Turboresumen UD1

1. Diferenciar los distintos modelos de ejecución de código en el servidor (back-end) y en el cliente (front-end) web.

1.1 Arquitectura Cliente/Servidor

En una arquitectura Cliente/Servidor, los clientes son un componente consumidor de servicios, proporcionados por un servidor.

1.1.1 Front-End (Del lado del cliente)

Parte de un sitio web que interactua con los usuarios. Tecnologías de diseño y desarrollo web que corren en el navegador.

1.1.2 Lenguajes del lado del cliente

- HTML
- CSS
- JavaScript / TypeScript / etc.

1.1.3 Back-End (Del lado del servidor)

Parte de un sitio web que trata con el servidor y base de datos. Capa de acceso a datos y la lógica de una aplicación web que no es directamente acesible por los usuarios.

1.1.4 Lenguajes del lado del servidor

- PHP
- Python
- Go
- Ruby

2. Identificar los mecanismos de ejecución de código de los navegadores web.

2.1 Arquitectura del navegador

La arquitectura de los navegadores puede ser variable, pero todos presentan una serie de elementos comunes. Pueden ejecutarse como un solo proceso, o un conjunto de procesos que se comunican entre sí.

2.2 Procesos principales

- **Navegador**

Se encarga de la aplicación en sí, las solicitudes de red y su interfaz de usuario.

- **Renderizador**

Controla todo lo que se visualiza dentro del área de la ventana / pestaña.

- **Plugins**

Controla los complementos usados por el sitio web.

- **GPU**

Maneja de forma aislada las tareas de la GPU.

2.3 Subprocesos del navegador

2.3.1 Subproceso de interfaz de usuario

Dibuja la interfaz gráfica del navegador: Marcos, botones y campos de entrada del navegador.

2.3.2 Subprocesos de renderizado

- **Subproceso de composión**
- **Suproceso de rástrer**

2.3.3 Subproceso de red

Se ocupa de la pila de red para recibir datos de internet. Implementa protocolos de documentos y ficheros (HTML / FTP) e identifica la condificación de los datos obtenidos en función de su tipo.

2.3.4 Subproceso de almacenamiento

Controla acceso a los archivos para los principales subsistemas del navegador: Historiales, marcadores, certificados de seguridad, cookies...

2.3 Flujo de navegación

2.3.1 Iniciando la navegación

El usuario accede a una URL; La entrada es manejada por la interfaz de usuario, que inicia una llamada de red para obtener el contenido del sitio.

2.3.2 Leer la respuesta

Se produce una respuesta por parte del sitio, enviando varios encabezados, que procesa el hilo de red y redirige a los hilos que corresponda.

2.3.3 Encontrar proceso de renderizado

Desde que la interfaz de usuario envía una URL al hilo de red, esta comienza a buscar o iniciar un proceso de renderizado para ahorrar tiempo mientras se recibe respuesta del sitio web.

2.3.4 Confirmar la navegación

Cuando los datos estén listos, se envía un IPC (Inter Process Communication) desde el proceso del navegador al proceso de renderizado para comenzar la navegación.

2.3.5 Carga inicial completa

El proceso de renderizado continúa cargando recursos y dibuja la página. Cuando éste finaliza*, envía un IPC de vuelta al navegador.

> * : El código JavaScript aún podría cargar recursos adicionales y generar nuevas vistas (DOM).

2.4 El proceso de renderizado

2.4.1 Construcción de un DOM

Cuando el proceso de renderizado recibe una solicitud, empieza a recibir datos HTML, analiza las cadenas de texto y lo convierte en un Modelo de Objetos del Documento (DOM).

- **DOM**

Esta es la representación interna de la página de un navegador, así como su estructura de datos y la API con la que el desarrollador puede interactuar mediante JavaScript.

2.4.2 Carga de subrecursos

A la vez que se construye el DOM, se ejecuta un escáner de precarga para obtener recursos externos, como imágenes, CSS y scripts de JavaScript.

2.4.3 Programando el estilo

El hilo principal analiza el código CSS y determina el estilo programado para cada elemento del DOM.

2.4.4 Diseño (Layouts)

El hilo principal recorre el DOM y los estilos programados para crear el árbol de diseño, que tiene información detallada como coordenadas y tamaños de los cuadros delimitadores.

2.4.5 Dibujado y composición

Mediante la composicón, se separan las partes de una página en capas, que se rasterizan por separado y luego las compone mediante el hilo del compositor. Como las capas ya están rastrerizadas, pueden adaptarse rápidamente a modificaciones.

3. Caracterizar los distintos lenguajes de programación en clientes web.

3.1 Lenguajes de programación en entornos cliente

Lenguaje	Definición
HTML (HyperText Markup Language)	No es un lenguaje de programación, es un lenguaje de marcas, interpretado, basado en un conjunto cerrado de etiquetas. Permite hipervínculos.
CSS (Cascading Style Sheet)	Lenguaje de diseño gráfico para definir la presentación de un documento web. Con ellas se separa el formato de la estructura de un sitio web.
JavaScript	Lenguaje de programación multipropósito, interpretado, y que permite ser integrado en un documento HTML. Aporta características dinámicas a un sitio web. Estándar especificado por la ECMA (ECMAScript).

3.2 Características de los lenguajes de script (guiones)

Un script es un fragmento de código que define una rutina concreta de tareas, usados para automatizar tareas ejecutadas en un intérprete de comandos.

Gracias a su formato, es posible integrar estos fragmentos de código para aportar funciones dinámicas. Debido al potencial de estos scripts, algunos pueden conformar auténticos programas.

3.2.1 Lengaujes de scripts populares

- JavaScript
- PHP
- Python
- Shell

4. Verificar los mecanismos de integración de los lenguajes de marcas con los lenguajes de programación.

4.1 Código JS dentro de HTML

Mediante el uso de las etiquetas `<script> y </script>`.

Puede colocarse en la etiqueta `<head>` ó `<body>`. Si está en `<head>` es importante tener en cuenta que no puede hacer referencia a elementos que aún no se hayan cargado.

4.2 Código JS en archivos separados

Mediante el uso de la etiqueta `<script>` y el uso del atributo `src`.

Obtenemos una serie de ventajas:

- Código HTML independiente.
- Código más facil de mantener.
- Carga más rápida.

5. Analizar, instalar y configurar las distintas herramientas y utilidades de programación para lenguajes en entorno cliente.

5.1 Herramientas y utilidades de programación

- **VSCode**
  - **Extensiones de VSCode**
- **Node**
- **Atajos de teclado**
- **Utilidades de desarrollo del navegador**

6. Utilizar los navegadores web para ejecutar, inspeccionar y depurar el código en lenguaje de programación en entorno cliente.

6.1 Utilidades de desarrollo del navegador

Los navegadores incluyen conjuntos de herramientas para facilitar labores de desarrollo.

Herramienta	Función
Modo responsivo	Simula la visualización de una página en pantallas de otras dimensiones.
Elementos	Se visualizan los elementos del DOM. Se pueden seleccionar elementos como variables, para modificar en tiempo real a través de la consola.

Herramienta	Función
Consola	Permite ejecutar código JavaScript. En el se vuelcan los mensajes envíados por funciones como `console.log()`.
Orígenes y recursos	Aparecen los dominios a los que se realizan peticiones y los recursos que se obtienen mediante ellas.
Red	Muestra las peticiones (HTTP) y sus características.
Rendimiento	Muestra el rendimiento de una pagina web.
Aplicación	Muestra las cookies y los *localStorage* de la página web.