



# ArcGIS Runtime SDK for Java: Building Apps

Mark Baird

Jen Merritt

Simone Claridge

2021 ESRI  
DEVELOPER SUMMIT

# Let's make a map-based desktop application!

- **Getting started with 100.10**
- **Java 11 / JavaFX**
- **API Keys for ArcGIS Online resources**
- **Base maps, layers and loadable pattern**
- **Graphics overlays**
- **Offline maps**
- **Licensing and deployment**

# Getting started with 100.10

The background features a vibrant, abstract composition. On the left side, there are several overlapping shapes: a blue globe-like form, a red and yellow curved band, and a blue shape with a white grid pattern. The bottom half of the image is filled with a complex network of red, blue, and yellow lines and shapes, suggesting a data visualization or a network diagram. The overall color palette is dominated by deep blues, bright reds, and vibrant yellows, set against a dark blue gradient background.

# Developers website : developers.arcgis.com

</> ArcGIS Developers Documentation Features Pricing Support

🔍 Search  Sign In

## APIs, Tools, and Location Services

Making it easy to build mapping apps and solutions

Start building for free >





# SDK Installation

- **SDK is available in 2 ways:**
  - Zip / tgz file with all required files
  - Files will be available in a Maven repository
    - Use Gradle scripting
    - Use Maven scripting



***Maven***<sup>TM</sup>

# Java 11, but moving on to 17 soon!

- **100.4 works with Java 8 and 11**
- **Subsequent releases for Java 11+ only**
- **You can choose your JRE**
  - **AdoptOpenJDK**
  - **OpenJDK**
  - **Oracle JDK**



# API Keys

# API Keys

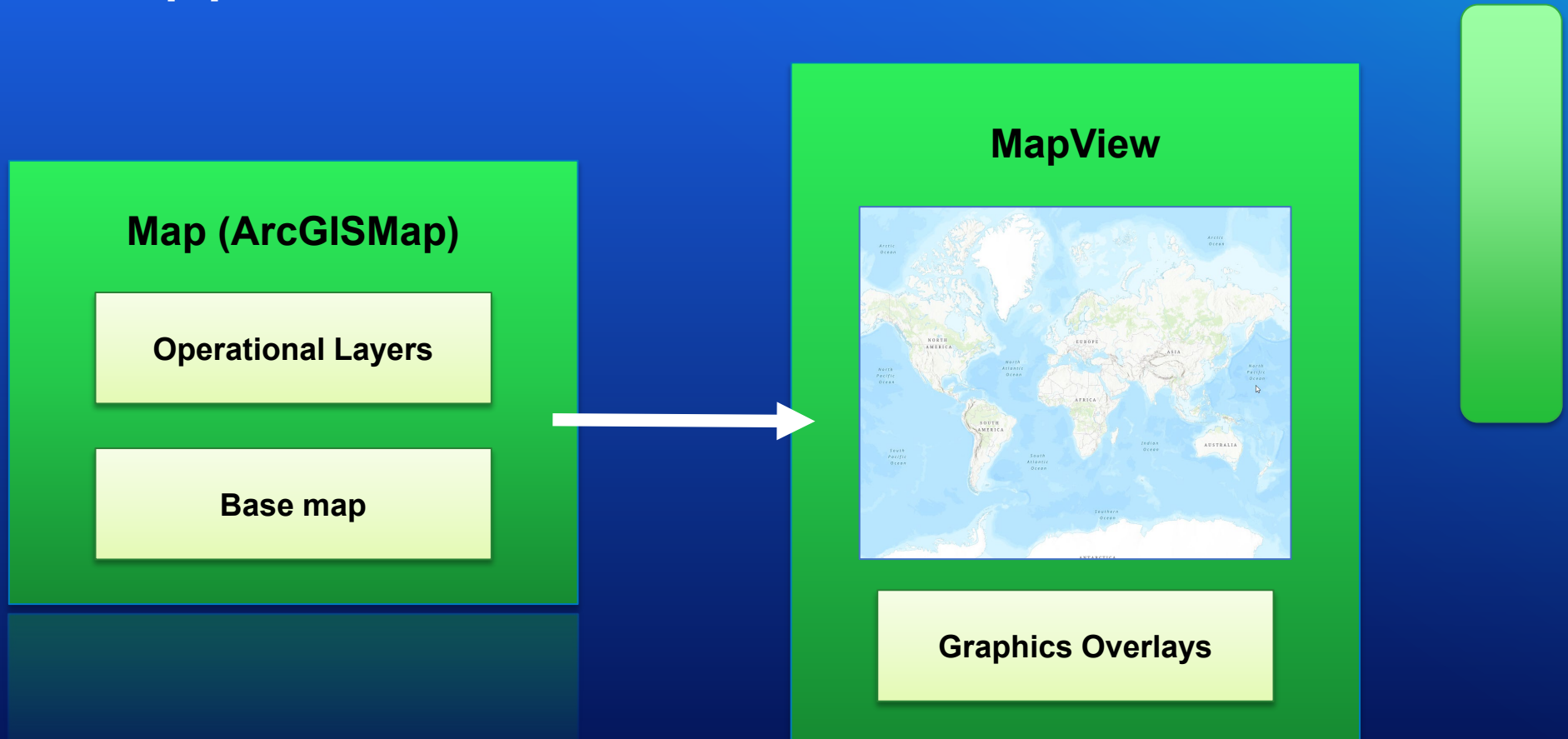
- **For authenticating an app to access ArcGIS Platform services**
  - Basemaps
  - Geocoding services
  - Routing services
- **Not needed for your own data or services**
- **Available for Runtime application use and other APIs**
  - JavaScript web apps
  - Leaflet, MapBox GL JS or OpenLayers
- **Created through developers site dashboard**



