

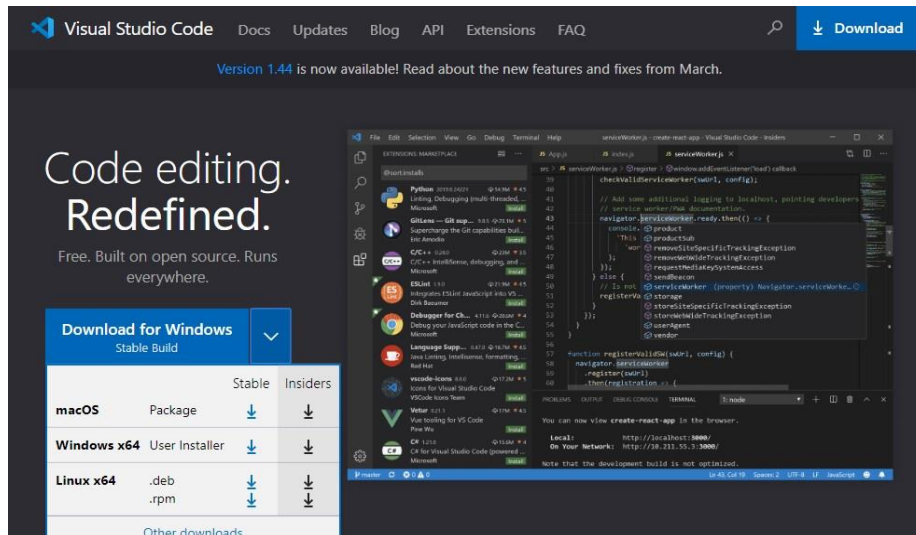
Trabajo de Visual Studio Code

Alex Chalo

**Instalación del editor VS
Code**

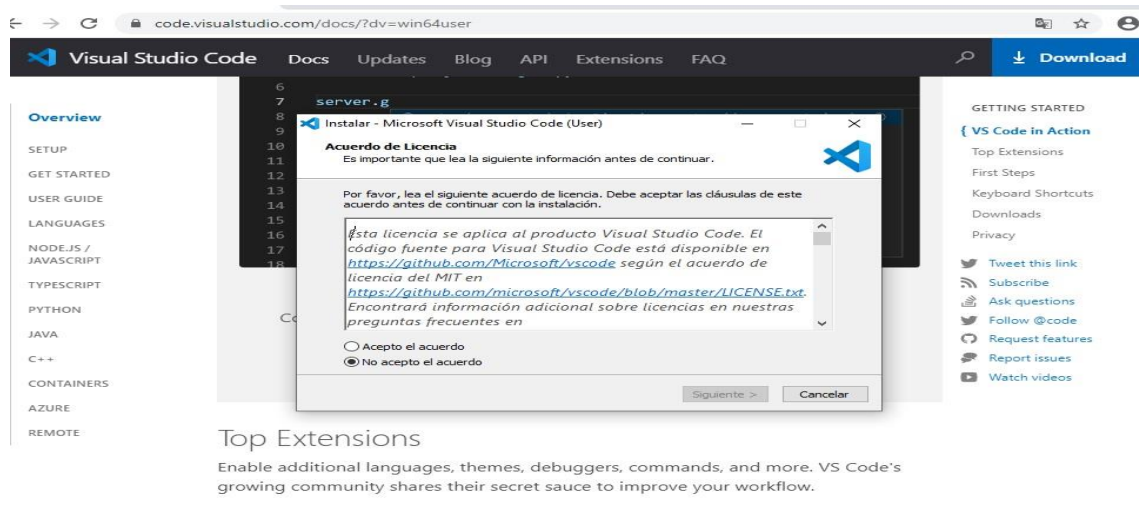
VS Code

El editor de texto VSCode ha sido creado y es mantenido por Microsoft. Lo distribuye con licencia Open Source y por lo tanto es gratuita. Lo podemos descargar de la página oficial y cómo podemos comprobar está disponible tanto para Windows, Mac y Linux:

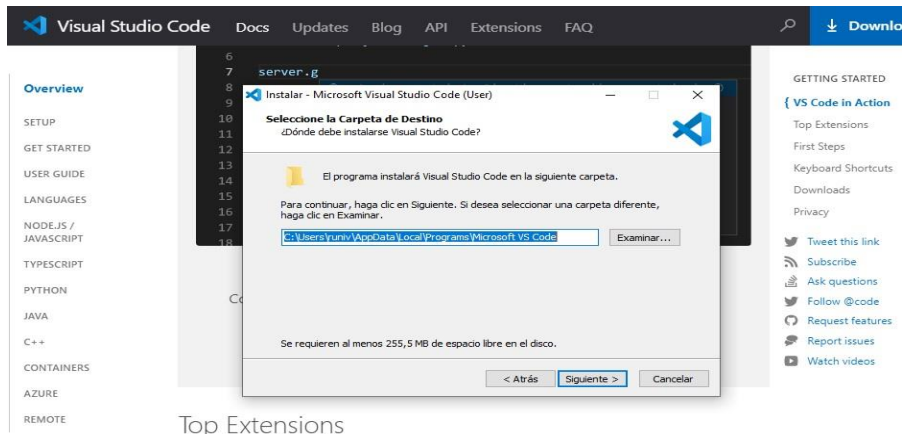


Una vez descargado procedemos a instalarlo siguiendo los pasos del asistente:

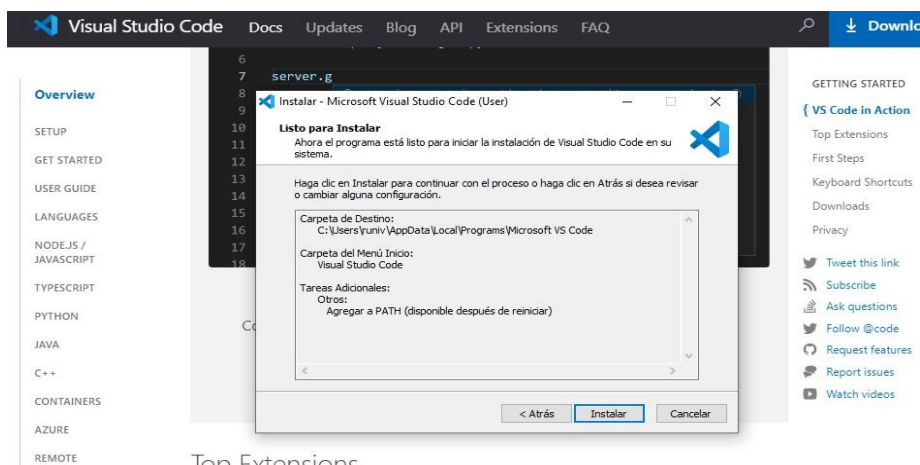
Le damos a siguiente, siguiente hasta finalizar la instalación.



VS Code

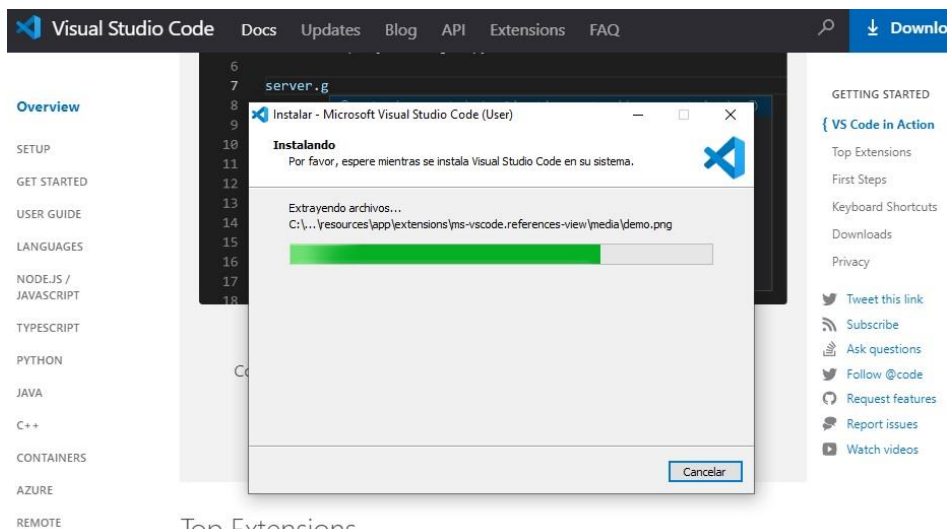


Top Extensions



Top Extensions

Enable additional languages, themes, debuggers, commands, and more. VS Code's

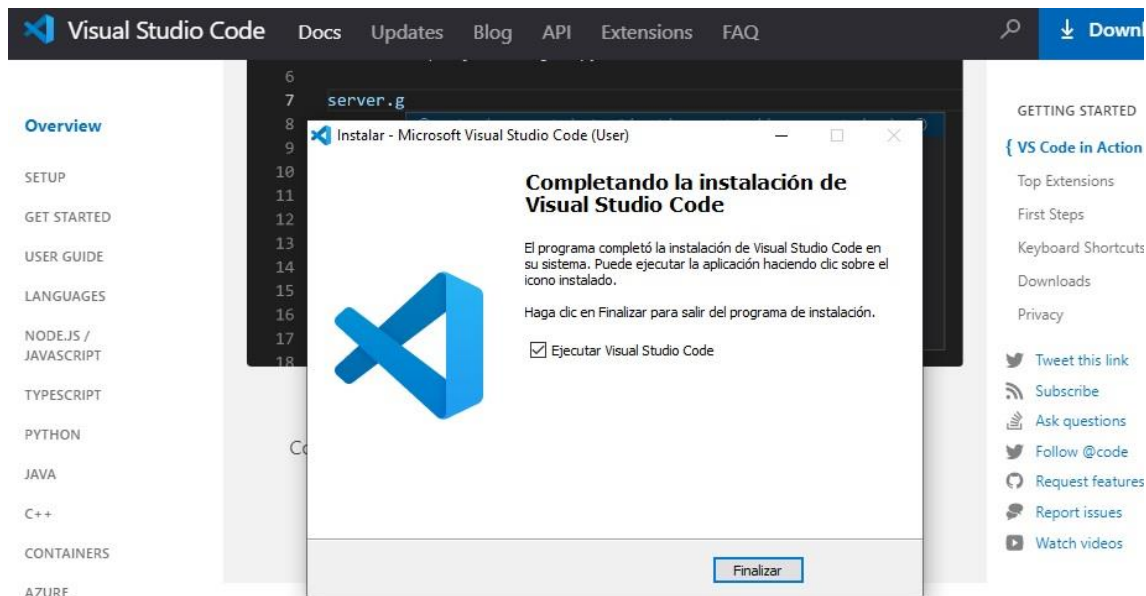


Top Extensions

Enable additional languages, themes, debuggers, commands, and more. VS Code's growing community shares their secret sauce to improve your workflow.

VS Code

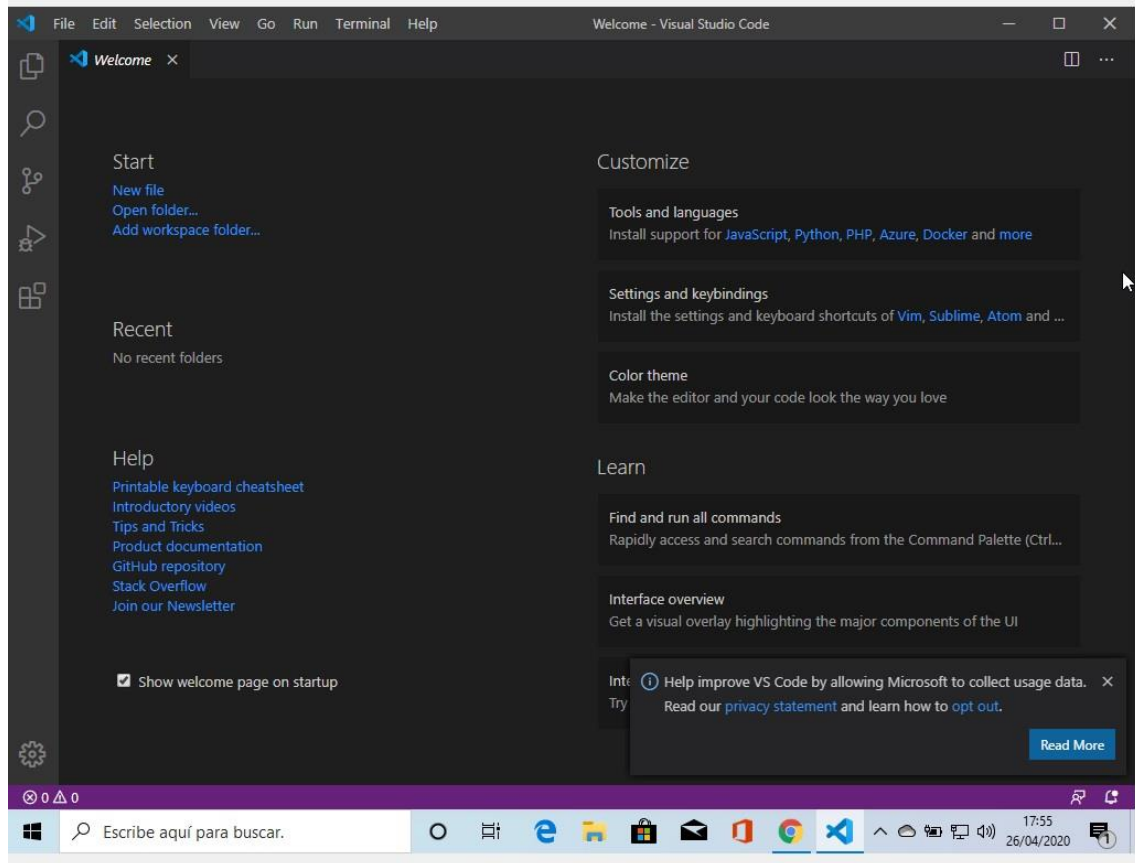
VS Code



Top Extensions

Enable additional languages, themes, debuggers, commands, and more. VS Code's growing community shares their secret sauce to improve your workflow.

