



First steps with Capacitor... in the real world

Horacio Gonzalez
@LostInBrittany

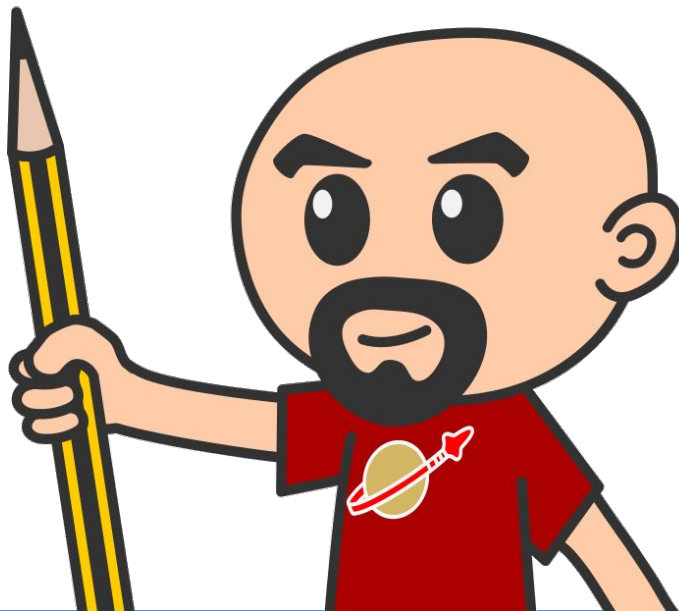


Horacio Gonzalez



@LostInBrittany

Spaniard lost in Brittany, developer,
dreamer and all-around geek



OVH : Key Figures



1.3M Customers worldwide in **138** Countries

1.5 Billions of investment over five years

+ 2 000 Employees

18 Years of Innovation

19 Countries

300k Dedicated Servers

200k Private cloud VMs running

500k Public cloud Instances created in a month

27 Datacenters

4 DC under construction

2 DC in project

13 Tbps bandwidth

33 Points of presence

4 Anti DDoS capacity

Hosting capacity : **1.3M** Physical Servers



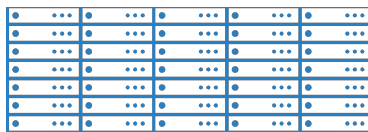
OVH: A Global Leader on Cloud



200k Private cloud
VMs running

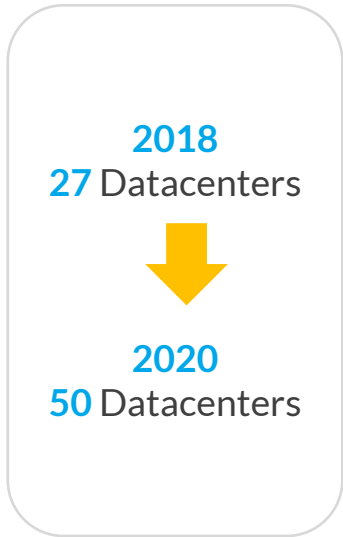
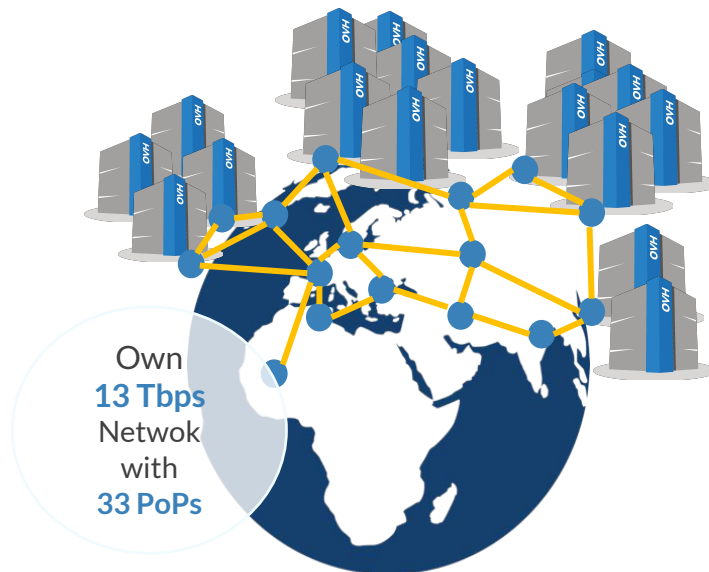


Dedicated IaaS
Europe



Hosting capacity :
1.3M Physical
Servers

360k
Servers already
deployed



> **1.3M** Customers in **138** Countries



First steps with Capacitor...
in the real world

@LostInBrittany



Ranking & Recognition



1st **European** Cloud Provider*

1st **Hosting** provider in Europe

1st **Provider** Microsoft Exchange

Certified vCloud Datacenter

Vmware **Global Service Provider** 2013-2016


Veeam Best Cloud Partner of the year (2018)

