Esri Developer Summit

March 8–11, 2016 | Palm Springs, CA

Overview of the ArcGIS Runtime Quartz API

Euan Cameron

Will Crick

Justin Colville

Reasons for API redesign

- New internal architecture requiring a new way of doing things
- Allow new capabilities
  - e.g. Map authoring
- Ease of use
  - Easy for new users, easier for existing users
    - Better OO / use of established patterns
  - Reduce areas of confusion
    - E.g. Difference between a graphics layer vs a feature layer
  - Consistency across the api
    - E.g. standard pattern for asynchronous loading of resources
- Robustness and performance
  - E.g. Draw performance / Geometry manipulation
- Opportunity to reflect on our evolving platform

ArcGIS Runtime Common API

- Common conceptual model across platforms
  - Capabilities, names and parameters will be the same
- Not a generic API
  - Leverage individual platform strengths

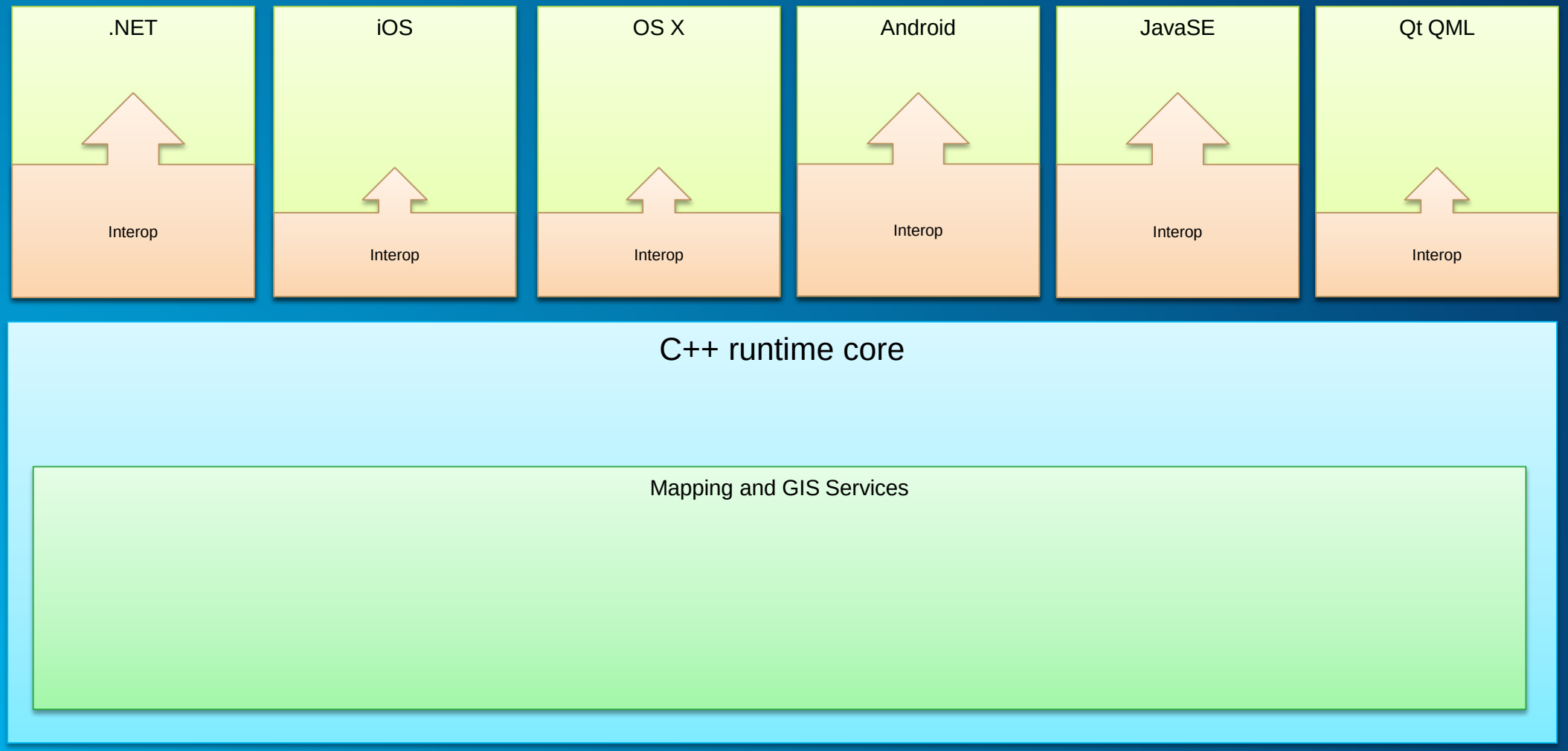
Quartz changes

- Internal architecture
- By reference semantics
- Mapping API
  - Map and MapView
  - Viewpoint
  - Graphics Overlays
  - Layers name and composition changes
  - Advanced symbology
- Geometry API
- Portal API
- Authentication
- 3D API

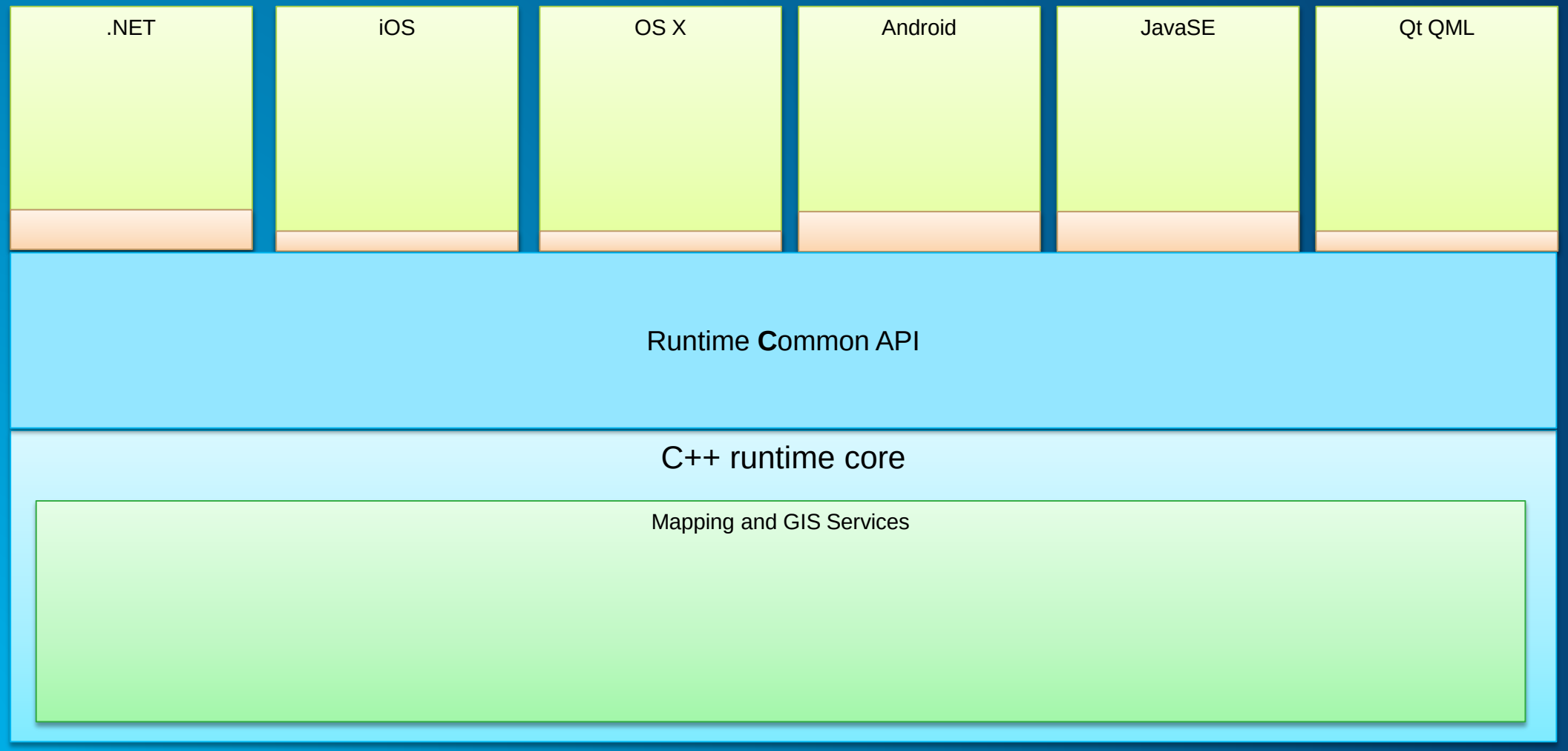
.NET 10.2.7 looks like this now!

