

**DESARROLLO DE APLICACIONES WEB AVANZADO**

**Electron JS**

**CONDORI CONDORI JOHNN BILL**

**Electron JS**

**Objetivos:**

Al finalizar el laboratorio el estudiante será capaz de:

* Entender el funcionamiento de Electron JS
* Utilizar Electron JS para creación de enlatados
* Implementación correcta de Electron JS

**Seguridad:**

* Ubicar maletines y/o mochilas en el gabinete del aula de Laboratorio.
* No ingresar con líquidos, ni comida al aula de Laboratorio.
* Al culminar la sesión de laboratorio apagar correctamente la computadora y la pantalla, y ordenar las sillas utilizadas.

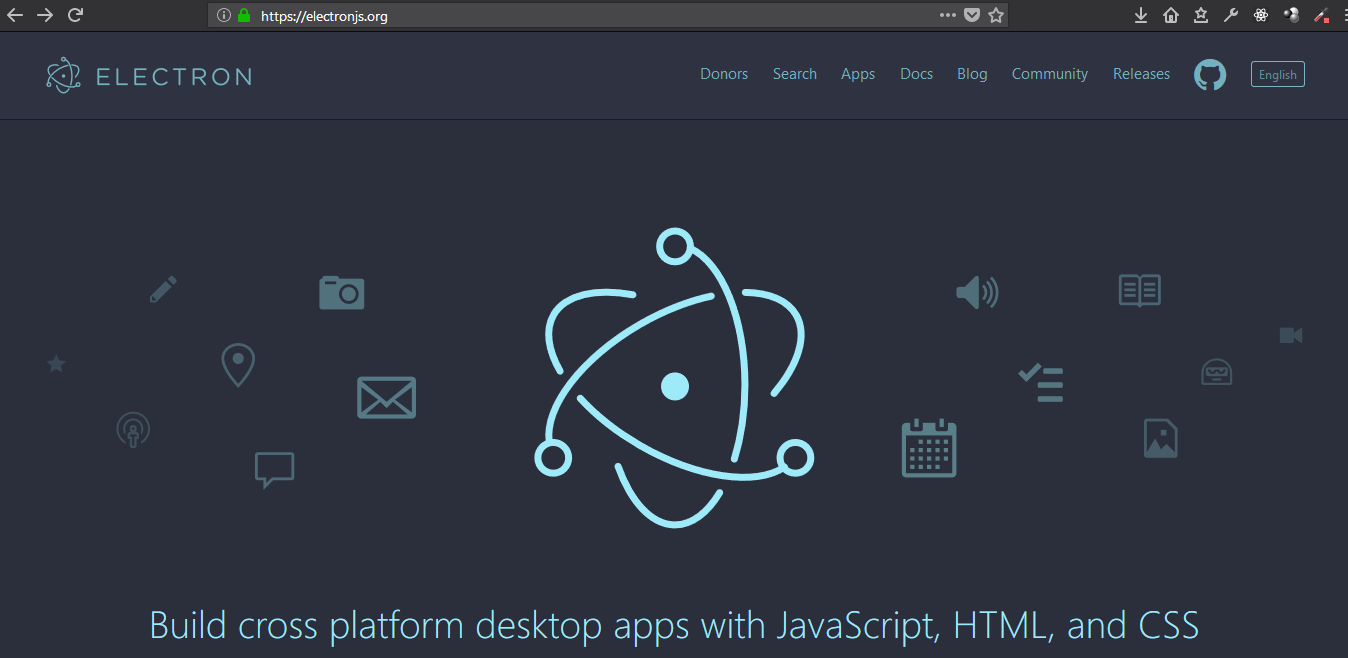
**Equipos y Materiales:**

* Una computadora con:
* Windows 7 o superior
* VMware Workstation 10+ o VMware Player 7+
* Conexión a la red del laboratorio
* Máquinas virtuales:
* Windows 7 Pro 64bits Español - Plantilla
* Instalador de node.js