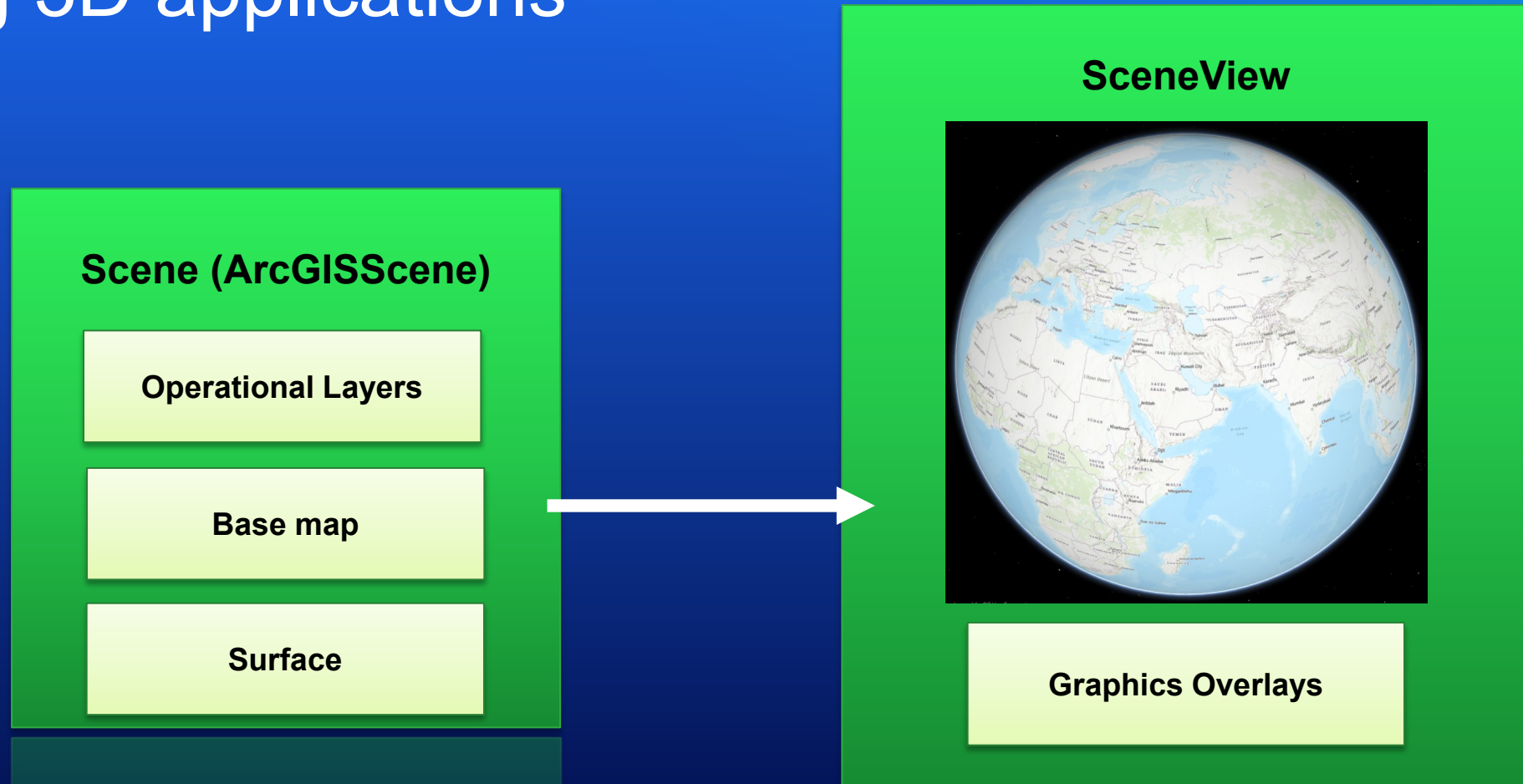
The background features a vibrant, abstract composition. On the left, there are overlapping layers of a map, with a prominent red and yellow shape. Below this, a globe is partially visible. The bottom half of the image is filled with dynamic, flowing shapes in shades of blue, red, and yellow, creating a sense of movement and depth. The overall aesthetic is modern and digital.

