

# Introducció a Visual Studio Code

Unitat 2. Entorns de desenvolupament

Entorns de desenvolupament



## Continguts

<b>1. Introducció a Visual Studio Code</b>	<b>3</b>
1.1. Descàrrega i instal·lació de VS Code . . . . .	3
Instal·lació de code a Ubuntu . . . . .	3
1.2. Presa de contacte amb VSCode . . . . .	4
<b>2. Instal·lació de plugins en VSCode</b>	<b>6</b>
3. Visual Studio Code i Java . . . . .	7
3.1. Comprovació de l'entorn . . . . .	7
3.2. Instal·lació de les extensions bàsiques de Java per a VSCode . . . . .	8
2.3. Configuració de l'entorn . . . . .	9
2.4. Hola VSCode. Un passeig per l'editor. Depuració . . . . .	12
Execució i Depuració . . . . .	12
Establir Breakpoints . . . . .	13

## 1. Introducció a Visual Studio Code

Visual Studio Code és un editor de codi lleuger multiplataforma, de gran popularitat en el món del desenvolupament multiplataforma en general i de serveis i web en particular. Per defecte suporta llenguatges com JavaScript, TypeScript i Node.js, i és extensible a altres llenguatges, com C++, C#, Java, Python, etc.

D'entre les principals característiques de l'editor, podem destacar:

- Ofereix una interfície simple i neta.
- Disposa d'una paleta d'ordres des d'on podem accedir a totes les opcions de l'editor.
- Ofereix una terminal integrada.
- Ofereix suport a Git per al control de versions.
- Integra Emmet (Zen Coding), per generar codi web de forma àgil.
- Cursors múltiples, per a l'edició massiva de codi.
- Fàcilment personalitzable a través d'un fitxer JSON.
- Autocompletat i ressaltat de sintaxi amb IntelliSense.
- Suporta un gran nombre de temes i extensions que amplien la seua funcionalitat.

També cal remarcar que no es tracta d'un entorn de desenvolupament (IDE) com a tal, com puga ser Eclipse, Netbeans o IntelliJ, pel que no ofereix tota la funcionalitat que ofereixen aquests (construcció, compilació, depuració, frameworks). Tot i això, es tracta d'un entorn molt més lleuger que aquests, i que amb les extensions adequades, podem fer que la seua funcionalitat siga pràcticament la mateixa que ofereix un IDE.

### 1.1. Descàrrega i instal·lació de VS Code

#### Instal·lació de code a Ubuntu

Tal i com es descriu a la documentació, podem obtenir VS Code des del mateix repositori de Microsoft. Per a això seguirem els passos següents. Potser siga necessari instal·lar prèviament el paquet `curl`:

1. Obtenim la clau GPG amb la que estan signats els paquets de MS i la incorporem al sistema (si no tenim instal·lat prèviament el paquet `curl` caldrà instal·lar-lo amb `sudo apt-get install curl`):

```
1 $ curl https://packages.microsoft.com/keys/microsoft.asc | gpg --  
   dearmor > packages.microsoft.gpg  
2 $ sudo install -o root -g root -m 644 packages.microsoft.gpg /usr/share  
   /keyrings/
```

2. Creem el fitxer `vscode.list` dins la carpeta de fonts de programari del sistema amb la línia referent al repositori de VSCode:

```
1 $ sudo sh -c 'echo "deb [arch=amd64] https://packages.microsoft.com/
  repos/vscode stable main" > /etc/apt/sources.list.d/vscode.list'
2 $ sudo sh -c 'echo "deb [arch=amd64 signed-by=/usr/share/keyrings/
  packages.microsoft.gpg] https://packages.microsoft.com/repos/vscode
  stable main" > /etc/apt/sources.list.d/vscode.list'
```

3. Ara només ens quedarà actualitzar la memòria caché de paquets i instal·lar VSCode amb:

```
1 $ sudo apt-get install apt-transport-https
2 $ sudo apt-get update
3 $ sudo apt-get install code
```

Si feu ús de la capa de programari de Justix (<https://ieseljjust.github.io/Bionic/>), ja disposeu de VSCode instal·lat per defecte.

Per a més informació, disposeu de la documentació oficial sobre la instal·lació en Linux i Windows en:

- <https://code.visualstudio.com/docs/setup/linux>
- <https://code.visualstudio.com/docs/setup/windows>

## 1.2. Presa de contacte amb VSCode

Una vegada instal·lat, tenim el Visual Studio Code accessible des del menú principal del sistema, dins de *Programació*.

