THE JOURNEY
“Progressive web app” (PWA) is both a general term for a new philosophy toward building websites and a specific term with an established set of three explicit, testable, baseline requirements.
Baseline Criteria

• It must run under HTTPS.
• It must include a Web App Manifest.
• It must implement a Service Worker.
Baseline Criteria

A PWA should be striving to satisfy the following set of attributes:
- Responsive
- Connectivity independent
- App-like-interactions
- Safe
- Discoverable
- Re-engageable
- Installable
- Linkable
Why should you build one?

• You're a JavaScript developer but not a native developer
  - Offline capabilities (business apps)
  - I want something in the store! (consumer apps)
• A PWA is the answer to a lot of customer questions in one solution
  - One app/codebase for all platforms
  - Including a website
• Easy updates
• Our JavaScript API is one of the most function rich API's we have
The 3 main components of a PWA

- The JS/HTML code
- App Manifest
- Service Workers
PWA capabilities

- Offline
- Push
- Location
- Camera

A website can do this too, but can’t be put in the App Stores
Step 1: Developing an application

- Small Footprint
- Device Toolbar
- Remote Debugging
- Sensors (Geolocation)
- Throttling
- ...
- ...
- Basically like any other webapp
Step 2: Adding manifest and service worker

- **Manifest.json**
  - Title
  - ColorsScheme
  - Icons
  - Use [https://app-manifest.firebaseapp.com/](https://app-manifest.firebaseapp.com/) to create one

- **Service Worker**
  - Tasks runners of a web application
  - Caching data for offline use
  - Other background tasks
Step 2: Adding manifest

```json
{
  "short_name": "PicoV2019092701",
  "name": "EsriNLDevTeamPico",
  "icons": [
    {
      "src": "images/icon_72.png",
      "type": "image/png",
      "sizes": "72x72"
    },
    {
      "src": "images/icon_96.png",
      "type": "image/png",
      "sizes": "96x96"
    },
    {
      "src": "images/icon_128.png",
      "type": "image/png",
      "sizes": "128x128"
    }
  ],
  "start_url": "index.html",
  "background_color": "#FF6300",
  "display": "standalone",
  "scope": "../",
  "theme_color": "#FF6300"
}
```
Step 2: Adding serviceworker

```javascript
const cacheName = 'v1';
const cacheAssets = [
  'index.html',
  'main.js',
  'esri.js'
];

// Call Install Event
self.addEventListener('install', e => {
  console.log('Service Worker: Installed');
  e.waitUntil(
    caches
      .open(cacheName)
      .then(cache => {
        console.log('Service Worker: Caching Files');
        cache.addAll(cacheAssets);
      })
      .then(() => self.skipWaiting())
  );
});

// Call Activate Event
self.addEventListener('activate', e => {
  console.log('Service Worker: Activated');
  // Remove unwanted caches
  e.waitUntil(
    caches.keys().then(cacheNames => {
      return Promise.all(
        cacheNames.map(cache => {
          if (cache !== cacheName) {
            console.log('Service Worker: Clearing Old Cache');
            console.log(cacheName + ' cache is removed');
          } else {
            return caches.delete(cache);
          }
        })
      );
    })
  );
});
```
Step 4: Getting the app to the stores

Enter URL and go!

… isch
Step 3: creating something to submit

- .apk
- .app
- .appx
Step 4: Getting the app to the stores

Available on the App Store

Android App on Google Play

Windows Store
"Turning a web app into a Progressive Web App (PWA) and submitting it to 3 app stores requires about a month of work, a few hundred dollars, and lots of red tape."

PWA Builder was founded by Microsoft as a community guided, open source project to help move PWA adoption forward.

Everything you need to make your PWA

If you haven’t already, download the content below and publish it to your website. Making these changes to your website is all you need to become a PWA. You may also want to publish your PWAs to the different app markets, you will find the packages for each of these on the right.

**Windows**

You’ll get a side-loadable version of your PWA (requires Win10 in dev mode) to test your PWA right away. To generate an AppX PWA package and submit to the Microsoft Store, click here.

**Samsung**

PWAs are available through the browser on Samsung devices, however your PWA can also be submitted to the Galaxy store by submitting the package you get here.

**Android**

PWAs are available through the browser on Android, however your PWA can also be submitted to the play store by submitting the package you get below.

**iOS**

PWAs are available through the browser. You can submit this app package to the iOS App Store.

**MacOS**

You can use Xcode to build this package to produce an app that runs on MacOS.
Google Play Store

- Android Studio vs Command Line Tools
- Build your app from the command line
  - Build a release APK
  - Build an app bundle
  - Sign your app from command line
Notes:
  - for us JDK 12 and JDK 11 didn’t work, we used JDK 8
  - align twice using “zipalign”

Google Play Store

- App review
- App testing
- App approval
Apple App Store

- You need a Mac (sigh…)
- Apple documentation (sigh…)
- The Apple App Store (sigh…)
- The Apple app business model (sigh…)
- Apple (sigh…)

THIS IS APPLE !!!
The Xcode experience….

- Select Generic iOS Device
- Product -> Archive
- you are not registered as an apple developer……..
- But we are!
- Try it with a different account
- you are not registered as an apple developer……..
- But we are!
- OK, time to go get some coffee……..
The Xcode experience…

- Oh, now it works!
- Upload app to app store
- No certificate
- Register certificate
- Upload app to app store
- No suitable application records were found. Verify your bundle identifier
- Register app id on App Store Connect (browser)
- First app id was not allowed anymore
- Registered another app id .2
The Xcode experience....

• Wait, what?!
The Xcode experience....