# Base maps, layers and loadable pattern

# Writing 2D applications



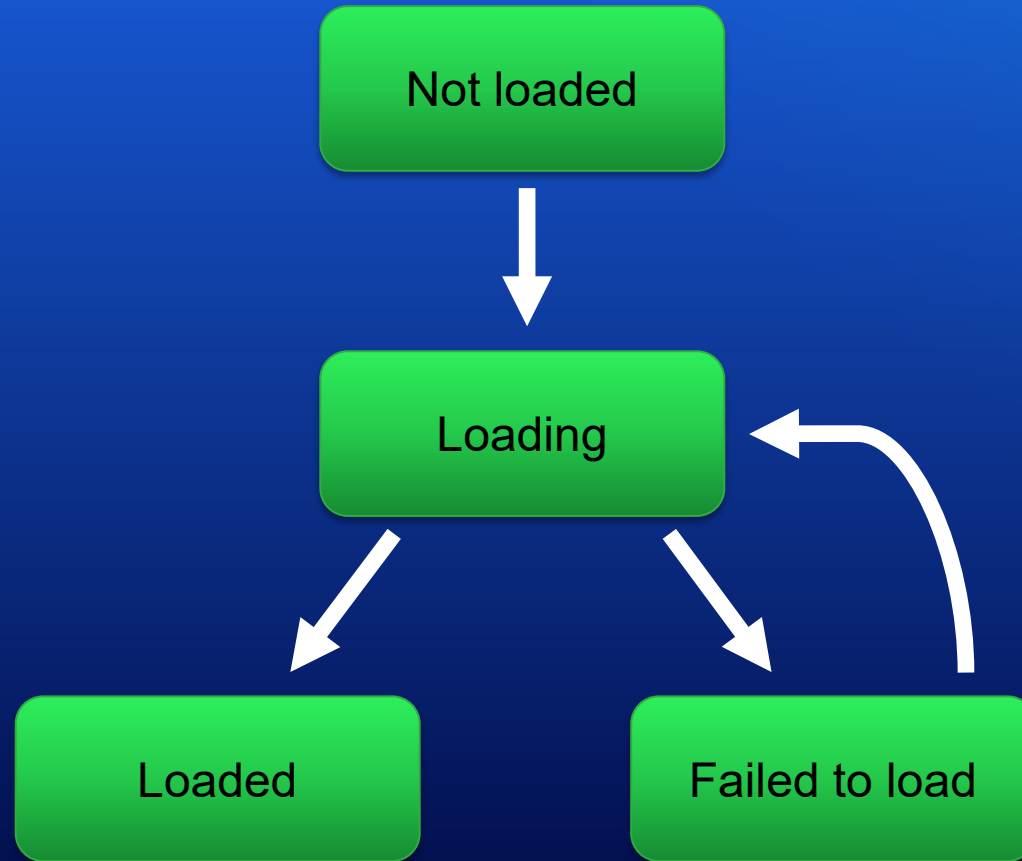
# Writing 3D applications



# Operational layers

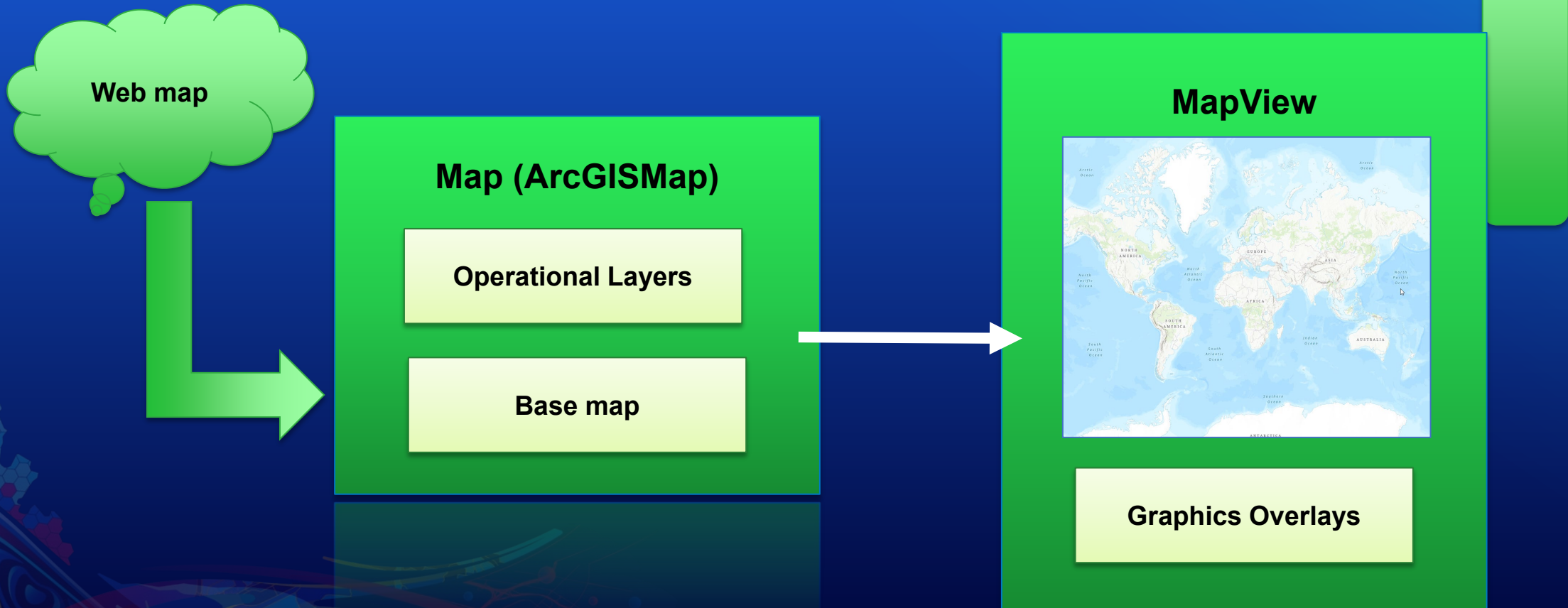
- **Your business data**
  - **Points, lines, polygons, text attributes, file attachments**
- **Local file formats**
  - **GeoPackages**
  - **Shapefiles**
  - **Raster data**
- **Server sources**
  - **ArcGIS Online**
  - **ArcGIS Server**
  - **OGC data (WFS etc)**

# Loadable Pattern





# Webmaps

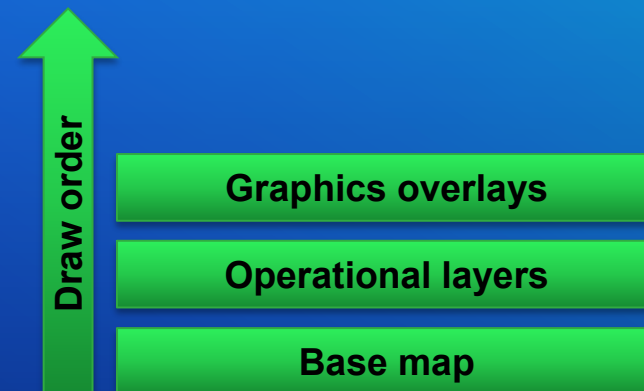


# Graphics Overlays

The background features a vibrant, abstract composition of overlapping shapes and lines. On the left side, there are prominent curved forms in shades of blue, red, and yellow. These elements are layered, creating a sense of depth and movement. The overall aesthetic is modern and digital, with a focus on bold colors and geometric patterns. The right side of the image is dominated by a smooth, light blue gradient that transitions from a deeper blue at the top to a lighter, almost white-blue at the bottom.

