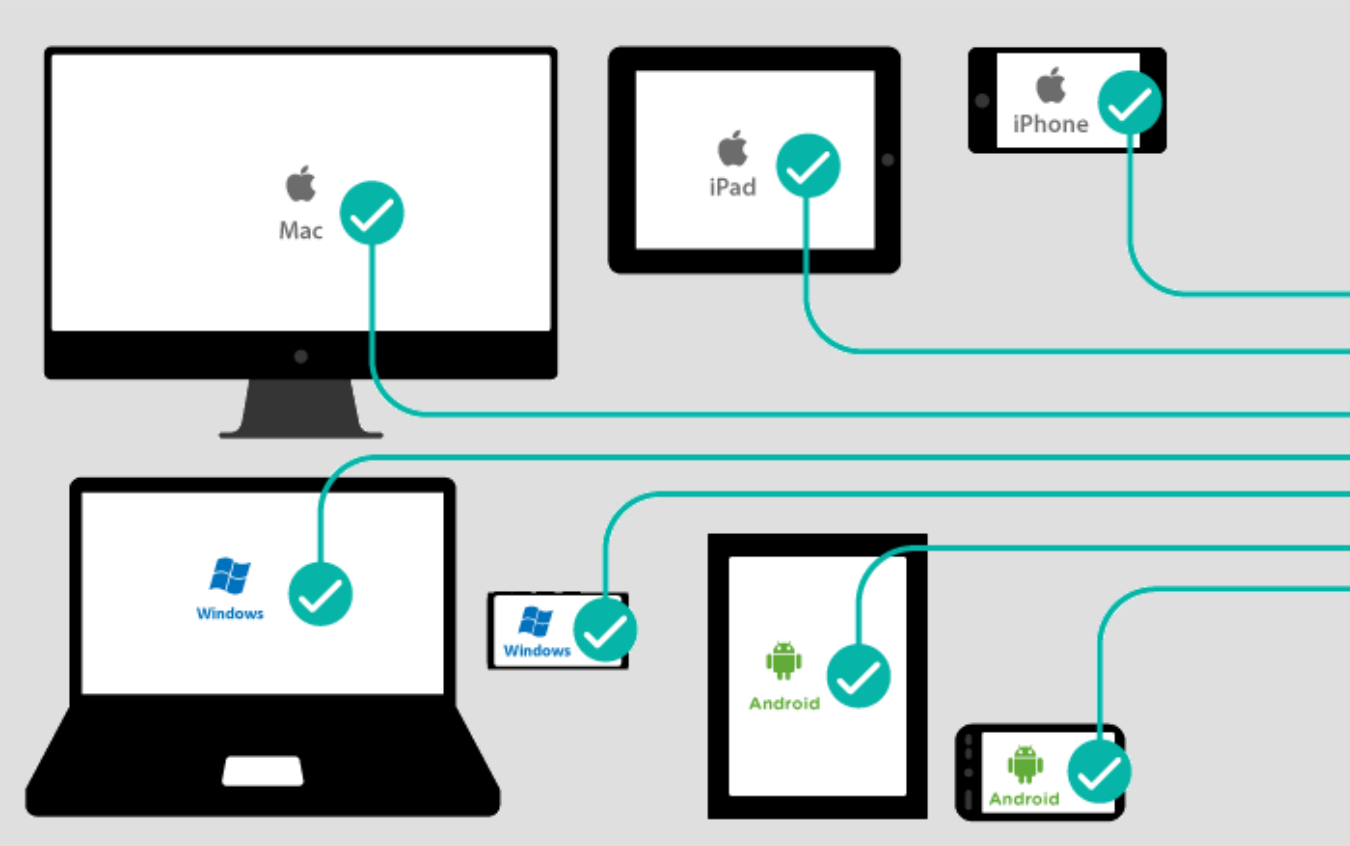
CROSS-PLATFORM MOBILE DEVELOPMENT



APACHE CORDOVA™

Apache Cordova is a open-source mobile development framework. It allows you to use standard web technologies such as HTML5, CSS3 and JavaScript for cross-platform development, avoiding each mobile platform's native development language.

Uses Apache Cordova:

- To develop an application across more than one platform, without having to re-implement it with each platform's language and tool set.
- To deploy a web app that's packaged for distribution in various app store portals.
- To mix native application components with a WebView that can access device-level APIs, or if you want to develop a plugin interface between native and WebView components.

