Desarrollo Web en Entorno Cliente  
UD 13. Cordova y Electron

short line

Actualizado Diciembre 2020

Licencia

**Reconocimiento – NoComercial - CompartirIgual (BY-NC-SA)**: No se permite un uso comercial de la obra original ni de las posibles obras derivadas, la distribución de las cuales se debe hacer con una licencia igual a la que regula la obra original.

Nomenclatura

A lo largo de este tema se utilizarán distintos símbolos para distinguir elementos importantes dentro del contenido. Estos símbolos son:

**📖 Importante**

**❕ Atención**

**💬 Interesante**

**Índice de contenido**

**1.** [**Introducción**](#_ge43nvu7ywtj) **3**

**2.** [**Instalación de Cordova y Electron**](#_63w9la2rzwtw) **3**

**3.** [**Usando Cordova**](#_tjvw91hmlvof) **3**

**3.1.** [**Preparamos el proyecto**](#_6op53u35w7ug) **3**

**3.2.** [**Colocamos nuestra aplicación web en el sitio indicado**](#_xin9h7z2ghx3) **4**

**3.3.** [**Generamos el fichero APK**](#_ocxuv2hyuxon) **4**

**3.4.** [**Firmando el fichero APK**](#_pqcorng164) **4**

**4.** [**Usando Electron**](#_5ivfzjxvi22) **5**

**4.1.** [**Primero pasos**](#_ehj3fqx9sjwq) **5**

**4.2.** [**Empaquetando una aplicación Electron para distribuirla**](#_128nb5rl0pzi) **5**

**4.3.** [**Empaquetando la aplicación en un directorio**](#_mn6g4xf62k36) **6**

**5.** [**Acceso a funciones especiales en Cordova y Electron**](#_i1a78i3aaw0x) **6**

**6.** [**Enlaces interesantes sobre Cordova Y Electron**](#_fflziav5c94n) **6**

**7.** [**Bibliografía**](#_q7ukkh657iu5) **7**

**8.** [**Autores (en orden alfabético)**](#_g1qlmy5ta6mu) **7**

UD13. Cordova y Electron

# Introducción

En esta unidad vamos a ver dos herramientas para crear aplicaciones multiplataforma.

En primer lugar, hablaremos de **Cordova**, que permitirá que vuestras aplicaciones HTML/Javascript se puedan meter en paquetes APK y verse como aplicaciones móviles, además de permitir el acceso a funciones del móvil (cámara, posicionamiento, etc.).

En segundo lugar, hablaremos de **Electron**, que permitirá que vuestras aplicaciones HTML/Javascript se pueden usar como aplicaciones de escritorio en Windows, Linux y Mac mediante el framework Electron, permitiendo incluso a funciones de acceso al sistema que no se permiten habitualmente en Javascript Web (por ejemplo, el acceso a ficheros).

Para que conozcáis la potencia de Electron, deciros que existen gran cantidad de aplicaciones que han sido creadas usándolo. Podéis ver la lista en <https://electronjs.org/apps>

Entre ellas destacan algunas como el editor Atom, Skype o Visual Studio Code.

# Instalación de Cordova y Electron

Instalar Cordova y Electron es tan sencillo como seguir las instrucciones:

* Cordova <https://cordova.apache.org/>
  + ***“npm install -g cordova”***
  + Cordova para generar APK requiere que se instale el kit de desarrollo Android.
    - La forma más sencilla de instalarlo y dejarlo todo configurado es instalando Android Studio <https://developer.android.com/studio>
* Electron <https://electronjs.org/>
  + ***“npm install -g electron”***

Recordamos que usando npm, la opción “-g” indica que instalamos dichos paquetes para todo el sistema (si lo ponemos sin -g, nos lo instala únicamente para el proyecto el directorio actual).

# Usando Cordova

Supongamos que tenemos una aplicación Web que queremos empaquetar en una APK para Android. Dicha aplicación puede ser una página o varias, incluir recursos como ficheros y bibliotecas JS, imágenes, etc. **(Importante: la inicial debe llamarse index.html)**.

Para facilitar el tutorial, suponemos que todo el contenido lo tenemos comprimido en el fichero **"nuestraPagina.zip"**.

## Preparamos el proyecto

Para crear un nuevo proyecto Cordova, utilizaremos la siguiente orden ***"cordova create MiProyecto com.pruebas.miproyecto MiProyecto"***, donde el primer parámetro es el nombre de la aplicación, el segundo es la ruta Android de la app y el tercero es el directorio a crear con el proyecto.

Una vez hecho eso, tendremos un directorio "MiProyecto". Entramos a esa carpeta "cd MiProyecto" y ahí escribimos "cordova platform add android". Esto sirve para decirle que genere código Android (un APK). Cordova no se limita a APK, también se puede genera código IOS por ejemplo.