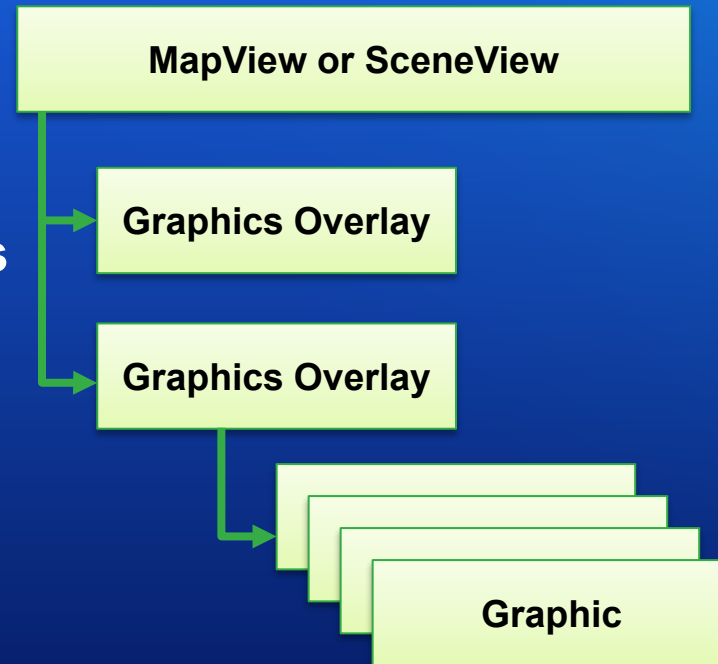
# Graphics overlays

- Used to display temporary or fast updating items
- Can be used to display point, lines or polygons
- Can be rendered in the same way as feature layers
  - Symbols, Unique value renderers, etc
- Graphics overlays are part of the MapView or SceneView
- Not persisted



# Graphics Overlays

- Found MapViews or SceneViews
- Can have multiple Graphics Overlays
- Graphics overlay is a container for Graphics
- Graphic is defined by:
  - Geometry (Point, line or polygon)
  - Attributes (optional)
  - Symbol
    - Specified for each graphic
    - Defined from a renderer on
- Very good for rapidly updating pictures





# Demo: Graphics Overlays

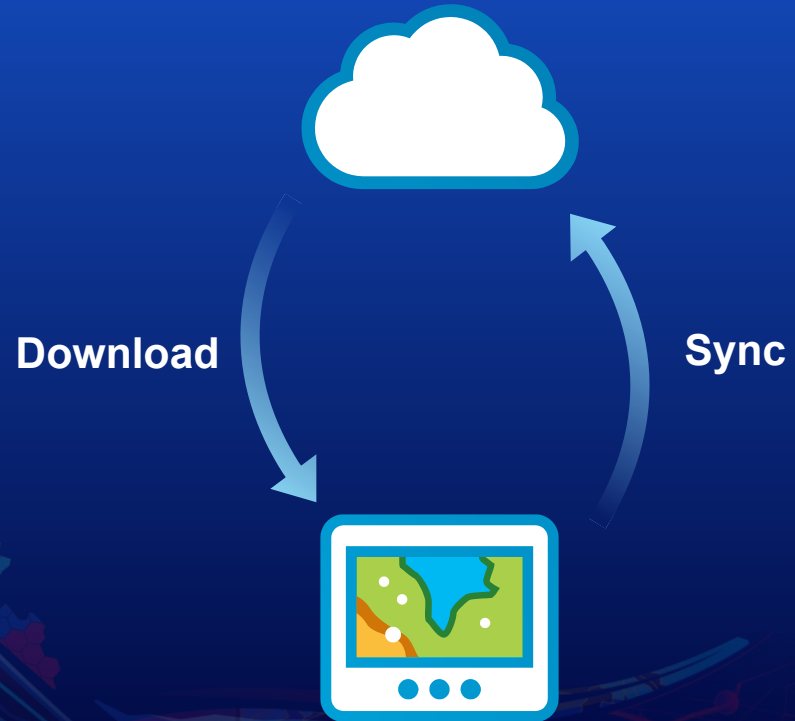


# Offline Data

The background features a vibrant, abstract composition. On the left side, there are several overlapping elements: a stylized globe with a grid pattern, a cluster of blue and green hexagons, and various flowing, ribbon-like shapes in red, yellow, and blue. The right side of the image is dominated by a smooth, horizontal gradient from a deep blue on the left to a bright cyan on the right. The overall aesthetic is modern and digital.