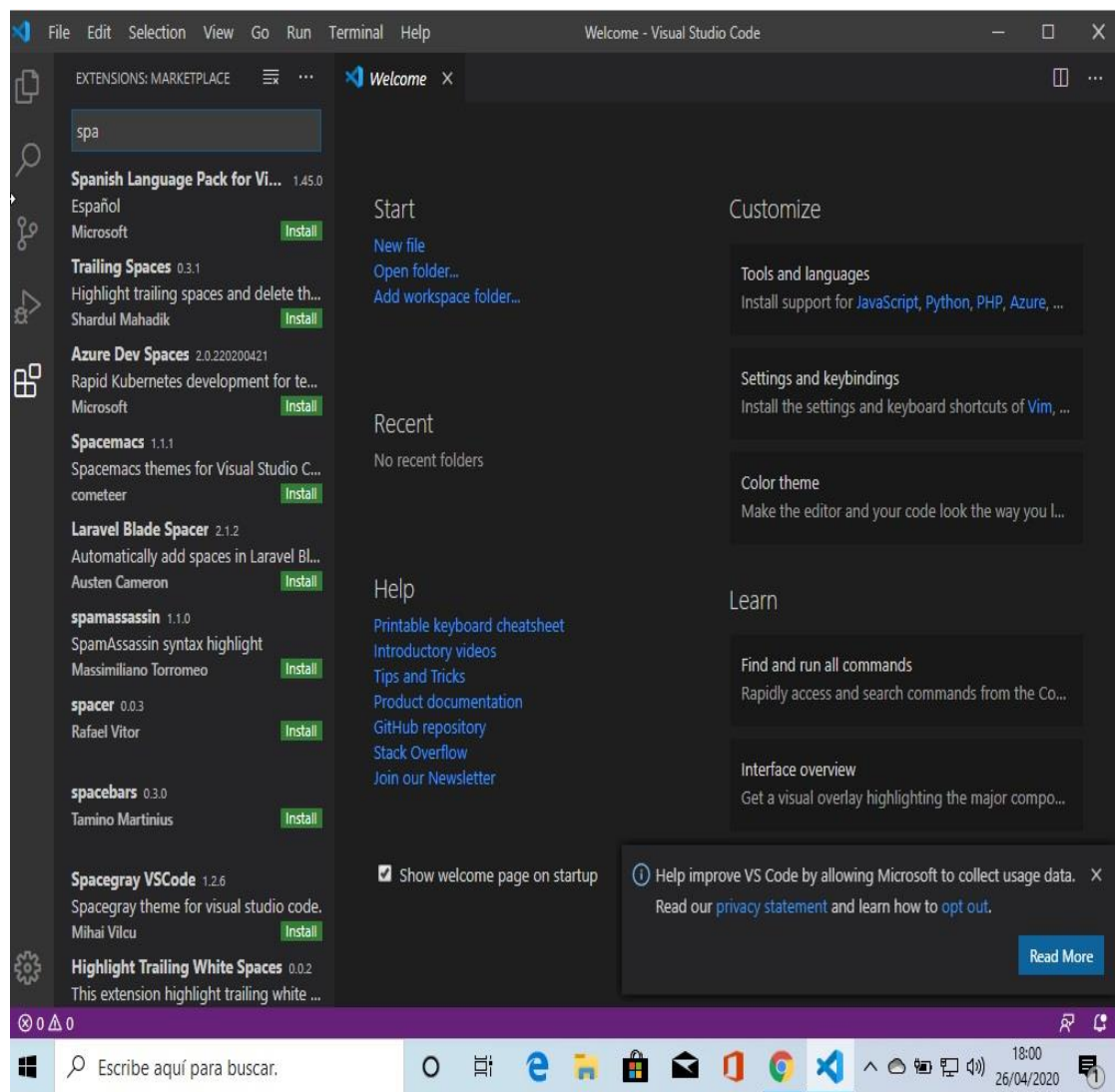
Ya tenemos instalado y funcionando VS Code:



Cambio de lenguaje

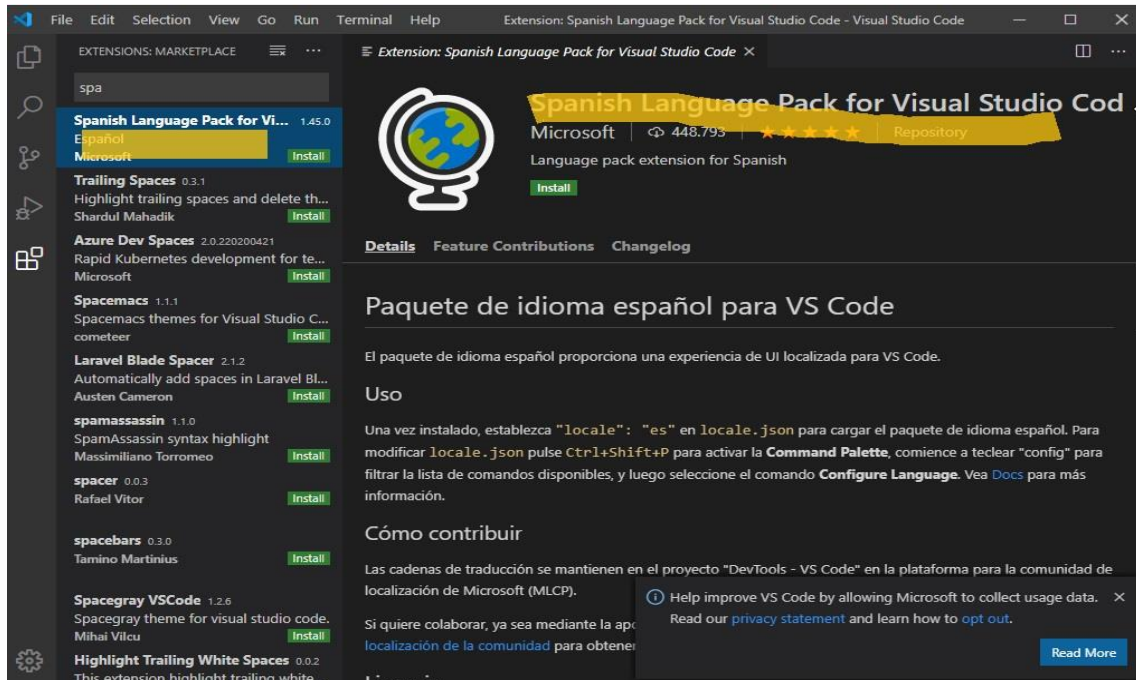
El lenguaje por defecto es el inglés, lo primero que haremos es cambiar a español, instalando la extensión de VS Code para tal fin. Presionamos el ícono de la izquierda para administrar extensiones y buscamos 'spanish'.

Como podemos ver en las imágenes.

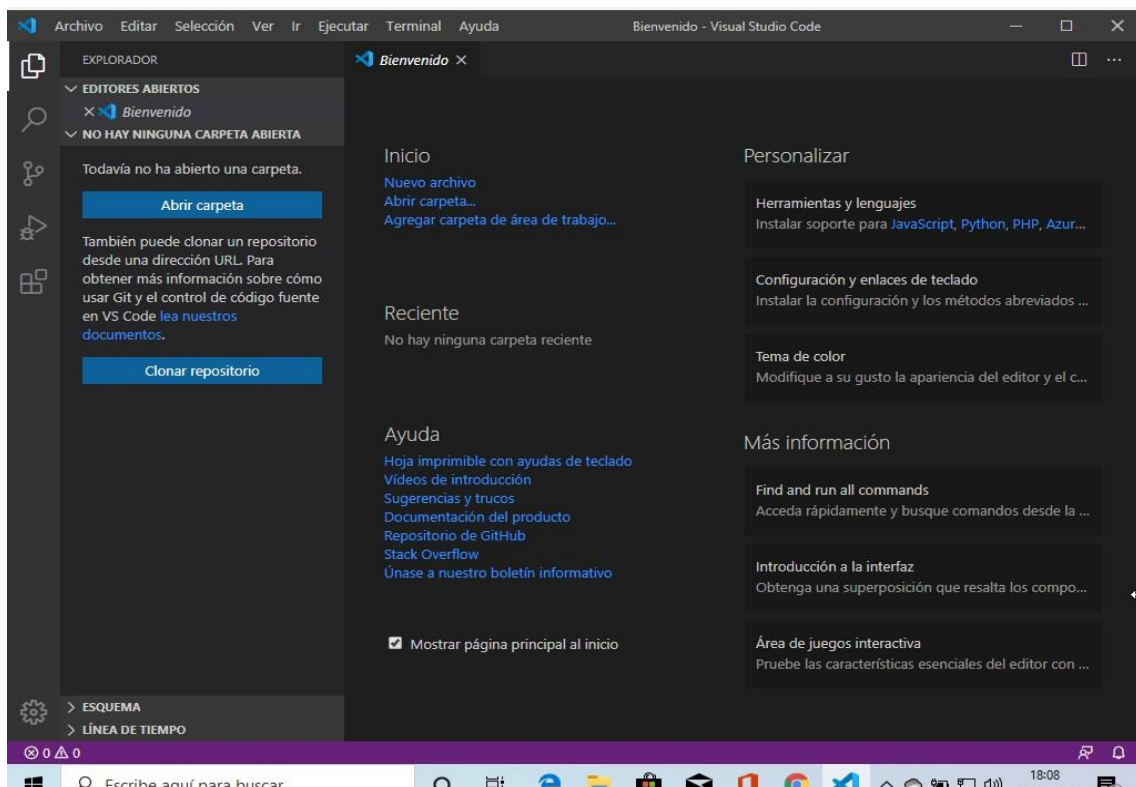


VS Code

Si queremos cambiar de lenguaje hacemos lo siguiente:
instalamos el plugin para el cambio de idioma, como se puede
ver en la imagen.



Como podemos ver ya está en español.



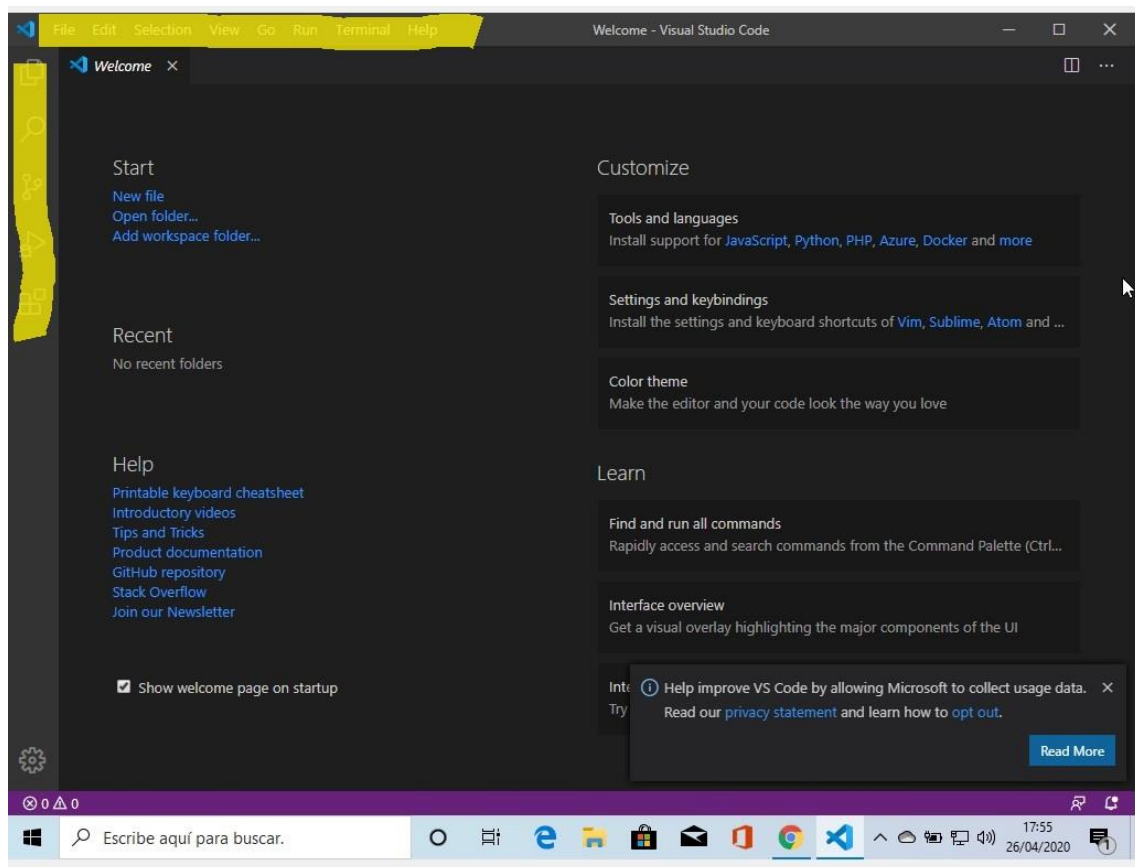
Partes que se ven en vs code

Al iniciar nuestro vs code podemos ver los siguientes:

Los menús en la parte superior.