## Colocamos nuestra aplicación web en el sitio indicado

Una vez hecho esto, en “MiProyecto” hay una carpeta "www". Podéis borrar su contenido y colocar aquí el código de vuestra aplicación web. **Recordad que debe haber una página "index.html"**.

## Generamos el fichero APK

Una vez todo listo, creamos el APK con la orden ***"cordova build"*** para la versión "debug" o para sacar la versión de producción ***"cordova build --release"***

Así se creará el fichero APK. Al finalizar nos indicará donde se aloja el APK. Generalmente (aunque podría cambiar según versión) se encontrará en la carpeta del proyecto "platforms/android/build/”.

## Firmando el fichero APK

Si queremos que la aplicación puede ejecutarse en un teléfono real, subirse a Google Play, etc. debemos firmar la aplicación.

Aquí algunos enlaces que explican el proceso:

* Cómo firmar manualmente una APK:
  + <https://professor-falken.com/programacion/android_/como-firmar-manualmente-una-aplicacion-android-apk/>
* Cómo firmar una APK para Google Play
  + <https://developer.android.com/studio/publish/app-signing?hl=es-419>

A continuación, algunos comandos de utilidades Java relacionadas con la firma de APK:

* **Generación de claves:**
  + Se hace una sola vez y se usa siempre la misma.
  + ¡No perder la clave generada!
  + ***"%JAVA\_HOME%\bin\keytool" -genkey -v -keystore tuapp.keystore -alias tuapp -keyalg RSA -keysize 2048 -validity 10000***
* **Firmado usando la clave:**
  + Firmamos primero y después alineamos APK.
  + ***"%JAVA\_HOME%\bin\jarsigner" -verbose -sigalg SHA1withRSA -digestalg SHA1 -keystore tuapp.keystore tuapp-apk-nofirmado.apk tuapp***
  + ***"%ANDROID\_HOME%\build-tools\30.0.0\zipalign" -v 4 tuapp-apk-nofirmado.apk tuapp-release-firmada.apk***

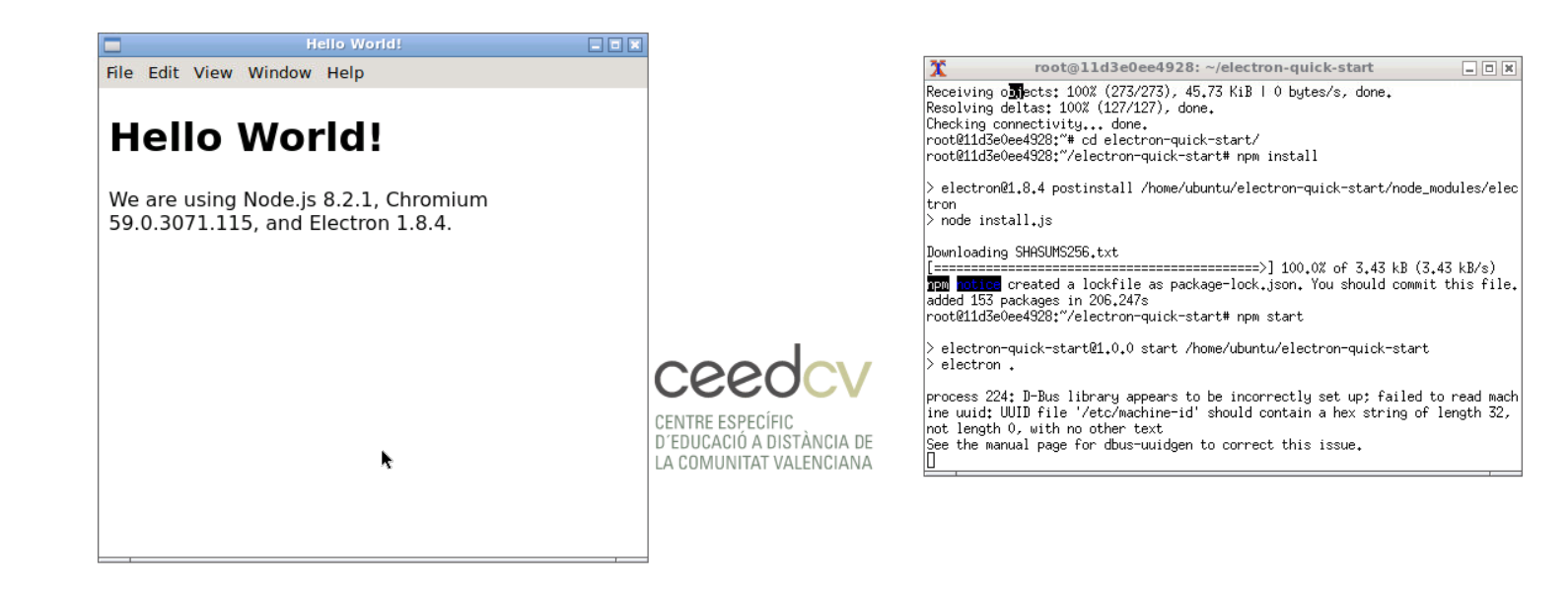
# Usando Electron

## Primero pasos

Comenzar con Electron es tan sencillo como seguir estos primeros pasos

|  |
| --- |
| *# Clona el repositorio con el ejemplo base de Electron git clone https://github.com/electron/electron-quick-start # Accede al repositorio clonado cd electron-quick-start # Instala dependencias de la aplicación (si las hay) npm install # Lanza la aplicacion npm start* |