# Offline

## Services Pattern



## Desktop Pattern





# Licensing and deployment

# License your application

- Choose a license level:

- Lite
- Basic
- Standard
- Advanced

- See <https://developers.arcgis.com/java/license-and-deployment/license/>

- Choose license method

- Named user
- License code



# License code

- Use the `setLicense` method before your app uses ArcGIS functionality

```
public static void main(String[] args) {  
    // set license  
    ArcGISRuntimeEnvironment.setLicense("runtimelite,1000,ru  
  
    // launch application  
    Application.launch(args);  
}
```



# Named user licensing

- Log into your portal to get the license key

```
// connect to ArcSIS Online or a portal with a named user  
// Note this is very insecure code and you should consider using  
// something like oAuth instead for example!  
UserCredential credential =  
    new UserCredential( username: "username", password: "password");  
  
// connect to the portal with my credential  
portal = new Portal( portalUrl: "https://your-org.maps.arcgis.com/");  
portal.setCredential(credential);  
  
// load the portal  
portal.loadAsync();
```

# Named user licensing

- Log into your portal to get the license key

```
// connect to ArcSIS Online or a portal with a named user  
// Note this is very insecure code and you should consider using  
// something like OAuth instead for example!  
UserCredential credential =  
    new UserCredential( username: "username", password: "password");  
  
// connect to the portal with my credential  
portal = new Portal( portalUrl: "https://your-org.maps.arcgis.com/");  
portal.setCredential(credential);  
  
// load the portal  
portal.loadAsync();
```

# Named user licensing

- Log into your portal to get the license key

```
// connect to ArcSIS Online or a portal with a named user  
// Note this is very insecure code and you should consider using  
// something like OAuth instead for example!  
UserCredential credential =  
    new UserCredential( username: "username", password: "password");  
  
// connect to the portal with my credential  
portal = new Portal( portalUrl: "https://your-org.maps.arcgis.com/");  
portal.setCredential(credential);  
  
{ // load the portal  
  portal.loadAsync();  
}
```

# Named user licensing (part 2)

```
// listen for the portal loaded event on a new thread
portal.addDoneLoadingListener(()-> {
    // check if the load status was successful
    if (portal.getLoadStatus() == LoadStatus.LOADED) {

        //get the license info from the server
        ListenableFuture<LicenseInfo> licenseInfoFuture = portal.fetchLicenseInfoAsync();

        licenseInfoFuture.addDoneListener(() -> {
            try {
                // get the license info
                licenseInfo = licenseInfoFuture.get();

                // apply the license to the app
                ArcGISRuntimeEnvironment.setLicense(licenseInfo);

                // save for offline use later
                savedLicenseJSON = licenseInfo.toJson();
            }
        });
    }
});
```

# Named user licensing (part 2)

```
// listen for the portal loaded event on a new thread
portal.addDoneLoadingListener(()-> {
    // check if the load status was successful
    if (portal.getLoadStatus() == LoadStatus.LOADED) {

        //get the license info from the server
        ListenableFuture<LicenseInfo> licenseInfoFuture = portal.fetchLicenseInfoAsync();

        licenseInfoFuture.addDoneListener(() -> {
            try {
                // get the license info
                licenseInfo = licenseInfoFuture.get();

                // apply the license to the app
                ArcGISRuntimeEnvironment.setLicense(licenseInfo);

                // save for offline use later
                savedLicenseJSON = licenseInfo.toJson();
            }
        });
    }
});
```



# Named user licensing (part 2)

```
// listen for the portal loaded event on a new thread
portal.addDoneLoadingListener(()-> {
    // check if the load status was successful
    if (portal.getLoadStatus() == LoadStatus.LOADED) {

        //get the license info from the server
        ListenableFuture<LicenseInfo> licenseInfoFuture = portal.fetchLicenseInfoAsync();

        licenseInfoFuture.addDoneListener(() -> {
            try {
                // get the license info
                licenseInfo = licenseInfoFuture.get();

                // apply the license to the app
                ArcGISRuntimeEnvironment.setLicense(licenseInfo);

                // save for offline use later
                savedLicenseJSON = licenseInfo.toJson();
            }
        });
    }
});
```

