# Ferramentas de programación

# Introdución

Para poder desenvolver páxinas web é necesario coñecer como funciona Internet. De forma resumida, **Internet** é un conxunto de equipos (de todo tipo: portátiles, ordenadores de sobremesa, móbiles, tabletas, servidores, reloxos intelixentes, etc.) conectados entre si. Cando se solicita unha páxina web nun navegador usando o seu enderezo URL, créase unha petición (**REQUEST**) ao servidor web, que será procesada e devolta en forma de resposta (**RESPONSE**) ao navegador. Esta resposta está formada polas tecnoloxías **HTML**, **CSS** e **JavaScript**, e será interpretada polo navegador mostrando visualmente a páxina web.

O proceso de desenvolvemento web pode separarse en dúas parte claramente diferenciadas: **frontend** e **backend**. Facendo un símil con un restaurante, este ten dúas zonas separadas, a cociña e o salón, e as dúas deben funcionar correctamente para que o restaurante teña éxito.

Cando se solicita unha páxina web, por exemplo unha consulta en Google, esta petición chega aos servidores de Google que deben construír a resposta. Nos servidores de Google execútase o procesamento para construír unha lista ordenada cos resultados da busca en función de moitos parámetros (a resposta de Google é diferente en función da información da persoa que pregunta). Todo este procesamento ocorre nos servidores de Google, que constitúen o **backend** do desenvolvemento web (sería a cociña nun restaurante). Este procesamento pode ser en diferentes linguaxes de programación: PHP, Python, C, Java, etc.

Os servidores web, **backend**, constrúen unha resposta en HTML, CSS e JavaScript que será executada no navegador, é dicir no lado cliente ou **frontend**. Usando o símil do restaurante sería o que ocorre no salón.

No desenvolvemento das páxinas web utilízanse diferentes ferramentas e tecnoloxías. Os seguintes apartados conteñen información de configuración das ferramentas e tecnoloxías a usar durante o curso.

# Visual Studio Code

Visual Studio Code, tamén chamado VS Code, é un editor de código lixeiro que permite editar programas en múltiples linguaxes de programación.

Desde a páxina oficial pode descargarse o instalable para diferentes sistemas operativos.

VS Code é facilmente personalizable e ten múltiples plugins e extensións que facilitan a codificación de programas en diferentes linguaxes de programación.

A páxina web oficial ofrece vídeos introdutorios para coñecer mellor o funcionamento deste editor.

VS Code ten unha interface de usuario simple e mostra á esquerda un explorador e á dereita a zona de edición. Esta interface ten as seguintes características:

- Proporciona un layout simple e intuitivo que se divide en diferentes áreas: editor, barra lateral, barra de estado, barra de actividades (á esquerda da barra lateral). Ademais, debaixo do editor poden colocarse diferentes paneis.
- Permite editar ficheiros un ao lado do outro.
- Contén un minimapa no lado dereito que da unha visión rápida do código.
- Na parte superior ten a ferramenta migas de pan para navegar.
- Ten unha paleta de comandos á que se accede pulsando **Ctrl+Shift+P**. Permite acceder a múltiples funcións usando o teclado.

Visual Studio Code ofrece soporte para a maioría das linguaxes de programación, algúns dos cales veñen configurados por defecto e outros necesitan instalar algunhas extensións. Entre as características que proporciona están:

- Sintaxe resaltada e emparellamento de corchetes mediante cores diferentes (función integrada en VS Code).
- IntelliSense: completado de código intelixente.
- Análise e correccións
- Depuración.
- Renomeado automático de etiquetas: cando se cambia unha etiqueta, tamén se cambia automaticamente o seu par, ben de apertura ou de cerre.
- Navegación no código: navegación entre ficheiros, ir á definición do símbolo, ir ao símbolo, etc.

VS Code permite configurar o tema a usar: pode pulsarse na roda dentada situada na parte inferior esquerda, seleccionar “Tema de color” e escoller un dos dispoñibles. Tamén é posible instalar novos temas.

Tamén se pode configurar o tema das iconas: roda dentada -> Tema de icona de arquivo -> seleccionar, por exemplo, “Seti”.