* Netcraft 2017 -




OVH: Our solutions




 **Cloud**


VPS
Public Cloud
Private Cloud
Serveur dédié
Cloud Desktop
Hybrid Cloud

 **Mobile Hosting**

Containers
Compute
Database
Object Storage
Securities
Messaging

 **Web Hosting**

Domain names
Email
CDN
Web hosting
MS Office
MS solutions

 **Telecom**

VoIP
SMS/Fax
Virtual desktop
Cloud HubiC
Over theBox





Capacitor

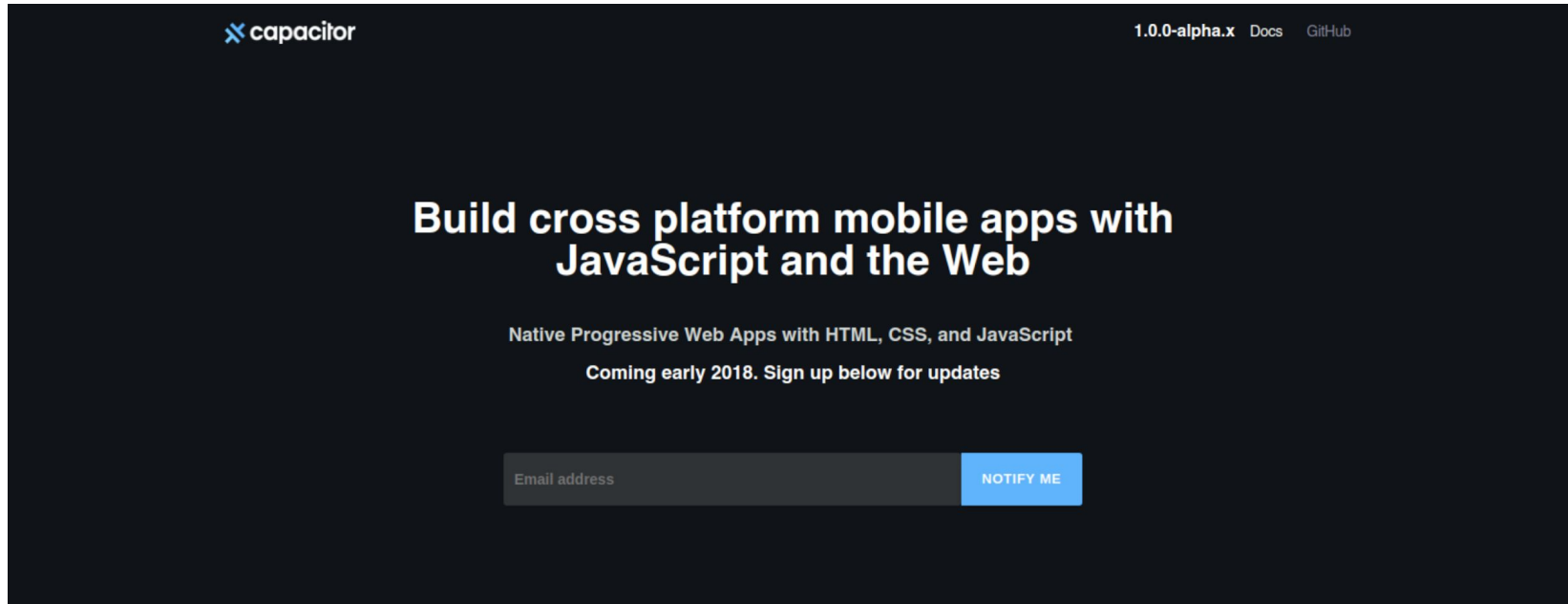
The next generation Cordova



What's Capacitor?



Cross-platform app runtime making it easy to build web apps that run natively on iOS, Android and web



Spiritual heir to Apache Cordova



Evolution, not revolution



Spiritual heir to Apache Cordova



Support for many Cordova plugins



Extensible and evolutif



- Close to web-standards
- Plugin API
 - Swift on iOS
 - Java on Android
 - JavaScript for the web.



Developer Friendly



Easy to get started
Works on any framework



You still need the platform tools



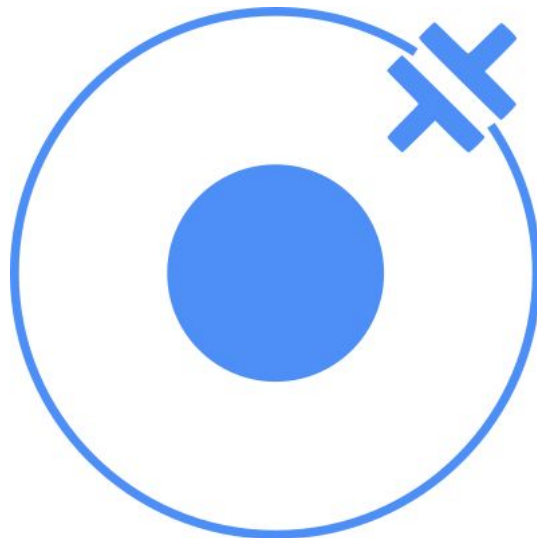
Android Studio and/or Xcode
to build the native packages





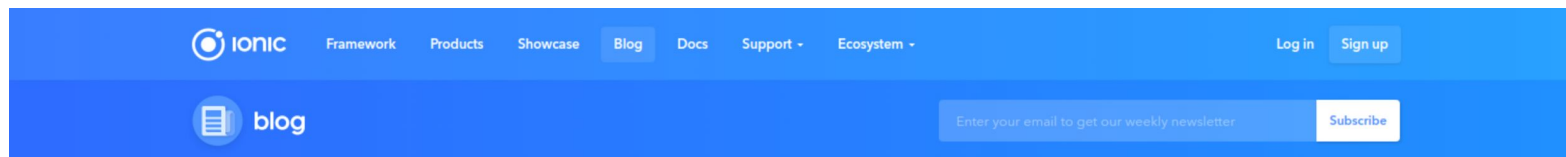
First steps testing Capacitor

Capacitor in an Ionic app





To replace Cordova in next Ionic



Announcing Capacitor 1.0.0 Alpha



By Max on February 27, 2018

INTRODUCING



Today we are incredibly excited to announce the alpha release of a major new open source project: [Capacitor](#).



