

Guía de instalación y deploy de React en Windows y Linux

Esta guía pretende ayudar en la instalación del software necesario para crear y desplegar (to deploy) nuestras aplicaciones React, tanto en Windows como en Linux



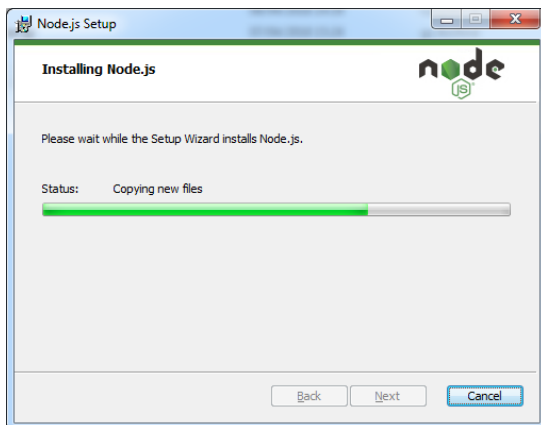
React es una librería de JavaScript para construir interfaces de usuario. No es un framework que trate de encargarse de todo, sino que trata de proveer solamente la “V” en los modelos MVC (Model-View-Controller), o sea, nos permite construir vistas de los modelos, que los llama componentes.

La guía tiene tres partes:

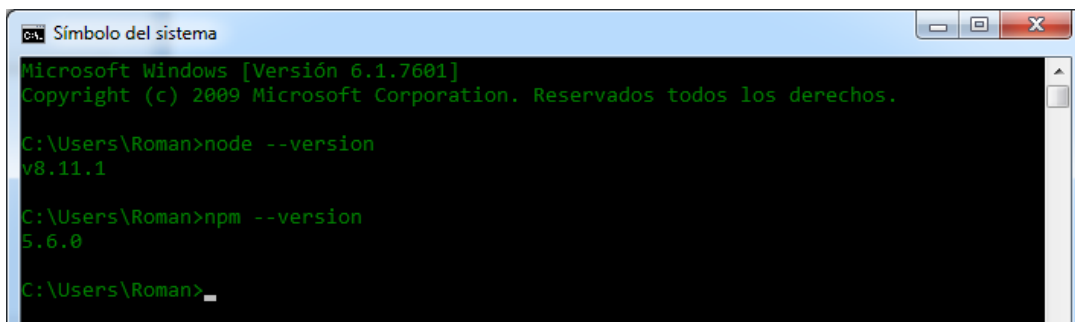
- a) Cómo crear y ejecutar proyectos React en Windows
- b) Cómo crear y ejecutar proyectos React en Linux (Mint)
- c) Cómo hacer deploy de nuestros proyectos en el servidor gratuito ZEIT

CÓMO CREAR Y EJECUTAR PROYECTOS REACT EN WINDOWS¹

1. Bajar e instalar la versión "Recomended for Most Users" de **node.js** de la página <https://nodejs.org>. El **npm** (node package manager) es el manejador de paquetes de node.js y viene incluido en esta versión. Npm permite instalar, actualizar, y desinstalar librerías y frameworks para usarlos en node.js. Esto se debe hacer una sola vez por máquina.

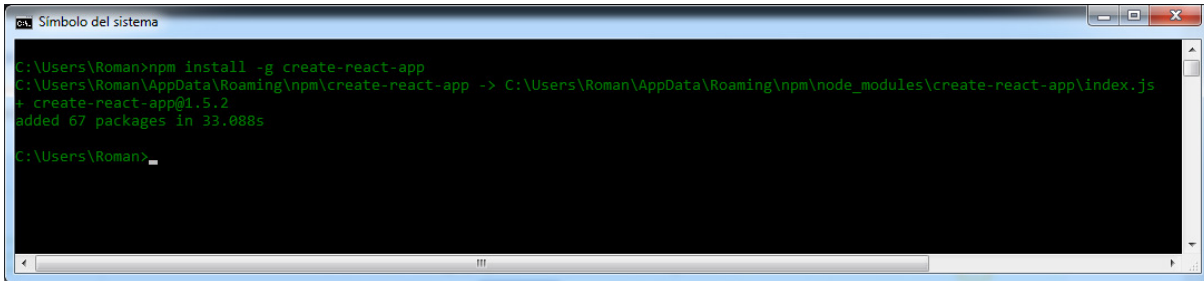


En la línea de comando verificar que se instaló correctamente. Para ello tipeamos:
node --version y **npm --version**