En toda la parte izquierda encontraremos lo siguiente:

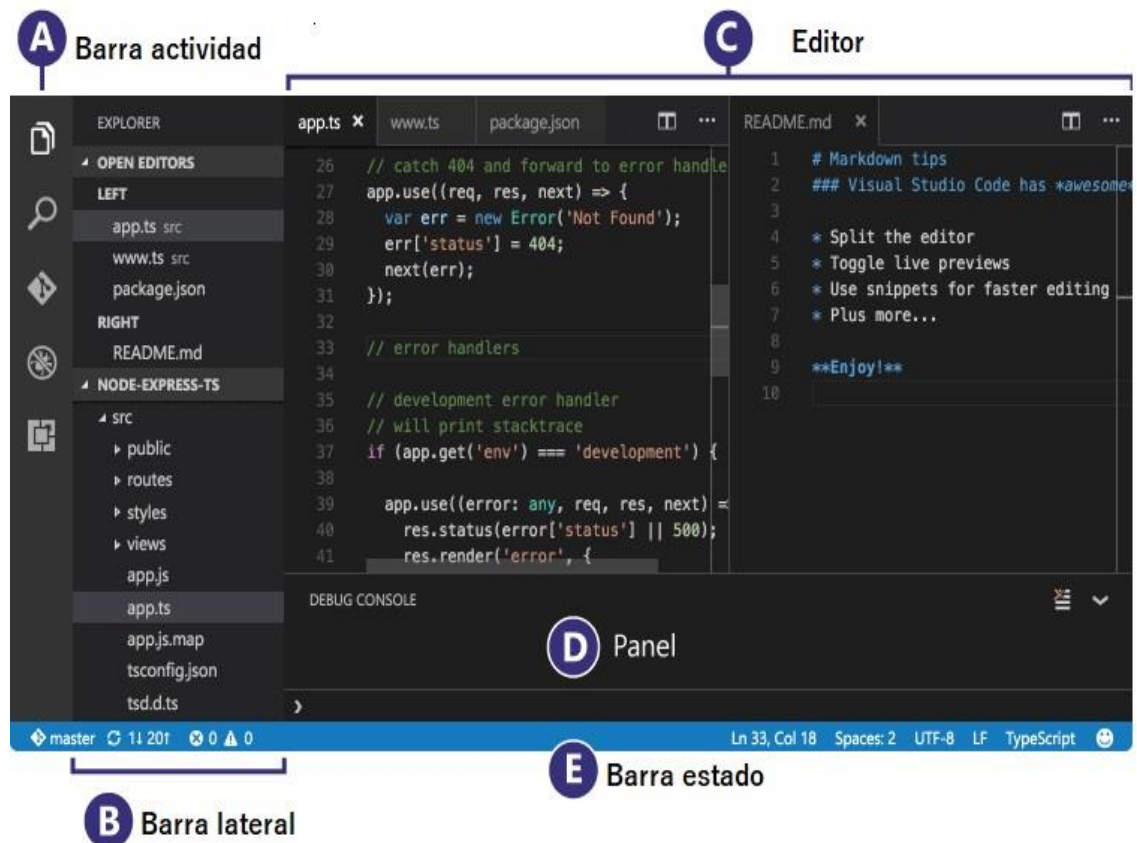
Explorador, buscador de archivos, control de código fuente, depurador y el instalador de extensiones.



Diseño básico

VS Code viene con un diseño simple e intuitivo que maximiza el espacio provisto para el editor mientras deja un amplio espacio para navegar y acceder al contexto completo de su carpeta o proyecto. La interfaz de usuario se divide en cinco áreas:

VS Code



- **Editor**: el área principal para editar sus archivos. Puede abrir tantos editores como desee, uno al lado del otro, vertical y horizontalmente.
- **Barra lateral**: contiene diferentes vistas, como el Explorador, para ayudarlo mientras trabaja en su proyecto.
- **Barra de estado**: información sobre el proyecto abierto y los archivos que edita.
- **Barra de actividad**: ubicada en el extremo izquierdo, le permite cambiar de vista y le brinda indicadores adicionales específicos del contexto, como la cantidad de cambios salientes cuando Git está habilitado.
- **Paneles**: puede mostrar diferentes paneles debajo de la región del editor para obtener información de salida o depuración, errores y advertencias, o un terminal integrado. El panel también se puede mover hacia la derecha para obtener más espacio vertical.

VS Code

Cada vez que inicia VS Code, se abre en el mismo estado en que estaba la última vez que lo cerró. La carpeta, el diseño y los archivos abiertos se conservan.

Los archivos abiertos en cada editor se muestran con encabezados con pestañas (Pestañas) en la parte superior de la región del editor.

Consejo: puede mover la barra lateral hacia el lado derecho (Ver > **Mover barra lateral derecha**) o alternar su visibilidad (**Ctrl + B**).

Como crear un proyecto java en vs code

1. Descargue VS Code y Apache Maven e instale ambos.
2. Instale el paquete de extensión de Visual Studio para Java; por ejemplo, pegue esta URL en un navegador web: `vscode:extension/vscjava.vscode-java-pack` y luego haga clic en el botón verde Instalar después de que se abra en el Código VS.
3. Si es necesario, el arquetipo de inicio rápido de Maven se podría usar para generar un nuevo proyecto de Maven en una carpeta local apropiada: `mvn archetype:generate -DgroupId= -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false` Esto creará una carpeta de nombre de aplicación con el diseño de directorio estándar de Maven (es decir, `src/main/java/com/companyname/appname` y `src/main/test/com/companyname/appname` para comenzar y un archivo Java de muestra "Hello World!" Llamado ***appname***.java y la prueba de unidad asociada llamada ***appname*** Test.java).

VS Code

4. Abra la carpeta del proyecto Maven en VS Code a través del menú Archivo -> Abrir carpeta ... y seleccione la carpeta ***appname*** .
5. Abra la paleta de comandos (a través del menú Ver o haciendo clic con el botón derecho) y escriba y seleccione Tasks: Configure task luego seleccione Create tasks.json from template .
6. Elija maven (“Ejecuta comandos comunes de Maven”). Esto crea un archivo tasks.json con las tareas “verificar” y “probar”. Se pueden agregar más correspondientes a otras fases de Maven Build Lifecycle . Para abordar específicamente su requisito de que las clases se construyan sin un archivo JAR, una tarea de “comstackción” deberá agregarse de la siguiente manera:

```
{ "label": "compile", "type": "shell", "command": "mvn -B compile", "group": "build" },
```
7. Guarde los cambios anteriores y luego abra la Paleta de comandos y seleccione “Tareas: Ejecutar tarea de comstackción”, luego seleccione “comstackr” y luego “Continuar sin escanear el resultado de la tarea”. Esto invoca a Maven, que crea una carpeta de target al mismo nivel que la carpeta src con los archivos de clase comstackdos en la carpeta de target\classes .

Funcionalidades de vs code

Visual Studio Code es un editor de código fuente que permite trabajar con diversos lenguajes de programación, admite gestionar tus propios atajos de teclado y refactorizar el código. Es gratuito, de código abierto y nos proporciona una utilidad para descargar y gestionar extensiones con las que podemos personalizar y potenciar esta herramienta.

VS Code

Las **extensiones de Visual Studio Code** nos otorgan infinidad de opciones, como colorear tabulaciones, etiquetas o recomendaciones de autocompletado. También hay extensiones que nos ayudan con el lenguaje de programación que vayamos a usar, como por ejemplo para Python, C / C++, JavaScript, etc.

A continuación, se detallan algunas extensiones interesantes para el desarrollo web en Visual Studio Code:

- **Better Haml**: Extensión que nos ayuda con la sintaxis y autocompletado de Haml.
- **YML (Yseop Markup Language)**: Provee soporte para YML, dando color y atajos.
- **HTML Snippets**: Añade colores a las etiquetas HTML y atajos para este lenguaje.
- **Beautify**: Deja un código más atractivo y facilita su lectura.

Funcionalidad de las pestañas

Estas son algunas de las funcionalidades y novedades que incluye esta versión de Visual Studio Code:

- **Pestañas**: Un editor de pestañas que permite navegar entre archivos y organizar nuestra área de trabajo.
- **Extensiones**: nos permite gestionar, ver e instalar extensiones
- **Área de trabajo**: Mejorada para arrastrar y soltar, posibilidad de guardar los paneles de edición.
- **Editor**: Búsqueda global, remplazar, guías, problemas en los paneles para ver errores y advertencias.
- **Lenguajes**: Emmet dispone de un soporte mejor y más completo y para la Atom JavaScript
- **Debug**: Mejoras incluyendo el soporte para adjuntar Node.js a los procesos en funcionamiento y cambiar valores en las variables mientras se realiza una sesión de debug.

- **Autorización a las extensiones:** Un nuevo punto de contribución para las barras de menú y los menús contextuales. Hay nuevas APIs para abrir recursos que no sean de texto y se ha añadido un editor de adjuntos.

Atajos de teclados vs code

A continuación, veremos una **selección** de atajos de teclado de vs Code:

- Abrir archivo: CTRL + O
- Abrir carpeta: CTRL + K + CTRL + O
- Guardar archivo: CTRL + S
- Crear un nuevo archivo: CTRL + N
- Cerrar archivo abierto: CTRL + W
- Deshacer: CTRL + Z
- Rehacer: CTRL + Y
- Buscar en archivo abierto: CTRL + F
- Buscar en archivos: CTRL + SHIFT + F
- Reemplazar: CTRL + H
- Seleccionar todo: CTRL + A
- Pantalla completa: F11
- Acercar Zoom: CTRL + +
- Alejar Zoom: CTRL + -
- Abrir paleta de comandos: CTRL + SHIFT + P
- Abrir explorador de archivos: CTRL + SHIFT + E
- Acceder a extensiones (instaladas y para instalar): CTRL + SHIFT + X
- Debug: CTRL + SHIFT + D
- Corte de palabras (Word Wrap): ALT + Z
- Abrir terminal integrado: CTRL + SHIFT + Ñ
- Ver/editar atajos de teclado: CTRL + K + CTRL + S

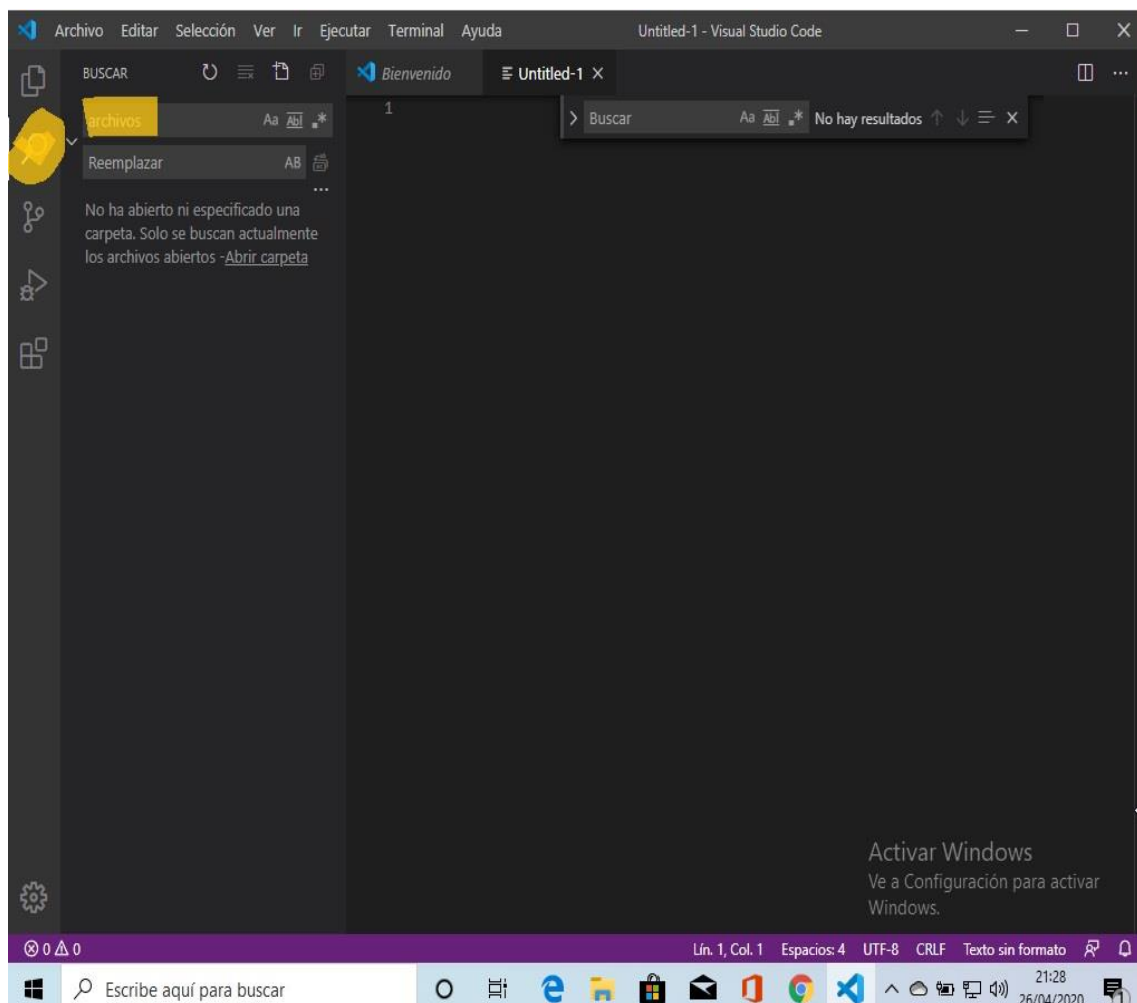
VS Code

Vale aclarar que tal como vemos en el último ítem, desde Visual Studio Code es posible acceder para ver todos los atajos del teclado y también para editarlos y configurarlos a nuestro gusto.