Official example built on Ionic



Showcasing the upcoming Ionic 4



Let's try the official example



This repository Search Pull requests Issues Marketplace Explore

ionic-team / capacitor Watch 90 Star 964 Fork 61

<> Code Issues 75 Pull requests 1 Insights

Branch: master capacitor / example / Create new file Upload files Find file History

README.md

Capacitor Example App

This example app can be used to develop and test Capacitor.

This project contains a modified Ionic app as the web app (source code in `src`, build in `www`), that demos the use of the Capacitor APIs (from `@capacitor/code`). The project is also already set up with two native projects, `ios` and `android`, that can be used to build and debug native apps for those platforms.

The installation instructions and native projects are set up in such a way, that `@capacitor/core` and the Capacitor Android and iOS libraries that are used in the native projects are loaded from the local (parent) directory, instead of as an external dependency through the normal distribution mechanism (npm, CocoaPods and Gradle/Bintray):

- `@capacitor/core = ../core`
- Capacitor iOS = `../ios`
- Capacitor Android = `../android`

This way you can make direct changes to all those and use them in the native apps, allowing quick iteration of development.



Using directly the Capacitor repository



Custom built project, not a production app

- @capacitor/core and native libs loaded from local directory



Let's build the Android version



1. Build Capacitor Core Module

Start by building the Capacitor Core Module in `/core` :

```
cd ../core  
  
npm install  
npm run build  
npm link
```

2. Build Example App

Switch back over to this example project in `/example` where you first install dependencies and link in the `@capacitor/core` you just built in the step before, then build the app and copy the build files to the correct `public` directories for both the iOS and Android example apps:

```
cd ../example  
  
npm install  
npm link @capacitor/core  
  
npm run build  
npm run copy
```

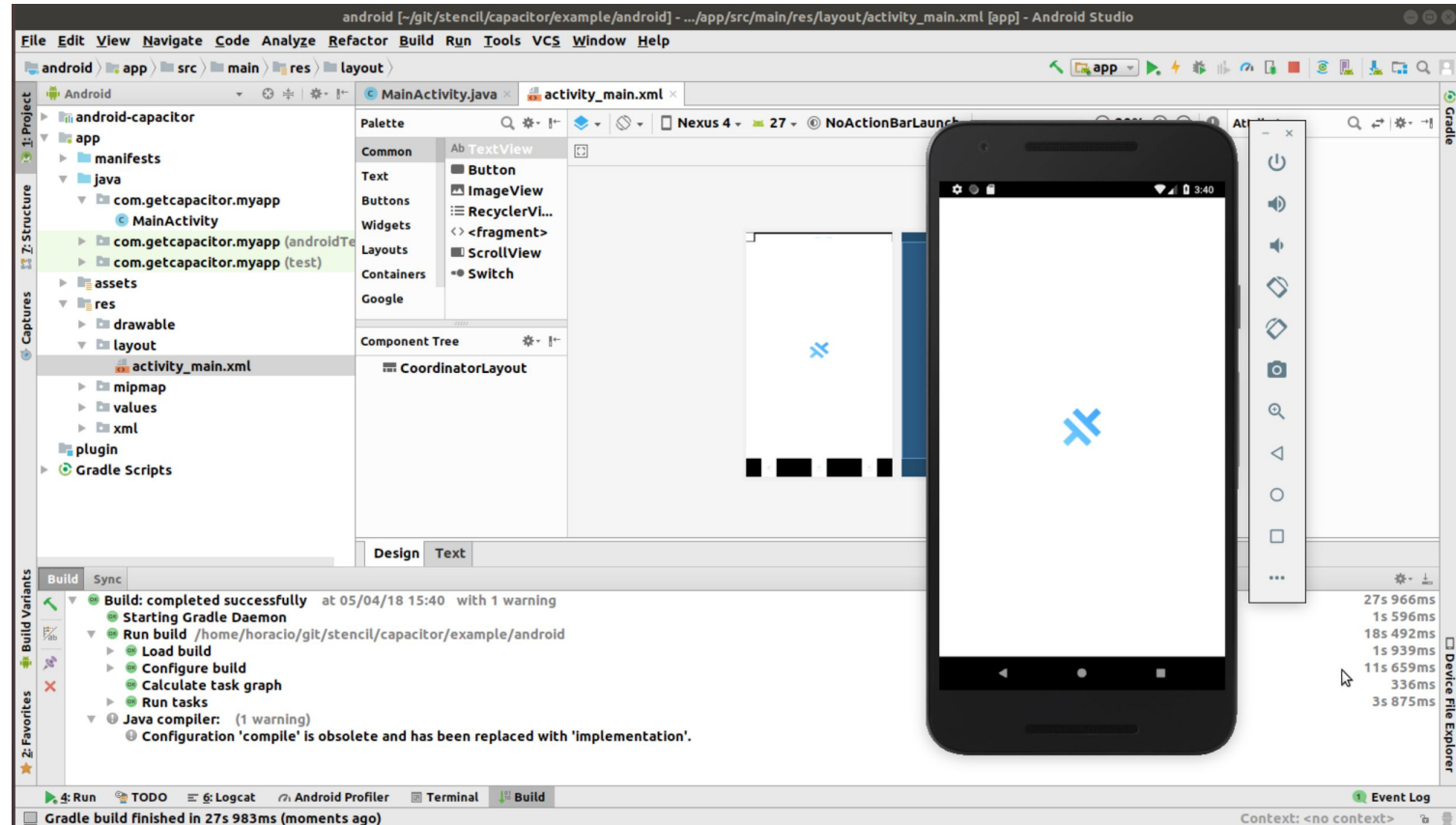
3. Build and run the native Capacitor Apps

Now that everything is in place you can build the native Capacitor Apps:

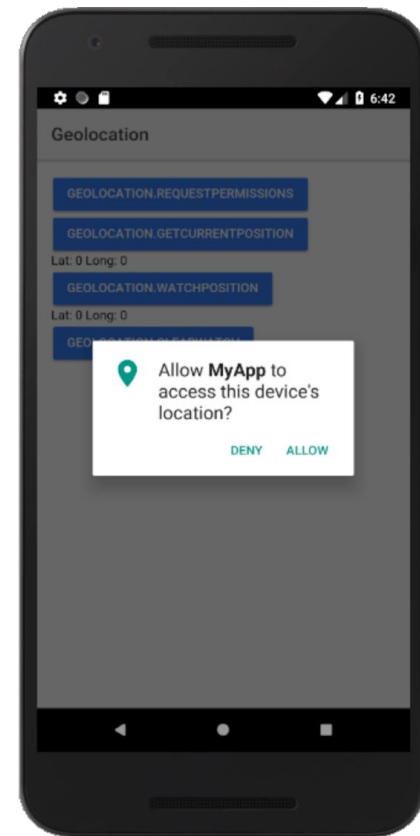
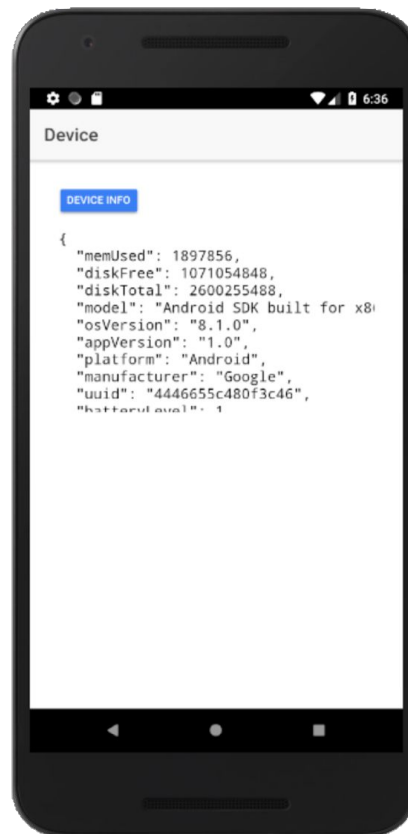
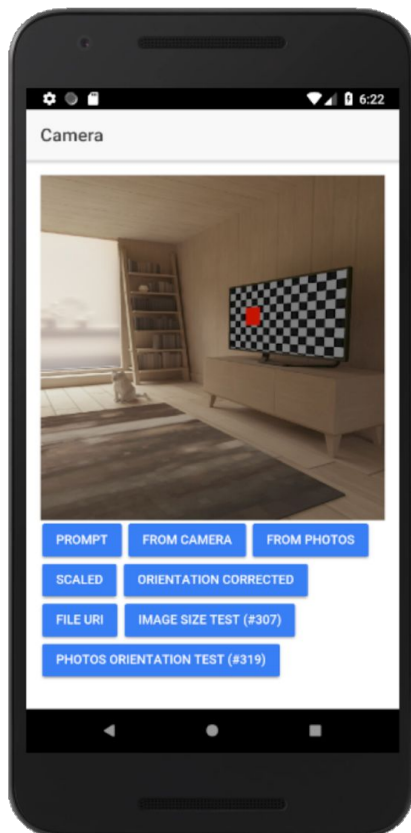
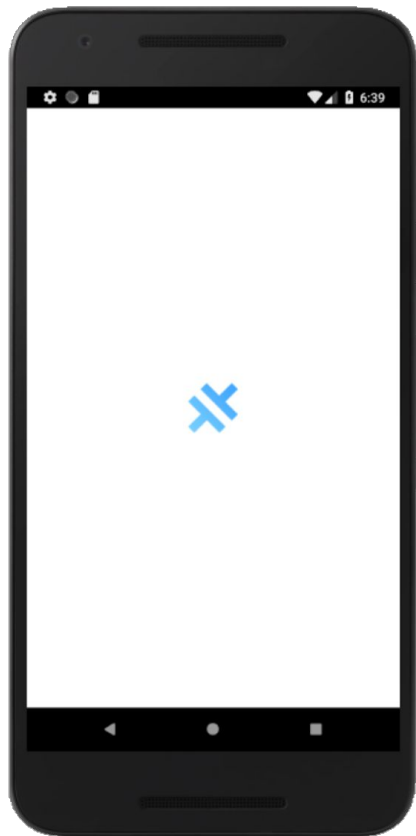
Some pre-building and building needed...



We now have an Android project



Easy and painless Android app



And in PWA mode?