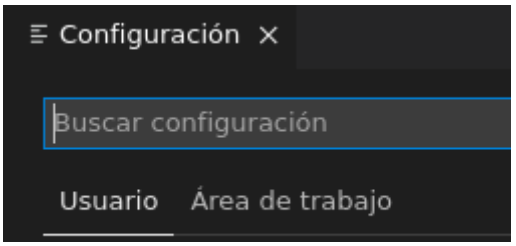
## Configuración de VS Code

É posible configurar VS Code ao gusto de cada persoa xa que ten múltiples parámetros de configuración e case todo pode ser configurado.

A configuración pode ser aplicada en dous ámbitos:

- **Configuración de usuario:** configuración aplicada á conta de usuario.
- **Configuración do espacio de traballo:** configuración almacenada para un proxecto. Normalmente almacénase na carpeta **.vscode** da raíz do proxecto.

Para acceder á configuración hai que pulsar **Arquivo -> Preferencias -> Configuración**. Tamén é posible acceder dende a roda dentada da parte inferior esquerda -> Configuración. Fixarse que é posible indicar en que ámbito se aplica a configuración, no usuario ou na área de traballo:




Normalmente, cando se traballa en equipo chégase a un acordo común de configuración para manter uniformidade no código. A continuación indícase a configuración recomendada neste curso para VS Code:

- **Auto Save (files.autoSave)**: “onFocusChange”. -> autogardado automático.
- **Editor: Font Size (editor.fontSize)** -> permite cambiar o tamaño de letra.
- **Word Wrap (editor.wordWrap): on.**
- **Format On Save (editor.formatOnSave): activado**. -> o código formátase automaticamente ao gardar (debe haber un formatador dispoñible -> configurar o “editor.defaultFormatter”).
- **FormatOnType (editor.formatOnType)**: formatar código a medida que se escribe.
- **FormatOnPaste (editor.formatOnPaste):** formatar código cando se pega.
- **Editor: Default Formatter (editor.defaultformatter):** Máis adiante neste documento instálase a extensión Prettier e configúrase este formatador.
- **Workbench > Editor: Enable Preview (workbench.editor.enablePreview)**. Se está activo, cando se abre un ficheiro este non permanece aberto, é dicir, o tabulador onde se abre reutilízase ata que o ficheiro se abre explicitamente mediante dobre clic, por exemplo. O nome do ficheiro aparecerá en cursiva no tabulador, para indicar que é unha previsualización. Se se desactiva, o arquivo permanecerá aberto.

A configuración feita almacénase nun ficheiro **settings.json** que pode editarse directamente dende VS Code dende a paleta de comandos “Abrir configuración de usuario (JSON) “.

## Atallos de teclado

Manter as mans no teclado cando se está codificando é crucial para manter unha alta produtividade. VSCode ten un conxunto amplo de atallos de teclado preconfigurados ao mesmo tempo que permite personalizalos.

A continuación indícanse documentos de referencia para os atallos de teclado en diferentes plataformas:

- Linux
- Windows
- macOS

Algúns dos atallos máis interesantes son:

- **Formateo automático** do código fonte: (Ctrl+Shift+I)
- **Terminal** integrado: Ver -> Terminal (Ctrl+`)
- **Paleta de comandos**: Ver -> Paleta de comandos (Ctrl+Shift+P, F1)
- **Múltiples cursores**: Alt+Clic
    - Shift+Alt+Up ou Shift+Alt+Down: engadir un novo cursos encima ou debaixo da posición actual.
    - Ctrl+Shift+L: seleccionar todas as ocorrencias da selección actual.
- Activar **IntelliSense**: mostrar suxestións de edición (Ctrl+Space).
- Cambiar nome a un símbolo. Pulsar **F2** cando o cursor está sobre o símbolo.

## Extensións

VS Code está pensado para ser ampliado. Ten unha grande variedade de extensións que permiten engadir novas personalizacións e mellorar as súas características. Hai extensións para instalar novos temas, engadir compoñentes, dar soporte a novas linguaxes de programación, etc.

A medida que se traballa co editor, este proporcionará suxestións de extensións en función da linguaxe de programación usada.

Algunhas extensións interesantes:

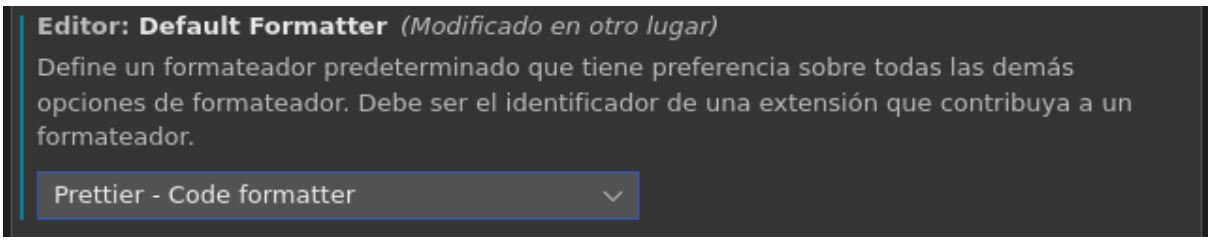
- Paquete de idioma español para VS Code. Haberá que acceder á paleta de comandos e “Configurar idioma de pantalla”
- Live Server: proporciona un servidor local para poder visualizar en tempo real os cambios nunha páxina web. Para executalo hai varias posibilidades: pulsar no botón 

na parte inferior dereita do editor ou pulsar co botón dereito na área de edición dun arquivo -> **Abrir con Live Server**.

  Unha vez instalado, comprobar cal é o navegador configurado por defecto na configuración de VS Code (**liveServer.settings.CustomBrowser**).

- Prettier: Prettier é un formatador de código.

  Para usar Prettier, hai que configurar VS Code para que o use como ferramenta de formatado por defecto. Para iso acceder ao menú **Arquivo -> Preferencias -> Configuración** e buscar “Default Formatter” (editor.defaultFormatter) para establecer o seu valor a “Prettier - Code formatter”.



Ademais, poden configurarse máis parámetros de configuración de Prettier. A forma máis fácil de facelo é crear un arquivo **.prettierrd** na raíz do proxecto e engadir a configuración. Por exemplo:

```
{
  "singleQuote": true,
  "arrowParens": "avoid"
}
```

- HTML CSS Support: autocompletado de sintaxe de HTML e CSS.
- CSS Peek: permite visualizar as propiedades CSS das clases que están no documento HTML.

  Pulsar **F12** sobre unha clase ou identificador CSS, permite acceder directamente ao ficheiro onde está definido.

  Pulsar **Ctrl+Shift+F10** sobre unha clase ou identificador CSS, permite ver a definición do elemento en liña, sen cambiar de arquivo.

- Image preview: mostra unha previsualización da imaxe no marxe ou ao pasar o rato por riba.
- Auto Close Tag.
- IntelliCode: desenvolvemento asistido por intelixencia artificial. Aforra tempo poñendo na primeira opción da lista de suxestións a que é máis probable que se use.
- Dash: integración da documentación Dash (API de documentación no navegador) en Visual Studio Code. É dicir, poderá usarse Zeal para consultar a documentación dende Visual Studio Code.

  Para usalo simplemente hai que colocar o cursor no texto a buscar e pulsar **ctrl + h.** Despois abrirase Zeal mostrando a páxina buscada. (teñen que estar descargados os docsets en Zeal).

## Emmet en VS Code

VS Code inclúe soporte para os fragmentos de código Emmet sen necesidade de ningunha extensión adicional. Emmet 2.0 proporciona soporte para a maioría das accións Emmet incluíndo as abreviaturas e fragmentos de código Emmet.

Cando se comeza a escribir unha abreviatura Emmet, despregarase na lista de suxestións.

Se se quere usar a tecla **Tabulador** para expandir a abreviatura Emmet, hai que configurar a seguinte chave:

**"emmet.triggerExpansionOnTab": true**

Pode comezarse a aprender Emmet consultando a sintaxe das abreviaturas e as accións dispoñibles.

**NOTA**: algunhas das accións dispoñibles non teñen asignado ningún atallo de teclado ou teñen outro diferente.

Na súa páxina web atópase a chuleta de emmet cun resumo dos seus comandos.

Pode consultarse toda a información na documentación de Emmet

## Fragmentos de código

Tamén é posible crear fragmentos de código accedendo ao menú **Arquivo -> Preferencias -> Configurar fragmentos de usuario.** Pode seleccionarse un dos arquivos existentes ou crear un **Novo arquivo de fragmentos globais** (opción que seleccionamos). Asignamos un nome ao arquivo a crear (**configuracionGlobal**).

Neste caso vaise crear un fragmento para automatizar a escritura de console.log. En realidade, o código de exemplo que aparece no ficheiro realiza esta función, polo que só hai que quitar os comentarios e facer uns **pequenos cambios**:

```
{
  "Print to console": {
    "scope": "javascript,typescript",
    "prefix": "cl",
    "body": ["console.log($1);"],
    "description": "Log output to console"
  }
}
```

Agora cada vez que se escriba **cl** na consola, pode pulsarse **ENTER** e completarase co comando **console.log**.

**NOTA**: a extensión JavaScript (ES6) code snippets inclúe o fragmento anterior coa abreviatura **clg** e tamén outros anacos de código JavaScript.

# JavaScript en Visual Studio Code

Visual Studio Code inclúe automaticamente JavaScript IntelliSense, depuración de código, formateo, navegación no código, refactorización e outras características avanzadas da linguaxe.

VS Code inclúe características interesantes para traballar con JavaScript. Algunhas delas son:

- VS Code inclúe anacos de código básicos de JavaScript.

- Cando se pon o rato enriba de calquera símbolo de JavaScript pode verse información e documentación relevante do símbolo.
- Axuda coa sinatura das funcións: mostra información sobre a función e os parámetros.
- Navegación no código: permite navegar de forma rápida no código do proxecto. Por exemplo F12 vai ao código fonte da definición dun símbolo.
- Renomeado: F2 permite cambiar o nome do símbolo onde está posicionado o cursor.
- Linters: os linters son ferramentas de análise de código estático empregadas para marcar erros de programación. VS Code non inclúe ningún automaticamente, sen embargo existen extensións: ESLint, jshint, Flow Language Support, StandardJS, etc.

Extensións interesantes para traballar con JavaScript:

- JavaScript (ES6) code snippets: anacos de código JavaScript. Na ligazón poden verse as abreviaturas e os anacos de código asociados que se xerarán automaticamente ao pulsar a tecla TAB.

As extensións anteriores son as que se van usar no curso, aínda que hai máis dispoñibles. Hai que ter coidado con conflitos entre extensións cando se instalan varias coas mesmas funcionalidades.

Outras extensións interesantes:

- Babel JavaScript: resaltado da sintaxe
- TabNine: autocompletado de fragmentos de código
- SonarLint: é un linter (ferramenta de análise de código estático empregada para marcar erros de programación).
- ESLint: Integra ESLint en VS Code. ESLint é unha ferramenta que analiza o código para detectar problemas, moitos dos cales poden ser solucionados automaticamente.

  Esta extensión usa a librería instalada no espacio de traballo do directorio. Se o directorio non proporciona a instalación, búscase unha versión global.

  Para que funcione é necesario ter Node.js instalado.

  ```
  # Instalar ESLint no directorio de traballo
  npm install eslint

  # Pode ser necesario crear un ficheiro de configuración
  npm init @eslint/config

  # NOTA: o comando anterior asume que existe un ficheiro package.json. Se non é
  # así, haberá que executar npm init antes para crealo.
  # O ficheiro de configuración de ESLint é .eslintrc.{js,yml,json}
  ```

# Git en VS Code

Visual Studio Code ten integrado un sistema de control de versións e inclúe soporte para Git. Máis información.

# Editores en liña

Tamén é posible usar editores en liña que permiten ver o código e o resultado á vez. Normalmente teñen varias pestanas ou seccións da páxina onde poñer o código HTML, CSS, JavaScript e ver o resultado.

Algúns dos máis coñecidos son Codesandbox, Fiddle, Plunker, CodePen, … aínda que hai moitos máis.

# Ferramentas de desenvolvemento do navegador

Todos os navegadores web modernos inclúen un potente conxunto de ferramentas para persoas desenvolvedoras. Estas ferramentas permiten dende, inspeccionar HTML, CSS e JavaScript xa cargados, ata mostrar que activos solicitou a páxina e canto tempo tardaron en cargarse.

As DevTools no navegador poden activarse dende o menú, pulsando ***Ctrl+Shift+I*** ou **menú contextual -> Inspeccionar**.

Pode obterse máis información sobre as ferramentas de desenvolvemento web dos diferentes navegadores:

- Inspector de páxinas de Firefox.
- Documentación para desenvolvedores de Microsoft Edge.
- Chrome.
- Safari inspector e explorador de estilos.

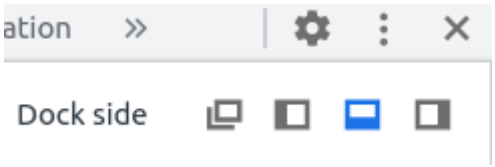
# Chrome DevTools

Chrome DevTools é un conxunto de ferramentas que permiten editar páxinas dinamicamente e diagnosticar problemas de forma fácil.

Hai varias formas de abrir Chrome Dev Tools:

- **Abrir o panel de elementos para inspeccionar o DOM ou CSS**: pulsar co botón dereito do rato sobre un elemento da páxina -> **Inspeccionar.**
- **Abrir o panel de Consola para ver mensaxes de log ou executar JavaScript: Control+Shift+J**

Para seleccionar a posición onde se coloca a ventá de DevTools pode facerse dende a paleta de comandos ou pulsando nos tres puntos á dereita da roda dentada:




Unha vez aberta, a ferramenta ten un panel que permite executar comandos. Para abrilo hai que pulsar **Ctrl + Shift + P**.

Na páxina web de Chrome DevTools teñen titoriais interactivos para aprender o básico. Por exemplo:

- Ver os nodos do DOM:
  - Inspeccionar un nodo.
  - Navegar pola árbore DOM co teclado.
  - Scroll into view.
  - Buscar nodos.
- Editar o DOM
  - Editar o contido.
  - Editar atributos.
  - Editar o tipo de nodo.
  - Editar como HTML.
  - Reordenar nodos.
  - Forzar o estado.
  - Borrar un nodo.
- Acceder a nodos na consola:
  - Facer referencia ao nodo actual con $0.
  - Almacenar como variable global.
  - Copiar o path JS.
- CSS
  - Ver o CSS dun elemento.
  - Engadir unha declaración CSS a un elemento.
  - Engadir unha clase CSS a un elemento.
  - Engadir un pseudoestado a unha clase.
  - Cambiar as dimensións dun elemento.
- CSS Reference.
  - Ver só o CSS aplicado actualmente a un elemento.
  - Ver o modelo caixa dun elemento.
- Simular dispositivos móbiles co *Device Mode*: este modo permite simular como se verá e cal será o rendemento dunha páxina nun dispositivo móbil. Simplemente é unha simulación, pois hai aspectos dos dispositivos móbiles que non poden ser simulados como a arquitectura, que é moi diferente á dun ordenador de sobremesa.

# Consola de JavaScript

A consola de JavaScript é unha ferramenta moi útil para depurar JavaScript. Para acceder a ela dende as DevTools hai que seleccionar a pestana “Consola”.

Na consola poderanse ver os erros e advertencias que xera o código e todas as mensaxes que se inclúan para facer depuración (comandos **console.log** e **console.error**).

Ademais, poderanse escribir en consola instrucións JavaScript que se executarán mostrando o seu resultado. Tamén será usada para mostrar o valor das variables e probar código.

- Titorial para mostrar mensaxes na consola.
- Titorial para executar JavaScript na consola.
- Ver en tempo real o valor dunha expresión.
- Formatear e aplicar estilo ás mensaxes da consola.

Pode consultarse a páxina de referencia das características de consola e a páxina de referencia da API para unha información máis completa.

Chrome DevTools tamén proporciona unha API con funcións útiles para usar na consola.

Pode obterse máis información sobre a consola de JavaScript nos diferentes navegadores:

- Consola Web de Firefox.
- Consola de JavaScript Edge.
- Consola JavaScript de Chrome.
- Consola en Safari.

# Panel Sources

O Panel “Sources” pode usarse para ver ficheiros, editar CSS e JavaScript, depurar código JavaScript, etc.

- Titorial para depurar JavaScript.
- Debugging in the browser

Para máis información pode consultarse como pausar o código con breakpoints ou a guía de referencia de depuración de código de JavaScript que inclúe como facer lexible un código minificado.

Pode obterse máis información sobre o depurador de JavaScript nos diferentes navegadores:

- Depurador de JavaScript en Firefox.
- Depurador de Microsoft Edge.
- Depurador de Safari.

# Referencias

Para a elaboración deste material utilizáronse, entre outros, os recursos que se enumeran a continuación:


- https://code.visualstudio.com/learn
- https://code.visualstudio.com/docs/languages/javascript
- Chrome DevTools
- Retos de codificación: https://www.codewars.com/