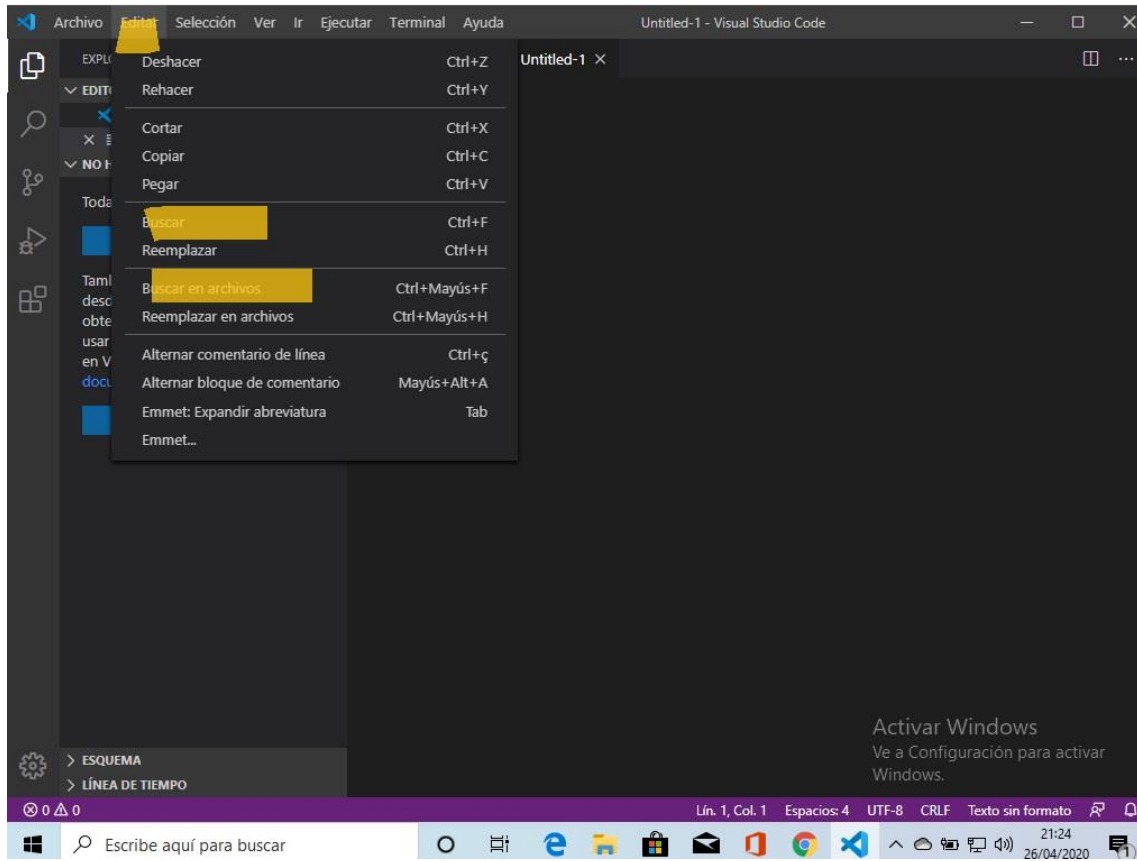
Búsqueda de archivos en vs code

Tengo conocimientos de dos formas de búsquedas de archivos y son:

La primera: le damos un clic a la lupa como se ve en la imagen.



La segunda: le damos a editar buscar o buscar archivos.



Atajos de teclados

En Windows presione Ctrl+p o Ctrl+e

En la prensa de mac Cmd+p

en la prensa de Linux también Ctrl+p trabajos

Versiones anteriores de Mac Press Cmd+e

A continuación, escriba el nombre del archivo

Historial de cambio de archivo

Visual Studio Code tiene varios conjuntos de comandos git para ejecutar y realizar varias tareas para su repositorio de código. Sin embargo, si está buscando visualizar el historial de git, puede usar la Extensión de historial de Git de Don Jayamanne. Esta es una extensión excelente y ampliamente utilizada para Visual Studio Code que muestra detalles de confirmación, ver copias anteriores de los archivos, cambios a la línea activa en el editor, etc.

Con esta extensión, encontrará varios conjuntos de características para su repositorio Git, como

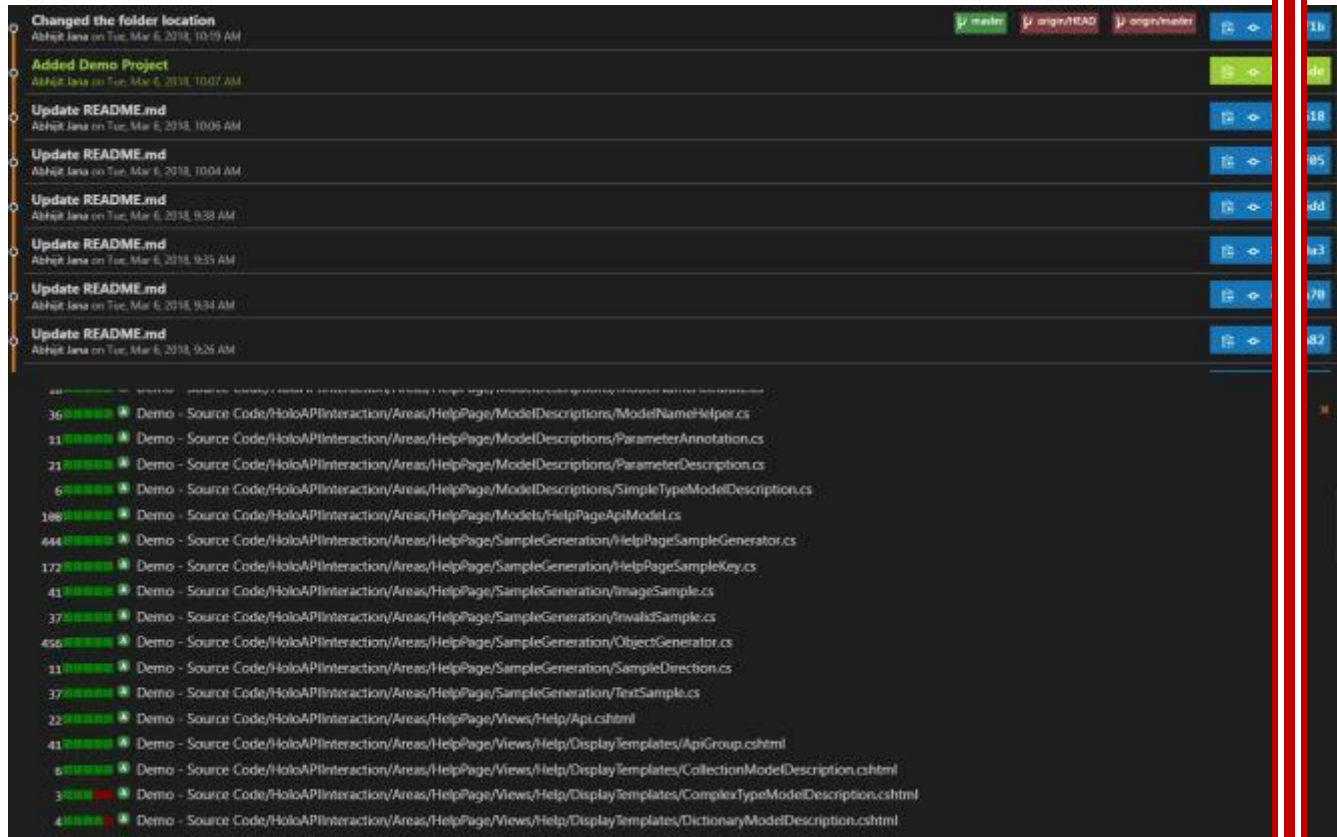
- Ver detalles de cada confirmación en una jerarquía de visualización
- Ver el historial de un archivo o línea
- Compare una copia anterior de un archivo con la versión del espacio de trabajo local.
- Ver los cambios en la línea activa

Cuando tiene el repositorio abierto en el código de Visual Studio, puede ejecutar el comando Git: Ver historial (git log) desde el comando Plate. Esto abrirá la ventana del historial de Git con todos los registros de cambios para el repositorio.



VS Code

Luego puede seleccionar confirmaciones individuales para registros de cambios detallados para cada una de ellas.



Esta extensión también te ayuda en Ver y buscar en el registro de git junto con los detalles.

Comentarios y formateos

Comentarios

1. Shift + Alt + A. ...
2. Selecciona las líneas que quieres **comentar**/descomentar.
...
3. El **atajo** de teclado para **comentar** múltiples en Windows es shift + alt + A
4. Cómo **comentar** hacia fuera multilínea/línea única en el Código **VS**: ...
5. En mi caso, Ubuntu, el **atajo** es ctrl + shift + A .

Como formatear

El formato del código está disponible en Visual Studio Code a través de los siguientes accesos directos:

- En Windows Shift + Alt + F
- En mac Shift + Option + F
- En Ubuntu Ctrl + Shift + I

Alternativamente, puede encontrar el acceso directo, así como otros accesos directos, a través de la funcionalidad de búsqueda provista en el editor con Ctrl + Shift + P (o Command + Shift + P en Mac), y luego busque ***documento de formato***.

Refactor

Esta refactorización se aplica a lo siguiente:

- C#
- Visual Basic

Qué: Le permite cambiar el nombre de los identificadores de símbolos de código, como campos, variables locales, métodos, espacios de nombres, propiedades y tipos.

Cuándo: Desea cambiar con seguridad algo sin tener que buscar todas las instancias y copiar y pegar el nuevo nombre.

Por qué: Es probable que copiar y pegar el nuevo nombre en todo un proyecto dé lugar a errores. Esta herramienta de refactorización llevará a cabo con precisión la acción de cambio de nombre.

Procedimiento

1. Resalte o coloque el cursor de texto dentro del elemento cuyo nombre se va a cambiar:

VS Code

- **C#:**

```
static void Main(string[] args)
{
    double r = 1.23;

    double area = Math.PI * r * r;
}
```

- **Visual Basic:**

```
Dim r As Double = 1.23
```

```
Dim area As Double = Math.PI * r * r
```

2. A continuación, realice alguno de los siguientes procedimientos:

- **Teclado**

- Presione **CTRL+R** y, a continuación, **CTRL+R**. (Tenga en cuenta que su método abreviado de teclado puede ser diferente en función del perfil que haya seleccionado).