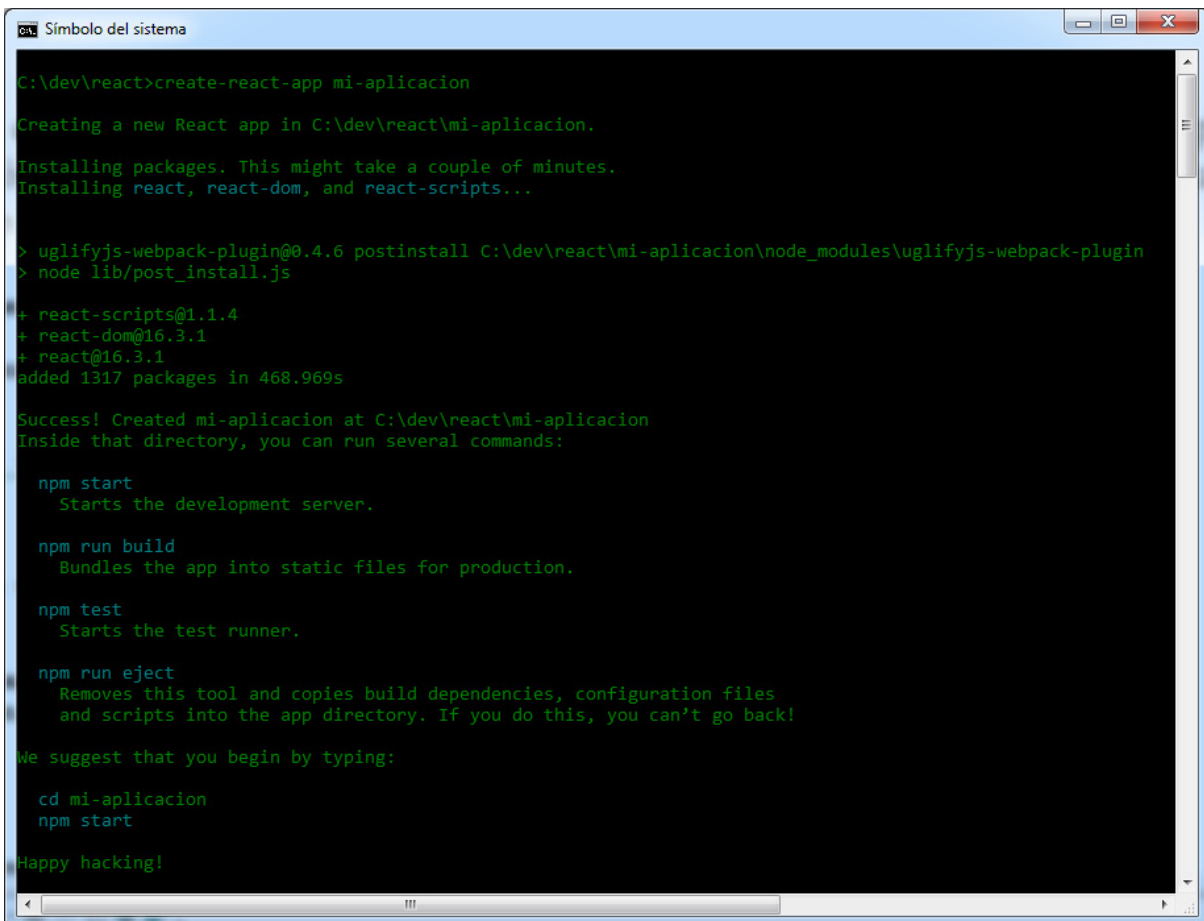
¹ Esta guía está basada una versión simplificada de <https://www.robinwieruch.de/react-js-windows-setup/> que he probado en mis máquinas.