- Update xcode
- Version MacOS is not compatible with version Xcode, grrrrrr
- Update MacOS
- ... ... ... ... ... ... ...
The Xcode experience....

- Validate app....
- Missing provisioning profile
- Change build system to “legacy”
The Xcode experience....

- Validate app....
- Missing provisioning profile
- Change build system to “legacy”
Validating archive:

Validating PicoV2019092701.ipa...
App "PicoV2019092701" successfully validated. Your app successfully passed all validation checks.
Select a method of distribution:

- **App Store Connect**
  Distribute on TestFlight and the App Store.
- **Ad Hoc**
  Install on designated devices.
- **Enterprise**
  Distribute to your organization.
- **Development**
  Distribute to members of your team.
App Store Connect distribution options:

- Upload your app's symbols to receive symbolicated reports from Apple
  Crash logs and other diagnostic information from your customers will be symbolicated and viewable within Xcode.

  Cancel

Previous  Next
Re-sign "PicoV2019092701":

"PicoV2019092701" needs to be re-signed for App Store Connect distribution. Select one of the following signing options to continue.

- Automatically manage signing
  Xcode will create and update profiles, app IDs, and certificates.

- Manually manage signing
  Select certificates and profiles from your team.

Cancel
Previous
Next
The Xcode experience….
The Xcode experience….
Distribution completed with warnings:

App Store Connect Operation Warning

WARNING ITMS-90704: "Missing App Store Icon. iOS Apps must include a 1024x1024px App Store Icon in PNG format. Without providing the icon in the Asset Catalog or via App Store Connect, apps cannot be submitted for App Review or Beta App Review. Refer to https://developer.apple.com/ios/human-interface-guidelines/icons-and-images/app-icon/ for more information."
<table>
<thead>
<tr>
<th>Name</th>
<th>Creation Date</th>
<th>Version</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>PicoV2019092701</td>
<td>4 Oct 2019 at 16:01</td>
<td>1.0.1</td>
<td>Uploaded</td>
</tr>
<tr>
<td>PicoV2019092701</td>
<td>4 Oct 2019 at 15:59</td>
<td>1.0.0</td>
<td>Validation failed</td>
</tr>
<tr>
<td>PicoV2019092701</td>
<td>4 Oct 2019 at 16:55</td>
<td>1.0.0</td>
<td>Validation failed</td>
</tr>
<tr>
<td>PicoV2019092701</td>
<td>4 Oct 2019 at 14:41</td>
<td>1.0.0</td>
<td>Uploaded with warnings</td>
</tr>
<tr>
<td>PicoV2019092701</td>
<td>4 Oct 2019 at 14:37</td>
<td>1.0.0</td>
<td>Validation failed</td>
</tr>
<tr>
<td>PicoV2019092701</td>
<td>4 Oct 2019 at 12:00</td>
<td>1.0.0</td>
<td></td>
</tr>
<tr>
<td>Date</td>
<td>Version</td>
<td>Status</td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>---------</td>
<td>----------------------</td>
<td></td>
</tr>
<tr>
<td>at 16:01</td>
<td>1.0.1 (1.0.1)</td>
<td>Uploaded</td>
<td></td>
</tr>
<tr>
<td>at 15:59</td>
<td>1.0.0 (1.0.0)</td>
<td>Validation failed</td>
<td></td>
</tr>
<tr>
<td>at 15:55</td>
<td>1.0.0 (1.0.0)</td>
<td>Validation failed</td>
<td></td>
</tr>
<tr>
<td>at 14:41</td>
<td>1.0.0 (1.0.0)</td>
<td>Uploaded with warnings</td>
<td></td>
</tr>
<tr>
<td>at 14:37</td>
<td>1.0.0 (1.0.0)</td>
<td>Validation failed</td>
<td></td>
</tr>
<tr>
<td>at 12:00</td>
<td>1.0.0 (1.0.0)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
5 minutes later….

App Store Connect

Dear Developer,

We identified one or more issues with a recent delivery for your app, "Pico" 1.0.1 (1.0.1). Please correct the following issues, then upload again.

**ITMS-90713: Missing Info.plist value** - A value for the Info.plist key 'CFBundleIconName' is missing in the bundle 'nl.esri.devteam2'. Apps built with iOS 11 or later SDK must supply app icons in an asset catalog and must also provide a value for this Info.plist key. For more information see [http://help.apple.com/xcode/mac/current/#/dev10510b1f7](http://help.apple.com/xcode/mac/current/#/dev10510b1f7).

Best regards,

The App Store Team
*SIGH*

I'M DONE WITH THIS...
Microsoft Store

- Easy...
- Oh but why...
Considerations
When do you need a PWA?

- You’re a JavaScript developer
- Customer wants the App Stores
- You want one codebase for multiple platforms
Pro’s

• Build once, deploy often
• The app is automatically updated
• Runs on many platforms
Things to think about

- Offline maps vs Data updates
- Specific Apple hardware
- Your app depends on browser functionality
Performance considerations

• Use the standard ArcGIS JS API
  - Avoid using Web AppBuilder (loading times)

• Lazy loading
  - Pro: the app loads faster (initially)
    - ArcGIS JS API does that by default
    - For non-Dojo: https://github.com/Esri/esri-loader
  - Con: Offline functionality is limited

• Map-centric is not a requirement
  - ArcGIS has a lot of cool functionalities without a map

• JS Frameworks are not a requirement
  - can be useful
  - provide better UX/UI
  - https://github.com/tomwayson/esri-preact-pwa
Updates

• Versioning
• Cache
• Storage
Thank You

Questions?
Please Take Our Survey on the App

Download the Esri Events app and find your event

Select the session you attended

Scroll down to find the feedback section

Complete answers and select “Submit”