Architecture

ArcGIS Runtime 10.2.x Architecture



ArcGIS Runtime Quartz Architecture



Mapping API

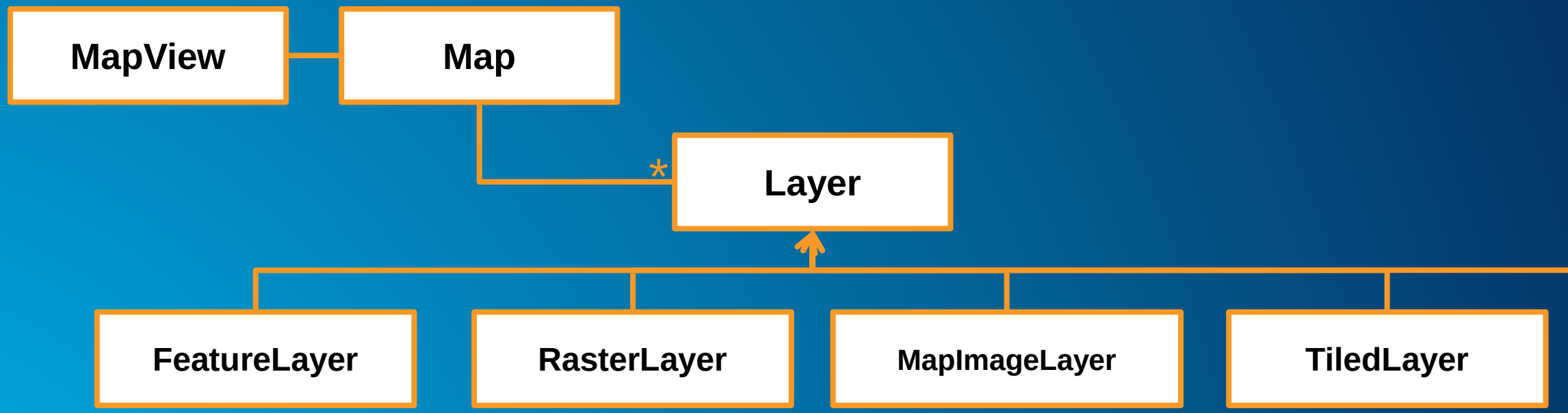
Refreshed Mapping API to Accommodate New Functionality

- API builds on what is already there
- Adds new capabilities
- Fully embraces *The Map* as a central component

Where do maps come from?

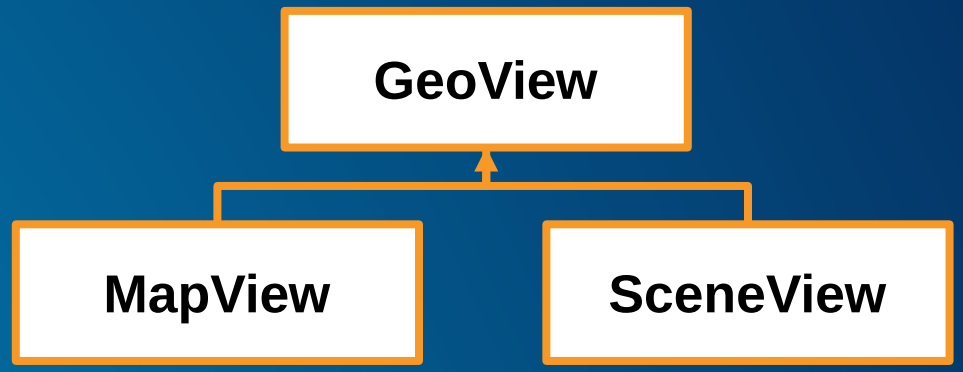
- Portal (web maps)
  - Create with a PortalItem or URL
- Pro (mobile maps)
  - Access maps inside a Mobile Map Package
- From disk (mobile maps)
  - Create with a LocalItem or a Path
- You!
  - Create a map in code
  - Save to a portal
    - Web maps and mobile maps will be different portal item types
  - Save locally

Maps and Layers



GeoView & MapView

- GeoView
  - Graphic Overlays
    - List<GraphicOverlay>
  - Location Display
  - Generic setViewpoint methods (Viewpoint only)
  - Draw events
  - Layer state events
- MapView
  - setViewpoint 2D methods (center, center & scale, geometry)
  - Visible Area changed event
  - Identify methods
    - Identify GraphicOverlays – returns Graphics
    - Identify Layers – returns GeoElements



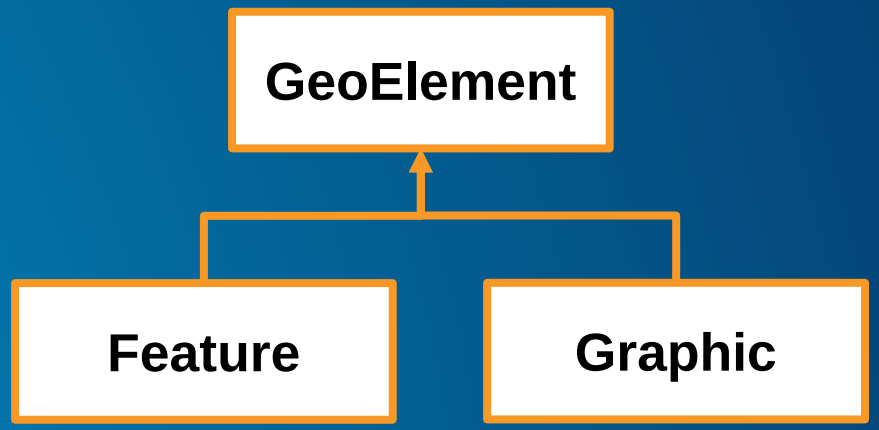
Viewpoint

- Describes the point of view for 2D and 3D
  - Center and scale
  - Bounding geometry
  - Camera

- Javascript api too:
  - http://jscore.esri.com/javascript/beta/api-reference/esri-Viewpoint.html

Graphics Overlay

- Temporary graphics added to MapView ONLY
- Contain Graphics
  - List<Graphic>
  - Graphic
    - Attributes, symbol, geometry
- Renderer
- Rendering mode
  - Dynamic – better UX for display
  - Static – good for lots of graphics



The Map

- Operational Layers
  - List<Layer>
- Basemap
  - ReferenceLayers – List<Layer>
  - BaseLayers - List<Layer>
  - Default constructors (Top, Streets, NatGeo etc…)
- Spatial Reference
- Initial Viewpoint
- Bookmarks
  - List<Bookmark>
- Item (Portal or Local)

Map
(from mapping)

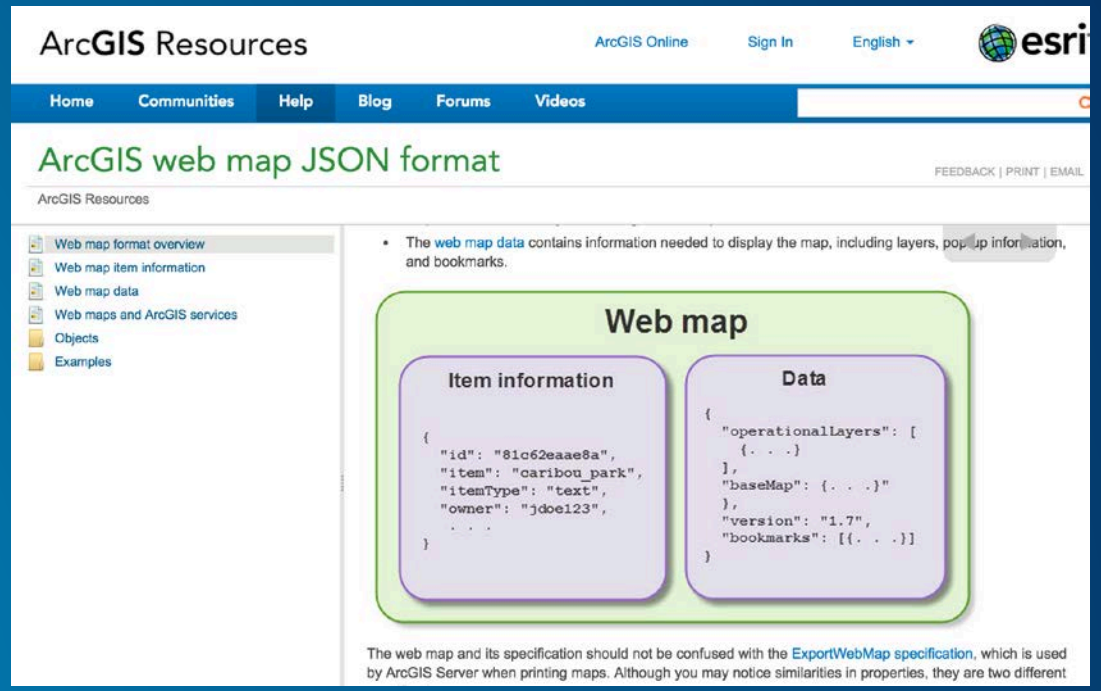
RequestConfiguration: RequestConfiguration
Basemap: Basemap
Bookmarks: Bookmark[*]
InitialViewpoint: Viewpoint
Item: Item {read-only}
OperationalLayers: Layer[*]
SpatialReference: SpatialReference {read-only}
Version: String {read-only}
MaxScale: Double
MinScale: Double
TransportationNetworks: TransportationNetworkDataset[*]

