

CORDOVA TOOLING - API

EXPERIMENTAL API FOR BETTER DEVELOPMENT WORKFLOWS

By Mark Koudritsky - kamrik@google.com / [GitHub](#)

[black theme](#) - [PDF](#) - [export as PDF](#) (Cmd-P, Chrome only)

CORDOVA CLI DOES TOO MANY THINGS!

- Some tasks better done by other tools (e.g. fetching)
- Difficult to combine with other webdev tools like Sass and CoffeScript
- Too prescriptive and wastes our time debating how exactly to over-specify Cordova user's environment
- Not very git friendly (platforms and plugins dirs)

REPHRASED POSITIVELY

I WANT CORDOVA TOOLING TO BE

1. Simpler, smaller and more flexible
2. Friendly to webdev tools and existing web apps
3. Very friendly to task runners like Grunt and Gulp

SIDE NOTE ABOUT GULP

I'll be using Gulp for code examples in this presentation. Since Gulp tasks are just JS functions, it can be trivially translated to pure JS scripts.

GULP CRASH COURSE

- Gulp is minimal, liberal and simple task runner
- Gulpfile is a JS file that can contain any code
- Way less declarative than Grunt
- Very easy to experiment and prototype with
- Based on **streams** but I don't really care



A typical "clean" task in Gulp.

```
gulp.task('clean', function(cb) {  
  del(['./build'], cb);  
});
```

Take a look at gulpjs.com or a [presentation](#) about Gulp

CORDOVA OOP API WRAPPER & DEMO APP

- github.com/kamrik/CordovaPlatformProject
- github.com/kamrik/cordova-api-example

Evolved from the previous iteration at:
github.com/kamrik/CordovaGulpTemplate

API WRAPPER

- A wrapper around cordova-lib that provides a coherent OOP API
- Centered around a PlatformProject object that deals with a single platform - represents what traditionally lives under `platforms/<platform>/*`

SAMPLE APP STRUCTURE

- package.json
- gulpfile.js
- www/
- CordovaConfig.xml - Cordova's config.xml file
- build/ - build artifacts, nuked on every "clean"
 - build/android
 - build/ios
 - ...

PACKAGE.JSON

```
"dependencies": {  
  "CordovaPlatformProject": "kamrik/CordovaPlatformProject",  
  
  "cordova-android": "latest",  
  "cordova-ios": "latest",  
  "cordova-plugin-dialogs": "latest"  
  ...  
}
```

GULPFILE.JS - SETUP

```
var pp = require('CordovaPlatformProject'); // The API Wrapper

var prjInfo = {
  platform: 'ios',
  paths: {
    root: 'build/ios',
    template: 'node_modules/cordova-ios',
    www: 'www'
  },
  cfg: pp.cdv.ConfigParser('CordovaConfig.xml');
};
```

GULPFILE.JS - CREATE & ADD PLUGINS

```
gulp.task('create', ['clean'], function() {  
  fs.mkdirSync('build');  
  
  return prj.init(prjInfo)  
  .then(function(){  
    return prj.addPluginsFrom('./node_modules');  
  });  
});
```

GULPFILE.JS - BUILD & RUN

```
gulp.task('build', [], function() {
  var prj = new pp.PlatformProject();
  return prj.open(platform, prjInfo.paths.root)
    .then(function() {
      return prj.updateConfig(prjInfo.cfg);
    })
    .then(function() {
      return prj.copyWww(prjInfo.paths.www);
      // Or copy yourself to prj.wwwDir
    })
    .then(function() {
      // This runs ./build/<platform>/cordova/run --device
      return prj.run({args: ['--device']});
    });
});
```

</platform>

USAGE

```
$ git clone https://github.com/kamrik/cordova-api-example.git .  
$ npm install  
$ gulp create  
$ gulp run # repeat after changing files in www
```

ADD / REMOVE A PLUGIN

```
$ gulp clean  
$ npm (un)install --save cordova-plugin-xyz  
$ gulp create
```

Note: Peer dependencies not yet added to plugins on npm

NOTABLE POINTS

- Using ConfigParser from cordova-lib directly. The loaded object can be manipulated in JS to customize workflow
- Using PluginInfo objects from cordova-lib directly
- Project hooks are no longer needed
- Plugin hooks might still be needed

NEXT STEPS

- Evolve PlatformProject and polish the API
- Minimize code duplication, though some duplication is unavoidable in this case

FUTURE IMPROVEMENTS

- ConfigParser from App manifest and other files
- Improved PluginInfo object and plugin handling functionality

QUESTIONS?

THE END

github.com/kamrik/cordova-api-example