- **Mouse**

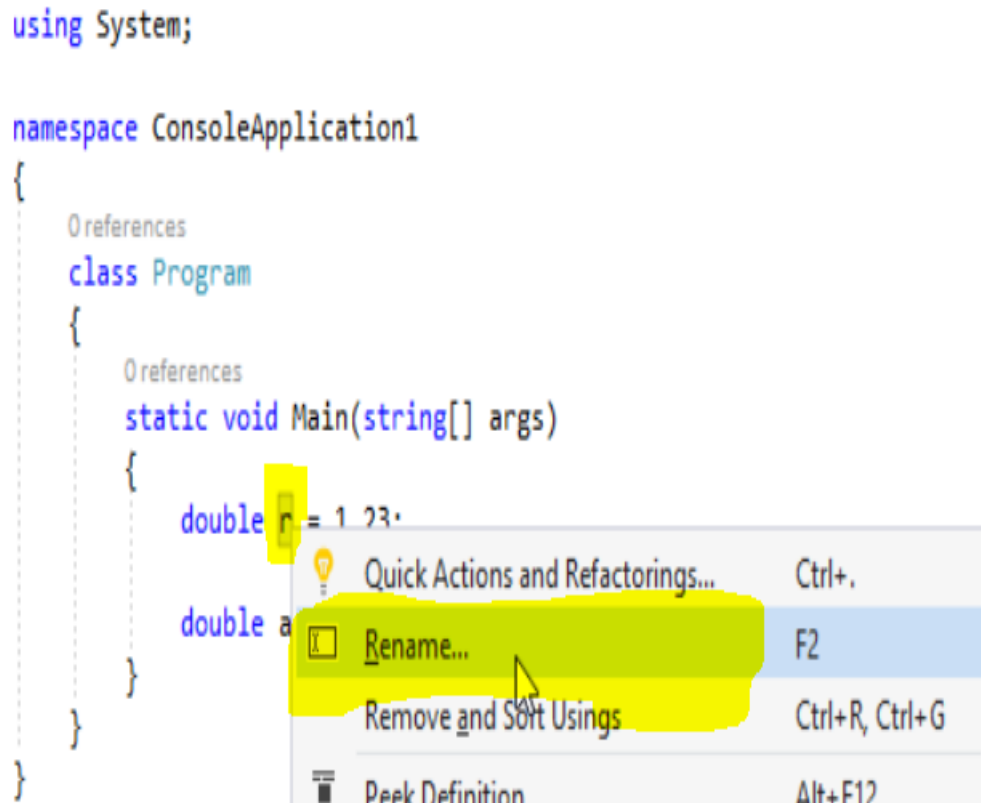
- Seleccione **Editar > Refactorizar > Cambiar nombre**.
- Haga clic con el botón derecho en el código y seleccione **Cambiar nombre**.

3. Cambie el nombre del elemento escribiendo el nuevo nombre.

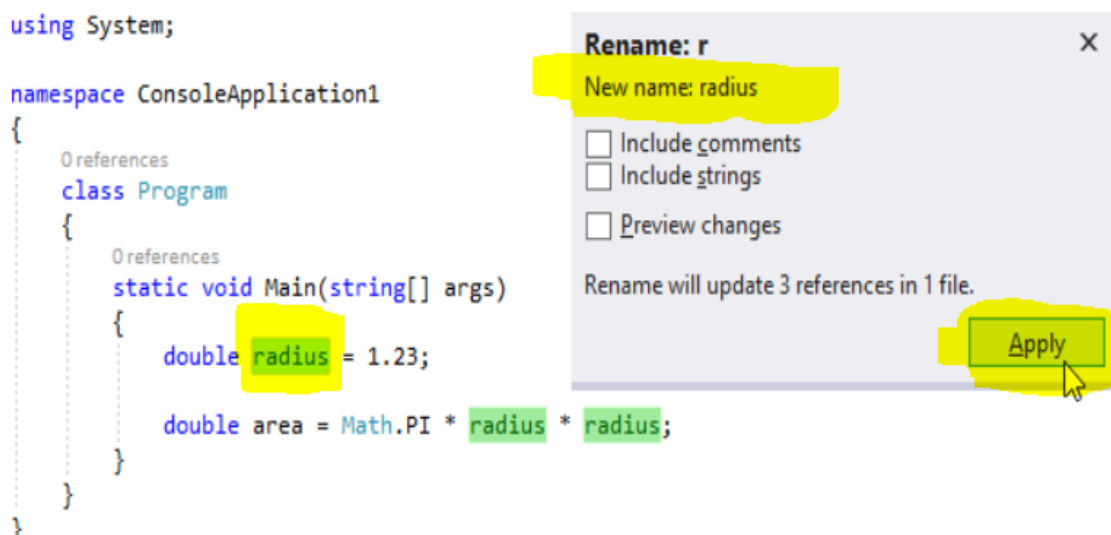
4. Como veremos a continuación.

Seleccionamos lo que queremos cambiar y le damos a rename.

VS Code



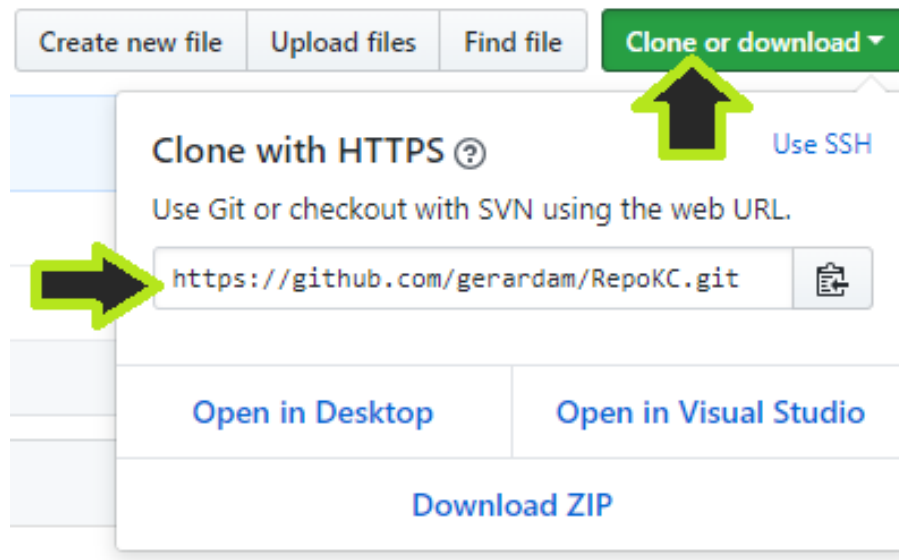
Una vez escrito lo que reemplacemos le damos a aplicar como vemos en la imagen.



Cuando esté satisfecho con el cambio, seleccione el botón **Aplicar** o presione **Entrar**. Los cambios se confirmarán.

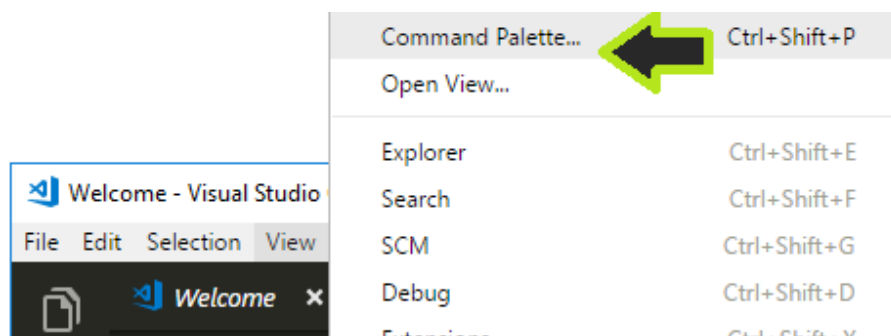
Como incluir y usar Git

Entramos a la página de GitHub y nos dirigimos al repositorio creado (como lo vimos en el post anterior), buscamos la opción de “Descargar” y copiamos la URL del repositorio.



Conectar vs Code con GitHub

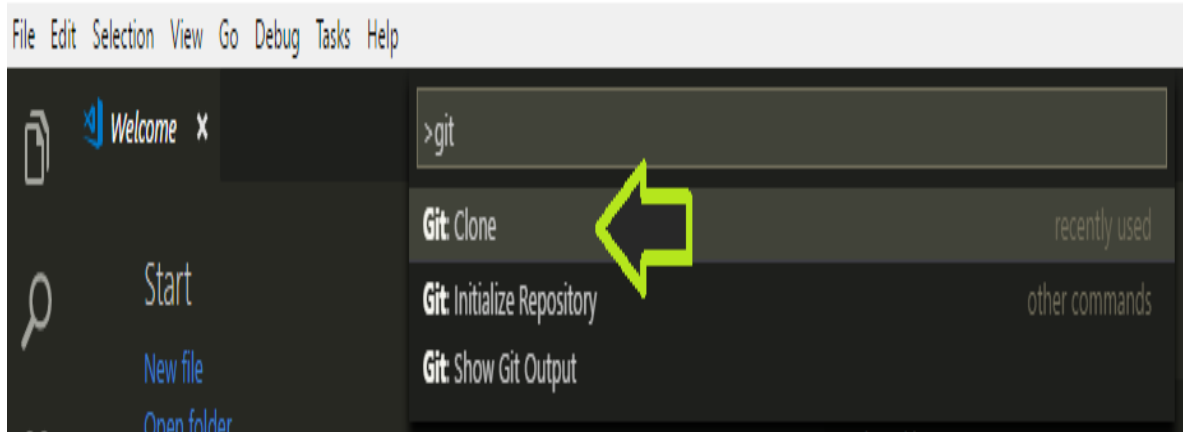
Abrimos visual studio code, en la barra de herramientas seleccionamos “vista”, buscamos y damos clic en la opción de “paleta de comandos”.



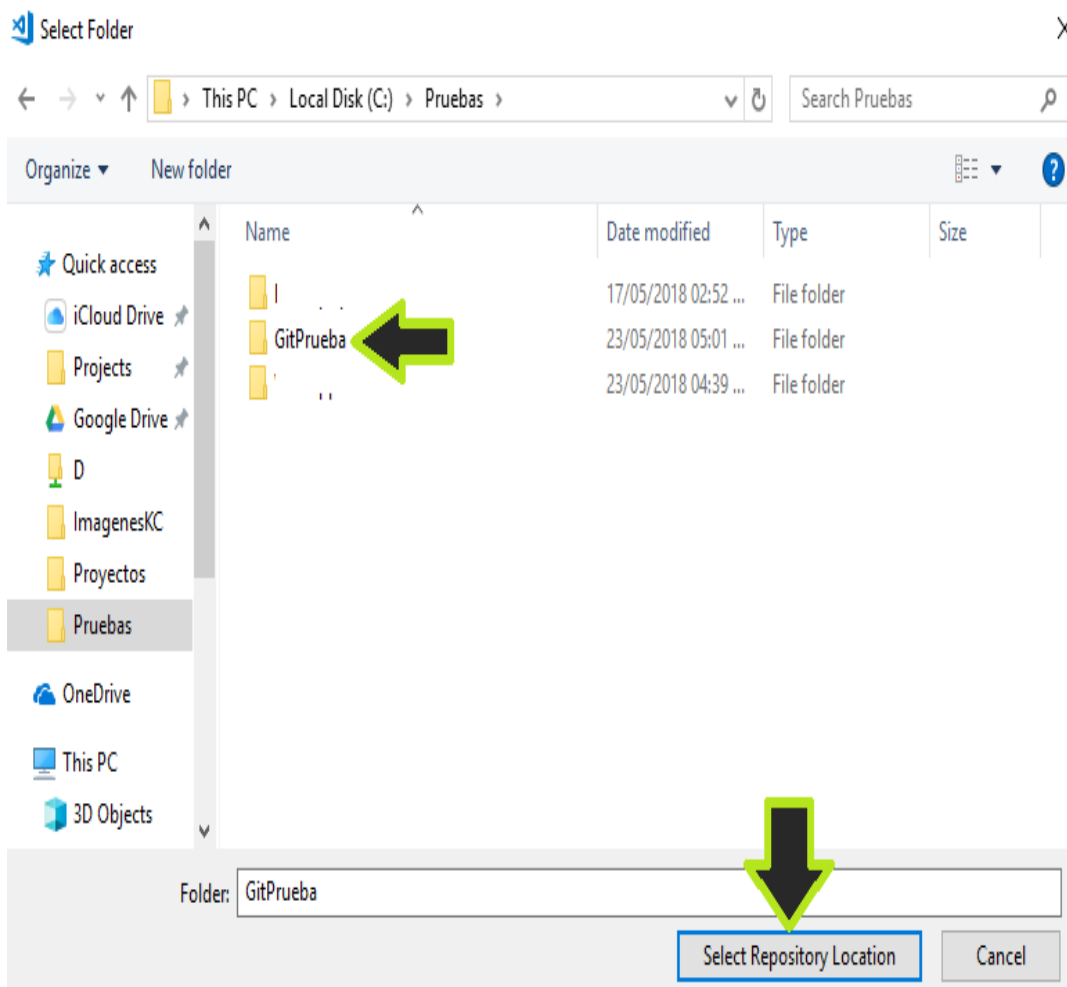
En el campo escribimos git, en los resultados seleccionamos la opción de “Git clone”.