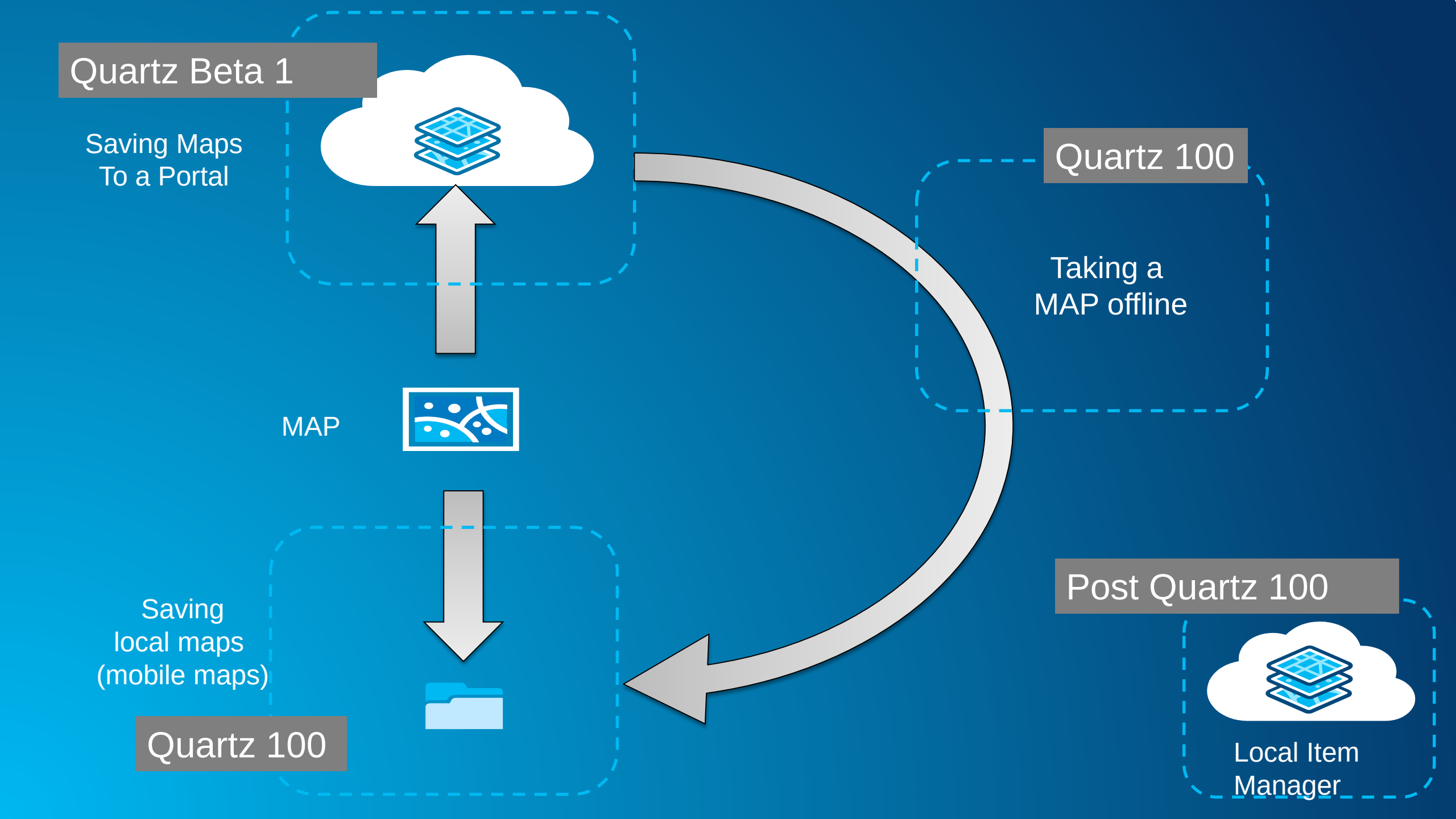
Map(): Map
Map(Basemap): Map
Map(SpatialReference): Map
Map(BasemapType, Float, Float, Integer): Map
Map(URI): Map
Map(Item): Map
saveAsync()
saveAsAsync(String, Portal, String[*], String, LoadableImage)
saveAsAsync(Item)

Saving maps

- **Update an existing map on the portal**
  Map.save()

- **Create and save a new map**
  Map.saveAs(title,portal,folder,...)
  Map.saveAs(portalItem)




Quartz Beta 1
Saving Maps
To a Portal
Quartz 100
Taking a
MAP offline
MAP
Saving
local maps
(mobile maps)
Quartz 100
Post Quartz 100
Local Item
Manager

Mobile map packages

- A container of Maps, Layers, their data
- Packages can also contain Locators
- Get maps
  - Access layers via a map
  - Maps also contain networks

MobileMapPackage
(from mapping)

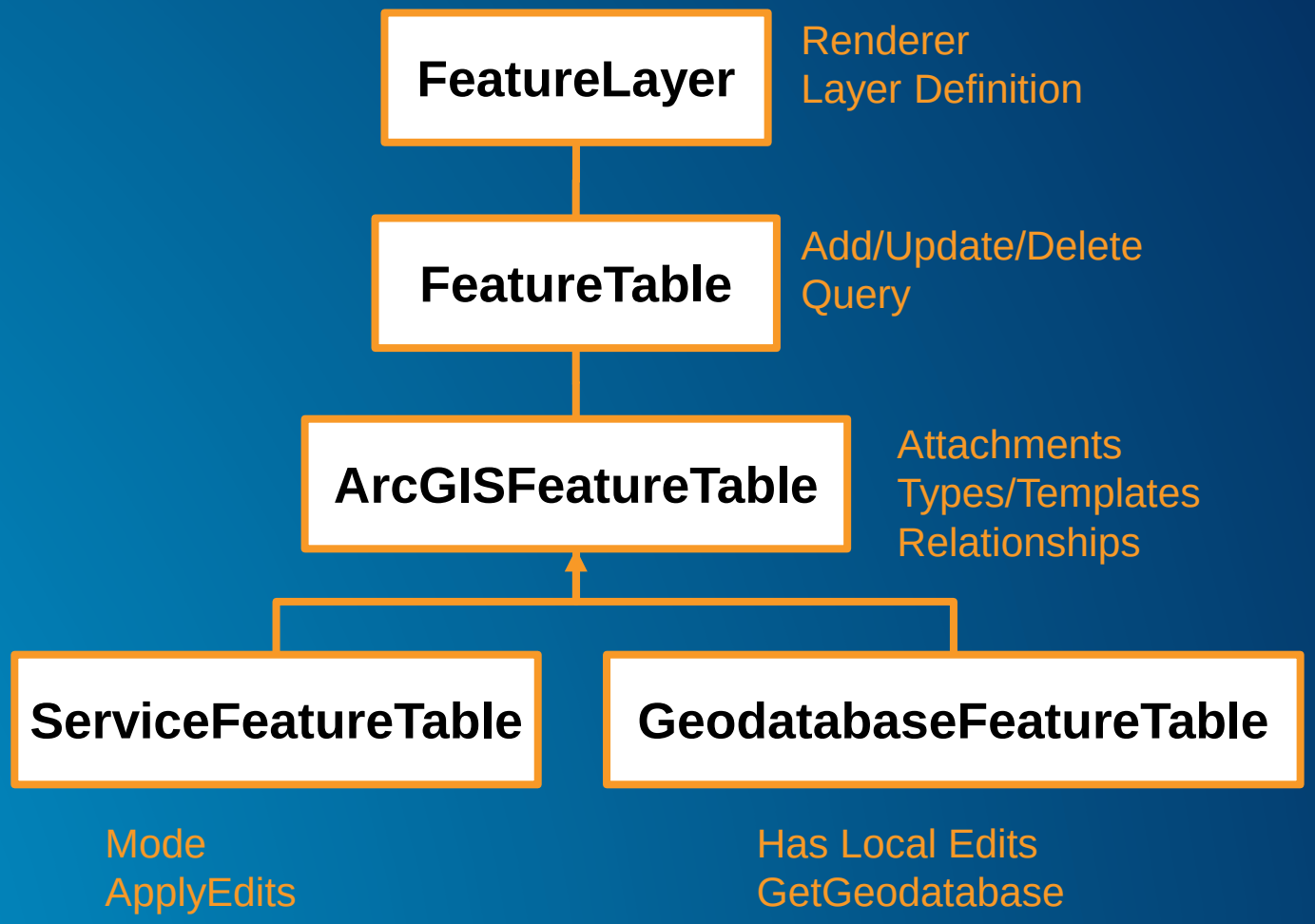
Maps: Map[*] {read-only}
Item: Item {read-only}
Locator: LocatorTask {read-only}
Path: String {read-only}

