

Promise(s)
rejected

Promises
fulfilled or
rejected

```
view.when()  
  .then((param1) => { return featureLayer.when(); })  
  .then((param2) => { console.debug(param2); })  
  .catch((error) => { console.error(error); })  
  .finally(() => { console.debug("All promises resolved."); });
```

Chained
promise

Promises
fulfilled

- Async / await

```
async viewAndLayerReady() {  
  try {  
    const param1 = await view.when();  
    const param2 = await featureLayer.when();  
  }  
  catch(error) { console.error(error); }  
  finally { console.debug("All promises resolved."); }  
}
```

Programming patterns (4)

- Widget viewModel pattern

```
const searchVM = new SearchViewModel({ view });  
  
view.on("click", (event) => {  
  searchVM.search(event.mapPoint);  
});
```

**viewModel
constructor**

**Use of
viewModel
method
without UI**



Demo.

Creating and loading WebMap



Content

- Basic concepts
- **ArcGIS REST API**
- Securing your data
- Widgets
- Integration with other frameworks
- Plenty to Get You Going



ArcGIS REST API

The screenshot shows the navigation menu of the ArcGIS Developer website. The top bar is dark blue with white text. It includes a home icon, 'ArcGIS Developer', 'Documentation', 'Features', 'Pricing', 'Support', a search icon, 'Search', and a user icon, 'Sign In'. Below this is a secondary menu with 'ArcGIS REST APIs', 'Home', 'Ready-to-use', 'Content management', 'All services' (which is underlined), and 'Enterprise administration'. A third bar contains 'Enterprise', 'Online', and 'Mission'.

<https://developers.arcgis.com/rest/>



Using the Services Directory





Content

- Basic concepts
- ArcGIS REST API
- **Securing your data**
- Widgets
- Integration with other frameworks
- Plenty to Get You Going



Authentication methods

- API Keys
- ArcGIS Identity
- Others

API Keys (New Jan 2021)

- Used for location services
- Publicly accessible
- Restricted to specific services
- Read private content (Developer accounts only)