# Named user licensing (part 2)

```
// listen for the portal loaded event on a new thread
portal.addDoneLoadingListener(()-> {
    // check if the load status was successful
    if (portal.getLoadStatus() == LoadStatus.LOADED) {

        //get the license info from the server
        ListenableFuture<LicenseInfo> licenseInfoFuture = portal.fetchLicenseInfoAsync();

        licenseInfoFuture.addDoneListener(() -> {
            try {
                // get the license info
                licenseInfo = licenseInfoFuture.get();

                // apply the license to the app
                ArcGISRuntimeEnvironment.setLicense(licenseInfo);

                // save for offline use later
                savedLicenseJSON = licenseInfo.toJson();
```

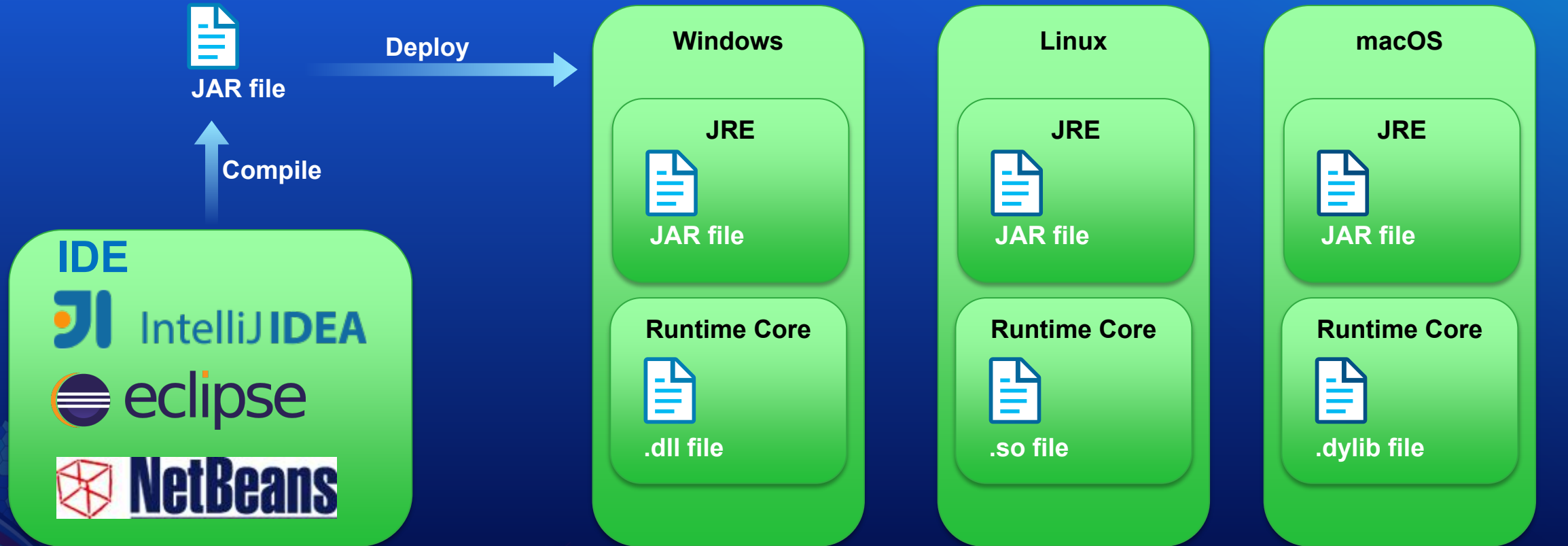


# Named user licensing (Offline)



```
// create a licenseInfo from the saved licenseJSON  
LicenseInfo licenseInfo = LicenseInfo.fromJson(savedLicenseJSON);  
  
// apply the license to the app  
ArcGISRuntimeEnvironment.setLicense(licenseInfo);
```

# Java Runtime development and deployment



Development environment

Cross platform deployments

# Demo: Deployment

The background features a vibrant, abstract digital graphic. It consists of various geometric shapes, including circles, lines, and polygons, in shades of blue, red, and yellow. The overall aesthetic is modern and tech-oriented, with a focus on clean lines and bright colors. The text 'Demo: Deployment' is centered in a bold, white, sans-serif font.