**Procedimiento:**

**Lab Setup**

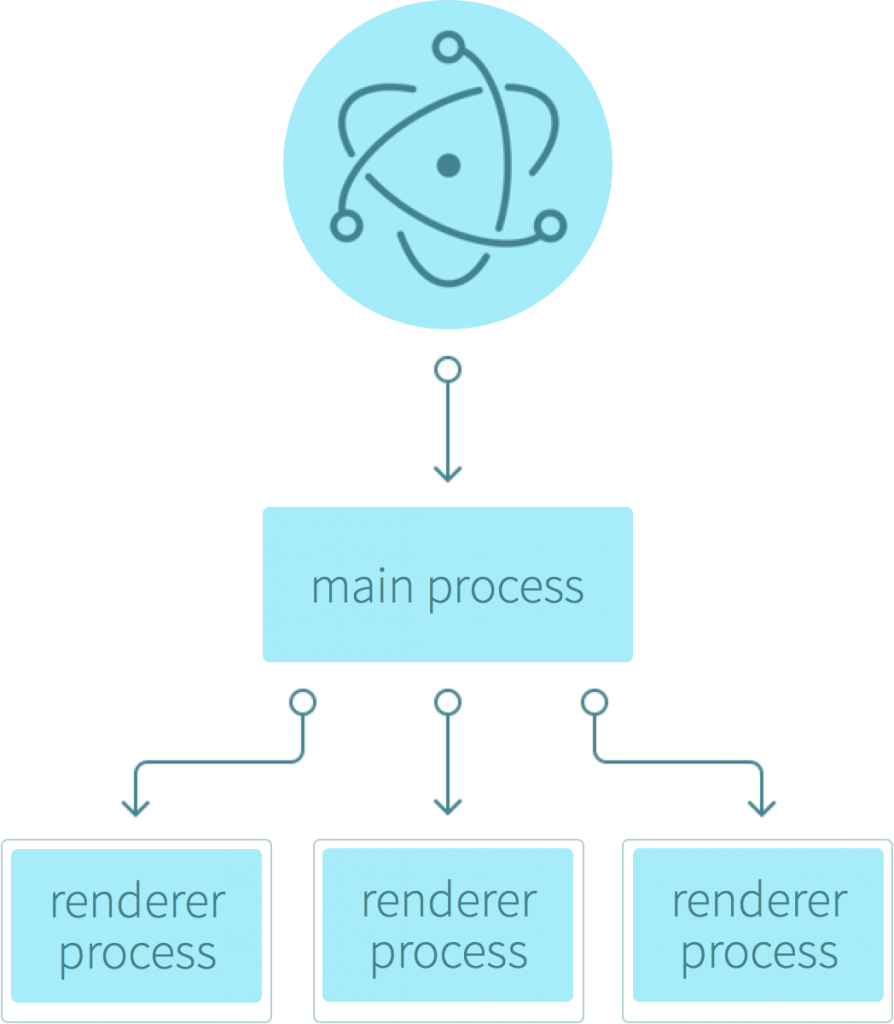
1. **Electron JS**



Electron.js es una plataforma para desarrollar aplicaciones de escritorio usando tecnologías web (HTML, CSS y JavaScript) creada y mantenida por Github. Electron.js funciona creando dos tipos de procesos, el proceso main y el proceso renderer. El primero es un proceso de Node.js, este es nuestro proceso principal, viene a ser nuestra aplicación en si misma, este proceso tiene acceso a varias API de Electron.js solo para este proceso que nos ayudan a comunicarnos con el SO y realizar distintas acciones o efectos.

El segundo (renderer) es un proceso de Chromium, con una diferencia, este Chromium tiene un Node.js incorporado y acceso a todos sus módulos y los que instalemos con npm (esto nos permitiría usar React.js, Angular.js, Polymer, etc. para desarrollar nuestra UI o cualquier otra librería básicamente), por lo que desde nuestrorenderer podemos usar módulos como fs para leer y escribir en el disco, o hacer peticiones a una base de datos directamente.

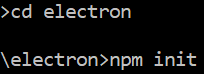
Además de acceder a los módulos de Node.js y npm, Electron.js nos da acceso a una pocas API en este proceso igual que hace con el procesomain.



* 1. Iniciaremos instalando de manera global el paquete de electron

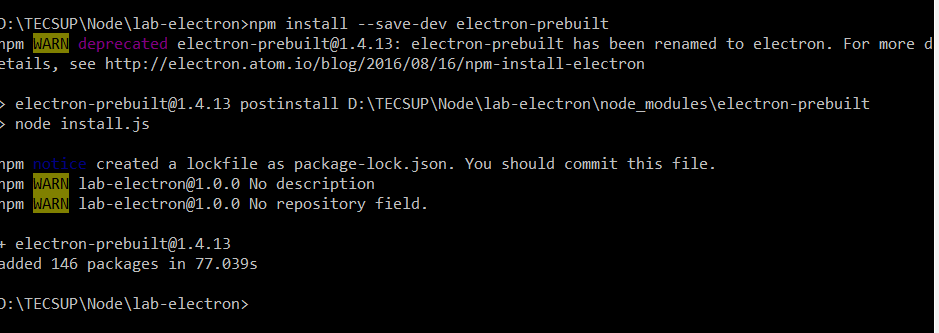


* 1. Crearemos una carpeta llamada electron e iniciaremos un proyecto npm con todas las opciones por defecto