Toast

SHOW

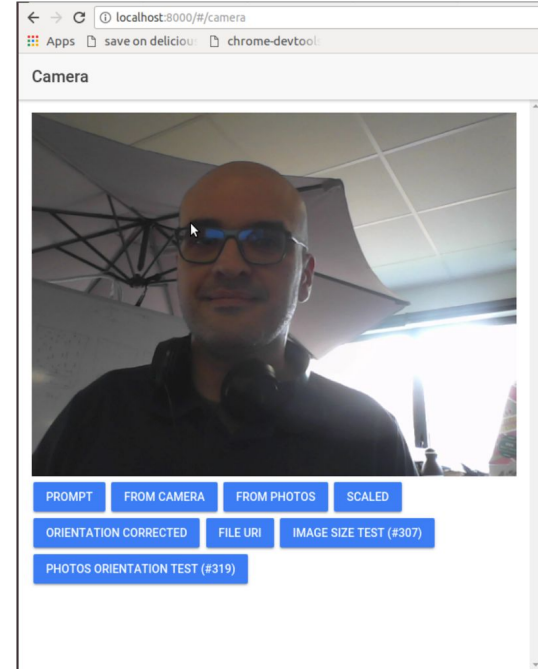
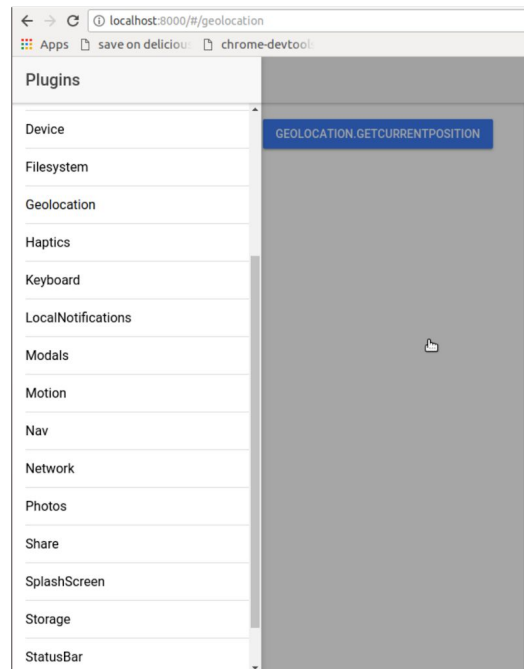
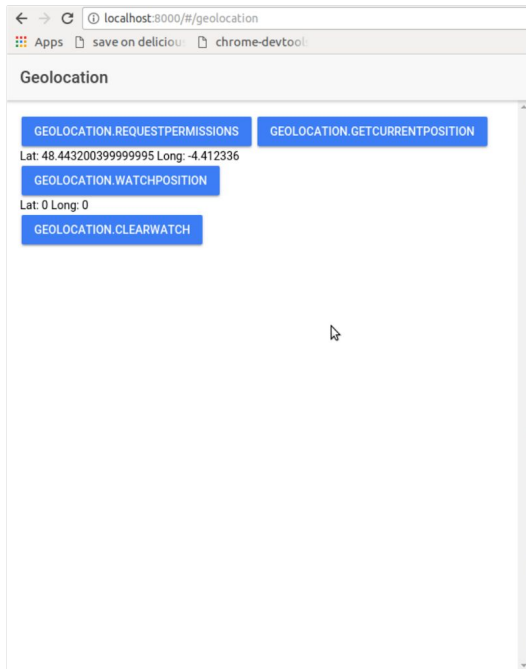
Angular is running in the development mode. Call enableProdMode() to enable the production mode. [core.js:3565](#)

- ▶ Error: Uncaught (in promise): SplashScreen does not have web implementation. [web-runtime.js:40](#)
 - at c (polyfills.js:3)
 - at Function.t.reject (polyfills.js:3)
 - at CapacitorWeb.webpackJsonp.248.CapacitorWeb.pluginMethodNoop (web-runtime.js:31)
 - at new MyApp (app.component.ts:40)
 - at createClass (core.js:12164)
 - at createDirectiveInstance (core.js:12011)
 - at createViewNodes (core.js:13449)
 - at createRootView (core.js:13339)
 - at callWithDebugContext (core.js:14740)
 - at Object.debugCreateRootView [as createRootView] (core.js:14041)
- ▶ Error: Uncaught (in promise): App does not have web implementation. [web-runtime.js:40](#)
 - at c (polyfills.js:3)
 - at Function.t.reject (polyfills.js:3)
 - at CapacitorWeb.webpackJsonp.248.CapacitorWeb.pluginMethodNoop (web-runtime.js:31)
 - at new MyApp (app.component.ts:47)
 - at createClass (core.js:12164)
 - at createDirectiveInstance (core.js:12011)
 - at createViewNodes (core.js:13449)
 - at createRootView (core.js:13339)
 - at callWithDebugContext (core.js:14740)
 - at Object.debugCreateRootView [as createRootView] (core.js:14041)
- ▶ Error: Uncaught (in promise): App does not have web implementation. [web-runtime.js:40](#)
 - at c (polyfills.js:3)
 - at Function.t.reject (polyfills.js:3)
 - at CapacitorWeb.webpackJsonp.248.CapacitorWeb.pluginMethodNoop (web-runtime.js:31)
 - at new MyApp (app.component.ts:51)
 - at createClass (core.js:12164)
 - at createDirectiveInstance (core.js:12011)
 - at createViewNodes (core.js:13449)
 - at createRootView (core.js:13339)
 - at callWithDebugContext (core.js:14740)
 - at Object.debugCreateRootView [as createRootView] (core.js:14041)

Some elements haven't web implementation (yet?)



But it still works!



It fails gracefully for unsupported plugins



First test: successful!



Capacitor 1 - Scepticism 0





Let's try something harder

stencil & capacitor



Capacitor without Ionic



I want to use it Capacitor
with my own toolset



I'm a Web Components guy



And I speak a lot about Polymer



But Polymer is in a transition phase



Polymer 2.x : bower based
Upcoming Polymer 3: npm based



So what to use?



Stencil, of course!

