

Realizado por Mario Liberato y Andrés Segovia

Memoria Proyecto ASIR

Índice

[Anteproyecto 2](#_Toc161339006)

[Herramientas 3](#_Toc161339007)

# Anteproyecto

La idea de nuestro proyecto es crear una plataforma de streaming basándonos en las plataformas de streaming existentes como Twitch, Netflix, etc. Añadiéndole nuevas funciones tales como creación de listas de reproducción personalizadas, sugerencias de nuevos contenidos que la plataforma considere que te puedan interesar en función de tus gustos.

Utilizaremos principalmente PHP y CSS para la base de nuestra página web y nos apoyaremos en otro lenguaje llamado React, almacenaremos los datos de nuestros usuarios en una base de datos con MySQL donde se añadirá la información que vaya recogiendo la web.



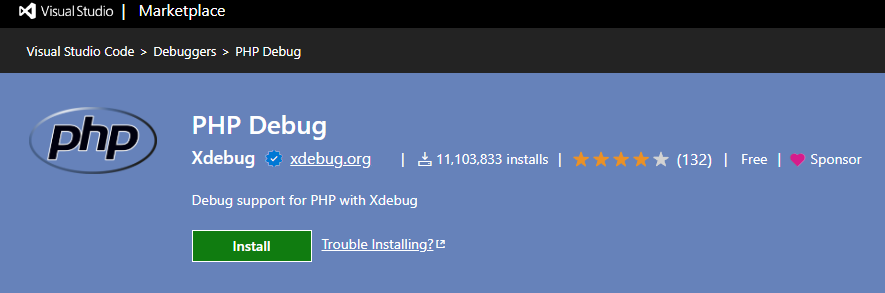
# Herramientas

Lo primero de todo tendremos que tener instalado **AppServ**, que nos proporcionará soporte tanto para Apache, PHP y MySQL.

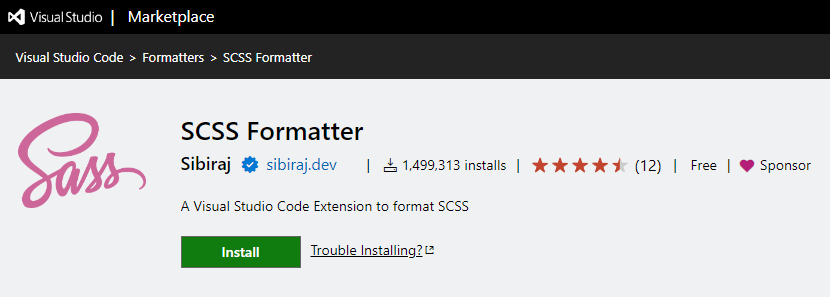
Podemos descargarlo desde la página principal de AppServ <https://www.appserv.org/en/>.

Como editor utilizaremos una herramienta llamada **Visual Studio Code**, que podemos descargar desde el siguiente enlace: <https://code.visualstudio.com/> y donde instalaremos varias extensiones que nos ayudarán a la hora de programar.

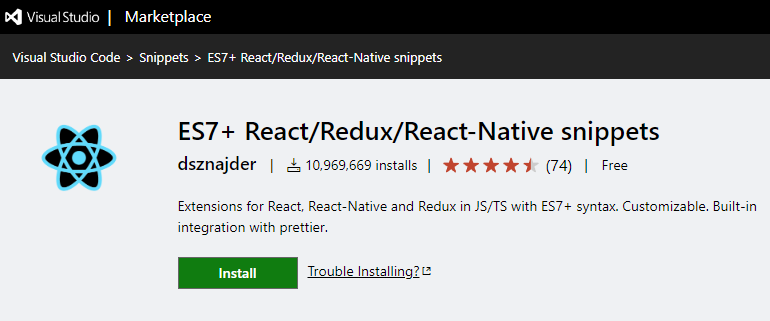
Las **extensiones** que vamos a utilizar son:

* **PHP Debug**: extensión que nos ayuda con la sintaxis de PHP.

<https://marketplace.visualstudio.com/items?itemName=xdebug.php-debug>.

* **SCSS Formatter**: extensión que nos ayuda con la sintaxis de archivos SCSS o SASS.

<https://marketplace.visualstudio.com/items?itemName=sibiraj-s.vscode-scss-formatter>.

* **ES7+ React/Redux/React-Native snippets:** extensión que nos ayuda con la sintaxis de archivos de React .jsx.

<https://marketplace.visualstudio.com/items?itemName=dsznajder.es7-react-js-snippets>.

Ahora que ya tenemos configurado nuestro editor vamos a instalar en nuestro equipo **Node.js**, que es un entorno de Javascript que nos permitirá junto con otras herramientas compilar los archivos .scss a .css y .jsx a .js.

Descarga: <https://nodejs.org/en>.

Abrimos una terminal e instalaremos 3 herramientas para poder compilar los archivos.

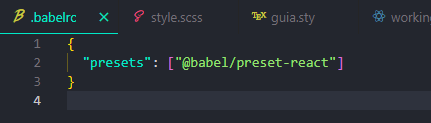
Las dos primeras que instalaremos serán **Babel** y **Nodemon**, que nos servirán para compilar los archivos de React a Javascript.