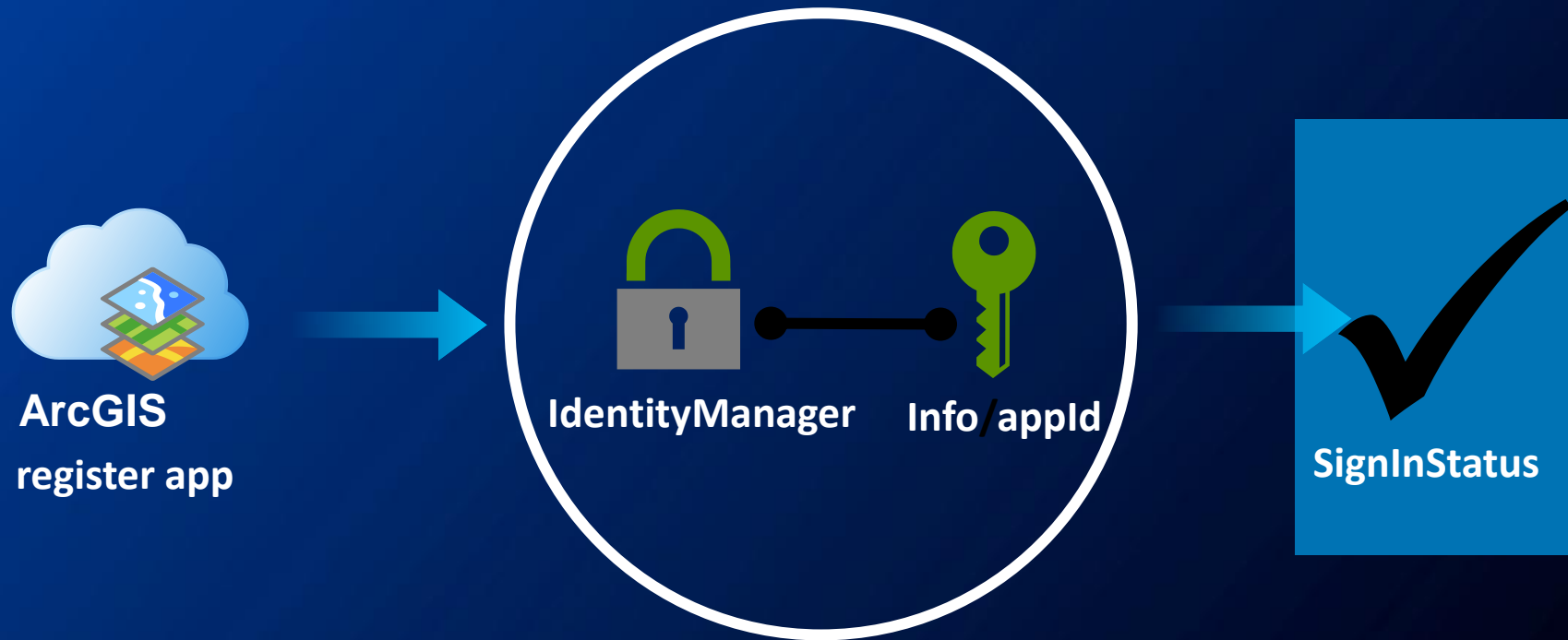
ArcGIS Identity

- Represents a user profile in ArcGIS Online/Enterprise
- Your apps can be authorized to access a user's identity via OAuth 2.0
- Represented by access token + expiration date