Quan l'obrim, veurem una interfície semblant a la **figura 1**, a la que podem distingir:



**Figura 1:** Interfície de VSCode

- La **barra d'activitats**, situada a l'esquerra, amb cinc activitats principals: L'**explorador de fitxers**, la de **Búsqueda de text**, la de **control de versions amb Git**, la de **depuració**, i la d'**extensions**. Cadascuna d'aquestes activitats, ens obrirà una altra barra lateral al costat amb les diferents possibilitats que ofereix l'activitat.
- La **finestra de benvinguda**, situada a la part principal de l'editor, que ocupa la part superior, i que ens ofereix algunes **opcions d'inici** com crear un fitxer nou, obrir una carpeta o afegir un espai de treball. A més, disposem també d'una secció de **fitxers recents**, inicialment buida, i de diversos **opcions de personalització**, com extensions per suportar altres llenguatges, dreceres de teclat, i temàtica de l'editor. En aquesta finestra, també disposem d'algunes opcions d'**Ajuda**, i alguns tips per aprendre, com la llista d'ordres o una ullada a la interfície.
- Diversos **panells**, situats al davall de la finestra principal de l'editor, i que mostraran informació sobre la depuració, errors, avisos, o bé la terminal integrada a VSCode.
- A la part inferior, distingim finalment la barra d'estat, amb informació sobre el projecte i els fitxers oberts.

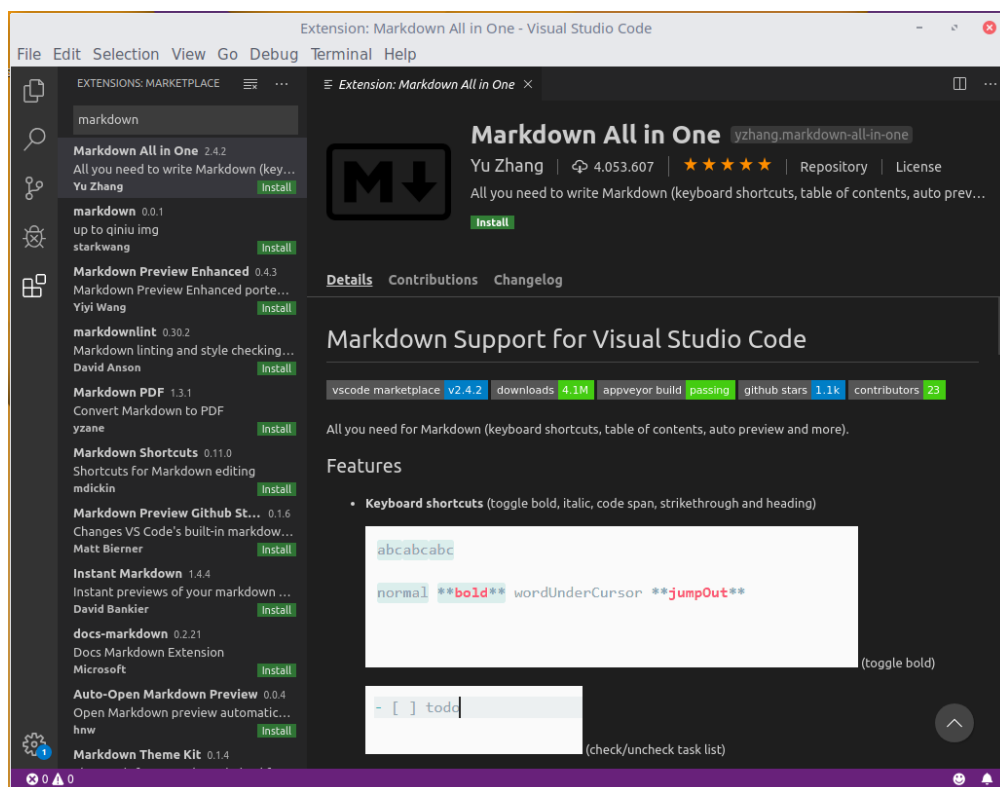
Disposeu de més informació sobre la interfície de VSCode en:

- <https://code.visualstudio.com/docs/getstarted/userinterface>

## 2. Instal·lació de plugins en VSCode

Com hem comentat, VSCode és un editor lleuger, i ahora extremadament flexible mitjançant plugins. A banda de suportar altres llenguatges de programació, també ens permet extensions, per treballar, per exemple amb el llenguatge Markdown.