# Summary

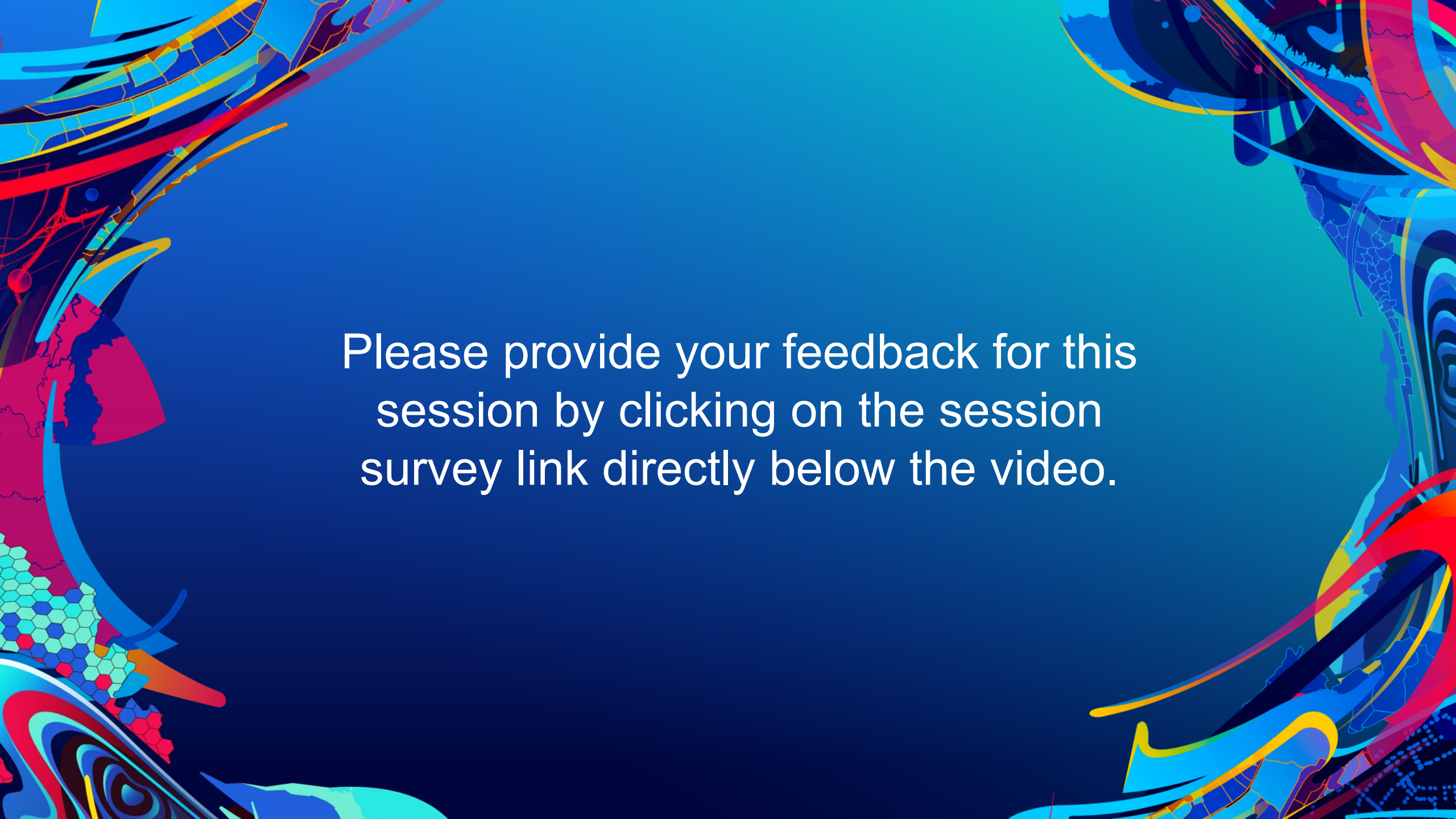
- **Cross platform Java 11 / Java FX desktop apps**
- **Key classes and techniques for mapping apps**
- **Graphics overlays**
- **Offline maps**
- **License and deployment**





esri®

THE  
SCIENCE  
OF  
WHERE®



Please provide your feedback for this session by clicking on the session survey link directly below the video.