educación IT
CODE YOUR FUTURE

Java Standard Web Programming

Módulo 8

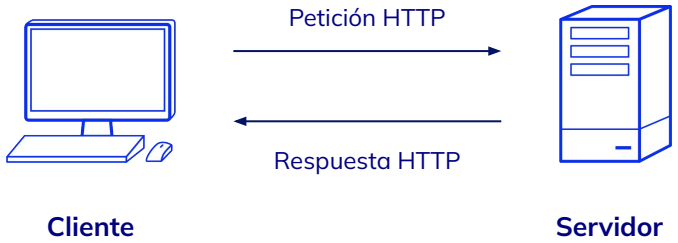
Cliente - Servidor

Protocolo HTTP

Es un protocolo *cliente-servidor*, lo que significa que el cliente envía una petición (`request`) al servidor y espera un mensaje de respuesta (`response`) del servidor.

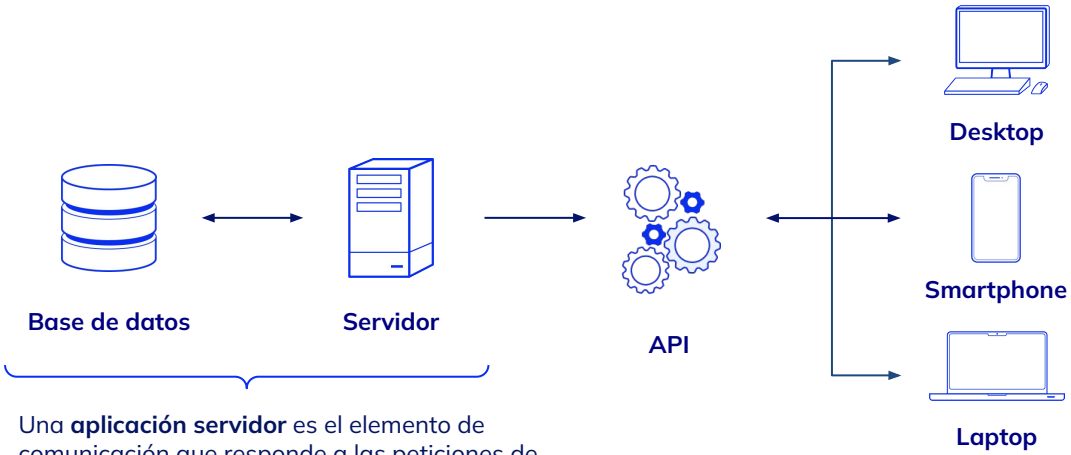
Es un protocolo **sin estado**, lo que significa que el servidor no guarda información del cliente, cada petición es independiente de las demás.

Las peticiones se realizan a través de verbos que definen la acción que se va a realizar en el recurso del servidor.



Cliente - Servidor

Una **aplicación cliente** es el elemento de la comunicación que pide o solicita un servicio de red.



Una **aplicación servidor** es el elemento de comunicación que responde a las peticiones de los clientes proporcionando el servicio requerido.

Ventajas

Escalabilidad

Se puede aumentar la capacidad de clientes y servidores por separado. Cualquier elemento puede ser aumentado (o mejorado) en cualquier momento. También, es posible agregar nuevos nodos a la red (clientes y/o servidores).

Seguridad

Permite administrar permisos de forma más localizada.

Fácil mantenimiento

Al estar distribuidas las funciones y responsabilidades entre varias computadoras independientes, se puede reemplazar, reparar, actualizar, o incluso trasladar un servidor, sin que sus clientes se vean afectados por el cambio. Actualmente, es frecuente también tener servidores en la nube.

Request

Para hacer las peticiones, el protocolo **HTTP** proporciona una serie de métodos llamados ***verbos*** para realizar una acción en el recurso determinado. Los **verbos** más utilizados son:

- **`GET`**: solicita una representación de un recurso específico. Las peticiones que usan este verbo solo deben recuperar datos, pues la información que se envía viaja a través de la URL.
- **`HEAD`**: pide una respuesta idéntica a la de una petición `GET`, pero sin el cuerpo de la respuesta.
- **`POST`**: se utiliza para enviar una entidad a un recurso en específico, causando a menudo un cambio en el estado o efectos secundarios en el servidor.
- **`PUT`**: reemplaza todas las representaciones actuales del recurso de destino con la carga útil de la petición.
- **`DELETE`**: borra un recurso en específico.

Response

Representa un recurso que generalmente es un HTML pero puede ser de cualquier otro tipo.

Además de devolver el recurso solicitado, si todo salió de forma exitosa, **la respuesta devuelve siempre un código que provee más información.**

- Respuestas informativas (100–199).
- Respuestas satisfactorias (200–299).
- Redirecciones (300–399).
- Errores de los clientes (400–499).
- Errores de los servidores (500–599).