The magical, reusable web component compiler



Let's begin with a simple example



☰ ✕ ☹ ☰ stencil & capacitor

Take a pic



A Camera app, working well on web mode
Using standard Media Capture and Streams API



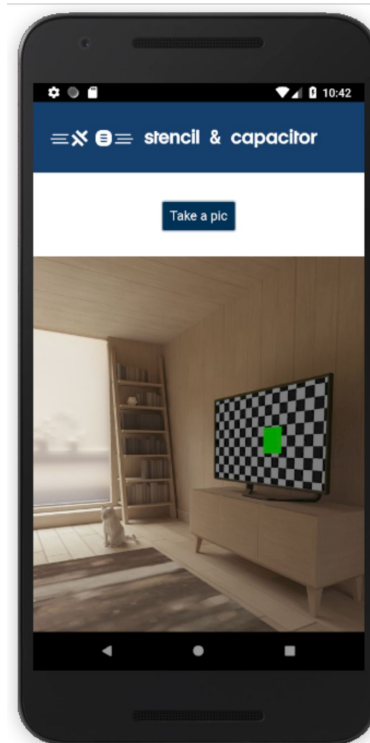
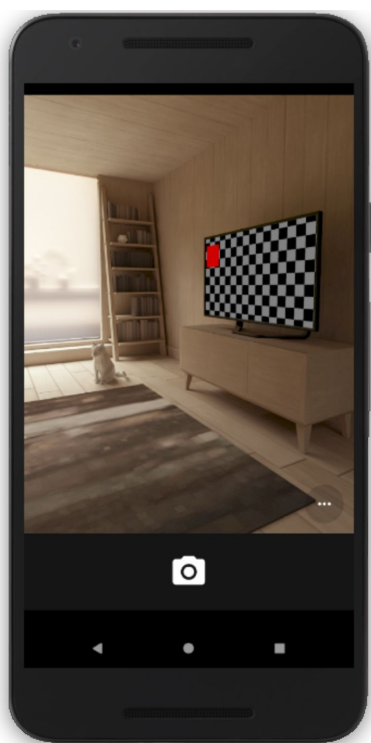
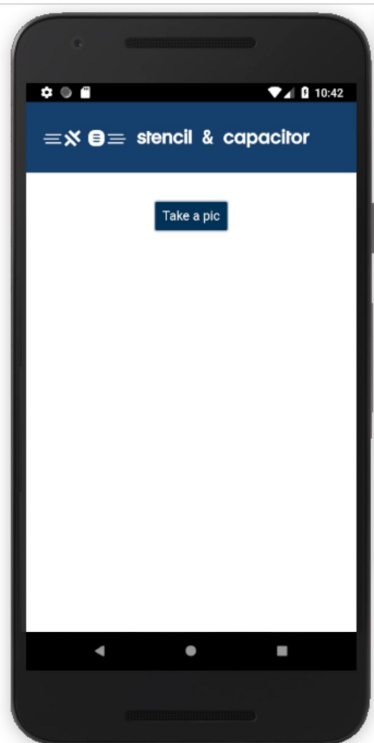
Let's charge it with Capacitor



We want the same behavior
But now in Android, iOS AND web



And does it work?



Yes it does... in native mode only!



But not in PWA mode :(



stencil & capacitor

Take a pic

```
takePicture cpctr-app.js:837
[{"label":0,"trys":[],"ops":[]} cpctr-app.js:409
ion-pwa-camera-modal cpctr-app.js:413
Uncaught (in promise) TypeError: cameraModal.componentOnReady is not a function cpctr-app.js:414
    at CameraPluginWeb.<anonymous> (cpctr-app.js:414)
    at step (cpctr-app.js:351)
    at Object.next (cpctr-app.js:332)
    at cpctr-app.js:325
    at new Promise (<anonymous>)
    at __awaiter (cpctr-app.js:321)
    at cpctr-app.js:406
    at new Promise (<anonymous>)
    at CameraPluginWeb.<anonymous> (cpctr-app.js:406)
    at step (cpctr-app.js:351)
```

And a big question: why?



Well, let's spot the differences...



Searching on the example code

hmmmm, @ionic/pwa-elements, what's that?





A non-documented private-repository w.i.p. Stencil-based project by Ionic team

The screenshot shows the npm package page for `@ionic/pwa-elements`. At the top, there is a red navigation bar with the text "Neovictorian Paisley Menswear" and links for "npm Enterprise", "Features", "Pricing", "Docs", and "Support". Below this is a search bar with the npm logo and the text "Search packages", and a "log in or sign up" link. A banner below the search bar says "Share your code. npm Orgs help your team discover, share, and reuse code. Create a free org »".

The package page for `@ionic/pwa-elements` is displayed. It is a public package with 2 dependencies, 0 dependents, and 11 versions. A "Built With Stencil" badge is visible. The main content area shows the package name, a "Readme" tab, and a description: "Stencil Component Starter". Below the description, it says "This is a starter project for building a standalone Web Component using Stencil. Stencil is also great for building entire apps. For that, use the `stencil-app-starter` instead."

On the right side, there is an "install" section with a code block: `> npm i @ionic/pwa-elements`. Below this is a "last 7 days" section with a bar chart showing 17 downloads. At the bottom right, there is a table with columns "version" and "license", showing "0.0.12" and "MIT" respectively.



Adding @ionic/pwa-elements



According to Stencil doc:

In a stencil-app-starter app

- Run `npm install my-name --save`
- Add `{ name: 'my-name' }` to your [collections](#)
- Then you can use the element anywhere in your template, JSX, html etc

But when trying to do that:

```
[ WARN ] As of v0.6.0, "config.collections" has been deprecated in favor of standard ES module imports. Instead of listing collections within the stencil config, collections should now be imported by the app's root component or module. The benefit of this is to not only simplify the config by using a standards approach for imports, but to also automatically import the collection's types to improve development. Please remove "config.collections" from the "stencil.config.js" file, and add this import to your root component or root module: import '@ionic/pwa-elements';
```