2. Bajar con **npm** un paquete que genera los esqueletos de los proyectos React sin necesidad de usar archivos de configuración. Escribir en la línea de comandos: **npm install -g create-react-app** Esto se debe hacer una sola vez por máquina (la opción **-g** indica que es global).



```
Símbolo del sistema
C:\Users\Roman>npm install -g create-react-app
C:\Users\Roman\AppData\Roaming\npm\create-react-app -> C:\Users\Roman\AppData\Roaming\npm\node_modules\create-react-app\index.js
+ create-react-app@1.5.2
added 67 packages in 33.088s
C:\Users\Roman>
```

3. Crear nuestro proyecto React. Para ello nos movemos con el comando **cd** hasta el directorio que queremos contenga nuestro proyecto. Se creará un subdirectorio por cada nuevo proyecto que contendrá el código fuente, las librerías locales, el archivo **README**, el archivo **.gitignore** para git, etc. Tipeamos: **create-react-app mi-aplicacion** (no se permiten mayúsculas). Después de un rato deberíamos tener una pantalla como esta:



```
Símbolo del sistema
C:\dev\react>create-react-app mi-aplicacion
Creating a new React app in C:\dev\react\mi-aplicacion.
Installing packages. This might take a couple of minutes.
Installing react, react-dom, and react-scripts...

> uglifyjs-webpack-plugin@0.4.6 postinstall C:\dev\react\mi-aplicacion\node_modules\uglifyjs-webpack-plugin
> node lib/post_install.js

+ react-scripts@1.1.4
+ react-dom@16.3.1
+ react@16.3.1
added 1317 packages in 468.969s

Success! Created mi-aplicacion at C:\dev\react\mi-aplicacion
Inside that directory, you can run several commands:

  npm start
    Starts the development server.

  npm run build
    Bundles the app into static files for production.

  npm test
    Starts the test runner.

  npm run eject
    Removes this tool and copies build dependencies, configuration files
    and scripts into the app directory. If you do this, you can't go back!

We suggest that you begin by typing:

  cd mi-aplicacion
  npm start

Happy hacking!
```

4. El desarrollo de nuestra aplicación estará centrado en el archivo **App.js** que se encuentra en el subdirectorio **/src**.

Para un primer ejemplo, recomiendo reemplazar el contenido del archivo App.js por el siguiente, que es una versión del ejemplo de Carlos Lombardi con unos pequeños cambios:

```
import React, { Component } from 'react'

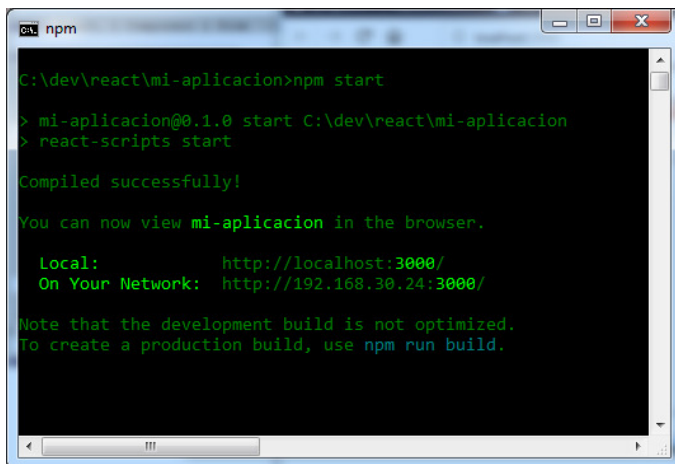
class App extends Component {
  constructor(props) {
    super(props);
    this.state = { texto: "Miren lo que va a pasar acá", tamañoFuente: "medium" }
  }

  render() {
    const theStyle = { fontSize: this.state.tamañoFuente }
    return (
      <div>
        <p><span style={ theStyle }>{ this.state.texto }</span></p>
        <button onClick={() => this.fillDemo()}>React magic</button>
      </div>
    )
  }

  fillDemo() { this.setState({ texto: "Hello React", tamañoFuente: "25px" }) }
}

export default App
```