Los **más comunes** que normalmente vemos:

- 404: Recurso no encontrado.
- 500: Error en el Servidor

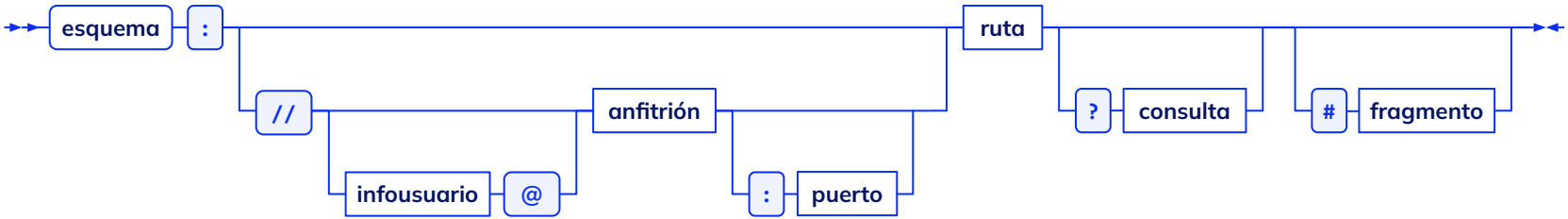
URL (Uniform Resource Locator)

Localizador de Recursos Uniformes es una dirección dada a un recurso único en la web.

En teoría, cada *URL* válida apunta a un único recurso.

- **Esquema**: define el protocolo a utilizar. Por ejemplo: `http:`, `https:`, `mailto:`, `ftp:`, y otros.
- **Host**: la *IP* o el nombre del servidor (Dominio) que se quiere acceder.
- **Puerto**: el puerto en el que está escuchando el servidor *HTTP*. Si se omite, se asume que es el puerto 80.
- **Ruta**: la ruta al recurso que se quiere acceder.
- **Consulta**: contiene información adicional para el servidor en forma de propiedades (`atributo=valor`). Las propiedades se separan por un ampersand **“&”**.
- **Fragmento**: la referencia a una ubicación interna del documento.

URL


esquema
:
//
infousuario
@
anfitrión
:
puerto
ruta
?
consulta
#
fragmento

Client-Side Processing

Client-Side Processing

En el desarrollo web, el "*lado del cliente*" también conocido como ***Front End***, se refiere a todo lo que, en una aplicación, se muestra o tiene lugar en el *cliente* (dispositivo del usuario final). Esto incluye **lo que ve el usuario**, como texto, imágenes y el resto de la interfaz de usuario. También, **las acciones que realiza una aplicación dentro del navegador del usuario.**

Para el desarrollo del lado del cliente, lo más utilizado es: **HTML** para darle estructura, ***CSS*** para estilizar y **JavaScript** para la lógica y validación.

Navegador

Un navegador web es un **software que te lleva a cualquier lugar de Internet**. Podemos decir que un navegador representa en la mayoría de los casos al cliente que realiza solicitudes al servidor. Adicionalmente, tiene la responsabilidad de "*comprender*" las respuestas provenientes del servidor, interpretarlas y visualizarlas en pantalla de modo gráfico.

Es importante tener presente que los navegadores **solo saben procesar texto plano, *HTML*, *CSS* y *JavaScript*.**

HTML

Hyper Text Markup Language (lenguaje de marcas de hipertexto) es el **lenguaje de marcado que usamos para estructurar y dar significado** a nuestro contenido web.

> **Por ejemplo**
>
> Definir párrafos, encabezados y tablas de datos, o insertar imágenes y videos en la página.

CSS

Las **hojas de estilo en cascada** (*CSS, Cascading Style Sheets*) son reglas de estilo que usamos para **dar forma** a nuestro contenido *HTML*,

> **Por ejemplo**
>
> Establecer colores de fondo y tipos de letra, y distribuir el contenido en múltiples columnas.

JavaScript

Es un **lenguaje de programación interpretado** que permite crear contenido de actualización dinámica, controlar multimedia, animar imágenes y prácticamente todo lo demás.

educación IT
CODE YOUR FUTURE

Server-Side Processing

Server-Side Processing

En el desarrollo web, el "*lado del servidor*" también conocido como ***Backend*** es todo lo que **sucede en el servidor**. Esto incluye los datos y lógicas que mostrará el cliente al recibir las respuestas por nuestros servidores.

Las tecnologías utilizadas en el lado del servidor son mucho más diversas que las del lado del cliente.

educación IT
CODE YOUR FUTURE

¡Sigamos trabajando!