Las instalamos con el comando *npm install --save-dev @babel/core @babel/cli @babel/preset-react nodemon.*

**Warning**: Es importante instalarlo dentro del directorio del proyecto por lo que antes de instalarlo desde la terminal debes moverte al directorio del proyecto.

Una vez instaladas podemos utilizarlas con el siguiente comando para que todos los cambios que vayamos haciendo en nuestro archivo .jsx se reflejen en un archivo .js: *npx babel working.jsx --out-file working.js.*

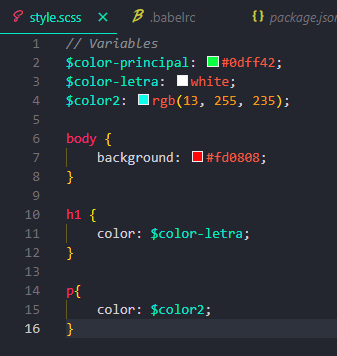
Donde working.jsx y working.js son los nombres de los archivos, este comando debemos utilizarlo cada vez que queramos añadir todos los cambios realizados en el archivo .jsx al archivo .js.

Después de instalarlo debemos crear dentro de nuestro directorio del proyecto un archivo .babelrc que podemos crear desde el Visual Studio y debemos añadir lo siguiente:

Con esto ya tenemos las librerías y herramientas necesarias para poder utilizar react y los archivos .jsx, ahora vamos a instalar las herramientas necesarias para poder utilizar los archivos .scss o .saas y convertirlos a archivos de css con extensión .css.

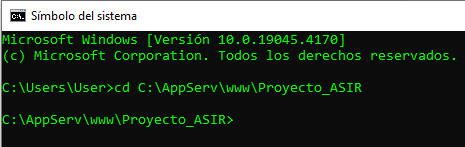
Lo primero que haremos será instalar **sass** en nuestro equipo, para ello ejecutamos desde la terminal *npm install –g sass* (no es necesario estar en el directorio del proyecto como con la herramienta anterior). Una vez instalado podemos ejecutar *sass –version* para verificar que se ha instalado.



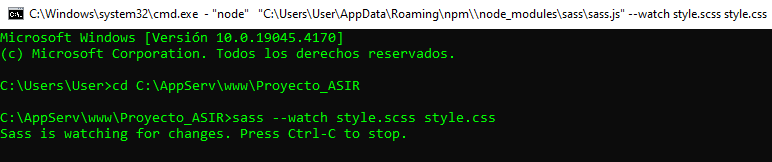
Una vez instalado vamos a crear un archivo style.scss donde podemos escribir algo de código para dar estilo a nuestra web, por ejemplo podemos escribir las siguientes líneas de código:

A simple vista podemos observar que estamos definiendo variables al comienzo, cosa que en CSS no se puede hacer. Esta es una de las muchas ventajas de utilizar este tipo de archivo.

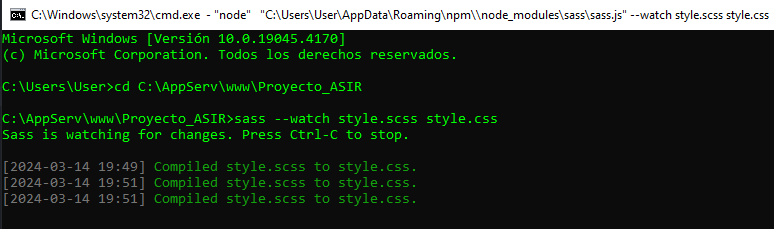
Ahora abrimos la terminal y nos movemos a la ruta donde tengamos el proyecto.



Y una vez dentro solo tendremos que ejecutar el siguiente comando *sass –watch style.scss style.css*.

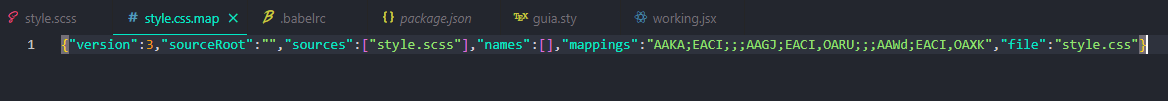
Nos aparecerá un mensaje como el siguiente:

También puede que la primera vez que ejecutemos este comando nos aparezca algún error del tipo “Error: Expected newline”, si nos apareceré este error no hay que darle importancia ninguna porque veremos que nos aparece debajo todas las líneas que introducimos anteriormente en nuestro archivo style.scss.

La terminal la podemos minimizar, pero no cerrar porque si no, se matará el proceso de conversión, si realizamos alguna modificación en el archivo style.scss veremos que en la terminal aparece un mensaje de que se ha compilado en el archivo style.css, y así con cada vez que modifiquemos el archivo se irá compilando los cambios sobre el archivo style.css.

Una vez que hayamos terminado de modificar el archivo style.scss podemos cerrar la terminal o finalizar el proceso con Ctrl + C.

Este proceso tendremos que repetirlo cada vez que vayamos a modificar el archivo style.cssc o cuando vayamos a crear otro archivo tendremos que ejecutar el comando anterior pero con el nuevo archivo .scss y el nuevo archivo .css a crear.

También veremos que aparte de crearnos el archivo de CSS nos creará un archivo con extensión .css.map que es un archivo de configuración para la compilación del archivo .scss a .css.