Material descargable

egg

SPRING FRAMEWORK

FUNDAMENTOS WEB

El funcionamiento de la web se basa en dos pilares fundamentales: el protocolo **HTTP** y el lenguaje de marcado **HTML**.

El protocolo **HTTP** permite establecer un sistema de comunicación eficiente para la transferencia de archivos, lo que facilita el funcionamiento de los servidores web. Gracias a su diseño, incluso servidores con recursos limitados pueden gestionar múltiples solicitudes de manera eficiente, reduciendo así los costos operativos.

Por su parte, el lenguaje **HTML** proporciona un mecanismo estructurado y eficaz para la creación de páginas web interconectadas, facilitando la organización y presentación de contenido en la web.

PROTOCOLO HTTP

Un protocolo es un conjunto de reglas que define cómo debe llevarse a cabo la comunicación entre dispositivos. En el contexto de la web, el Hypertext Transfer Protocol (HTTP) es el estándar utilizado para la transferencia de información entre clientes y servidores.

HTTP establece la conexión a través del **Transport Control Protocol (TCP)**, que garantiza una comunicación estable entre el navegador (cliente) y el servidor web. A través de esta conexión, se intercambian los datos necesarios para la visualización de una página web.

En términos sencillos, cuando ingresas una dirección web en el navegador, este envía una solicitud HTTP al servidor correspondiente. El servidor procesa la solicitud y responde enviando la página web solicitada. De esta manera, HTTP actúa como el lenguaje que permite la comunicación entre el navegador y el servidor.

HTTPS: SEGURIDAD EN LA TRANSFERENCIA DE DATOS

Existe una variante de HTTP denominada **HTTPS** (Hypertext Transfer Protocol

Secure), donde la "S" significa **secure** o **seguro**. HTTPS incorpora el protocolo de seguridad **SSL (Secure Socket Layer)**, que cifra y autentica la comunicación entre el cliente y el servidor.

Este protocolo es ampliamente utilizado en sitios web que manejan información confidencial, como plataformas de comercio electrónico y servicios bancarios en línea, ya que protege los datos frente a posibles ataques y accesos no autorizados.

FUNCIONAMIENTO DE HTTP

El protocolo HTTP opera bajo un modelo de solicitud-respuesta. El proceso básico de carga de una página web puede describirse en los siguientes pasos:

1. El usuario ingresa una URL en la barra de direcciones del navegador.
2. El navegador envía una **solicitud HTTP** al servidor que aloja el dominio solicitado.
3. El servidor recibe la solicitud, busca el recurso solicitado (por ejemplo, un archivo HTML) y responde con una **cabecera HTTP**, que indica el estado de la petición.
4. Si el recurso está disponible y el cliente ha solicitado recibirlo, el servidor envía el **cuerpo del mensaje**, que contiene el contenido de la página web.
5. El navegador interpreta la respuesta y renderiza la página web para su visualización.

Este proceso se repite cada vez que se accede a una nueva página web, asegurando la transferencia eficiente de información en la web.

MENSAJES HTTP

Las interacciones en HTTP se basan en el intercambio de **mensajes** entre el cliente y el servidor. Existen dos tipos principales de mensajes:

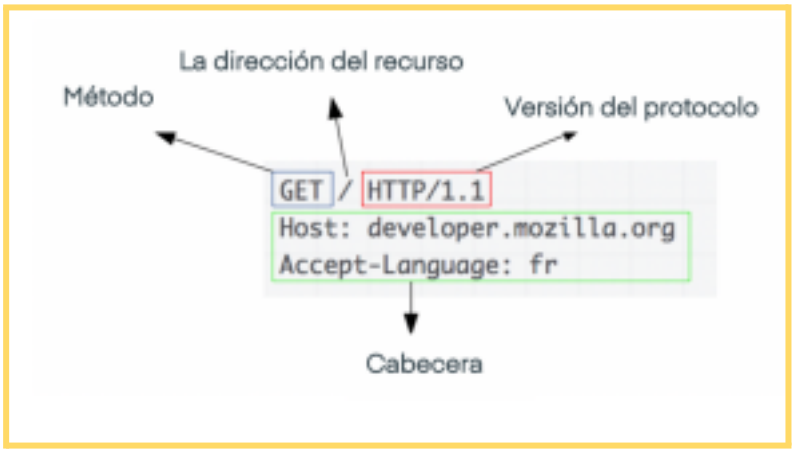
1. **Solicitudes HTTP:** Mensajes enviados por el cliente para solicitar un recurso o ejecutar una acción en el servidor.
2. **Respuestas HTTP:** Mensajes enviados por el servidor en respuesta a una solicitud, indicando si la petición fue exitosa o si hubo un error.