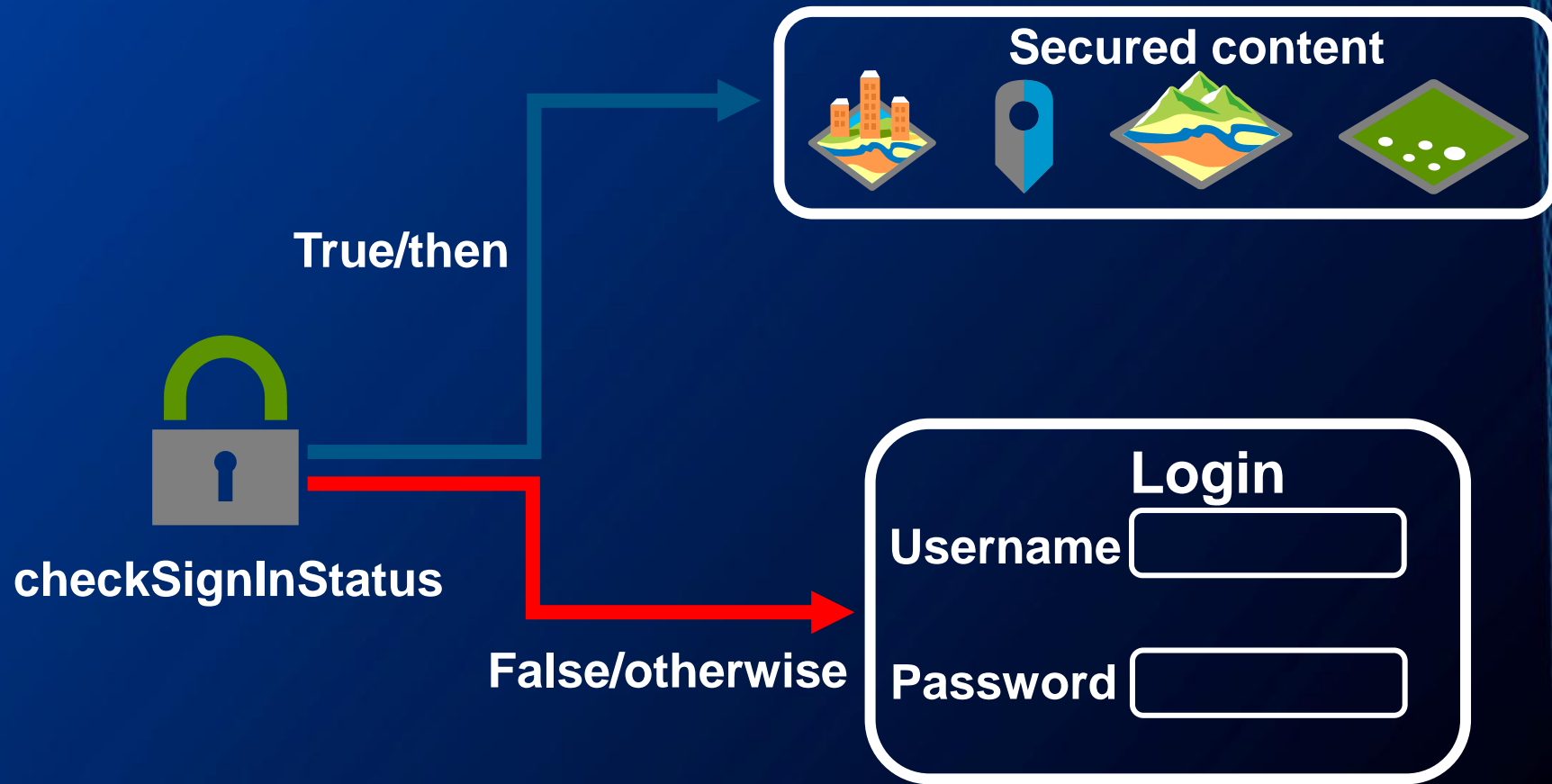


Securing your data

IdentityManager (1)



IdentityManager (2)





Demo.

Registering your application



Content

- Basic concepts
- ArcGIS REST API
- Securing your data
- **Widgets**
- Integration with other frameworks
- Plenty to Get You Going



Out of the box widgets

- 49 widgets available out of the box

AreaMeasurement2D - AreaMeasurement3D - Attribution - BasemapGallery - BasemapLayerList - BasemapToggle - Bookmarks - BuildingExplorer - Compass - CoordinateConversion - Daylight - Directions - DirectLineMeasurement3D - DistanceMeasurement2D - Editor - ElevationProfile - FeatureForm - FeatureTable - FeatureTemplates - FloorFilter - Home - LayerList - Legend - LineOfSight - Locate - Measurement - NavigationToggle - Popup - Print - ScaleBar - ScaleRangeSlider - Search - ShadowCast - Sketch - Slice - Swipe - TimeSlider - Track - UtilityNetworkTrace - Zoom



Widget development (1)

- TypeScript and JSX
- Accessor
- @arcgis/core
- @subclass()

```
1 import MyWidget  
  from "./MyWidget";
```

```
2 const myWidget = new  
  3 MyWidget({  
    view: view,  
    ...  
  });
```

```
4 view.ui.add(myWidget, {  
  position: "bottom-left"  
});
```

Widget development (2)

- Widget life cycle

constructor (params) - This is where the widget is initially created while setting any needed properties. Since the widget is derived from [Accessor](#), you get access to getting, setting, and watching properties as discussed in the [Working with properties](#) topic.

postInitialize() - This [method](#) is called after the widget is created but before the UI is rendered.

render() - This is the only [required method](#) and is used to render the UI.

destroy() - [Method](#) to release the widget instance.

Widget development (3)

- Extend the Widget class
- Implement properties and methods
- Render the widget
- Export module

```
class HelloWorld extends Widget {  
  
  constructor(params?: any) {  
    super(params);  
    this._onNameUpdate = this._onNameUpdate.bind(this);  
  }  
}
```

```
// Create 'name' property  
@property()  
name: string = "John Smith";
```

```
render() {  
  return (  
    <div>  
      {this._onNameUpdate()}  
    </div>  
  );  
}
```

```
export default HelloWorld;
```



Demo.

Creating custom widget



Content

- Basic concepts
- ArcGIS REST API
- Securing your data
- Widgets
- **Integration with other frameworks**
- Plenty to Get You Going



Supported frameworks



<https://github.com/esri>

Integration with other frameworks



Low or no code applications

- Configurable apps
- StoryMaps
- ArcGIS Experience Builder
- ArcGIS Web AppBuilder

Integration with other frameworks



Content

- Basic concepts
- ArcGIS REST API
- Securing your data
- Widgets
- Integration with other frameworks
- **Plenty to Get You Going**



What's next?

- Nearly 300 code samples <https://developers.arcgis.com/javascript/latest/>
- Open-source starter apps
- JSAPI Resources (GitHub) <https://github.com/Esri/jsapi-resources/>
- Step-by-step tutorials
- Community <https://developers.arcgis.com/javascript/latest/community/>



ESRI DEVELOPER
WORKSHOP 2022

Thank you!



May 13, 2022 | Tallink Spa & Conference Hotel | Sadama 7, Tallinn
<https://arcg.is/37GjtSR>