5. Ahora ejecutemos nuestro proyecto. En la línea de comandos tipeamos: **npm start**

A terminal window titled 'npm' showing the output of the 'npm start' command. The output indicates that the application has been compiled successfully and is now running on localhost:3000. It also provides the URL for the application on the local network and a note that the development build is not optimized.

```
C:\dev\react\mi-aplicacion>npm start
> mi-aplicacion@0.1.0 start C:\dev\react\mi-aplicacion
> react-scripts start

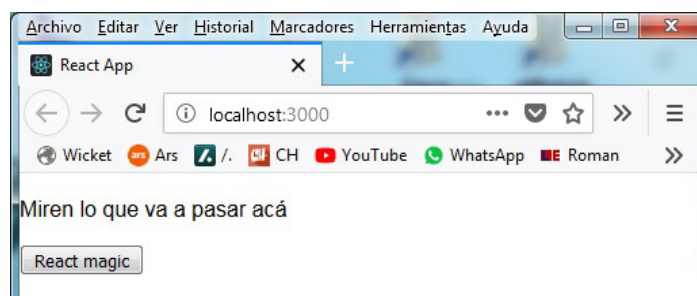
Compiled successfully!

You can now view mi-aplicacion in the browser.

Local:            http://localhost:3000/
On Your Network:  http://192.168.30.24:3000/

Note that the development build is not optimized.
To create a production build, use npm run build.
```

Esto creará un servidor de desarrollo en nuestra máquina en el puerto 3000 para nuestra aplicación y abrirá el browser por defecto a nuestra página. Lo mejor, es que *todos los cambios hechos y guardados en nuestra aplicación se actualizarán automáticamente en la página*.



CÓMO CREAR Y EJECUTAR PROYECTOS REACT EN LINUX MINT

En Linux Mint (un derivado de Ubuntu, y por lo tanto de Debian), los pasos son los siguientes.

- Para usar la última versión de node.js (9.11 en este momento):

```
curl -sL https://deb.nodesource.com/setup_9.x | sudo -E bash -  
sudo apt-get install -y nodejs
```

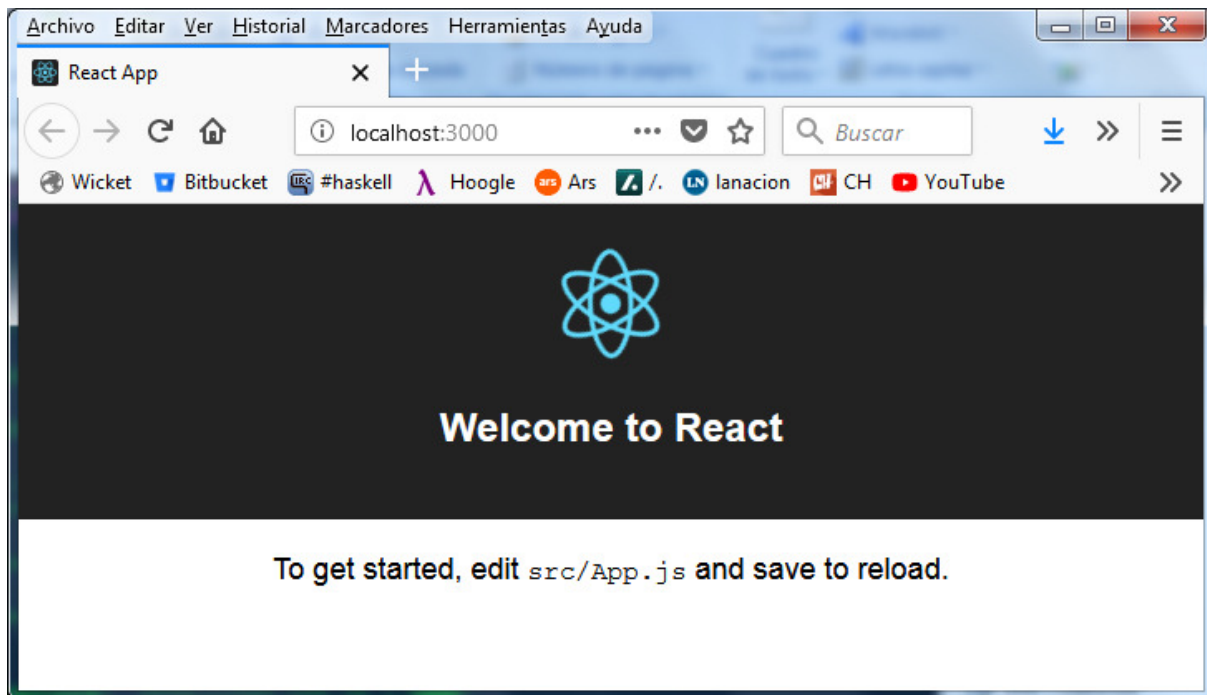
- Para usar la versión estable de node.js (8.11.1) (RECOMENDADA):

```
curl -sL https://deb.nodesource.com/setup_8.x | sudo -E bash -  
sudo apt-get install -y nodejs
```

Luego, instalar el script que genera proyectos React, crear un proyecto React, y levantarlo con un server de desarrollo

```
sudo npm install -g create-react-app  
create-react-app mi-aplicacion  
cd mi-aplicación  
npm start
```

Donde **mi-aplicacion** es el nombre del proyecto que queremos construir. Con estos comandos debe abrirse automáticamente el browser predeterminado en el puerto 3000 mostrando el proyecto por defecto.