VS Code

Welcome - Visual Studio Code



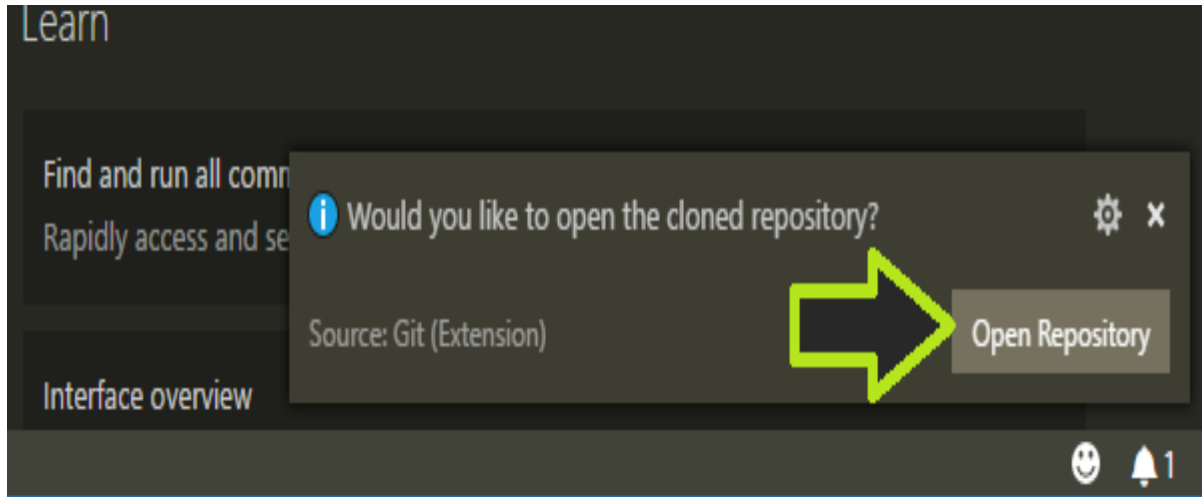
Aparecerá una ventana de Windows para seleccionar la carpeta donde se descarga el código fuente desde GitHub (Por ahora solo tiene el archivo README), seleccionamos la carpeta.



VS Code

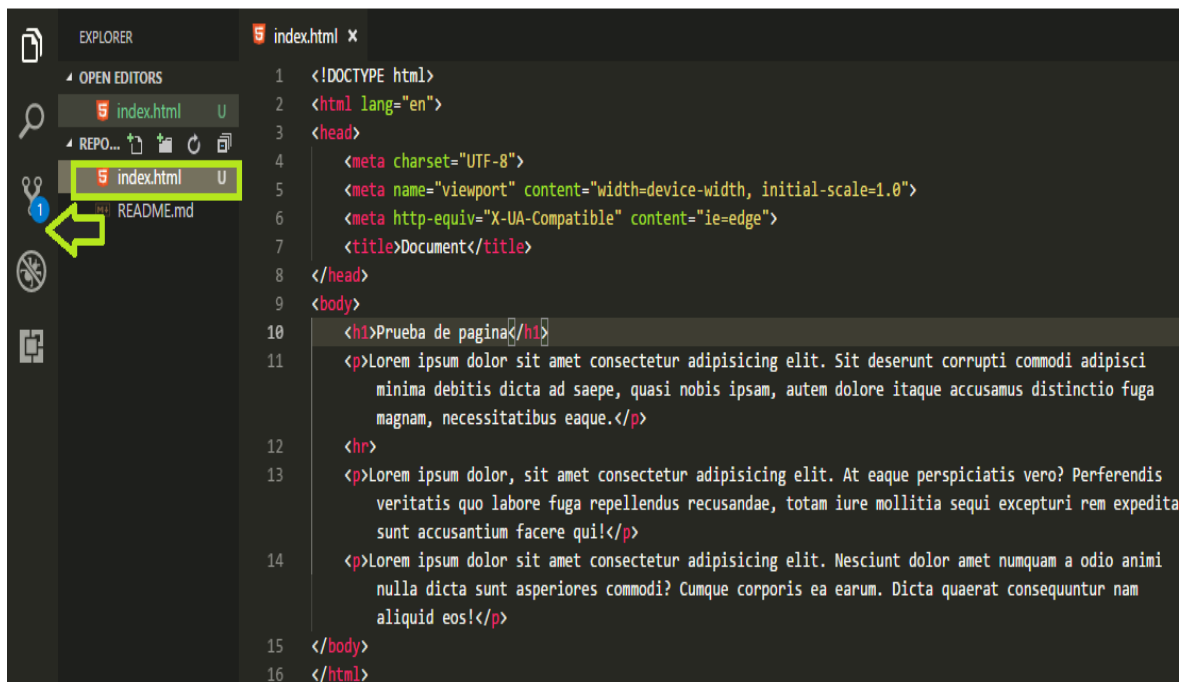
VS Code

Una vez terminado la conexión y descarga de archivos veremos una ventana indicando si queremos abrir el repositorio, daremos clic en “Abrir”.



Nos aparecerán todos los archivos existentes en el repositorio, para este ejemplo voy a crear un archivo HTML de nombre index y le agregare un poco de contenido, verán que inmediatamente aparece un globo azul indicando que existen cambios, damos clic en ese mensaje.

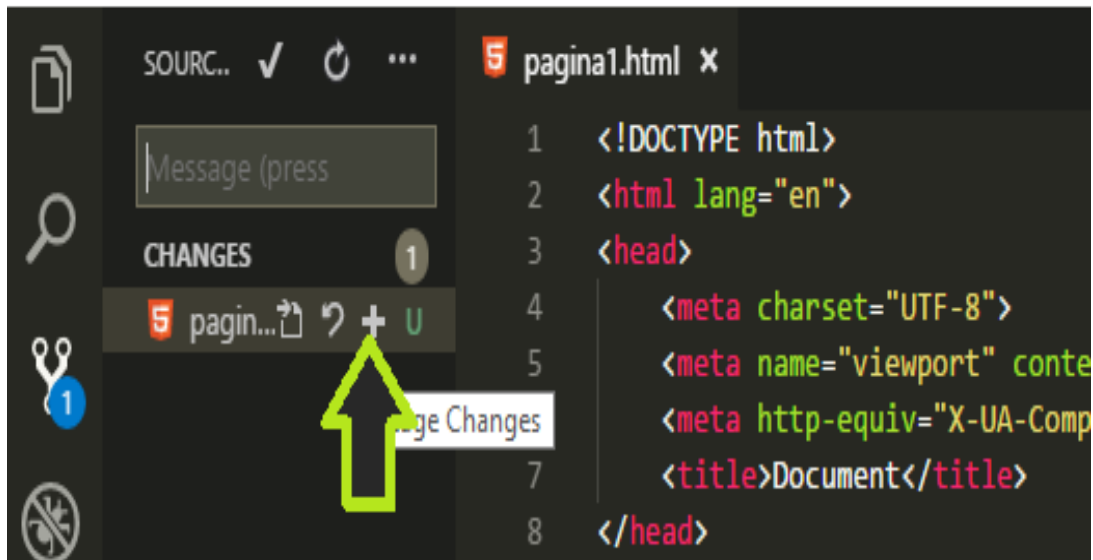
index.html - RepoKC - Visual Studio Code



VS Code

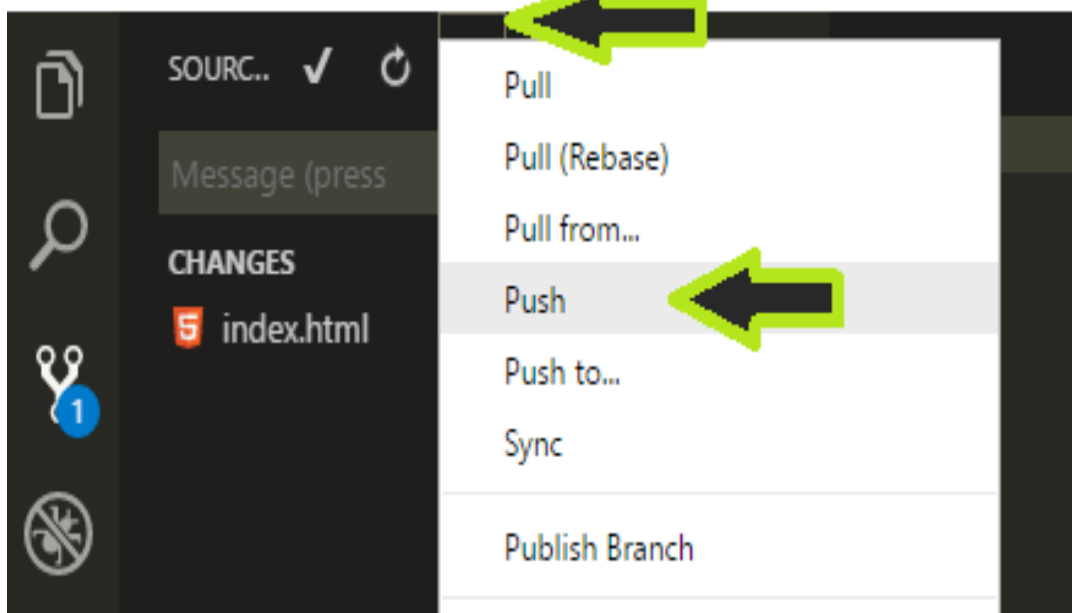
Veremos la lista de archivos que han sufrido cambios, seleccionamos la opción de agregar para subir los cambios.

pagina1.html - RepoKC - Visual Studio Code



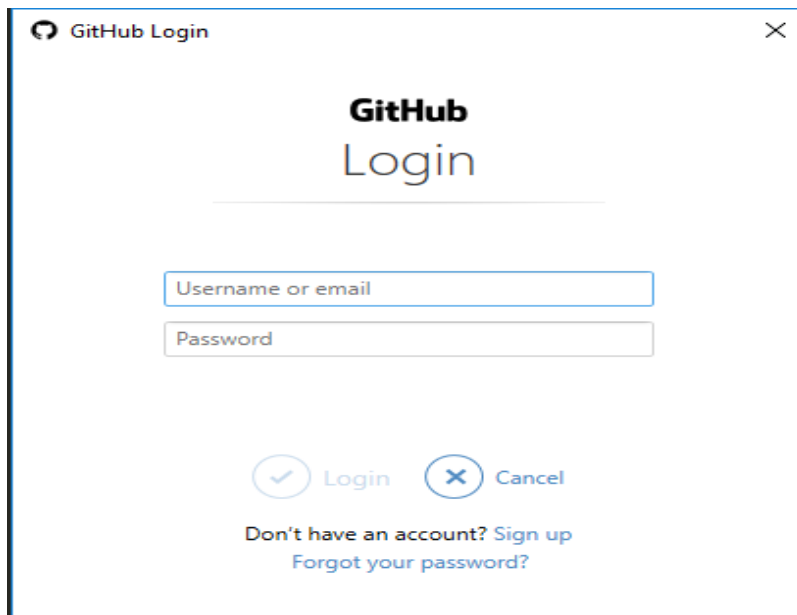
En la parte superior de la lista veremos una opción con tres puntitos, lo seleccionamos, veremos una lista de opciones, seleccionamos el elemento de “Push” para guardar los cambios.

index.html - RepoKC - Visual Studio Code



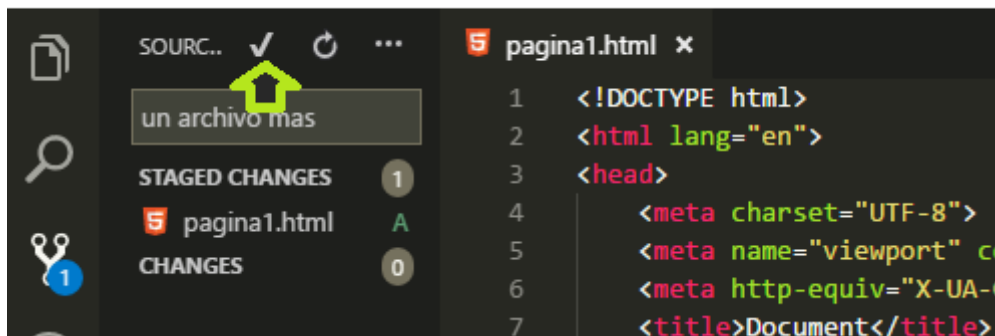
VS Code

Nos pedirá las credenciales de inicio de sesión en GitHub para realizar los cambios en el repositorio.

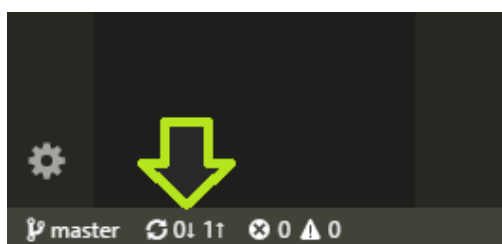


Una vez iniciado sesión se guardarán los cambios y en caso de que no seleccionamos la palomita de guardar cambios.

pagina1.html - RepoKC - Visual Studio Code

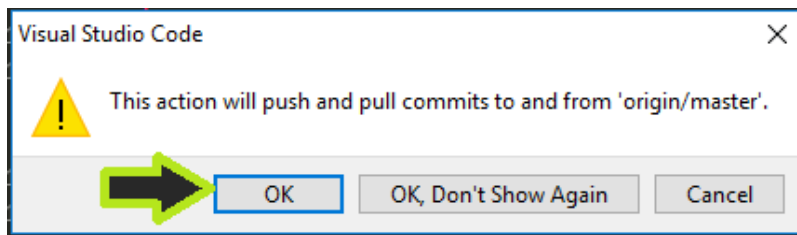


Con esta acción los cambios no se sincronizan con GitHub, si no que se mantiene de manera local, para sincronizar nos vamos a la parte inferior y damos clic en "Sincronizar cambios".