MobileMapPackage(String): MobileMapPackage

Mapping API - Layers

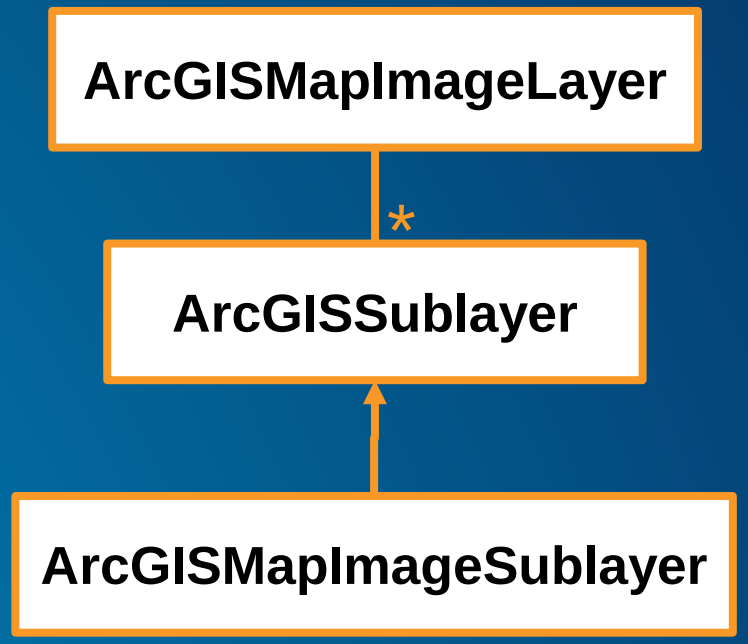
Feature Layer

- **ServiceFeatureTable**
  - **Feature Request Mode**
    - **On_Interaction_Cache**
    - **On_Interaction_No_Cache**
    - **Manual_Cache**
      - **populateFromService()**



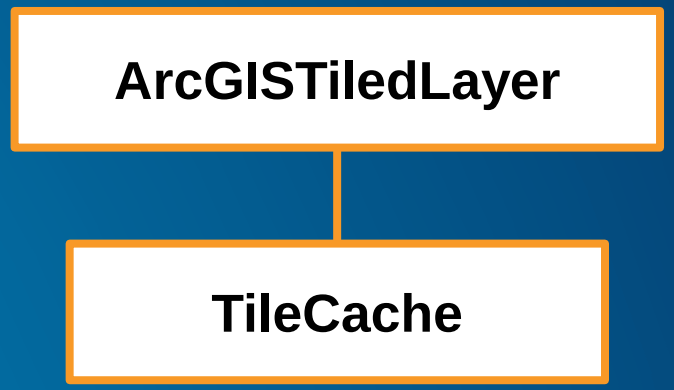
Map Image Layer

- ArcGISMapImageLayer
  - New name for ArcGISDynamicMapServiceLayer
- ArcGISSubLayers
  - Can get renderers
  - Common with Tiled layers
- ArcGISMapImageSubLayers
  - Can set renderers (dynamic layers)



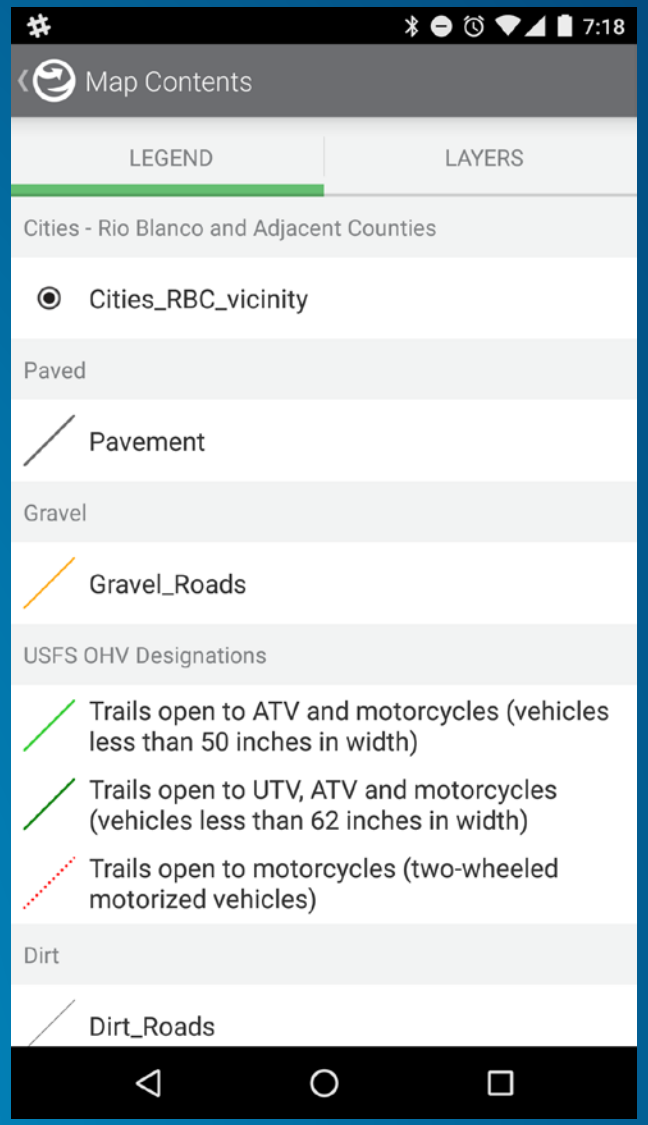
ArcGIS Tiled Layer

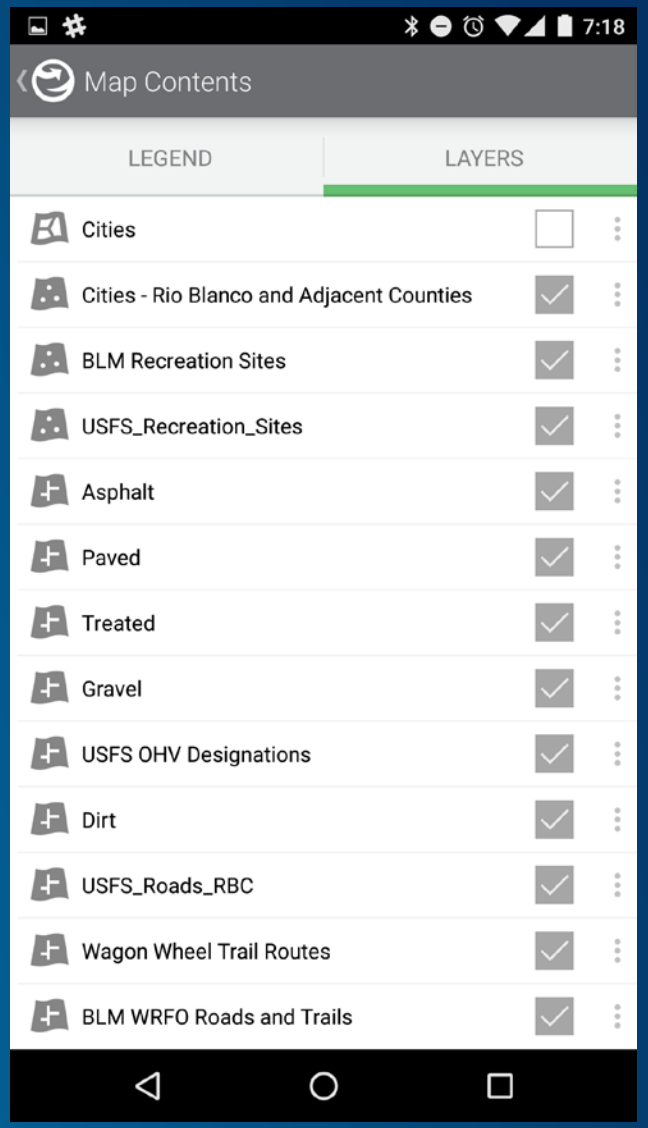
- Create from URL or TileCache
- Supports online and offline
  - Combines ArcGISTiledServiceLayer and ArcGISLocalTiledLayer
- ArcGISSubLayers
  - Can get renderers
  - Common with MapImageLayer



Layer Content

- Interface for building Layer Lists, Legends and Tables of Contents
- Visibility
- IsVisibleAtScale(scale)
- SubLayerContent
  - For map image layers, tiled layers, group layers etc
- LegendInfo
  - Symbol – can create a swatch
  - Name
- Content changed listeners (visibility)





Symbols and Renderers

- By reference
- Web symbology
  - Minor naming changes
  - Addition of new renderers/properties for smart mapping
    - Dot density renderer, sized by attributes - graduated/proportional symbols (visual variables)
- Advanced symbology API
  - CIM symbol objects
  - Military symbology and custom symbology
    - SymbolDictionary – from style file in Pro
    - DictionaryRenderer – apply to feature layer / graphics layer

Loadable

What is it?