BASIC COMPONENTS

There are 3 basic components in every Apache Cordova project:

- **config.xml** – A file that provides information about the app and specifies parameters affecting how it work.
- **index.html** – web page, that references whatever CSS, JavaScript, images, media files, or other resources are necessary for it to run. The app executes as a WebView within the native application wrapper, which you distribute to app stores.
- A **plugin interface** is available for Cordova and native components to communicate with each other. This enables you to invoke native code from JavaScript:
  - **Third-party plugins**, provide additional bindings to features not necessarily available on all platforms. (Plugin Registry)
  - **Additional plugins** developed by ourselves.

DEVELOPMENT PATHS

There are two basic workflows to create a mobile app. While you can often use either workflow to accomplish the same task, they each offer advantages:

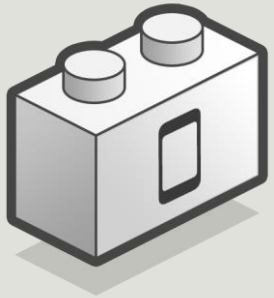
Cross-platform (CLI)
Used to develop an app that run on as many different mobile operating system as possible, this work around the cordova utility.

The CLI is a high-level tool that allows you to build projects for many platforms at once. Provides a common interface to apply plugins to your app.

Platform-centered
Used to focus on building an app for a single platform and need to be able to modify it at a lower level.

This allows greater access to development options provided by each SDK, and is essential for complex hybrid apps.

When first starting out, it may be easiest to use the cross-platform workflow to create an app, then switch to a platform-centered workflow to control the SDK provides.



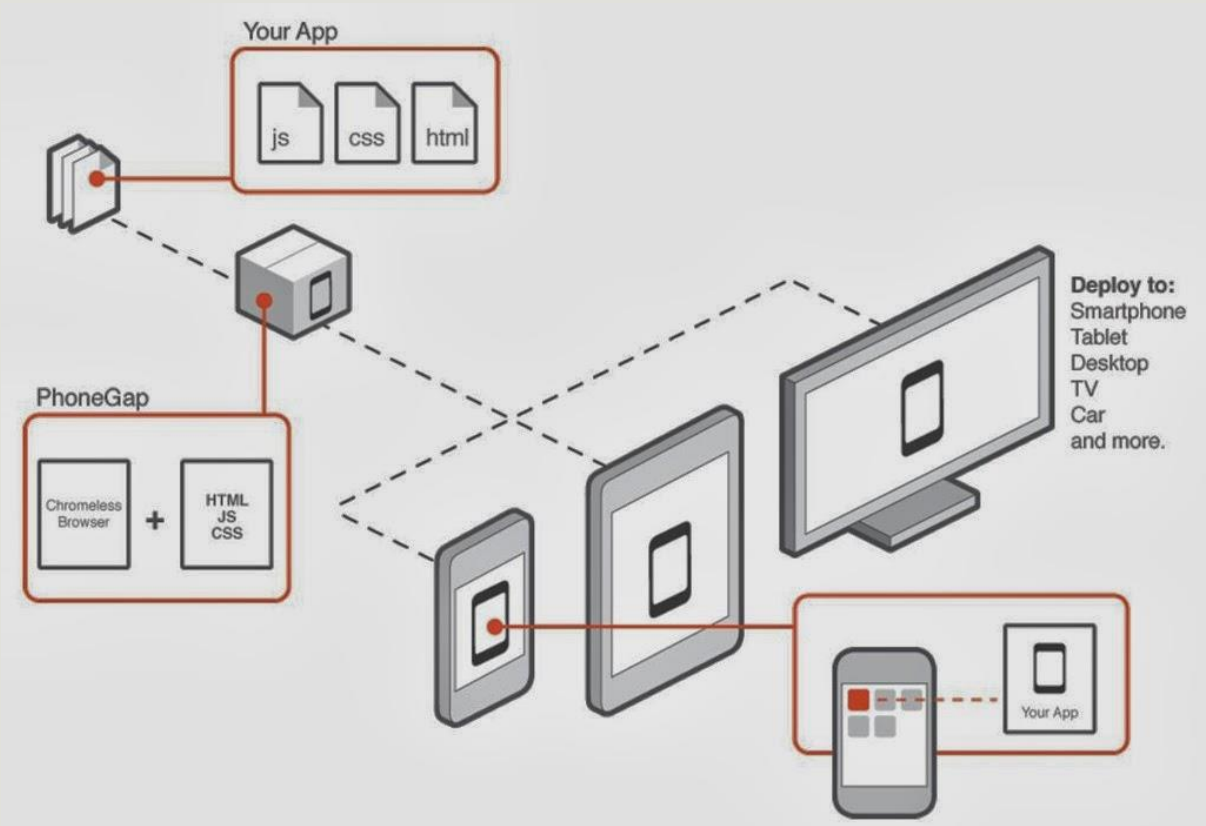
PLUGINS

PLUGIN Is a package of injected code that allows the Cordova webview within which the app renders to communicate with the native platform on which it runs.
Plugins provide access to device and platform functionality that is ordinarily unavailable to web-based apps. There is Registry of available plugins.
Plugins comprise a single JavaScript interface along with corresponding native code libraries for each supported platform. This hides the various native code implementations behind a common JavaScript interface.