Con esto lanzaremos la aplicación, con un resultado similar al de la imagen.



En este primer ejemplo que hemos probado hay 3 ficheros importantes:

* **package.json**: configurado con la información de los paquetes Node que necesita para funcionar Electron.
* **main.js**: Fichero que arranca la aplicación y que crea el navegador dentro de una ventana para que se pueda ver nuestra aplicación HTML/Javascript. Es el principal proceso de la aplicación.
* **index.html**: la Web que mostramos en nuestra aplicación. Este fichero lo sustituiremos por el index de nuestra aplicación.

Para saber más, aquí la guía de inicio de Electron <https://electronjs.org/docs/tutorial/quick-start>

## Empaquetando una aplicación Electron para distribuirla

Vamos a explicar como generar un directorio que contenga todo lo necesario para distribuir la aplicación. Deberemos generar un paquete por cada Sistema/Arquitectura que queremos que soporte y distribuir dichos directorios completos.

Vamos a presentar unos ejemplos de creación de paquetes para distribuir. Para estos ejemplos nos basamos en la información sobre “electron-packager” obtenida en

<https://www.christianengvall.se/electron-packager-tutorial/>

## Empaquetando la aplicación en un directorio

Podemos instalar la herramienta **“electron-packager”** con el comando ***“npm install electron-packager -g*”**. Una vez instalado, pongo aquí un ejemplo para generar un paquete desde Linux usando como destino una plataforma Linux de 64 bits.

|  |
| --- |
| electron-packager . miAPP --overwrite --platform=linux --arch=x64 |

Sobre este comando indicar:

* El “.” Inicial indica donde está el proyecto Electron (en este ejemplo, suponemos estamos en dicho directorio).
* “miAPP” es el nombre de la aplicación. Se generará un directorio formado por su nombre seguido de la plataforma y arquitectura. En dicho directorio es donde se guardará la aplicación con todo lo necesario para funcionar, incluyendo un ejecutable llamado ”miAPP”.
* --overwrite indica que si ya había un proyecto en el destino, lo sobreescriba.
* --platform indica la plataforma. En este caso Linux.
* --arch indica la arquitectura, en este caso x64

Otro ejemplo con el comando equivalente para crear un paquete para Windows de 32 bits.

|  |
| --- |
| electron-packager . miAPP --overwrite --platform=win32 --arch=ia32 |

**NOTA**: para crear aplicaciones Windows desde Linux, es obligatorio tener instalado Wine.

# Acceso a funciones especiales en Cordova y Electron

Tanto Cordova como Electron tienen acceso a funciones especiales de los dispositivos donde se ejecutan. Por ejemplo, Cordova permite en móviles acceder a partes del hardware tales como la cámara, el GPS, giroscopio, etc.

Electron permite acceder a funciones del sistema operativo como leer y escribir ficheros locales, lanzar procesos, etc.

# Enlaces interesantes sobre Cordova Y Electron

Aquí algunos enlaces interesantes sobre Cordova y Electron:

* Cordova
  + Instalar Cordova en Ubuntu.
    - <https://rolandocaldas.com/android/instalar-cordova-en-ubuntu>
  + Ejemplos de Cordova
    - <https://github.com/cfjedimaster/Cordova-Examples>
* Electron
  + Ejemplos de aplicaciones Electron y recurso (videos, tutoriales) creadas por la comunidad
    - <https://electronjs.org/community#boilerplates>
  + Tutorial inicio rápido
    - <https://electronjs.org/docs/tutorial/quick-start>
  + Aplicación hecha con Electron que… te enseña a usar Electron :)
    - <https://github.com/electron/electron-api-demos>

# Bibliografía

[1] Javascript Mozilla Developer <https://developer.mozilla.org/es/docs/Web/JavaScript>

[2] Javascript ES6 W3C <https://www.w3schools.com/js/js_es6.asp>

[3] Cordova <https://cordova.apache.org/>

[4] Electron <https://www.electronjs.org/>

# Autores (en orden alfabético)

A continuación ofrecemos en orden alfabético el listado de autores que han hecho aportaciones a este documento:

* García Barea, Sergi