- **A pattern for classes that asynchronously load metadata**
  - **Layers, Map, Portal Items, Geodatabase, Tile Cache, Features etc**

- **Formalizes, enhances, and improves upon concepts that we had before**
  - **Provides a generic API across different types of classes**
  - load()
  - retryLoad()
  - cancelLoad()
  - loadStatus
  - loadError

Loadable

What is it?

- State is explicit
  - Not Loaded
  - Loading
  - Loaded
  - Failed To Load
- State changes can be monitored easily
- Resources don’t auto-load.
  - Loaded directly by developer or indirectly through other objects
    - load()

Loadable features

- Features don’t always have all of their attributes
  - Load the feature to get all the data
  - Load the feature to edit
  - Option to load features for a query result
- Allows us to optimize feature queries for map rendering
  - Including generalized geometries in certain modes
- Reduces complexity of OutFields
- Removes parameters on Query which don’t return features (ReturnIdsOnly, ReturnCount)
- Reduces query parameters (ReturnZ, ReturnM)

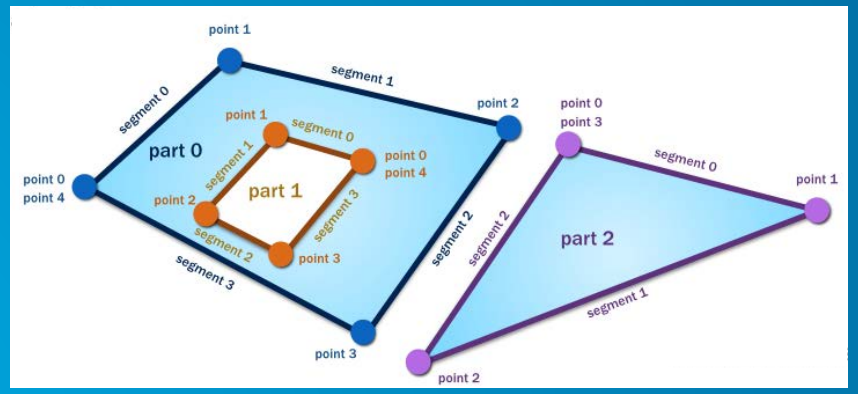
Loadable Feature Patterns

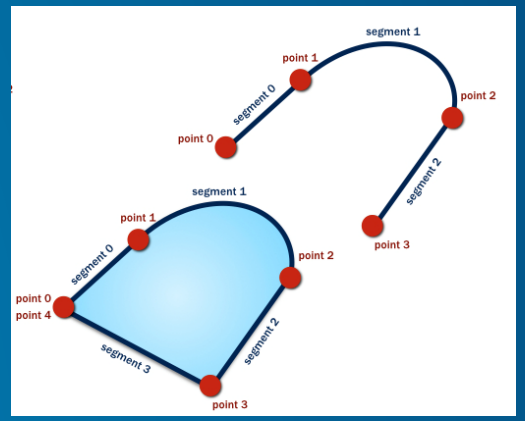
- Identify features on the map
  - Identify on the MapView returns a feature
  - Load the feature
  - Show in popup/callout
- Show a table view of features
  - Query the table, set QueryOptions to return loaded features
  - Show the features in a table view
- Edit features
  - Load the feature BEFORE editing

Geometry API

Geometry API

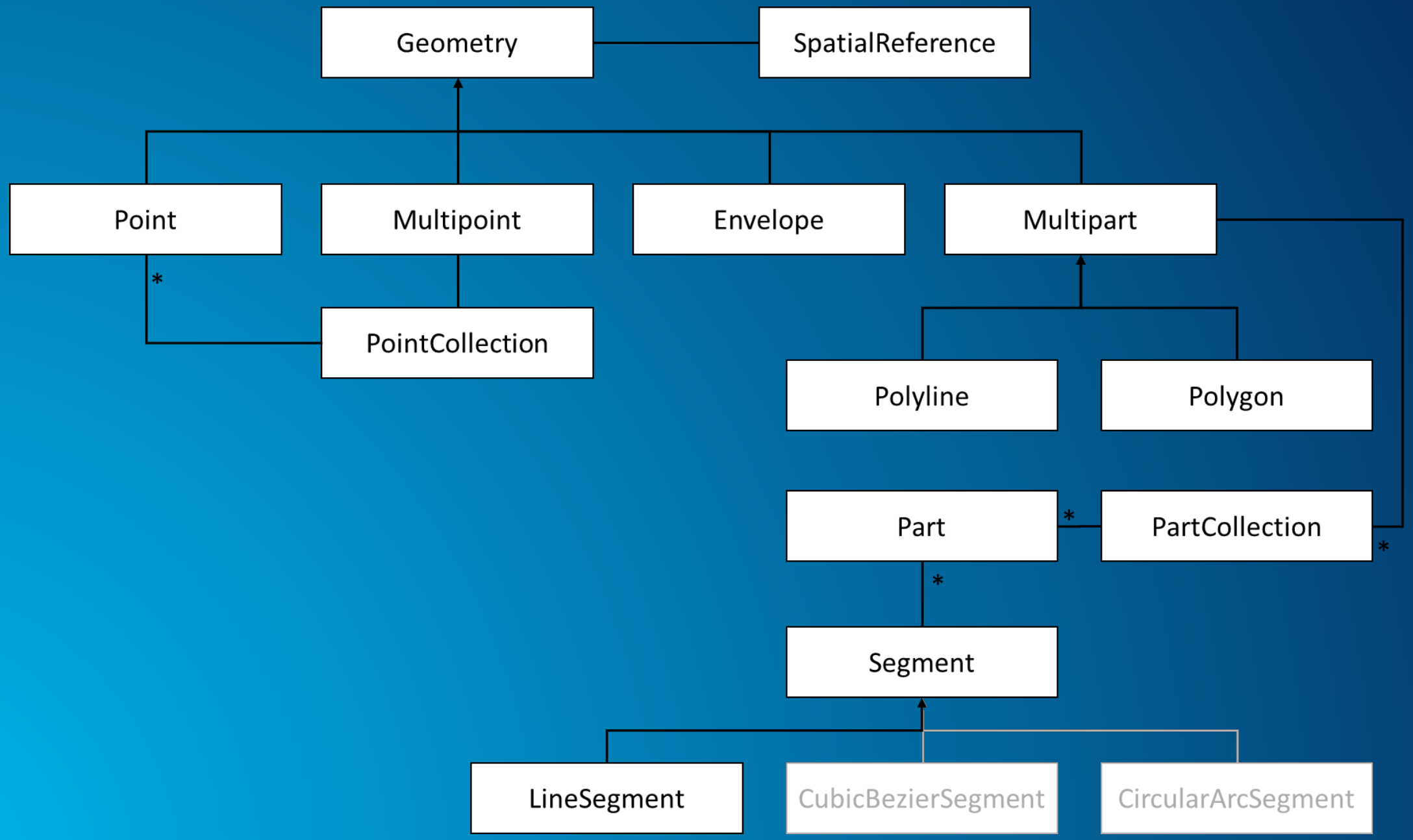
- Synchronized geometry model across all platforms
- Improved architecture and performance
  - Immutable geometries
- Can support curves





Geometry

- Immutable
- Has a spatial reference
- Create new with Builders
- E.g. PolygonBuilder
  - Constructors
    - Polygon
    - Point collection
    - Part
    - PartCollection
  - Methods
    - Add point/s
    - Add part/s
- Curves…


Geometry
SpatialReference
Point
Multipoint
Envelope
Multipart
*
PointCollection
Polyline
Polygon
Part
*
PartCollection
*
*
Segment
LineSegment
CubicBezierSegment
CircularArcSegment

