Apache Cordova/PhoneGap

What is PhoneGap

**PhoneGap** is an open source framework for quickly building cross-platform mobile apps using HTML5, Javascript and CSS. Wrap your app with **PhoneGap** Deploy to mobile platforms! Build your app with web technology Wrap your app with **PhoneGap** Deploy to mobile platforms!

**[Apache Cordova](http://cordova.apache.org/" \t "_blank) (known by many as “[PhoneGap](http://phonegap.com/" \t "_blank)“) holds the top slot in developer mindshare**. Cordova/PhoneGap developers write their mobile applications using HTML, JavaScript and CSS. These assets run in a “WebView” inside a native application container on the target platform. It is, conceptually, a web application packaged within a native application container where your JavaScript has access to device-level APIs that normal web applications would not (more on that below).

The name “PhoneGap” is quite possibly one of the more recognizable names in this space. Originally created by Nitobi, the name was changed to “Apache Cordova” when it was donated to the Apache Software Foundation. Adobe purchased Nitobi – including rights to the PhoneGap name – and now distributes Cordova under that name.

**Pros**

* Regardless of server side platform & language experience, a significant number of developers have experience with HTML, JavaScript and CSS. Apache Cordova allows developers to immediately leverage these existing skills. The value of this can’t be overstated – as it reduces training and can enable a quick-to-market stance in companies ready to adopt it.
* Cordova apps install just like a native application, and are able to leverage app store discoverability.
* Cordova follows a plugin architecture, which means that access to native device APIs can be extended in a modular way. There are a lot [Cordova](http://plugins.cordova.io/)/[PhoneGap](https://github.com/phonegap/phonegap-plugins" \t "_blank) plugins to choose from – enabling developers to focus on the web-based skills they already have. (This is a weakness as well, as we’ll see in a moment.)
* Cordova is open source *and* free, so there are no licensing costs (also a potential weakness, mentioned below).
* Cordova/PhoneGap solutions existed in this space early on, and have matured to the point where value-add offerings on top of the basic CPT are the norm. For example, both [Adobe’s PhoneGap Build](https://build.phonegap.com/) and [Telerik’s Icenium](http://www.icenium.com/" \t "_blank) enable developers to build for supported target platforms in the cloud, without local SDKs (meaning non-Mac users can build iOS applications). In addition to Icenium’s cloud build services, Telerik also provides [Kendo UI Mobile](http://www.kendoui.com/) (an MVVM framework targeted for performance on mobile), app analytics via [EQATEC](http://www.telerik.com/analytics) and a Backend-as-a-Service (BaaS) offering named [Everlive](http://telerik.com/everlive" \t "_blank). Adobe has integrated PhoneGap Build capabilities into [Brackets](http://brackets.io/) (a web based IDE) and [Dreamweaver](http://www.adobe.com/products/dreamweaver.html).

**Cons**

* Of course, being free is no guarantee of success. In fact, the emergence of PhoneGap Build and Icenium are clear demonstrations that a “bare bones” Apache Cordova is woefully incomplete. The strength of being open source – and leveraging the talents of a wide array of contributors – is both a blessing and curse. If you need to extend your app with a custom Cordova/PhoneGap plugin, odds are you will find one. Yet it may be out of date and not support the target platforms you need.
* The plugin architecture works well if you can find the plugins you need *or* if your web developers are capable of changing gears to write their own custom plugin(s) as needed. However, odds are that you chose Cordova, in part, to avoid the need for specialized native platform skills.
* The performance of Cordova/PhoneGap apps has often been criticized. Native UI will always outperform a hybrid solution, but improvements in device hardware and WebView implementations have narrowed the gap. Your web developers will need to pay close attention to performance, which means their knowledge of profiling tools as well as which web UI frameworks are mobile-friendly is essential.

Performance is not best

Graphical UI not good