VS Code

Tenemos que confirmar la sincronización con el origen de datos (GitHub), damos clic en "OK".



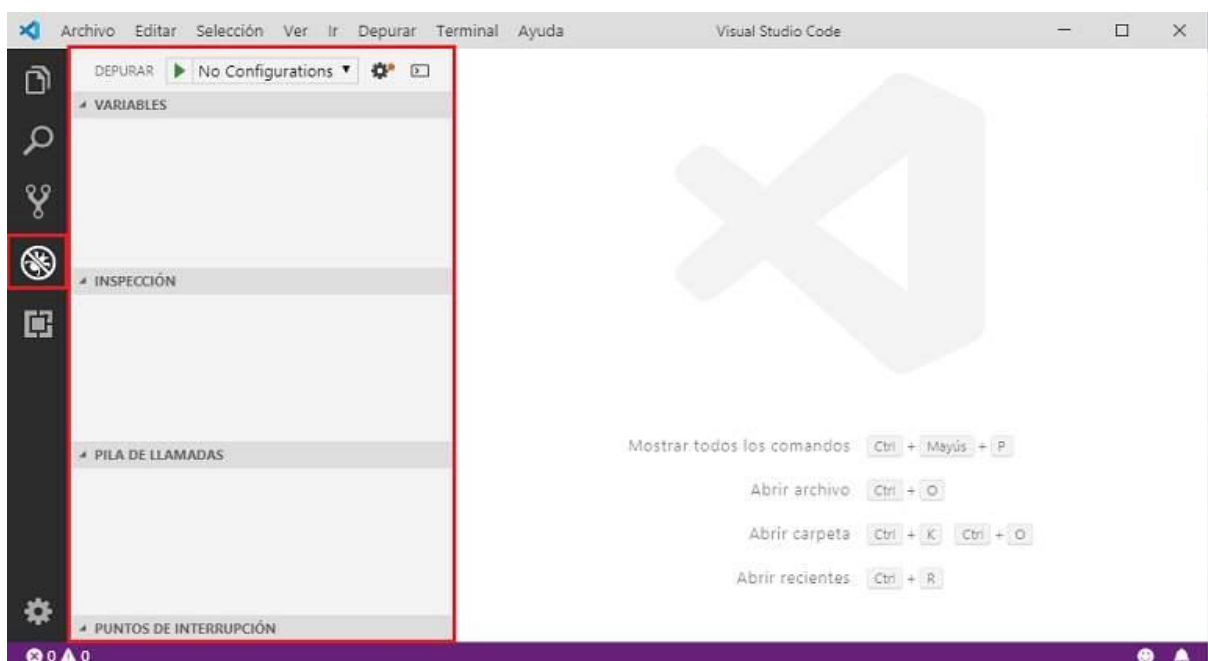
Para confirmar que los archivos se han subido, nos vamos a la página del repositorio en GitHub, podemos ver los archivos cargados y algunos detalles que la plataforma nos proporciona.

Uso del DEBUG

Otra área fundamental que VSCode nos trae dentro del editor es la depuración de código.

Recordemos que depurar es el proceso que hacemos a nuestros programas para identificar errores que hay en los mismos.

Dentro de la "Barra de actividades" de VSCode podemos identificar el ícono que nos abre el diálogo de depuración:



VS Code

Cada ambiente de trabajo tiene sus propias herramientas de depuración: Node.JS, Python, C# etc.

Prueba de depuración con Node.JS

Crearemos un pequeño proyecto con dos archivos y algunos errores en su interior para posteriormente depurarlos.

En una carpeta crearemos dos archivos que ejecutaremos luego desde el ambiente de Node.JS.

matematica.js

```
const PI = 2.14;

function sumar(x1, x2) {
    return x1 - x2;
}

function restar(x1, x2) {
    return x1 - x2;
}

function dividir(x1, x2) {
    if (x2 == 0) {
        mostrarErrorDivision();
    }
    else {
        return x1 / x2;
    }
}

function mostrarErrorDivision() {
```

VS Code

```
    console.log('No se puede dividir por cero');  
}
```

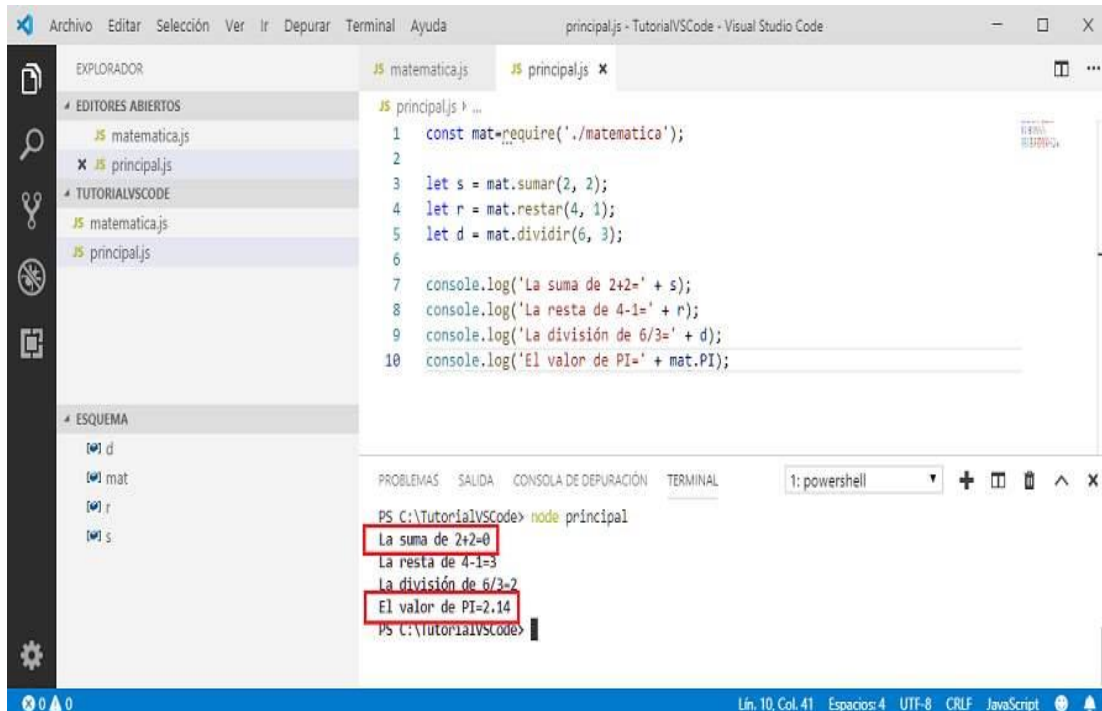
```
exports.sumar = sumar;  
exports.restar = restar;  
exports.dividir = dividir;  
exports.PI = PI;
```

principal.js

```
const mat=require('./matematica');  
  
let s = mat.sumar(2, 2);  
let r = mat.restar(4, 1);  
let d = mat.dividir(6, 3);  
  
console.log('La suma de 2+2=' + s);  
console.log('La resta de 4-1=' + r);  
console.log('La división de 6/3=' + d);  
console.log('El valor de PI=' + mat.PI);
```

VS Code

Podemos ejecutar la aplicación desde la consola y comprobaremos que algunos resultados son incorrectos (la suma y el valor de PI):



```
Archivo Editar Selección Ver Ir Depurar Terminal Ayuda principal.js - TutorialVSCode - Visual Studio Code
```

EXPLORADOR

- EDITORIOS ABIERTOS
 - matematica.js
 - principal.js
- TUTORIALVSCODE
 - matematica.js
 - principal.js
- ESQUEMA
 - d
 - mat
 - r
 - s

```
1 const mat=require('./matematica');
2
3 let s = mat.sumar(2, 2);
4 let r = mat.restar(4, 1);
5 let d = mat.dividir(6, 3);
6
7 console.log('La suma de 2+2=' + s);
8 console.log('La resta de 4-1=' + r);
9 console.log('La división de 6/3=' + d);
10 console.log('El valor de PI=' + mat.PI);
```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL 1: powershell

```
PS C:\TutorialVSCode> node principal
La suma de 2+2=0
La resta de 4-1=3
La división de 6/3=2
El valor de PI=2.14
PS C:\TutorialVSCode>
```

Lín. 10, Col. 41 Espacios: 4 UTF-8 CRLF JavaScript

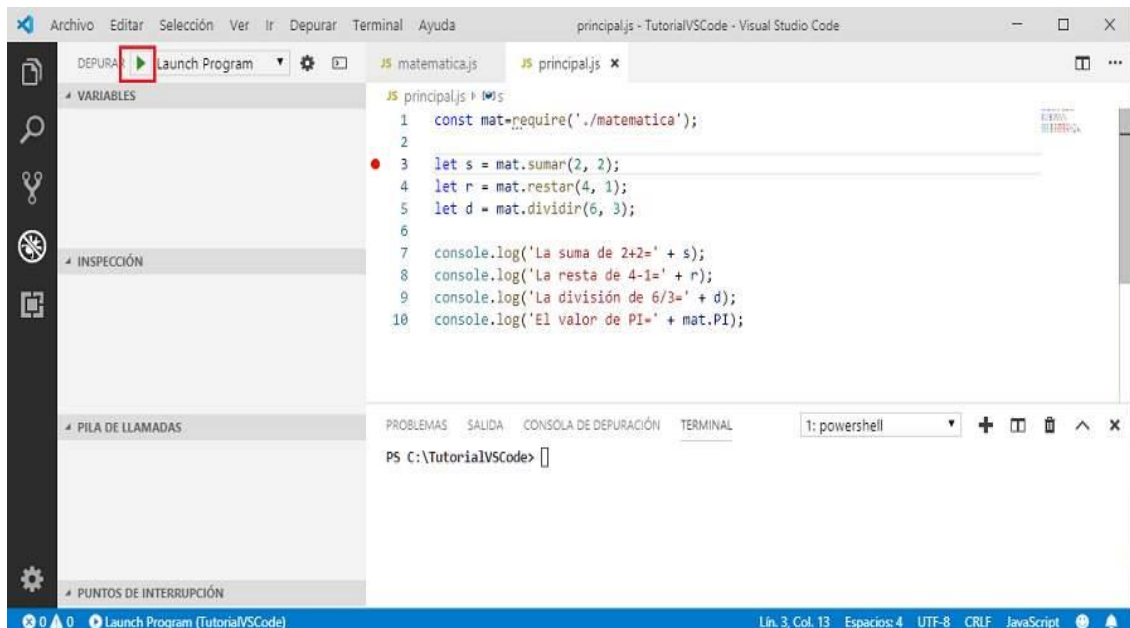
En la pestaña donde se llama a la función 'sumar' dispondremos un punto de ruptura. Esto se logra disponiendo la flecha del mouse a la izquierda del comando:



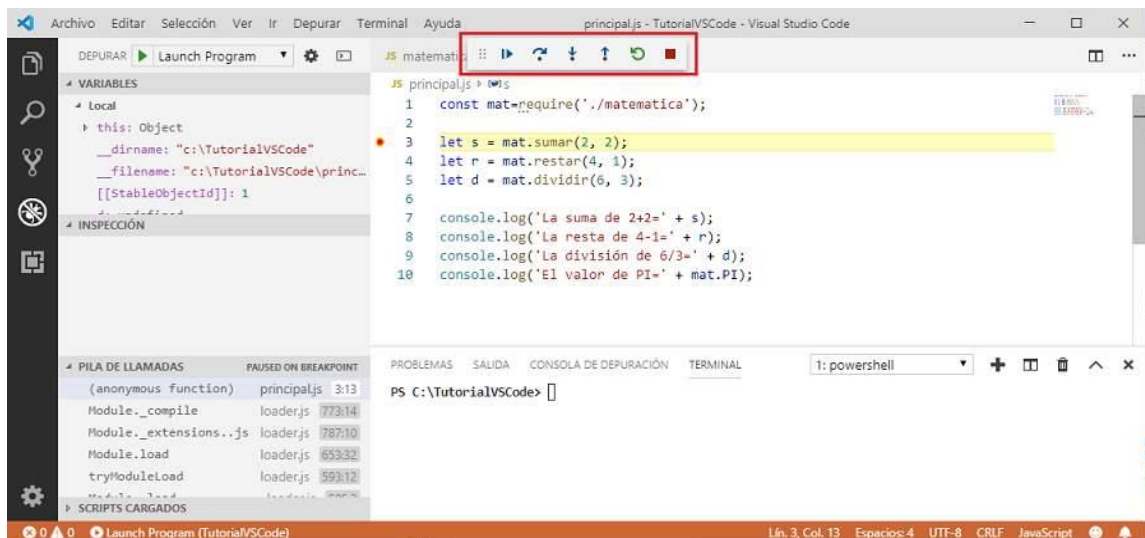
```
1 const mat=require('./matematica');
2
3 let s = mat.sumar(2, 2);
4 let r = mat.restar(4, 1);
5 let d = mat.dividir(6, 3);
6
7 console.log('La suma de 2+2=' + s);
8 console.log('La resta de 4-1=' + r);
9 console.log('La división de 6/3=' + d);
10 console.log('El valor de PI=' + mat.PI);
```