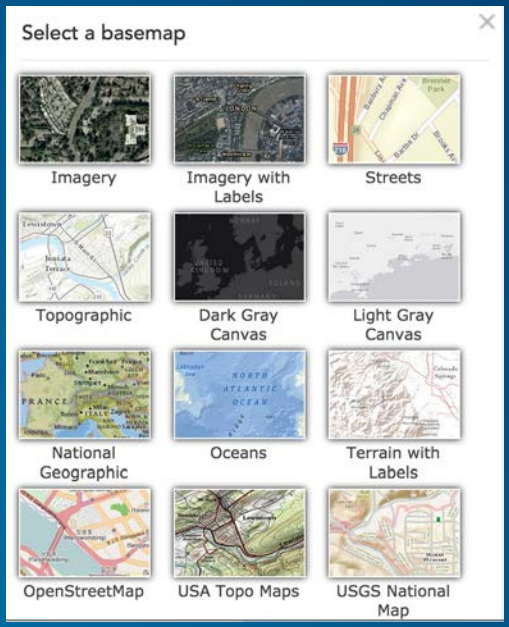
Portal API

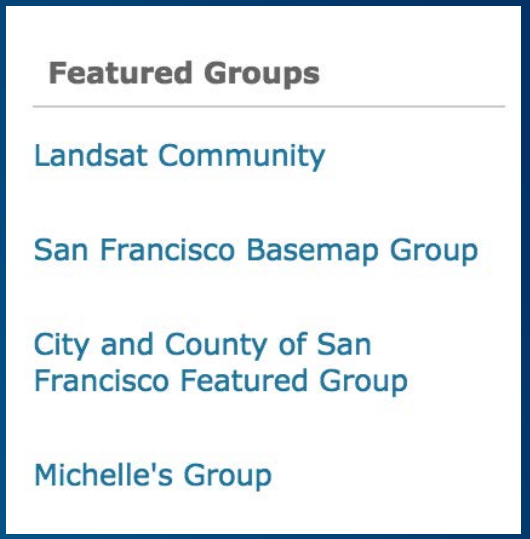
Runtime Portal API

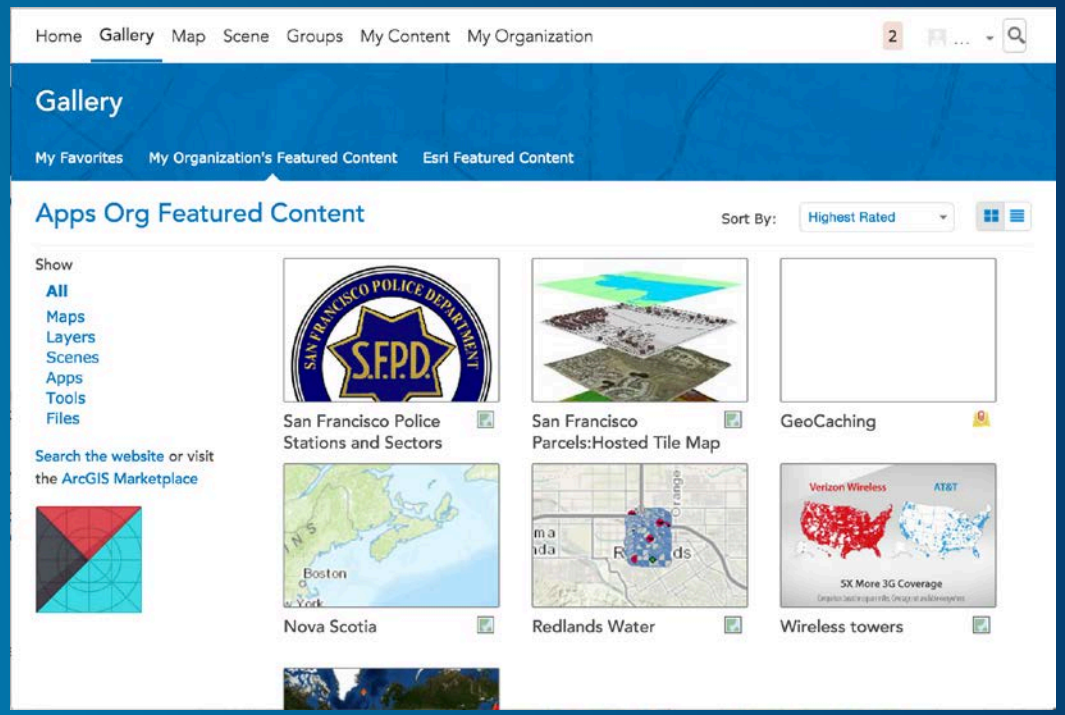
- **A developer’s gateway to the portal information model**
- **High-level, coarse-grained APIs to access, use, create, and share content**
- **Key Classes**
  - Portal
  - PortalUser
  - PortalItem
  - PortalGroup

Common Portal Workflows

- **Access foundational content of a portal**
  - **Basemaps**
    Portal.fetchBasemaps()
  - **Featured Groups**
    Portal.fetchFeaturedGroups()
  - **Featured Items**
    Portal.fetchFeaturedItems()

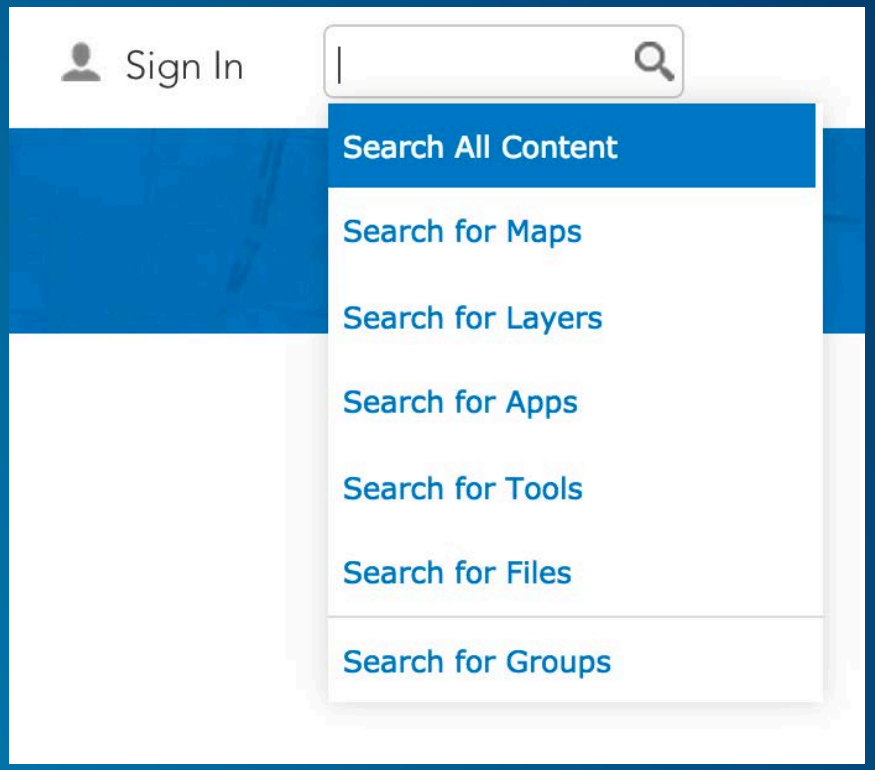




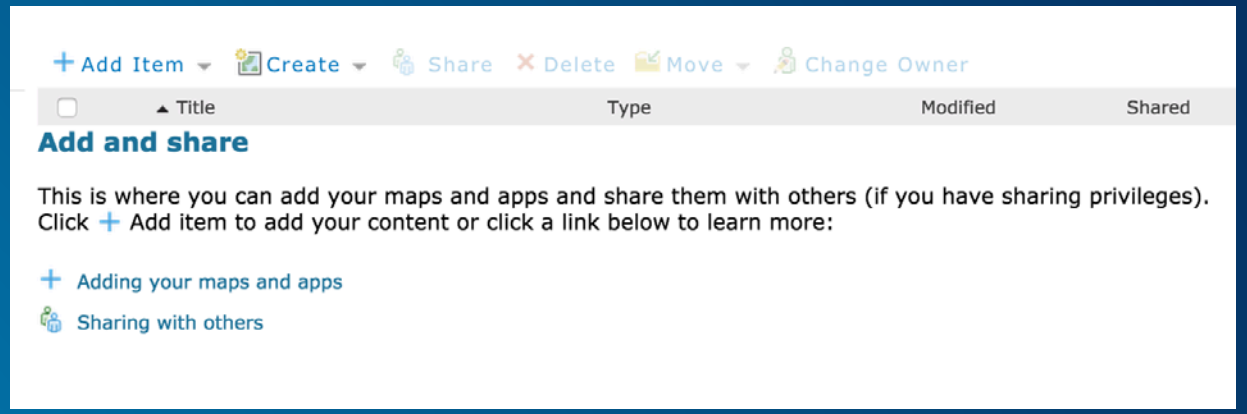


Common Portal Workflows

- **Search for content and groups**
  - Portal.findItems(portalQueryParams)
  - Portal.findGroups(portalQueryParams)