Cada mensaje HTTP tiene una estructura específica que incluye información esencial para la comunicación.

1. Peticiones HTTP

Una solicitud HTTP está compuesta por los siguientes elementos:

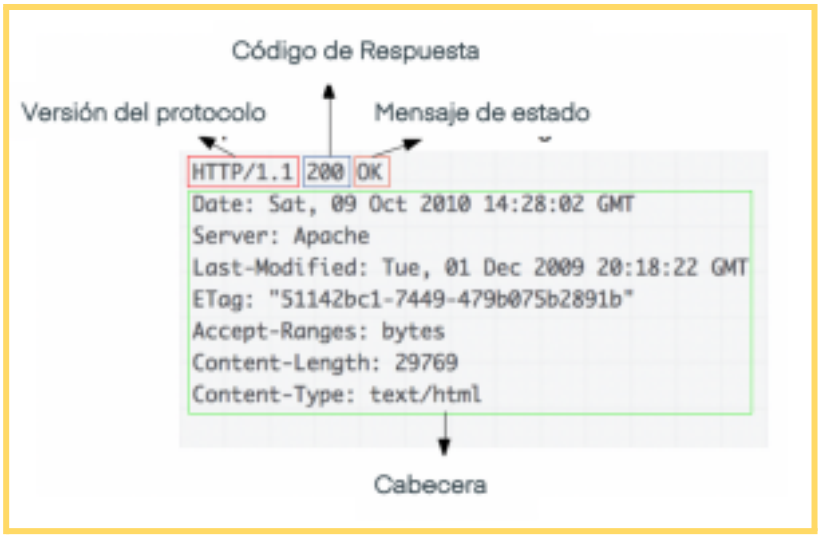
- **Método HTTP:** Indica la acción a realizar. Los métodos más comunes son **GET** (solicitar un recurso) y **POST** (enviar datos al servidor).
- **Dirección del recurso:** La URL del recurso solicitado, sin incluir elementos implícitos como el protocolo o el puerto.
- **Versión del protocolo HTTP:** Especifica la versión del protocolo utilizada en la solicitud.
- **Cabeceras HTTP:** Información adicional que el cliente puede incluir para indicar preferencias o detalles de la solicitud.



2. **Respuestas HTTP**

Las respuestas del servidor incluyen los siguientes elementos:

- **Versión del protocolo HTTP** utilizada en la comunicación.
- **Código de estado HTTP,** que indica el resultado de la solicitud (por ejemplo, `200 OK` si fue exitosa o `404 Not Found` si el recurso no fue encontrado).
- **Mensaje de estado,** que proporciona una descripción breve del código de estado.
- **Cabeceras HTTP,** que pueden incluir información sobre el servidor, la codificación del contenido, entre otros detalles.



MÉTODOS DE PETICIÓN EN HTTP

El protocolo HTTP define distintos **métodos de petición** para realizar diferentes tipos de solicitudes. Los más utilizados son:

1. **GET**

El método GET se emplea para solicitar un recurso del servidor, como una página web o un archivo.

Ejemplo:

GET /index.html HTTP/1.1
Host: www.ejemplo.com

Si el recurso está disponible, el servidor responde enviando el contenido solicitado.

2. **GET con Parámetros en la URL**

En ocasiones, es necesario incluir parámetros en la solicitud para que el servidor procese información adicional. Estos parámetros se agregan a la URL en formato **nombre=valor**, separados por &.

Ejemplo:

GET /search?platform=Windows&category=office HTTP/1.1

En este caso, el servidor recibe la información de la búsqueda y responde con los resultados correspondientes.

3. **POST**

El método **POST** se utiliza para enviar datos al servidor, como información de formularios. A diferencia de GET, los datos se envían en el cuerpo de la solicitud, en lugar de incluirse en la URL.

Ejemplo de un formulario HTML que envía datos mediante POST:

```
<html>
<body>
    <form action="/prueba " method="post">
        Name: <input type="text" name="name"><br>
        E-mail: <input type="text" name="email"><br>
        <button type="submit">
    </form>
</body>
</html>
```

El uso de POST es recomendado cuando se manejan datos sensibles, ya que evita exponer la información en la URL.