PLUGMAN Cordova utility to manage plugins.

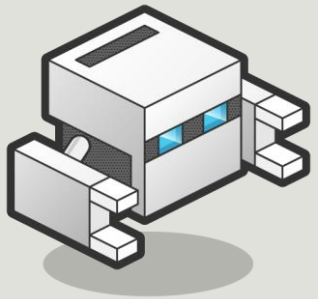
Cordova current version implements all device APIs as plugins, and leaves them disabled by default.
It also supports two different ways to add and remove plugins, depending on your choice of workflow.

PhoneGap



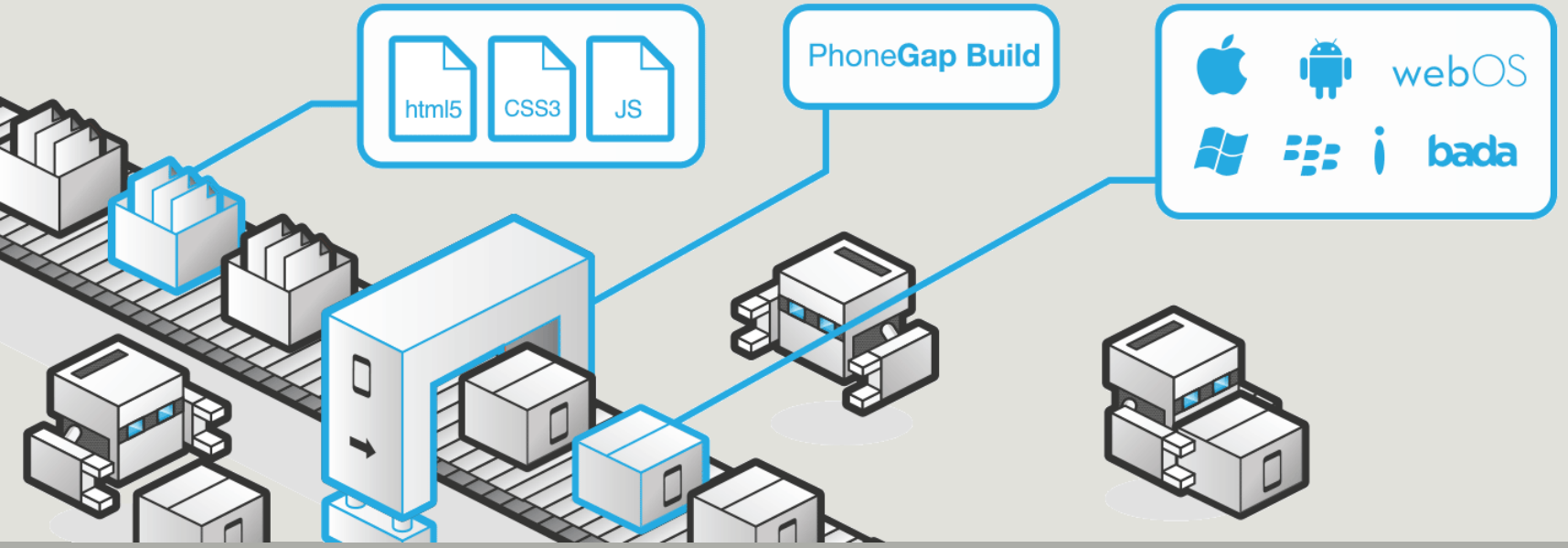
PhoneGap is a distribution of Apache Cordova. In other terms, Apache Cordova is the engine that powers PhoneGap.
Over time, the PhoneGap distribution may contain additional tools that tie into other Adobe services, which would not be appropriate for an Apache project.

WHY PHONEGAP?



PhoneGap:Build

PhoneGap Build is a cloud service for compiling PhoneGap applications.



ADVANTAGE

- **Manage Compilation and Signing**: Get app-store ready apps without the headache of maintaining native SDKs. PhoneGap Build will always be build against the required SDK for the platform you are targeting.

- **Multi-Platform Support**: Maximize productivity while minimizing production time. Target iOS, Android, Windows Phone all with a single codebase.

- **Work With Your Team**: Work collaboratively by adding team members and create roles within PhoneGap Build projects.

- **Quick Development Cycle**: Hydration speeds up debug and build cycles. The updates be pushed directly to the tester's previously installed apps, ensuring everyone is working on the most up-to-date version.