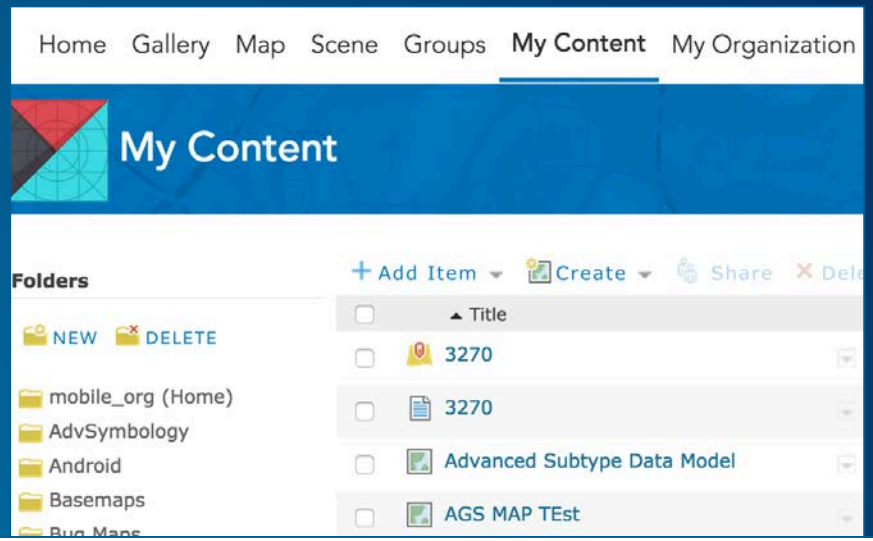
Common Portal Workflows

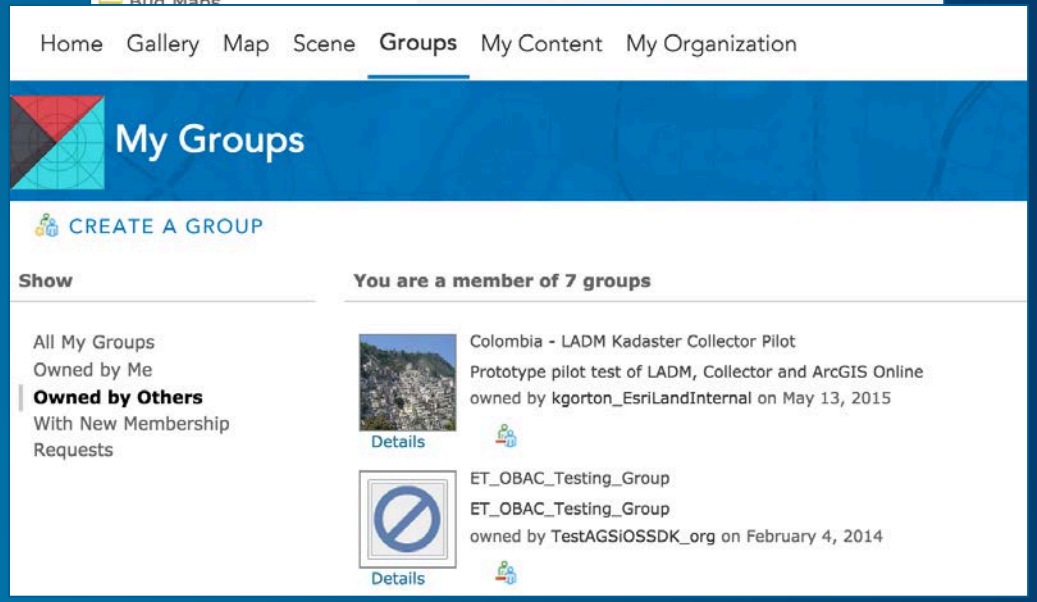


- **Add, modify, and share content**
  - PortalUser.addPortalItem(item, params, folder)
  - PortalItem.updateData(json)
  - PortalItem.updateData(file)
  - PortalItem.shareWithGroups(groups)

Common Portal Workflows

- **Access user's content and groups**
  - PortalUser.fetchContent()
  - PortalUser.fetchContentInFolder(folder)
  - PortalUser.groups





Using portal content

- **Portal API integrates seamlessly with Mapping API**
- PortalItem **can represent a Web Map, Basemap, or Layer**

```
//If the portal item represents a webmap
Map map = Map(portalItem)
...
//If the portal item represents a basemap
Basemap basemap = Basemap(portalItem)
...
//If the portal item represents a layer
ArcGISMapImageLayer layer = ArcGISMapImageLayer(portalItem)
```

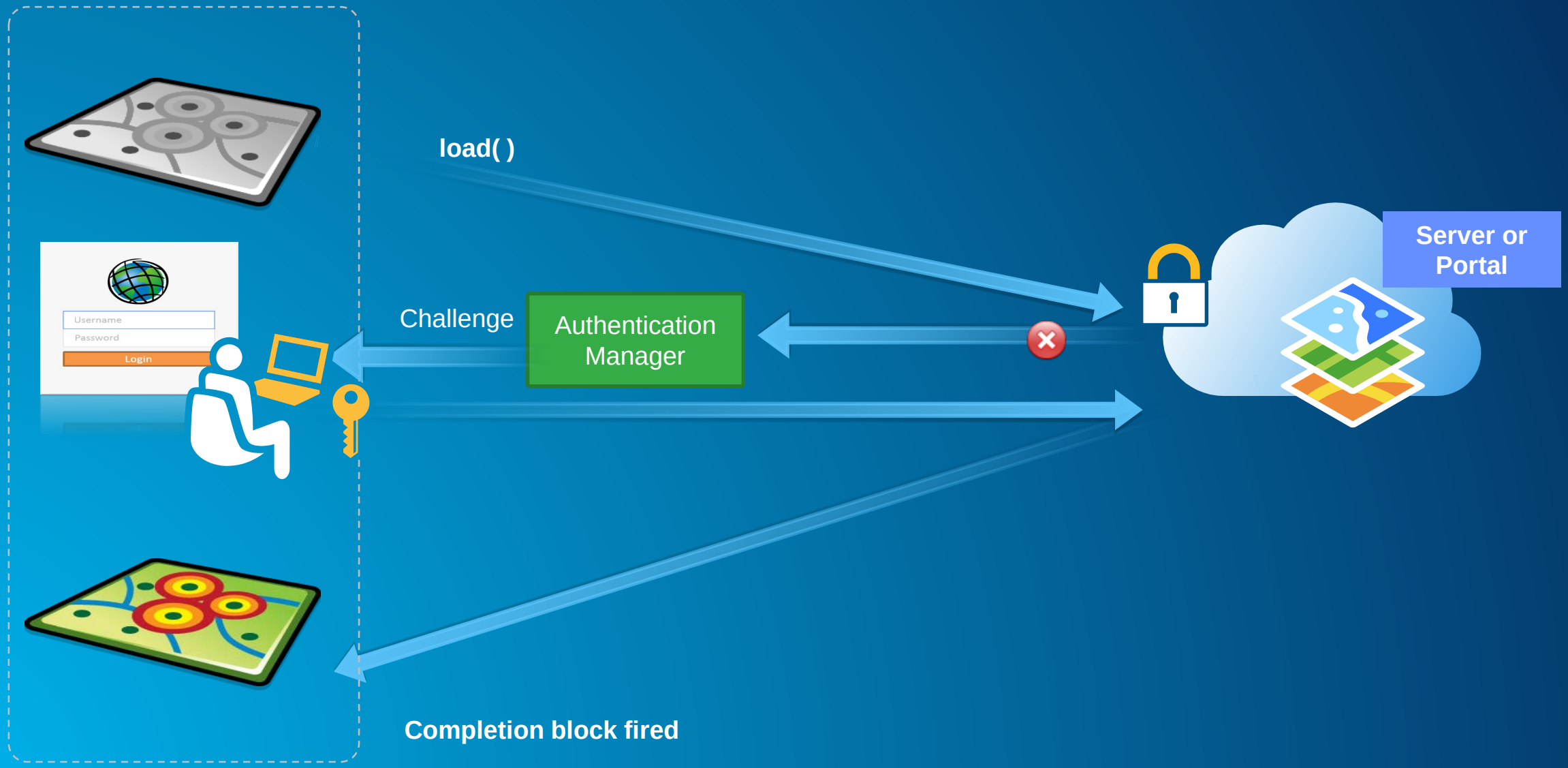
Authentication

Authentication Manager

(aka IdentityManager)