Per a això, fem clic a l'**activitat d'extensions**, ubicada cap a la part inferior de la barra d'activitats, i al quadre de cerca d'extensions escrivim *Markdown*.



**Figura 2:** Extensions markdown

Veurem que ens apareix gran quantitat d'extensions per a Markdown. Podeu donar un cop d'ull a totes elles, però de moment, instal·larem l'extensió `markdownlint`, que ens ajuda a comprovar la correcta sintaxi del codi Markdown i ens ofereix la possibilitat de previsualitzar el resultat al temps que escrivim. Per a això, només haurem de **fer clic** en el botó **Install** que apareix bé al costat de l'extensió o en la descripció d'aquesta. Si ho desitgeu, podeu instal·lar també l'extensió *Markdown PDF* que us permet guardar els fitxers creats en format *md* a *pdf*.

Per tal de començar a treballar amb algun exemple, podem anar a l'activitat de l'*Explorador de fitxers* i prémer el botó *Open Folder* per tal d'obrir una carpeta de treball, o fer-ho directament a través del

menú *File > Open Folder*.

Així doncs, obriu la carpeta on teníeu la primera pràctica i carregueu el fitxer en format Markdown que vàreu realitzar. Comproveu si el VSCode us detecta alguna incorrecció i quines suggerències us fa per corregir-les. Fixeu-vos també en què ha aparegut una icona a la part superior dreta de la finestra d'edició, amb una lupa davant una finestra, que ens servirà per dividir la vista de l'editor i veure a la part dreta el resultat del què estem escrivint en Markdown.

Per altra banda, doneu una ullada a les possibilitats que ens ofereix el gestor de fitxers: navegar per la carpeta, buscar fitxers o carpetes, i crear tant fitxers com carpetes noves dins l'estructura en la què estem.

### 3. Visual Studio Code i Java

Com hem comentat, VSCode és un editor que suporta múltiples llenguatges de programació a través de plugins. En aquest apartat, anem a veure com afegir a VSCode el suport per a Java, i com crear algun exemple senzill.

#### 3.1. Comprovació de l'entorn

Abans de començar amb la instal·lació de les extensions de Java, haurem de comprovar que disposem al nostre equip del JDK (*Java Development Kit*) per tal de poder compilar i executar les aplicacions en Java que desenvolupem.

Java i el JDK pot que no vinguin preinstal·lats al nostre sistema Ubuntu. Per tal de comprovar si el tenim instal·lat i si és el cas, saber quina versió tenim, des d'una terminal podem fer:

```
1 $ javac -version
```

Amb açò podem comprovar quina versió del JDK tenim instal·lada:

```
1 openjdk version "11.0.4" 2019-07-16
2 OpenJDK Runtime Environment (build 11.0.4+11-post-Ubuntu-1ubuntu218
  .04.3)
3 OpenJDK 64-Bit Server VM (build 11.0.4+11-post-Ubuntu-1ubuntu218.04.3,
  mixed mode, sharing)
```

O si no en tenim cap, se'ns avisarà dels diferents paquets que ens ofereixen l'eina, és a dir, dels diferents JDKs que tenim disponibles, i com instal·lar-los:

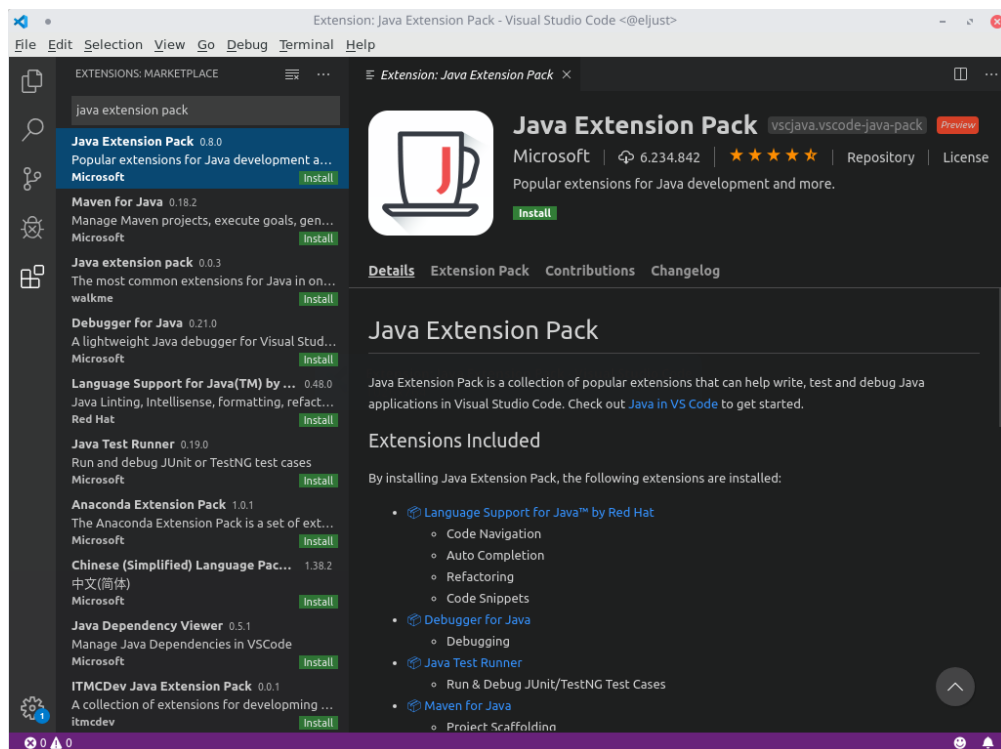
```
1 $ javac -version
2
3 Command 'javac' not found, but can be installed with:
4
```

```
5 sudo apt install default-jdk
6 sudo apt install openjdk-11-jdk-headless
7 sudo apt install ecj
8 sudo apt install openjdk-8-jdk-headless
9 sudo apt install openjdk-9-jdk-headless
```

### 3.2. Instal·lació de les extensions bàsiques de Java per a VSCode

Per tal d'instal·lar el suport per a Java en VSCode, farem clic al botó de l'activitat d'extensions per obrir de nou l'*Extension Marketplace*, que ens mostrarà de nou llistats d'extensions recomanades i d'extensions populars.

Busquem ara l'extensió `java extension pack`:



**Figura 3:** Java Extension Pack

L'extensió **Java Extension Pack** porta el següent conjunt d'extensions:

- *Language Support for Java(TM) by Red Hat*, amb el suport per a Java de l'IDE Eclipse. Ens ofereix: suport de projectes Maven i Gradle, comprovació de sintaxi i errors de compilació, autocompletat de codi, refactorització, suport a JavaDoc, ressaltat de la sintaxi, snippets de codi, etc.



- *Debugger for Java*, amb un depurador lleuger, que ens permet llençar les aplicacions, afegir breakpoints, pausar i continuar l'execució, seguiment de variables, etc.
- *Java Test Runner*, per tal d'executar diferents proves de funcionament. Té suport per a tests unitaris de JUnit, depuració, logs, etc.
- *\*Maven for Java*, per tal de gestionar projectes Maven. Ofereix explorador de projectes, execució d'ordres, generació de projectes, validació, compilació, realització de tests, desplegament, etc.

Per a més informació sobre cadascuna d'aquestes extensions, podeu llegir la que elles mateixa us proporcionen al Marketplace.

Bé, una vegada localitzat el pack d'extensions, només hem de fer clic en **Install**, i esperar que totes les extensions s'instal·len al nostre equip.

També cal dir, que hi ha moltes altres extensions de Java per a VSCode, i que si són necessàries, veurem en el seu moment oportú.