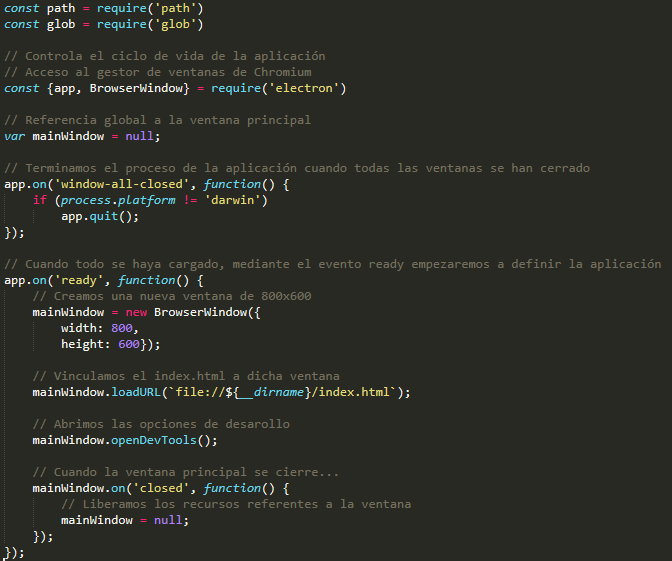


* 1. Después de haber iniciado nuestro proyecto, instalaremos electron como dependencia de desarrollo

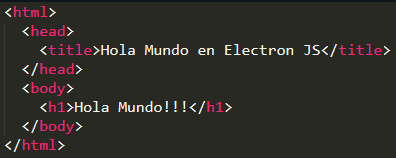




* 1. Para que Electron funcione de forma básica, debemos declarar un archivo principal. Crearemos el archivo index.js con el siguiente contenido, el cual ha tratado de ser comentado para tener la explicación de cada comando.

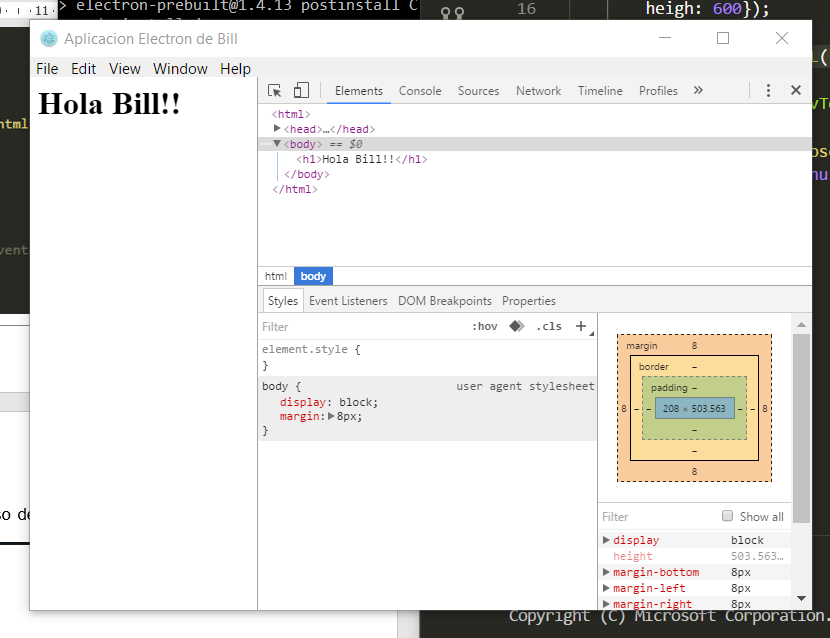


* 1. Como se declara en el archivo de arriba, haremos uso de un index.html, el cual tendrá el siguiente contenido



* 1. Listo! Ahora solamente debemos ejecutar el siguiente comando y ver el resultado de nuestra aplicación nativa.