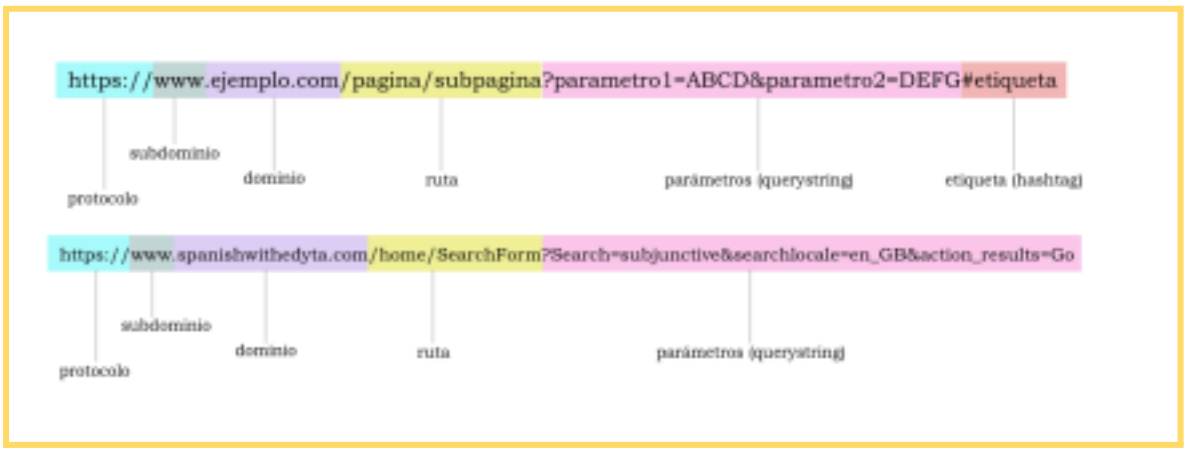
PARTES DE UNA URL

A una petición **GET** se le puede añadir información adicional para que el servidor la procese. Estos parámetros de URL se adjuntan a la dirección y pueden dividirse en distintas partes:

- **Ruta (Path):** Es la parte de la URL que sigue a la barra `/`. Indica páginas o subpáginas dentro de un sitio web.
    - Ejemplo: `www.ejemplo.com/otraPagina.html`

- **Parámetro (Query String):** Es la información que sigue al signo de interrogación `?`. Se compone de un nombre y un valor en el formato `nombre=valor`. Cuando hay varios parámetros, se separan con `&`.
    - Ejemplo: `GET /search?platform=Windows&category=office`
- **Etiqueta (Fragmento):** Aparece después del símbolo `#` y permite desplazarse a una sección específica dentro de la página.

Ejemplo de una URL completa:



¿CUÁNDO USAR GET O POST?

- **GET**: Se usa para obtener recursos del servidor. Es útil para personalizar páginas web, guardar búsquedas o configuraciones mediante la URL.
- **POST**: Se emplea para enviar datos al servidor, como formularios o archivos. Permite enviar información con una longitud ilimitada.

CÓDIGOS DE RESPUESTA HTTP

Cuando un cliente (como un navegador) realiza una petición al servidor, este responde con un **código de estado HTTP** de tres dígitos. Estos códigos indican si la solicitud fue procesada correctamente, si hubo un error o si se requiere autenticación.

Los códigos de estado HTTP se dividen en cinco categorías:

- **1xx – Información:** Indican que la petición sigue en curso.
- **2xx – Éxito:** La solicitud fue procesada correctamente.
- **3xx – Redirección:** Se requiere una acción adicional para completar la solicitud.
- **4xx – Errores del cliente:** La petición es incorrecta o no puede ser procesada.

- **5xx – Errores del servidor:** Ocurrió un problema interno en el servidor.

Códigos de estado HTTP más comunes:

- **200 – OK:** Petición procesada con éxito.
- **301 – Redirección permanente:** Indica al navegador que la dirección ha cambiado.
- **403 – Prohibido:** Acceso denegado por falta de permisos.
- **404 – No encontrado:** El recurso solicitado no existe.
- **500 – Error interno del servidor:** Fallo en la ejecución del servidor.

Para más información sobre los códigos de estado, puedes consultar el siguiente enlace.

Comprender el funcionamiento de los navegadores y servidores es fundamental para desarrollar programas que interactúen eficazmente con el protocolo HTTP.