VS Code

Procederemos ahora a abrir la opción de "Depurar" (Ctrl + Shift + D o el ícono de la barra de actividades) e iniciamos la depuración:



El programa empieza a ejecutarse y se detiene en la línea donde dispusimos el punto de ruptura (break point):



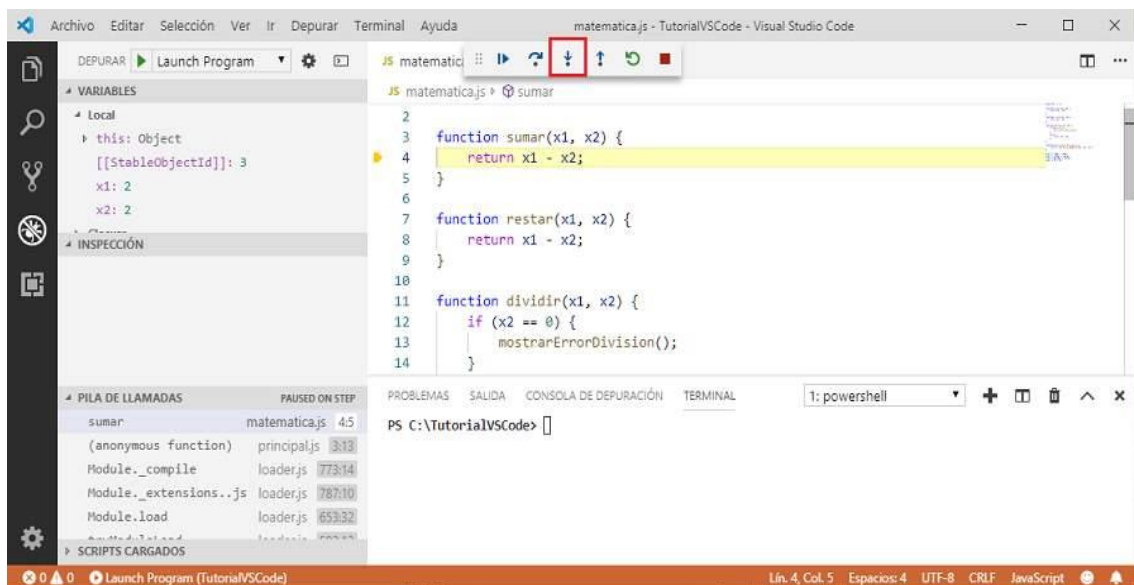
Como vemos aparece una barra de botones que nos permite:

- Continuar (F5) (el triángulo azul)

VS Code

- Depurar paso a paso por procedimiento (F10)
- Depurar paso a paso por instrucciones (F11)
- Salir de depuración (Shift + F11)
- Reiniciar (Ctrl + Shift + F5)
- Detener (Shift+F5)

En nuestro problema elegiremos "Depurar paso a paso por instrucciones" para analizar el algoritmo de la función sumar:



Disponiendo la flecha del mouse sobre el nombre de una variable o parámetro se nos muestra su contenido. Podemos identificar luego que la lógica de la función sumar es incorrecta:

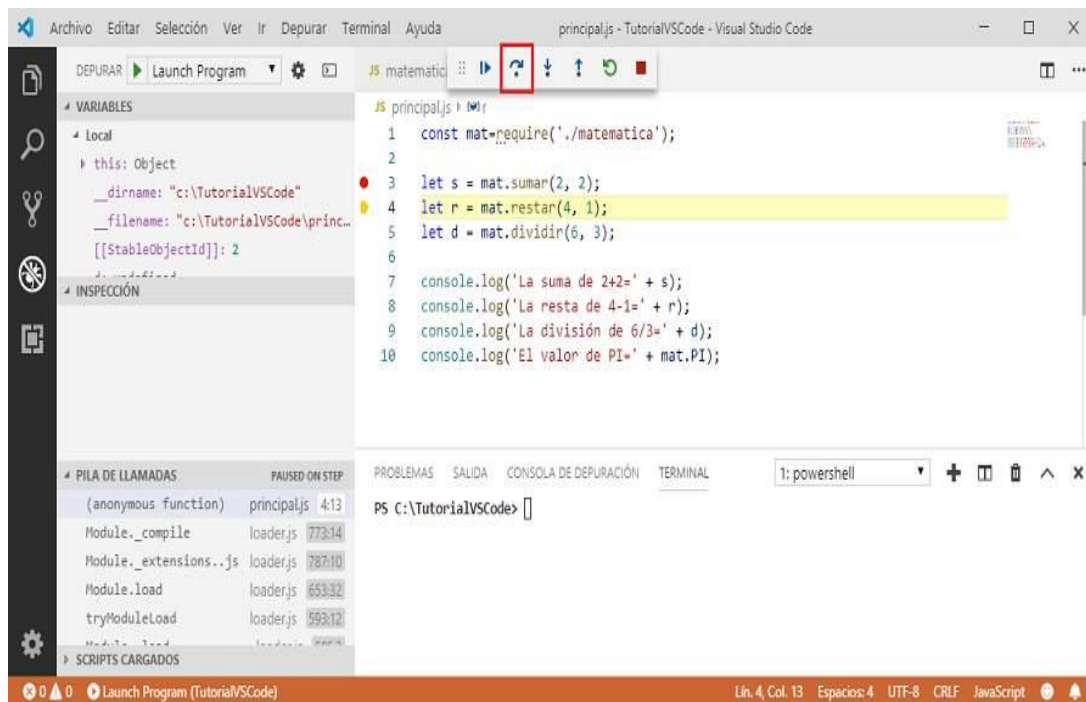
```
return x1 - x2;
```

Cuando debería ser:

```
return x1 + x2;
```

Reiniciar la depuración y probemos ahora de ejecutar "Depurar paso a paso por procedimiento", veremos ahora que la función sumar se ejecuta en forma completa, pero sin entrar y ejecutarla paso a paso sus instrucciones:

VS Code

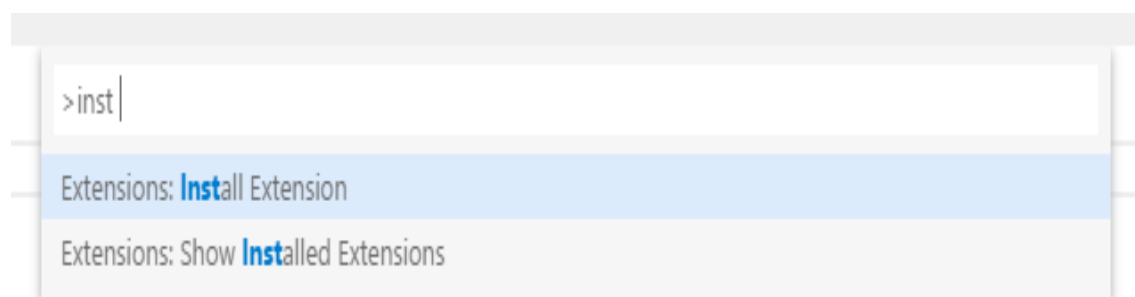


Gestionar plugins

Gestión de extensiones con Visual Studio Code

No existe de momento ninguna interfaz para gestionar las extensiones (algo que sí tiene, por ejemplo, Atom), por lo que tendremos que usar comandos para hacerlo. Por suerte no es nada complicado y el entorno nos ayuda para conseguirlo en un par de pulsaciones de teclas.

Para empezar con la instalación sacamos la paleta de comandos pulsando F1 (o CTRL+MAY+P) y en el cuadro de texto que aparece escribimos simplemente "inst":

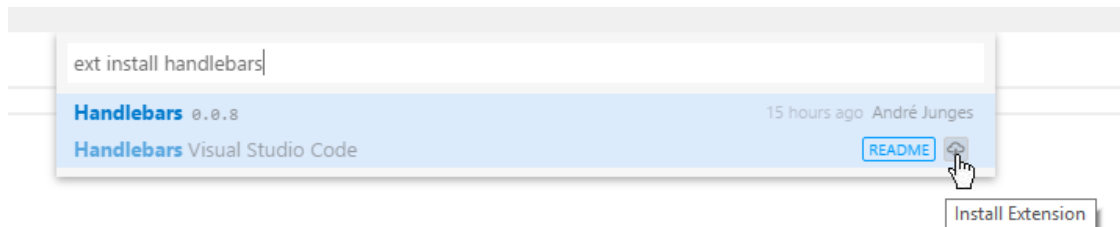


VS Code

Esto nos mostrará las opciones disponibles para instalación de extensiones: instalarlas o listar las existentes.

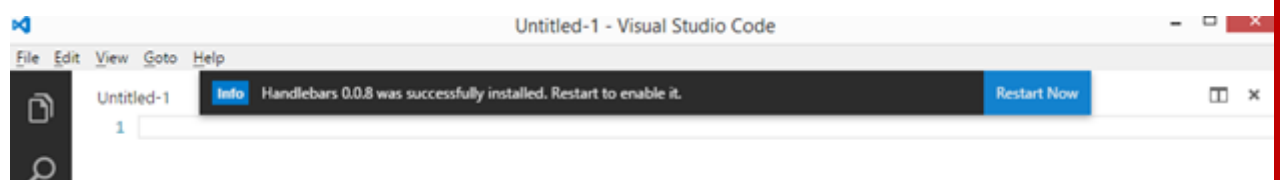
Lo primero que tenemos que hacer para instalar una extensión es **conocer su nombre aproximado**. O si no lo conocemos, saber más o menos cómo buscarlo.

Por ejemplo, supongamos que queremos instalar una extensión para que nos ayude con plantillas Handlebars. Pues escribimos las primeras letras de esa palabra justo después del comando, y ya nos sugiere las extensiones que encajen con ese texto o similar:



Al elegir una (y aparecer el comando completo) se nos ofrece información resumida de la extensión con un botón "Readme" para ir a la página de la extensión y saber más, y un icono de una "nubecita" que es el que debemos pulsar para instalar la extensión, como se ve en la figura. La verdad es que esto lo tienen que mejorar un poco.

Al pulsar la instalación y tras un rato de espera (la verdad es que van bastante lentas), se nos muestra una barra informando de que la instalación ha finalizado con éxito (pulsar para aumentar la imagen):



Tendremos que reiniciar Code para que funcione la nueva extensión. Si vamos a instalar muchas es mejor hacerlo con todas y luego reiniciar solo una vez.

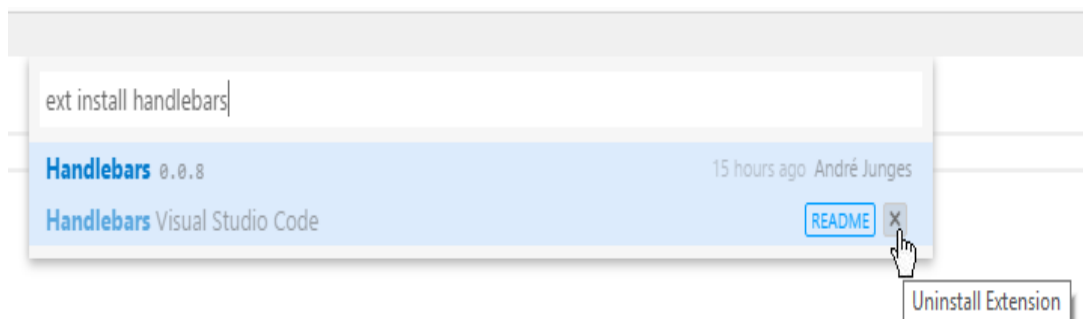
VS Code

La extensión está lista para ser utilizada.

Podemos ver cuáles tenemos instaladas con el otro comando que se nos ofrecía al escribir "inst" y que vimos en la primera figura.

Para desinstalar una extensión que no nos convenza o que ya no nos sea útil **el proceso es casi idéntico**.

Procedemos exactamente de la misma manera y hacemos lo mismo que si fuésemos a instalarla. Lo que ocurre es que en esta ocasión el botón de la nube cambia y se convierte en un aspa:



Pulsando esa aspa desinstalaremos la extensión y tras un nuevo reinicio del entorno nos habremos librado de ella.

Plugin que sería muy importante para ayudarnos

Color highlight

Esta extensión es de mis favoritas, me permite identificar con mayor facilidad los colores que estoy utilizando dentro de mis proyectos en HTML/CSS (También funciona con archivos de Stylus) y apreciar con mayor facilidad el color que estoy utilizando.

Import Cost

Esta extensión me permite saber el tamaño del paquete que estoy importando, así como detectar si es demasiado pesado para nuestros proyectos. Saber el tamaño de los recursos que estamos importando nos permitirá entender y trabajar con mayor calidad nuestros proyectos.

Trailing Spaces

Una extensión muy simple, pero me permite dejar mi código limpio y libre de espacios innecesarios, esta extensión nos marcara en rojo todos aquellos espacios que dejamos dentro de nuestro código.

Auto Close Tag

No puede faltar la extensión que nos ayude a cerrar los tags, es muy útil cuando estamos maquetando ya que nos brinda un rendimiento a la hora de crear código.