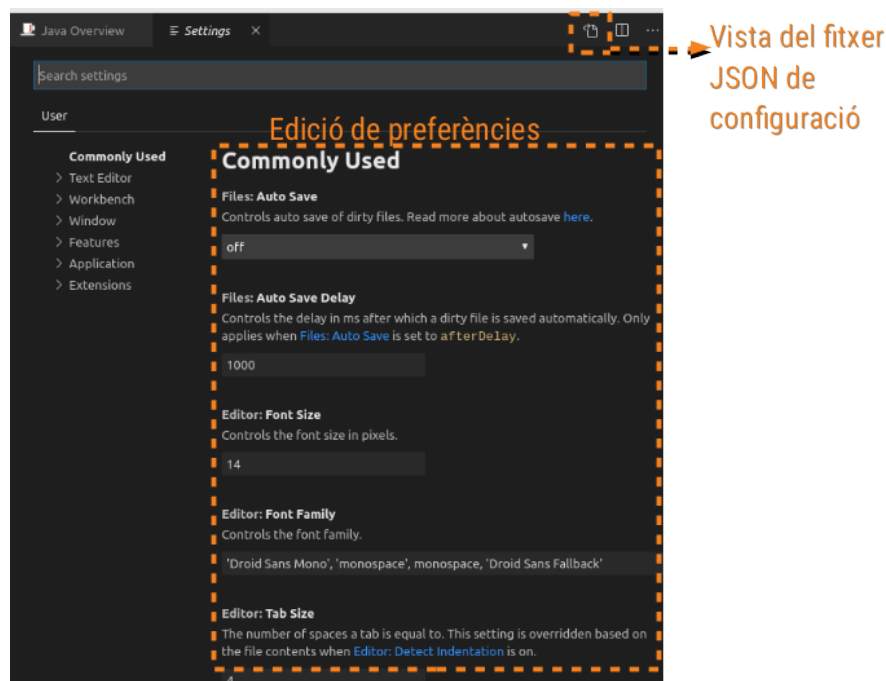
### 2.3. Configuració de l'entorn

Com hem comentat, VSCode és àmpliament personalitzable, a través d'un fitxer de configuració en format JSON. A més, ens permet establir dos àmbits diferents per a la configuració:

- **Ajustos d'usuari**, que s'apliquen de forma global a totes les instàncies de VSCode que llancem amb el nostre usuari, i
- **Ajustos de l'espai de treball**, associats dins de cada espai de treball, i només aplicable a aquest. En cas que tinguem definits ajustos d'usuari i de l'espai de treball, aquests últims sobreescriran els de l'usuari.

Per tal d'accedir als ajustos, ho fem a través de **File > Preferences > Settings**, o bé mitjançant **Ctrl+,,**, o amb la paleta d'ordres, accessible amb **Ctrl+Shift+P** i buscant *Preferences: Open User Settings* o *Preferences: Open Workspace Settings*.

Podem veure aquesta finestra de configuració a la **figura** .



**Figura 4:** Ajustos

En versions anteriors se'ns presentava directament el fitxer JSON de configuració. En les últimes actualitzacions de VSCode, ja ens ofereix una interfície més amigable per tal de modificar les preferències, ordenades en diverses seccions.

Com veiem, a la part superior disposem d'una barra per filtrar ajustos, i a sota les preferències que podem modificar, ordenades per seccions. A la imatge, de moment, només veiem les preferències de l'usuari (*User*), però si tinguérem un espai de treball obert, també se'ns mostrarien les preferències d'aquest *Workspace*.

Si desitgem veure directament el fitxer JSON de configuració, haurem de fer clic a la pestanya corresponent (veieu *figura*). Aquest fitxer està compost de parells `"clau": "valor"`, separats per comes, i tancats entre claus `{ }`. Els valors, a més de valors simples (cadena o números), també poden ser llistes de valors, tancats entre claudàtors `[]`, o bé altre conjunt de parells `"clau": "valor"`, tancats també entre claus `{ }`. Més endavant veurem en detall aquest format, però veiem un exemple autoexplicatiu de fitxer de configuració en aquest format:

```
1 {
2   "languageTool.language": "ca",
3   "[markdown]": {
4     "editor.tabSize": 2
5   },
6   "markdown-toc.depthFrom": 2,
7   "markdown-toc.depthTo": 3,
```

```

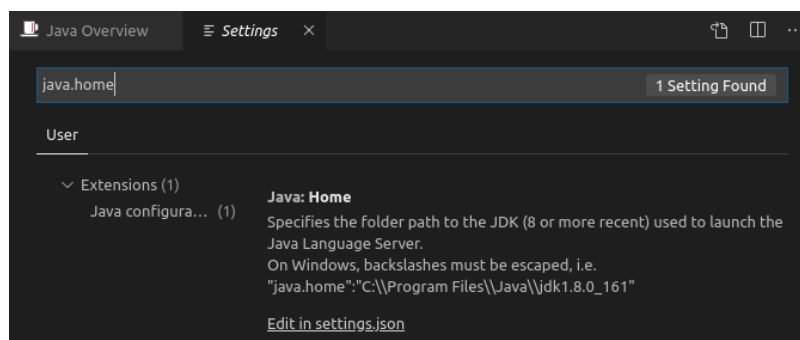
8     "window.zoomLevel": 0,
9     "editor.wordWrap": "on",
10    "markdown.styles": [
11        "CustomMDStyles.css"
12    ],
13    "java.home": "/usr/lib/jvm/openjdk-11"
14 }

```

Com veiem, tenim:

- diferents atributs en forma *clau : valor*, com `"editor.wordWrap": "on"`, per activar la separació de línies en arribar al final de l'editor;
- Atributs en *clau : llista de valors*, com `"markdown.styles"`, i
- Atributs en forma *clau : llista de parells clau:valor*, com el cas de `"[markdown]"`.