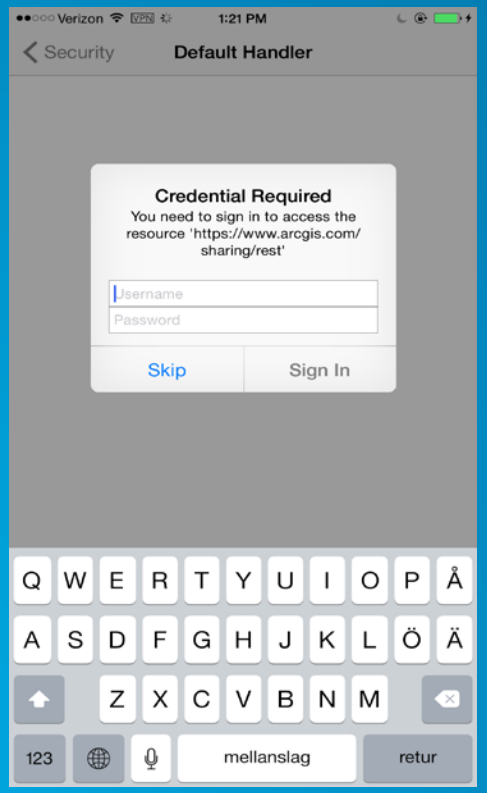
- AuthenticationManager
- Coarse-grained
  - Hides unnecessary details from the developer
- Go-to class for all security related configuration
- Issues challenges when authentication failures are encountered
  - In-built challenge handling to provide default behavior
  - Developer can customize challenge handling

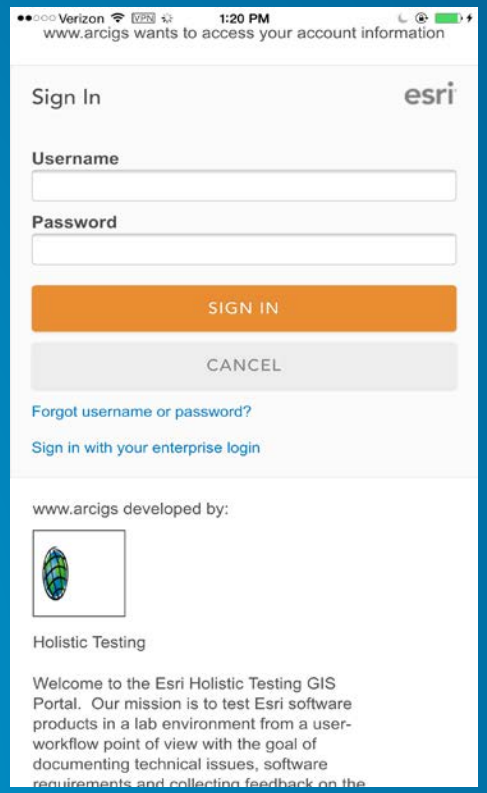
Issuing Challenge

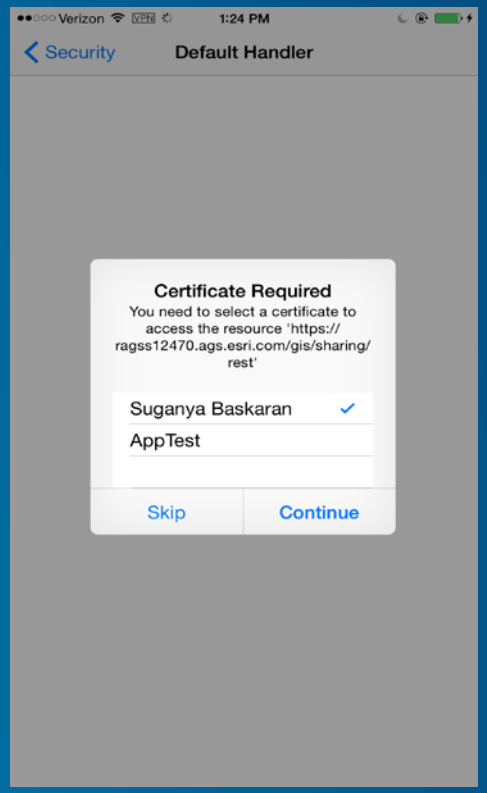


Responding to Authentication Challenges

- **In-built handling**
  - **Authentication Manager displays UI to capture user credentials**
  - **Retries loading with credentials**







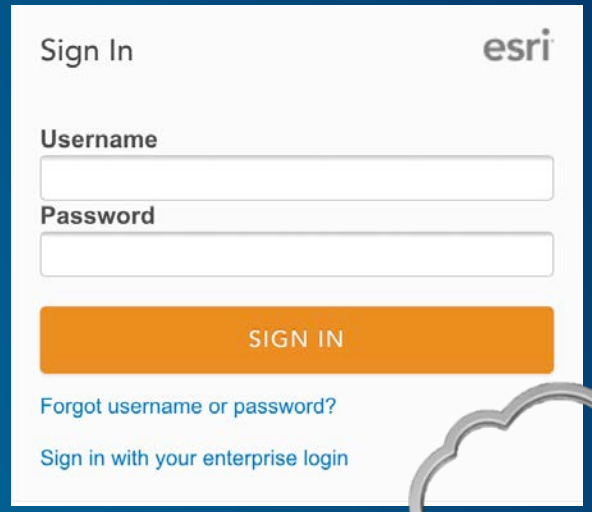
Responding to Authentication Challenges

- **Custom Handler**
  - **Get challenge from Authentication Manager**
  - **Implement customized UI**
    - **Call** continueWithCredential() **or** cancel()

```
AGSuthenticationManager.setAuthenticationChallengeHandler { (challenge) -> Void in
    //Display UI and get credentials from user
    challenge.continueWithCredential(...)
    //Or, cancel the challenge to abort loading resource
    challenge.cancel()
}
```

Reducing authentication challenges

- In-memory credential cache
  - To reuse credentials when accessing different resources from the same security realm



- Able to automatically persist credential cache to device keychain(ios only)
  - To remember between app launches
  - share between apps

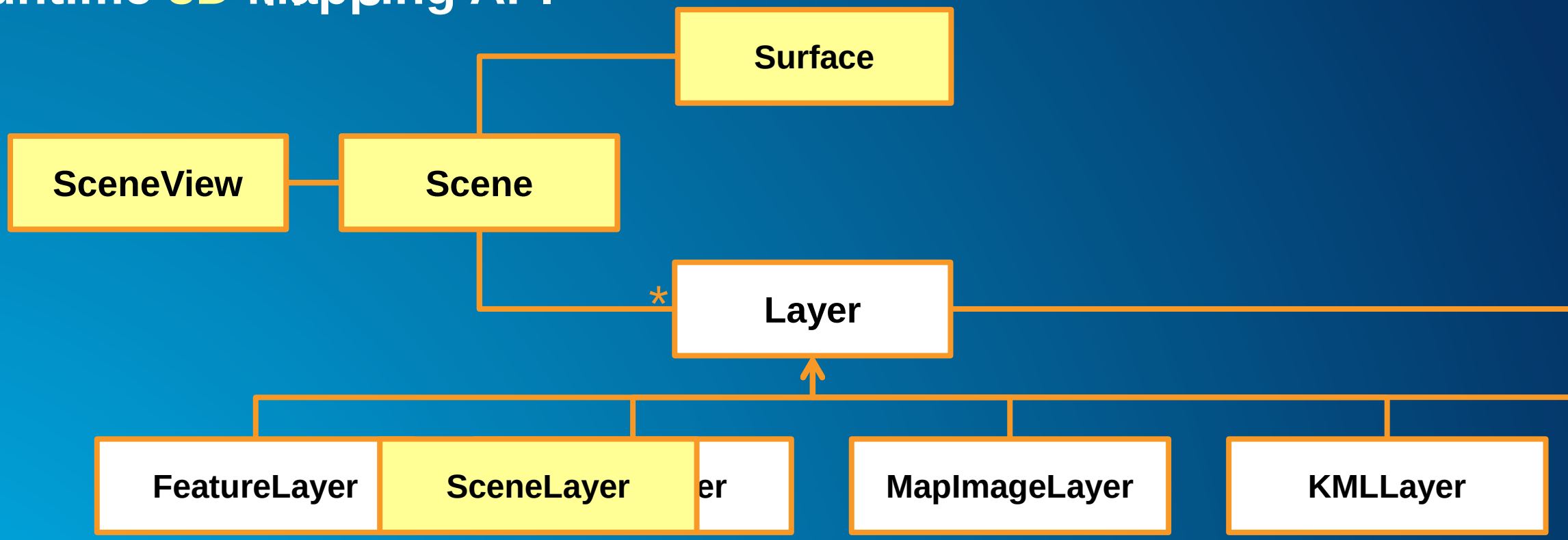
3D API

3D - Scenes

- Scenes are like maps, but for 3D
- Scenes created and published with ArcGIS Pro or ArcGIS Online
- Scenes can be viewed on the desktop, web or runtime apps

- 10.2.6 .Net SDK released the Quartz 3D API

Runtime 2D Mapping API



esri®