Cordova - Guide - Introduction

- Dr Nick Hayward

A brief overview and introduction to Apache Cordova.

Contents

What is Cordova?

Cordova has been designed to offer a simple, yet powerful set of API calls to JavaScript functions. These functions map native code for a device's OS to plugins and code in Cordova, thereby enabling access to core functionality for a device.

In effect, this means we can transfer, manipulate, and control data and resources from the native OS and device to the web view in our app. We can provide the same experience for our users, with a few low level caveats, and the user will be none the wiser.

Cordova also offers cross-platform support for multiple mobile OSs, and associated devices.

Cordova wouldn't be cross-platform if it failed to support many different OSs. Therefore, support has included the following mobile OSs since it began,

- Android
- iOS
- Windows 10 Universal platform
- Ubuntu
- LG webOS
- BlackBerry 10
- ...

It has now focused support on Android, iOS, OS X, and Windows 10.

Functionality and plugins

Cordova allows us to create native mobile applications using a set of common web technologies, including HTML5 (HyperText Markup Language), CSS (Cascading Style Sheet), and JavaScript. By providing an initial set of JavaScript APIs, it provides access to natively built core plugins. It currently offers many core APIs, which include some of the following native functionality,

- access the device's microphone for recording &c.
- capture photos using the device's camera
- photo retrieval from the OSs gallery/photo album
- retrieve device information
  - locale
  - various sensors such as motion, location, connection information, compass...
- retrieve device data, contact information...
- process files from/to storage

- ...

So, you should already be able to see how we can use these core APIs to help us construct an application. There are also many other plugins available, and we can always build a custom plugin for missing native functionality.

Documentation and APIs

The official *Cordova* API documentation is currently available at the following URL,

- Apache Cordova API
- Apache Cordova GitHub
- Android API
- iOS API
- ... & many others

Obviously, the above repositories require knowledge of GitHub. Now is a good time to learn how to use GitHub. At least, you should be able to clone and pull the contents of a repository.

Why choose it?

Cordova helps us solve some of the following issues with mobile application development.

- different programming languages for different mobile OSs
  - different programming philosophies, conventions, best practices, guidelines...
- unique problems inherent to each given mobile OS
  - e.g. handling and routing SMS requests, data storage, privacy features...
- developing, testing, and maintaining applications across multiple mobile platforms

One of the biggest headaches from the above brief list is the inherent difference in programming philosophies and characteristics from platform to platform. It's this switching from one language to another, one set of guidelines and practices to another, that often limits a developer's choice of mobile OS, and may ultimately define their development focus and application's design. Switching from native Android development to iOS takes a noticeable mental shift, and you'll often find yourself forgetting or confusing one best practice with another.

Overview of APIs

Whilst Cordova supports many different mobile OSs, for the sake of brevity, and our own general sanity, we're going to primarily consider it relative to Android and iOS, which can then be abstracted to include other available platforms where necessary.

So, a good starting point for these APIs, and their general feature sets, is to briefly consider the following table. It details the current plugins, and their general support in these particular OSs.

Initial impressions seem to indicate pretty good support for native functionality via plugins in Cordova. Again, if functionality is currently not available via existing plugins we can create a custom one for our project.

We also have access to W3C APIs, which are now increasingly supported by native OS webviews. For example, accelerometer support has now migrated from a Cordova Plugin to W3C API.

We'll start going through these plugins, and what they allow us to do with a native device, as we develop our Cordova test apps

Many of these mobile native function APIs should be self explanatory, but there are a few that would benefit from further consideration.

For example, `capture` allows us to record various media formats directly from the native device. So, this API allows us to capture audio using the native device's audio recording application, capture images with the device's camera application, and similarly capture video using the device's video recording application.

`connection` provides information about the device's wifi and cellular connections, particularly useful to allow us to switch data contexts within an application as required by connection parameters.

`device`, as the name suggests, provides useful information on a device's hardware and software, including the native device model, the current platform and its version, and receive the device's name.

`events` is a particularly important, and useful, API with many **life cycle** events from `deviceready` to `backbutton` to `batterystatus` and various volume events.

We can process files using the `file` API, and receive the device's location using GPS or network signals. We have access to various states of notifications, including tactile and visual, and various storage capabilities including local storage and temporary storage for the period of the open application.

We'll work our way through many of these APIs, simply to show what can be done with Cordova, and also as practice to help gain experience.

Further information can be found at the following URL,

- Apache Cordova Documentation - Platform Support