Com hem comentat, poc a poc anirem familiaritzant-nos amb aquest format. De moment, anem a buscar a la barra superior l'ajust **java.home** (figura 5)



**Figura 5:** Configuració del JDK

Com veiem, aquesta opció no es pot editar directament, i ens indica que ho fem de fer a través del fitxer JSON de configuració. Si fem clic en *Edit in Settings*, obrirem aquest fitxer, i indicarem a la propietat `java.home` el directori on tenim el nostre JDK. Per al cas de l'OpenJDK 11 serà:

```

1 {
2     "java.home": "/usr/lib/jvm/java-11-openjdk-amd64"
3 }

```

Amb açò hem modificat el fitxer de **preferències d'usuari**. Per tal que els canvis tinguin efecte, haurem de guardar el fitxer (*File > Save* o directament *Ctrl+S*), i reiniciar el component *Java Language Server*, tal i com se'ns indicarà quan guardem el fitxer.

Podeu trobar més informació sobre els ajustos a la web de VSCode:

- <https://code.visualstudio.com/docs/getstarted/settings>

## 2.4. Hola VSCode. Un passeig per l'editor. Depuració

Anem ja a posar-nos mans en l'obra, i veure una adaptació del clàssic *Hola Món* en VSCode.

Per a això, creem una carpeta per als exemples de java i un fitxer a dins anomenat HelloVSC.java:

```
1 ExemplesJava
2   |-- HelloVSC.java
```

Per tal d'obrir la carpeta, si hem tancat totes les pestanyes obertes, i no tenim res obert, tindrem disponible un botó amb el text *Open Folder* per obrir-la. També podem obrir la carpeta directament a través del menú *File > Open Folder*.

Una vegada tinguem la carpeta oberta, la veurem a l'explorador de fitxers. Si passem el ratolí per damunt, veurem que apareixen unes icones, per tal de crear nous fitxers dins la carpeta, noves carpetes, refrescar el contingut o col·lapsar la jerarquia de carpetes que tinguem expandides a la finestra. Per crear el nou fitxer, farem clic a la icona de *New File*, i indicarem el nom del nou fitxer (HelloVSC.java):

També podrem crear un fitxer nou a través de *File > New File*, o bé amb **Ctrl+N**, tenint en compte que haurem d'indicar el nom del fitxer i la seua ubicació posteriorment.

El contingut de HelloVSC.java és el clàssic de l'Hola Món, amb dos escriptures per pantalla.

```
1 public class HelloVSC {
2
3     public static void main(String[] args) {
4         System.out.println("Hola Visual Studio Code!");
5         System.out.println("Estem provant un nou editor!");
6     }
7
8 }
```

### Execució i Depuració

Per tal d'executar el nostre programa, seleccionarem *Debug > Start Without Debugging*, o bé premerem **Ctrl+F5**. Si volem depurar-lo, ho fem amb *Debug > Start Debugging*, o directament amb **F5**.

Una vegada iniciada la depuració, disposarem del resultat al panell de la consola de depuració (*Debug Console*), amb el corresponent resultat:

```
1 ...
2 Hola Visual Studio Code!
3 Estem provant un nou editor!
```

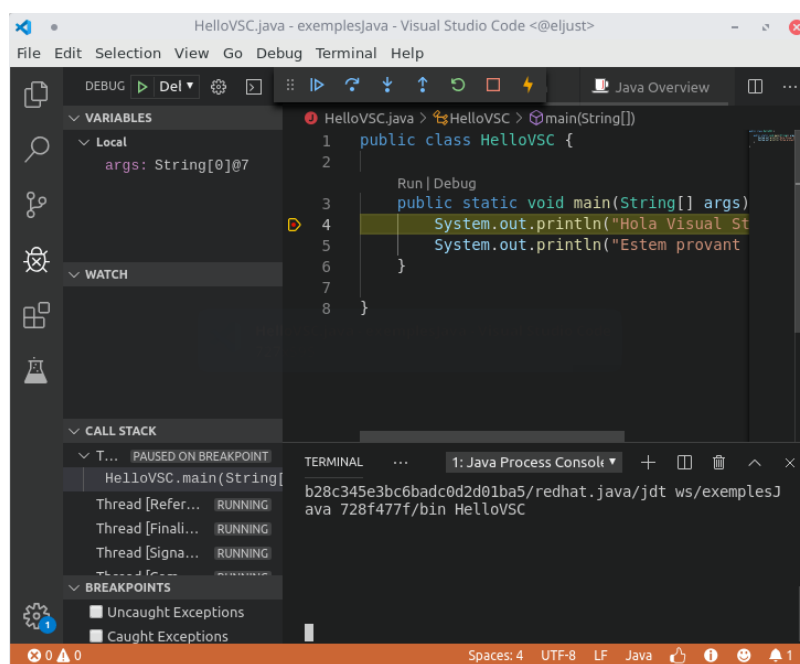
Si no ens apareix aquest panell directament, podem activar-lo, des de la barra lateral en l'activitat de depuració, fent clic en la icona de la consola, situada a la dreta.