Adding @ionic/pwa-elements



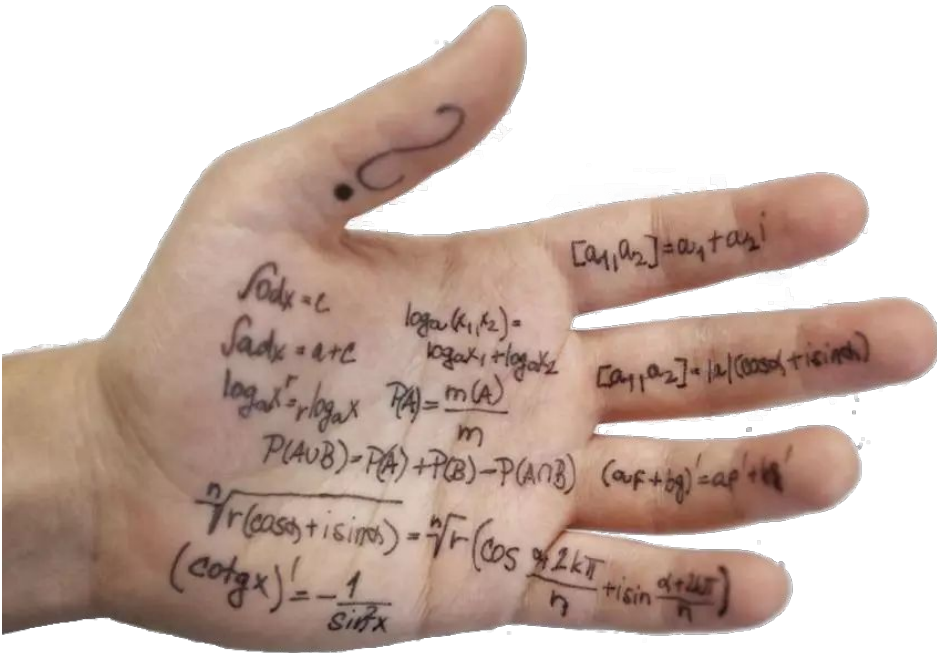
But when importing it in the main element

The screenshot shows a web application interface on the left and a browser developer console on the right. The application has a dark blue header with the text 'stencil & capacitor' and a 'Take a pic' button. The console shows a stack trace for an uncaught error:

```
takePicture cpctr-app.js:837
▶ [{"label":0,"trys":[],"ops":[]}] cpctr-app.js:409
▶ ion-pwa-camera-modal cpctr-app.js:413
▶ Uncaught (in promise) TypeError: cameraModal.componentOnReady is not a function cpctr-app.js:414
    at CameraPluginWeb.<anonymous> (cpctr-app.js:414)
    at step (cpctr-app.js:351)
    at Object.next (cpctr-app.js:332)
    at cpctr-app.js:325
    at new Promise (<anonymous>)
    at __awaiter (cpctr-app.js:321)
    at cpctr-app.js:406
    at new Promise (<anonymous>)
    at CameraPluginWeb.<anonymous> (cpctr-app.js:406)
    at step (cpctr-app.js:351)
```



Let's cheat!



Manually adding a

`<ion-pwa-camera-modal>`
element

Only to force it to be recognized...

-1



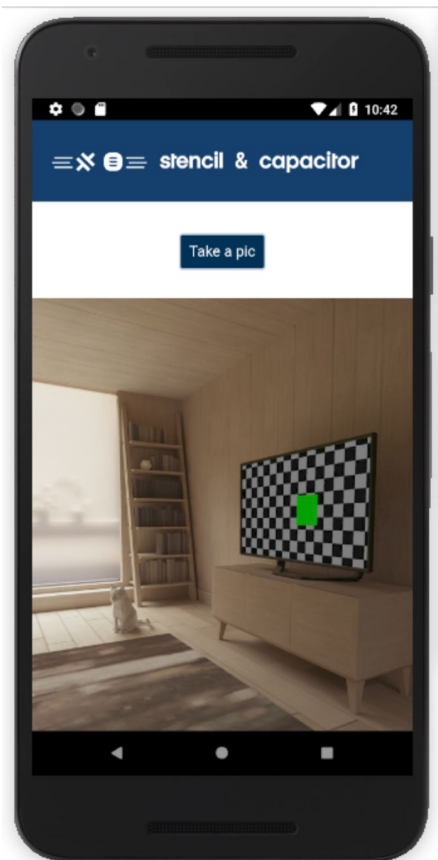
And then...



It's a kind of magic!



On Android and Web



☰ ✕ ☰ stencil & capacitor

Take a pic



Second test: successful!



Capacitor 2 - Scepticism 0





And now?