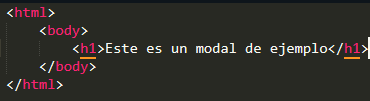
* 1. Como ya se mencionó, Electron no solamente permite crear una aplicación nativa desde una página web, sino que da funcionalidades de node a dichas páginas. Modificaremos el archivo index.html para que al iniciar de nuevo la aplicación, cargue el contenido del archivo package.json (el cual puede leer gracias al paquete fs de node)



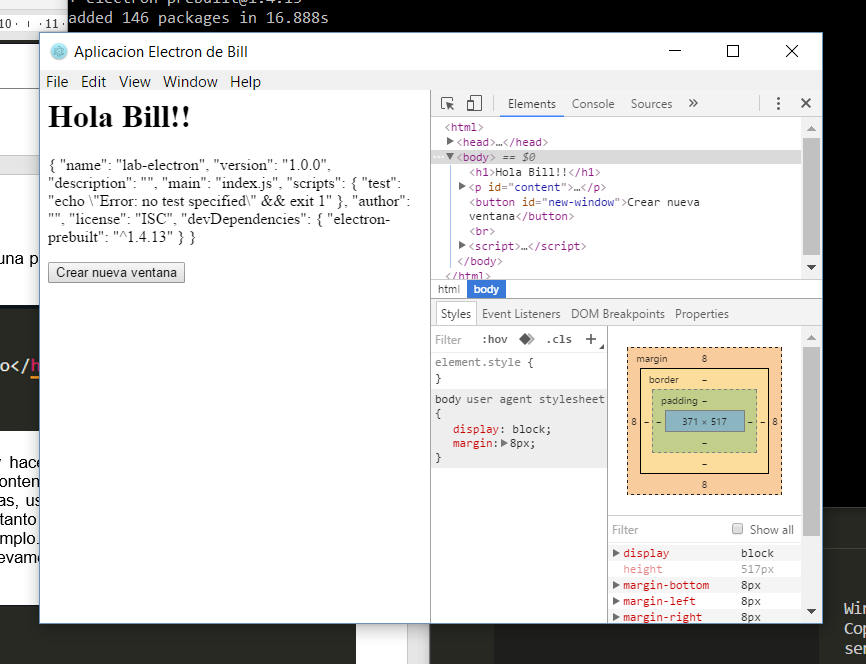
* 1. Así mismo, no estamos limitados a funciones de node sino también a funciones del sistema operativo. Crearemos una nueva ventana desde nuestra página web, por lo que modificaremos el index.html una vez más.

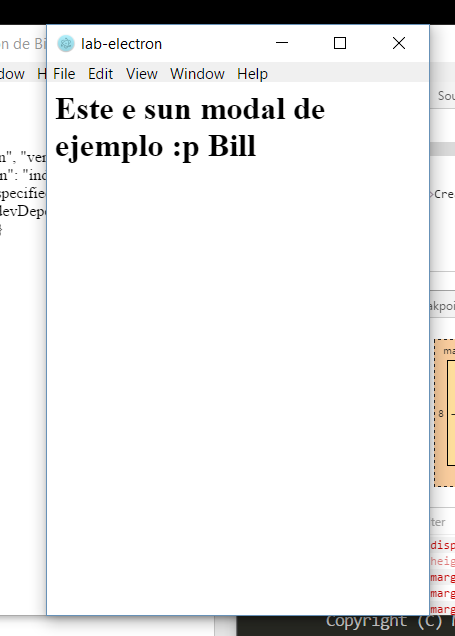


* 1. En la función antes agregada, estamos invocando a una página llamada modal.html, la cual crearemos con el siguiente contenido



* 1. Al ejecutar nuevamente nuestra aplicación, y hacer click en nuestro nuevo botón, veremos cómo se ejecuta una nueva ventana con el contenido indicado.