Per altra banda, veurem que a la jerarquia de fitxers se'ns ha creat una nova carpeta anomenada .vscode, amb un fitxer `launch.json` a dins. Aquest fitxer el crea l'extensió de depuració de Java, i conté informació per a la depuració.

## Establir Breakpoints

VSCode ens permet establir punts de ruptura en el codi. Per a això, disposem d'una columna, que ens permet establir els punts fent clic al costat esquerre del número de línia corresponent.

Quan fem clic, veurem com apareix un punt roig marcant el breakpoint. Ara podem executar el codi en mode depuració (F5) i veure quin és l'efecte que té. Ho podem veure a la **figura 6**



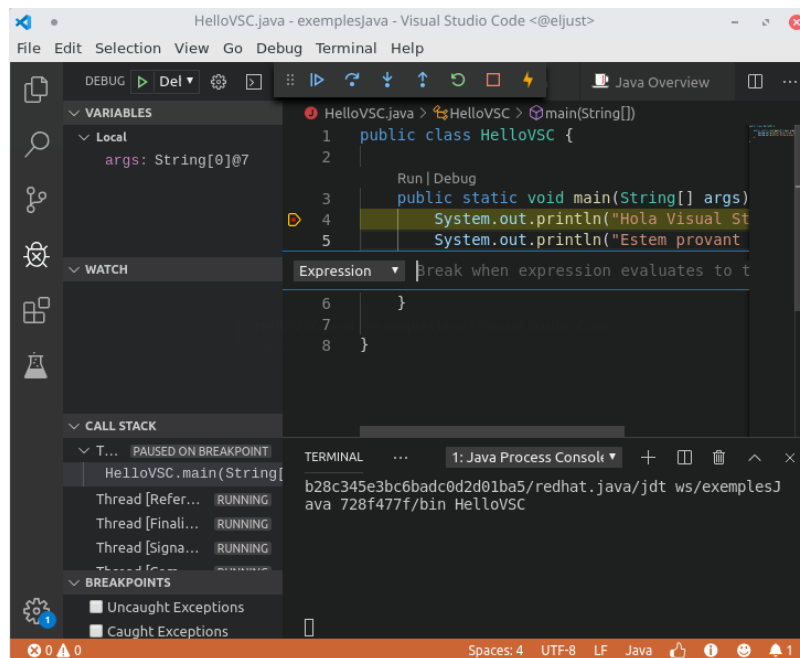
**Figura 6:** Depuració

A més, si ens ubiquem damunt d'aquesta columna de l'esquerra dels números de línia i fem clic amb el botó dret del ratolí, ens apareixerà un menú contextual amb tres opcions, en funció de si ja tenim un breakpoint a la línia o no:

- Si la línia ja disposa d'un Breakpoint, podem:
  - Eliminar el Breakpoint

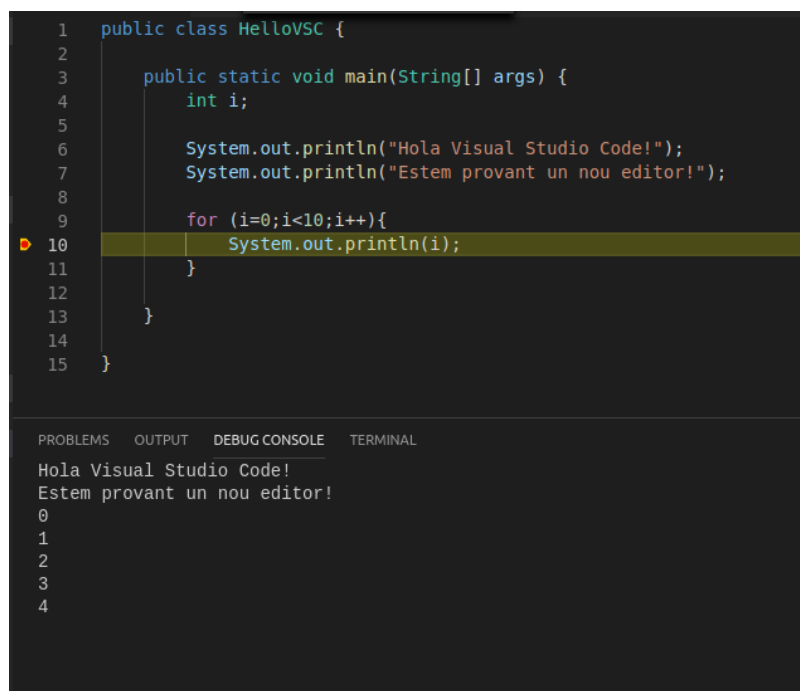
- Editar el Breakpoint
- Desactivar el breakpoint
- En canvi, si la línia no té cap breakpoint, podem:
  - Afegir un breakpoint
  - Afegir un breakpoint condicional
  - Afegir un breakpoint de log

Si volem afegir un punt de ruptura condicional, només hem de seleccionar l'opció d'afegir el breakpoint o editar-lo. En estos moments, ens demanarà una expressió condicional. L'escrivim i polsem **Enter** per a que els canvis tinguin efecte (**figura 7**).

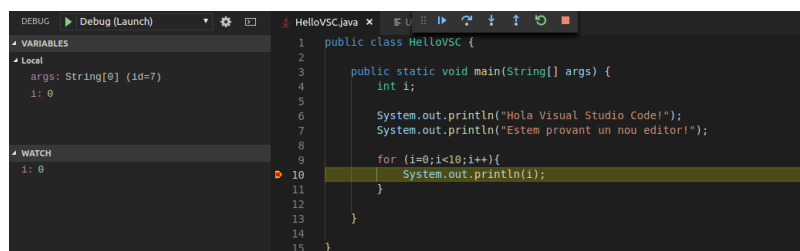


**Figura 7:** Breakpoints condicionals

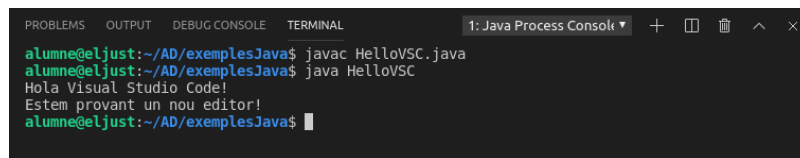
Fet açò, si tornem a depurar, veurem com atura l'execució en el moment que es compleix l'expressió que li hem indicat. Veiem un exemple més complet a la (**figura 8**).

**Figura 8:** Breakpoints condicionals 2

I finalment, recordar, que a la part de l'esquerra, tenim disponibles les finestres que ens mostren l'estat de la variable en el punt de ruptura que hem establert (Secció *Variables*), o bé que indiquem explícitament (*Watch*) (**figura 9**).

**Figura 9:** Estat de les variables

**Execució des de consola** Per altra banda, també cal destacar que podem compilar i llançar el nostre programa java des de la pròpia terminal integrada en VSCode o la terminal del sistema, com podem veure a la **figura 10**.

A screenshot of a Visual Studio Code terminal window. The terminal has tabs for 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', and 'TERMINAL'. The 'TERMINAL' tab is active, showing a command prompt with the username 'alumne@eljust' and the directory '~/AD/exemplesJava'. The user has entered 'javac HelloVSC.java' and 'java HelloVSC'. The output shows 'Hola Visual Studio Code!' and 'Estem provant un nou editor!'. The terminal window title is '1: Java Process Console'.

```
alumne@eljust:~/AD/exemplesJava$ javac HelloVSC.java
alumne@eljust:~/AD/exemplesJava$ java HelloVSC
Hola Visual Studio Code!
Estem provant un nou editor!
alumne@eljust:~/AD/exemplesJava$
```

**Figura 10:** Execució des de consola

Si voleu més informació sobre la depuració en Java podeu trobar-la a:

- <https://code.visualstudio.com/docs/editor/debugging>