What's next



Work in progress



Working on an app, needs to be ready for
next Warp 10 Meetup on April 17th



Over a Stencil+Polymer hybrid



Stencil and Polymer mix already worked on



And Capacitor already works



A true winner



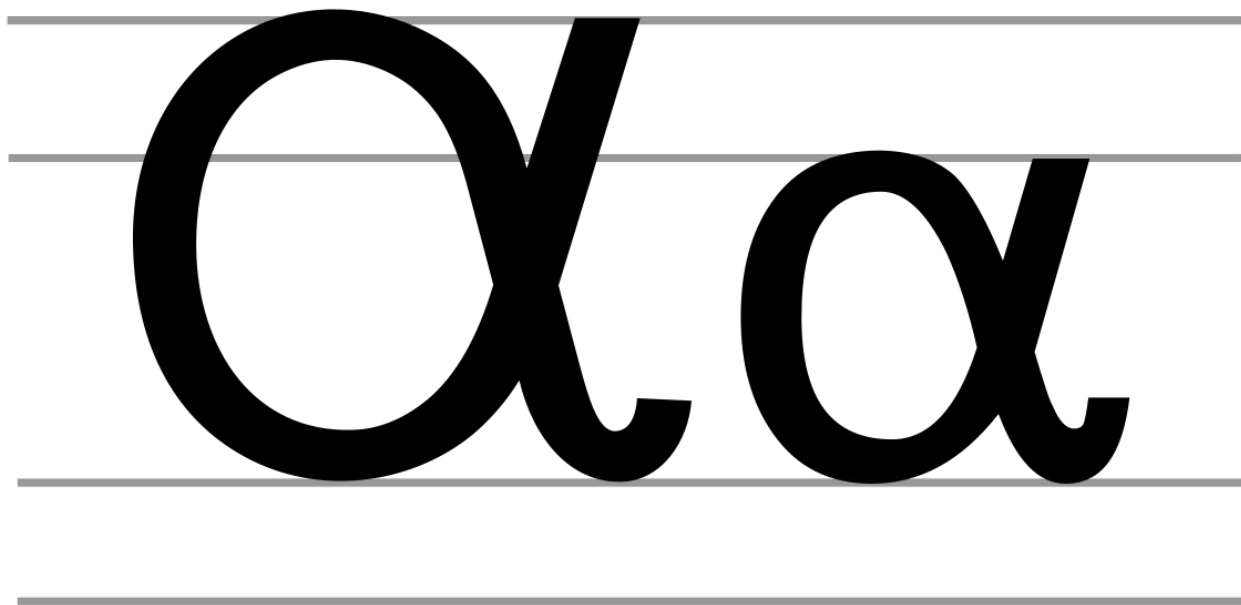


Conclusions

Capacitor or not?



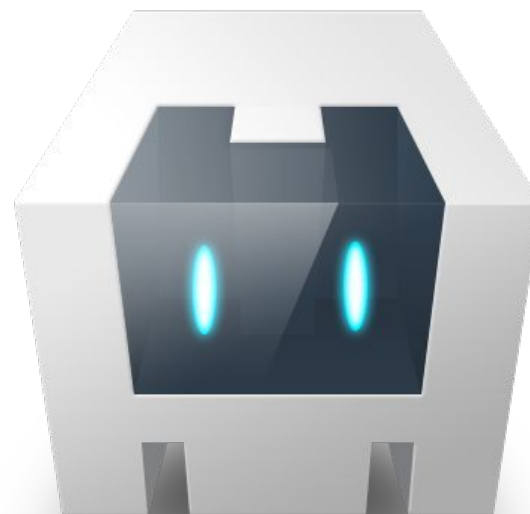
Paint is still wet



Some things are broken
Some things will change



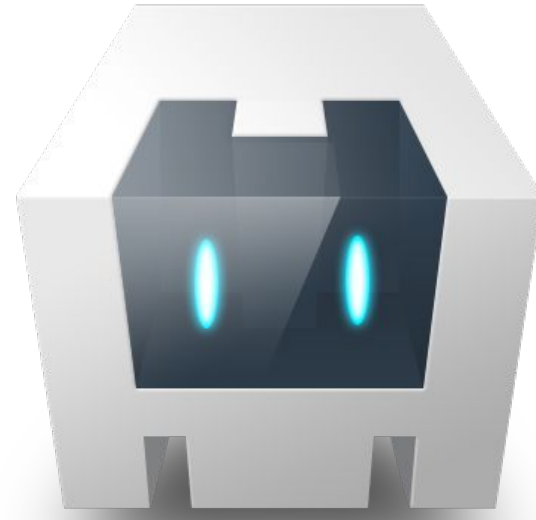
Easy to use



Friendlier than Cordova



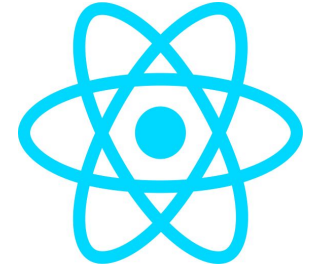
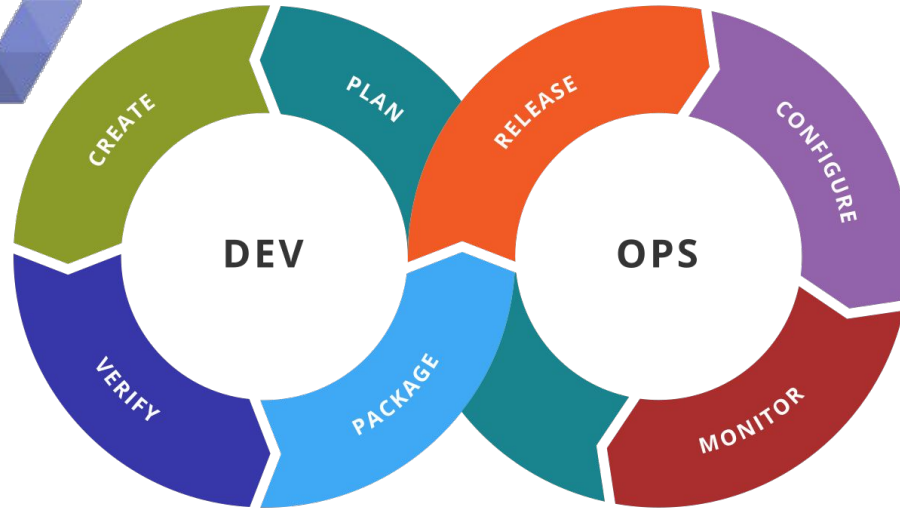
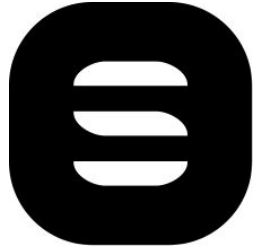
Yet extensible



You can use existing Cordova plugins



Not opinionated



Easy to use in any framework
Easy to integrate in any dev toolchain