Para comenzar a trabajar, editar el archivo **src/App.js**. Copiando el código del punto 5 de la guía para Windows, tenemos el primer ejemplo visto en clase.

CÓMO HACER EL DEPLOY DE NUESTROS PROYECTOS (WIN/LIN)



ZEIT es un servicio que permite instalar nuestras aplicaciones en los servidores “en la nube”. Para crearse una cuenta con servidores gratuitos debemos entrar en la página (<https://zeit.co/now>) donde nos preguntará por un email.

1. Instalar el comando **now** en nuestra máquina para subir nuestros proyectos a la nube ZEIT tipeando: `npm install -g now`

En la pantalla se puede ver dos intentos. En el primero se puede ver que vivo en Gral. Belgrano, pues se paró al 23% de la descarga. Si ello sucede, presionar CTRL-C y reintentar.

```
C:\Windows\system32\cmd.exe
C:\dev\react\mi-aplicacion>npm install -g now
C:\Users\Roman\AppData\Roaming\npm\now -> C:\Users\Roman\AppData\Roaming\n
> now@11.1.4 postinstall C:\Users\Roman\AppData\Roaming\npm\node_modules\n
> node download/install.js
> For the source code, check out: https://github.com/zeit/now-cli
> Downloading Now CLI 11.1.4 [=====] 23%¿Desea terminar el
C:\dev\react\mi-aplicacion>s
"s" no se reconoce como un comando interno o externo,
programa o archivo por lotes ejecutable.
C:\dev\react\mi-aplicacion>npm install -g now
C:\Users\Roman\AppData\Roaming\npm\now -> C:\Users\Roman\AppData\Roaming\n
> now@11.1.4 postinstall C:\Users\Roman\AppData\Roaming\npm\node_modules\n
> node download/install.js
> For the source code, check out: https://github.com/zeit/now-cli
> Downloading Now CLI 11.1.4 [=====] 100%
+ now@11.1.4
updated 1 package in 176.436s
C:\dev\react\mi-aplicacion>_
```

2. Ahora tipeamos **now** en la línea de comandos. La primera vez nos preguntará por el email que registramos la cuenta ZEIT. La segunda vez creará un servidor y nos copiará en el clipboard su dirección.

```
C:\Windows\system32\cmd.exe
C:\dev\react\mi-aplicacion>now
> Deploying C:\dev\react\mi-aplicacion under nykros
> Your deployment's code and logs will be publicly accessible because you are subscribed to the OSS plan.
> NOTE: You can use 'now --public' or upgrade your plan (https://zeit.co/account/plan) to skip this prompt
> Using Node.js 8.11.1 (default)
> https://mi-aplicacion-sefwybzwwl.now.sh [in clipboard] (sfo1) [16s]
> Synced 3 files (406.09KB) [16s]
> Building.
> ▲ npm install
> V Using "package-lock.json"
> ? Installing 3 main dependencies.
> ▲ npm run build
mi-aplicacion@0.1.0 build /home/nowuser/src
react-scripts build
Creating an optimized production build...
Compiled successfully.
File sizes after gzip:
  37.86 KB  build/static/js/main.bbe42d1c.js
  109 B    build/static/css/main.65027555.css
> The project was built assuming it is hosted at the server root.
> You can control this with the homepage field in your package.json.
> For example, add this to build it for GitHub Pages:
> "homepage": "http://myname.github.io/myapp",
> The build folder is ready to be deployed.
> You may serve it with a static server:
>   npm install -g serve
>   serve -s build
> Find out more about deployment here:
>
> http://bit.ly/2vY88Kr
> ▲ Snapshotting deployment
> ▲ Saving deployment image (34.7M)
> Build completed
> V Verified sfo1 (1) [33s]
> Success! Deployment ready!
C:\dev\react\mi-aplicacion>
```

3. Disfrutamos de nuestra súper aplicación:

