

Documentación Git y GitHub

1. Conexión SSH con GitHub y Git

La conexión SSH permite autenticar y asegurar la comunicación entre tu máquina y GitHub sin necesidad de ingresar credenciales repetidamente. Para configurarla:

1. Genera una clave SSH si no la tienes:
ssh-keygen -t rsa -b 4096 -C "tu-email@ejemplo.com"
2. Copia la clave pública:
cat ~/.ssh/id_rsa.pub
3. Agrega la clave pública en GitHub:
 - Ve a **Settings > SSH and GPG Keys** en GitHub.
 - Agrega la clave copiada.
4. Verifica la conexión:
ssh -T git@github.com

2. Creación de repositorios desde local y remoto

Para crear un repositorio en GitHub:

1. Ve a GitHub y haz clic en **New Repository**.
2. Asigna un nombre y configuración (público/privado).
3. Copia la URL del repositorio.

Para crear un repositorio en local y conectarlo con GitHub:

```
mkdir mi-repositorio
cd mi-repositorio
git init
git remote add origin https://github.com/usuario/mi-repositorio.git
```

3. Clonar archivos de repositorio remoto a local

Para copiar un repositorio en tu máquina local:

```
git clone https://github.com/usuario/mi-repositorio.git
```

Si usas SSH:

```
git clone git@github.com:usuario/mi-repositorio.git
```

4. Primer commit

Después de realizar cambios en los archivos, sigue estos pasos:

git add . – Agrega todos los cambios al área de staging
git commit -m "Primer commit" – Crea un commit con mensaje
git push origin main – Envía los cambios al repositorio remoto

5. Creación de rama (branch)

Las ramas permiten desarrollar nuevas funcionalidades sin afectar el código principal:

git branch nueva-rama – Crea la rama
git checkout nueva-rama – Cambia a la nueva rama

También puedes usar:

git checkout -b nueva-rama – Crea y cambia de rama en un solo paso

En versiones más actuales el comando checkout se está sustituyendo por switch, de manera que:

git switch otra-rama – Cambia a otra rama
git switch -c nueva-rama – Crea y cambia a la nueva rama

6. Fusión de ramas (merge)

Para fusionar una rama en main:

git checkout main – Cambia a la rama principal
git merge nueva-rama – Fusiona la nueva rama en main

Si hay conflictos, Git pedirá resolverlos manualmente antes de completar la fusión.

7. Merge vs Rebase

- **Merge:** Mantiene el historial de cambios con todos los commits.
- **Rebase:** Reaplica los cambios de una rama sobre otra, creando un historial más lineal.

8. Manejo de conflictos

Si al hacer merge o rebase hay conflictos, Git mostrará archivos en conflicto. Para resolverlos:

1. Edita los archivos con <<<<<<, ===== y >>>>>>.
2. Guarda los cambios y agrégalos seguimiento:
git add archivo-en-conflicto
3. Finaliza el proceso:
git commit – En caso de merge
git rebase --continue – En caso de rebase

¿Qué es Javadoc?

Javadoc es una herramienta utilizada para generar documentación en formato HTML a partir de comentarios en el código fuente de programas escritos en Java. Útil para detallar la información acerca de las clases que programemos de Java, y que así quienes la usen tengan acceso a las instrucciones necesarias para entender su funcionamiento e implementación.

Se debe incluir al documentar una clase:

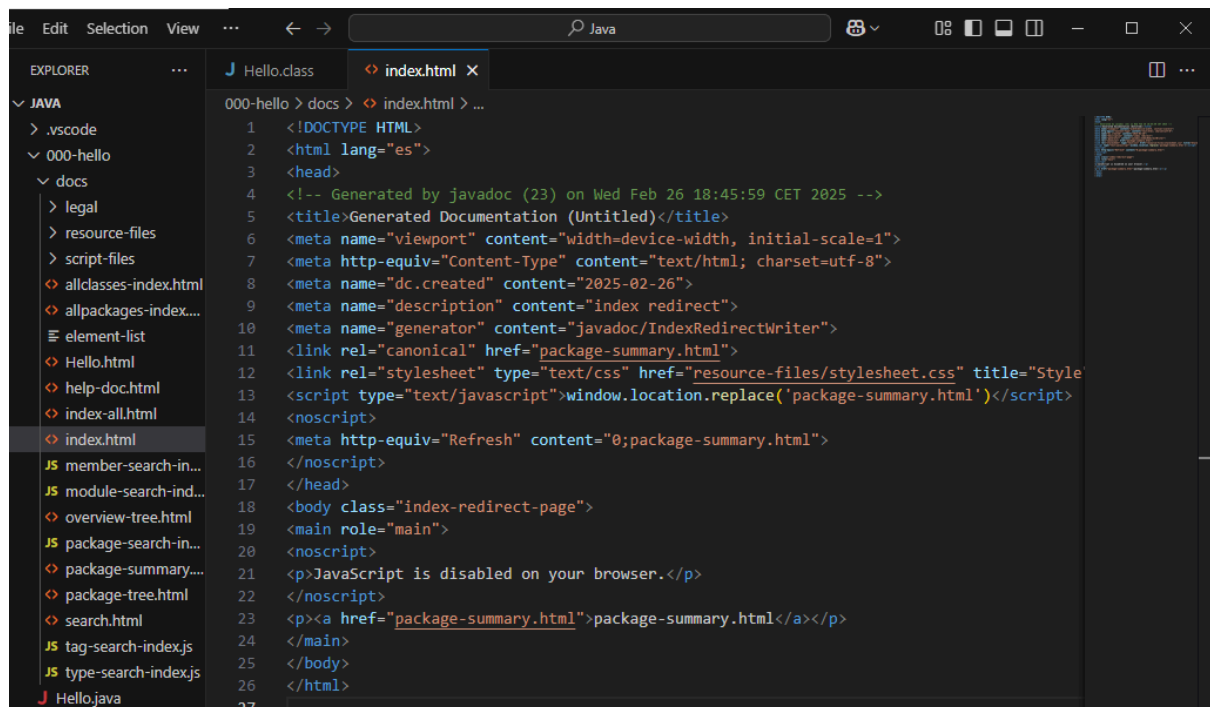
- Nombre de clase
- Descripción general
- Número de versión
- Nombres de autores
- Documentación de cada constructor o método

Para escribir comentarios en Javadoc los empezaremos con `/**` y los terminaremos con `*/`. Además, cada línea deberá empezar por `*`, y cada etiqueta por `@`.



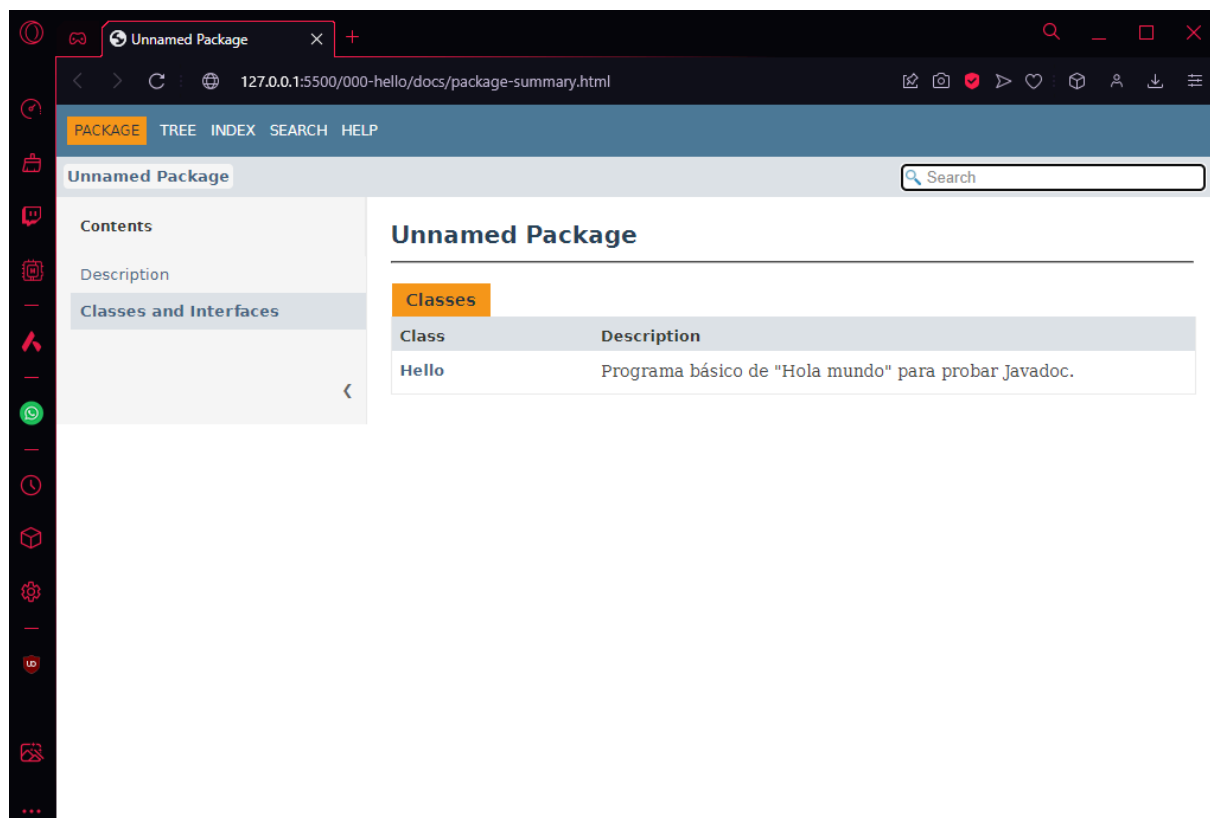
```
1  /**
2   * Programa básico de "Hola mundo" para probar Javadoc.
3   *
4   * @author Enrique García
5   * @version 1.0
6   */
7
8  public class Hello {
9
10     /**
11     *
12     * @param h1 - Primera cadena
13     * @param h2 - Segunda cadena
14     *
15     */
16
17     Run main | Debug main | Run | Debug
18     public static void main(String[] args) {
19
20         String h1 = "¡Hola, mundo!";
21         String h2 = "¿Cómo estás?";
22
23         System.out.println(h1);
24         System.out.print(h2);
25     }
26 }
```

Ejecutamos el comando `javadoc -d docs Hello.java`, con lo que guardaremos la documentación en `/docs/index.html`.



The screenshot shows the Visual Studio Code editor with the Explorer sidebar on the left. The Explorer shows a project structure with a folder named '000-hello' containing a 'docs' subfolder. The 'docs' folder contains several files, including 'index.html'. The main editor window displays the content of 'index.html', which is a generated Javadoc page. The code is as follows:

```
1 <!DOCTYPE HTML>
2 <html lang="es">
3 <head>
4 <!-- Generated by javadoc (23) on Wed Feb 26 18:45:59 CET 2025 -->
5 <title>Generated Documentation (Untitled)</title>
6 <meta name="viewport" content="width=device-width, initial-scale=1">
7 <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
8 <meta name="dc.created" content="2025-02-26">
9 <meta name="description" content="index redirect">
10 <meta name="generator" content="javadoc/IndexRedirectWriter">
11 <link rel="canonical" href="package-summary.html">
12 <link rel="stylesheet" type="text/css" href="resource-files/stylessheet.css" title="Style"
13 <script type="text/javascript">window.location.replace('package-summary.html')</script>
14 <noscript>
15 <meta http-equiv="Refresh" content="0;package-summary.html">
16 </noscript>
17 </head>
18 <body class="index-redirect-page">
19 <main role="main">
20 <noscript>
21 <p>JavaScript is disabled on your browser.</p>
22 </noscript>
23 <p><a href="package-summary.html">package-summary.html</a></p>
24 </main>
25 </body>
26 </html>
27
```



Method Details

main

```
public static void main(String[] args)
```

Parameters:

h1 -- Primera cadena

h2 -- Segunda cadena

Tipos de etiquetas

- **@author:** nombre del desarrollador
- **@version:** versión del método o clase.
- **@see:** asocia un método o clase con otro.
- **@param:** parámetro de un método, requerido para todos
- **@serial:** describe el significado del campo y sus valores aceptables
- **@since:** especifica desde qué versión está disponible un campo o método
- **@return:** lo que devuelve el método al llamarlo, no aplica en constructores o métodos “void” (ya que no devuelven nada)
- **@deprecated:** método obsoleto, no se recomienda su uso

Instalación de Javadoc (Windows)

1. Descargar JDK a través de este [enlace](#). Se puede instalar a través de archivos .zip o .exe pero consideramos que el archivo .exe es mucho más sencillo.
2. Copiar la ruta de la carpeta bin dentro de la carpeta de JDK (C:\Program Files\Java\jdk-23\bin).
3. Buscar “Variables de entorno” en el menú de inicio y abrirlo.
4. Buscar la variable “Path” y editarla.
5. Añade la ruta copiada anteriormente a una nueva entrada.
6. Guarda los cambios y reinicia la terminal.

¿Qué es PHPDoc?

PHPDoc es un estándar de documentación para código PHP que permite agregar comentarios estructurados a los archivos, clases, funciones y variables. Se basa en un formato similar a Javadoc (usado en Java) y ayuda a describir qué hace el código, los parámetros que recibe, los valores que devuelve y otras anotaciones útiles.

Los comentarios PHPDoc utilizan el formato de bloque de comentarios en PHP, comenzando con `/**` y cerrando con `*/`.

Etiquetas comunes en PHPDoc

Las etiquetas se utilizan para definir información adicional sobre los elementos documentados. Algunas de las más utilizadas son:

Etiqueta	Descripción
<code>@param</code>	Define el tipo y nombre de un parámetro de función.
<code>@return</code>	Especifica el tipo de dato que retorna la función.
<code>@var</code>	Indica el tipo de dato de una variable.
<code>@throws</code>	Describe excepciones que puede lanzar un método o función.
<code>@deprecated</code>	Marca un método o función como obsoleto.
<code>@author</code>	Nombre del autor del código.
<code>@version</code>	Versión del código.
<code>@see</code>	Referencia a otra parte del código.
<code>@since</code>	Indica desde qué versión está disponible la función o clase.

¿Qué es PHPDocumentor?

phpDocumentor es una herramienta de línea de comandos que permite generar documentación automática de proyectos PHP a partir de comentarios PHPDoc. Su propósito principal es transformar el código fuente en documentación estructurada en HTML, PDF u otros formatos.

Instalación PHPDocumentor

- **Descargamos el archivo phpdocumentor.phar**
- **Abrimos el cmd de Windows y ponemos la ruta donde esté nuestro proyecto**
- **Una vez direccionados, ponemos: php phpdocumentor.phar -d (directorio raíz donde tienes el archivo php) -t (directorio al que quieres crear la documentación)**
- **Por último, mira de nuevo el directorio al que indicaste para crearlo y verás que se creó la documentación**