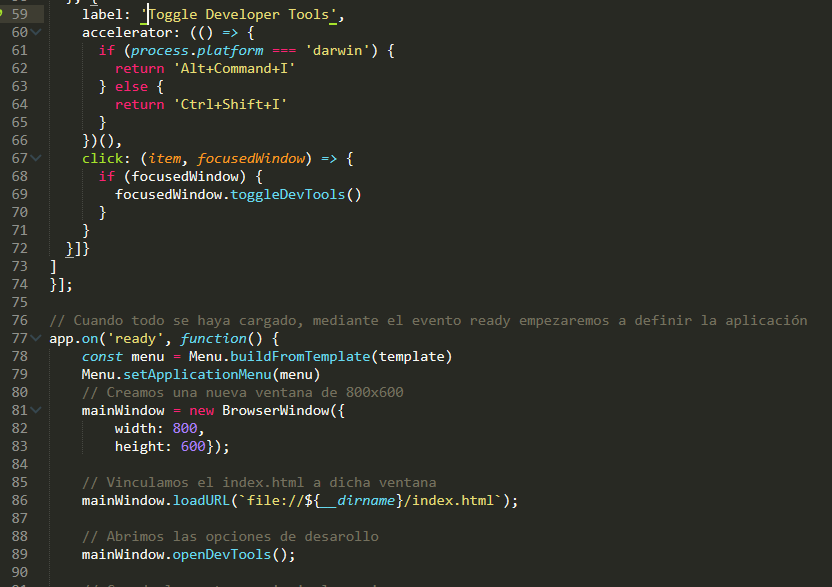


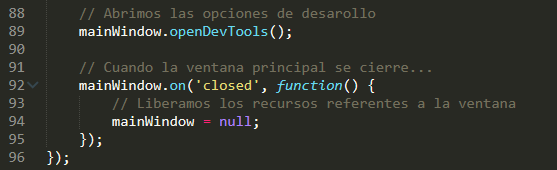


* 1. Así como hemos podido crear nuevas ventanas, usaremos una función adicional de Electron, que es Menu. Gracias a ella podemos crear tanto menus contextuales como menus de ventana, que es lo que haremos en el siguiente ejemplo. Modificaremos el archivo index.js para que luzca como se detalla y ejecutaremos nuevamente la aplicación para probar lo declarado.









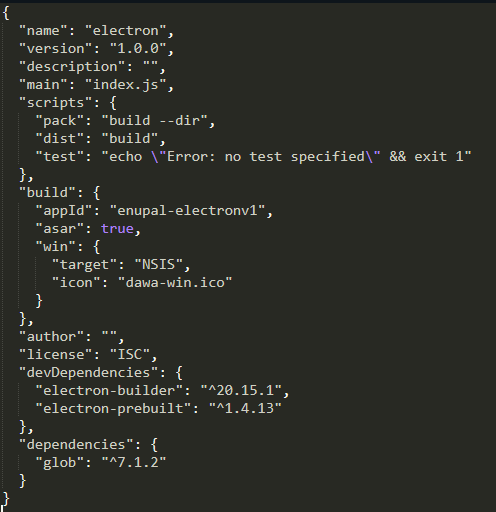
1. Creación de instalador
   1. Instalaremos el paquete electron-builder



* 1. Instalaremos adicionalmente el paquete glob



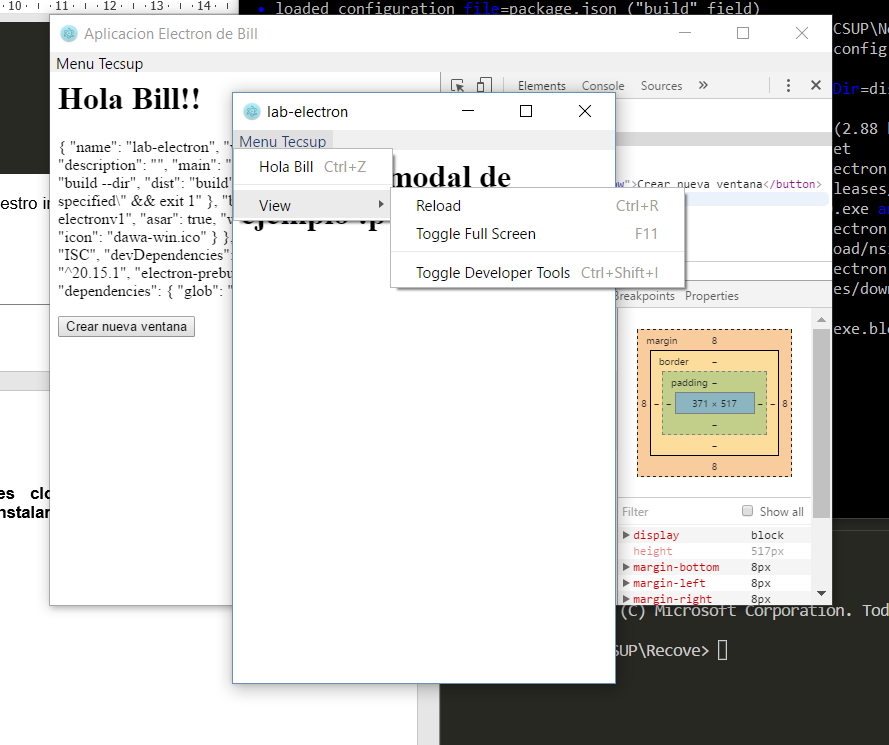
* 1. Modificaremos nuestro package.json para ue tenga la instrucción dist, que nos permitirá crear nuestro instalador



* 1. Finalmente, ejecutaremos el comando para crear nuestro instalador de aplicación, el cual se creará en la carpeta dist



**Para mayor información de electron, puedes clonar el siguiente repositorio** [**https://github.com/electron/electron-api-demos**](https://github.com/electron/electron-api-demos)**, instalarlo y ver en vivo cómo funcionan las principales características del proyecto**



1. **Finalizar la sesión**
   1. Apagar el equipo virtual
   2. Apagar el equipo