25-9-2023

Tema 1 Ejercicios

Desarrollo web en entorno servidor

ÓSCAR PASCUAL FERRERO
IES LOS SAUCES

Ejercicios

1. Protocolos de comunicaciones:

Los protocolos de comunicación IP, TCP, HTTP y HTTPS son fundamentales en el funcionamiento de Internet y la comunicación en la web. Aquí tienes una breve descripción de cada uno de ellos:

1. **IP (Protocolo de Internet):**
    - IP es el Protocolo de Internet. Es un protocolo de la capa de red que se utiliza para enrutar paquetes de datos entre dispositivos en una red, incluida la Internet.
    - IP asigna direcciones únicas a cada dispositivo conectado a una red y garantiza que los datos se entreguen a su destino correcto.
    - Hay dos versiones principales de IP en uso: IPv4 (versión 4) y IPv6 (versión 6). IPv6 se desarrolló para abordar la falta de direcciones IP en el sistema IPv4 y para proporcionar un espacio de direcciones más grande.
2. **TCP (Protocolo de Control de Transmisión):**
    - TCP es un protocolo de la capa de transporte que proporciona una comunicación confiable y orientada a la conexión entre dispositivos en una red.
    - TCP se encarga de dividir los datos en paquetes, enviarlos al destino, reorganizarlos en el orden correcto y verificar que lleguen sin errores ni duplicados.
    - Es ampliamente utilizado para la transferencia de datos en aplicaciones que requieren confiabilidad, como la navegación web, el correo electrónico y la transmisión de archivos.
3. **HTTP (Protocolo de Transferencia de Hipertexto):**
    - HTTP es un protocolo de la capa de aplicación que se utiliza para la transferencia de datos en la World Wide Web.
    - Define la estructura y el formato de las solicitudes y respuestas entre un cliente (generalmente un navegador web) y un servidor web.
    - Las solicitudes HTTP se utilizan para solicitar recursos web, como páginas HTML, imágenes y archivos, mientras que las respuestas HTTP contienen los datos solicitados.
4. **HTTPS (HTTP Seguro):**
    - HTTPS es una extensión segura de HTTP que utiliza cifrado SSL/TLS (Secure Sockets Layer/Transport Layer Security) para proteger la privacidad y seguridad de la comunicación.
    - HTTPS se utiliza para garantizar que los datos transmitidos entre el cliente y el servidor web estén encriptados y no sean accesibles para terceros.

- Es esencial en aplicaciones donde se manejan datos sensibles, como información de inicio de sesión, tarjetas de crédito y datos personales.

Estos protocolos trabajan juntos para permitir la comunicación efectiva en Internet y para garantizar la integridad, seguridad y confiabilidad de los datos transmitidos. IP se encarga del enrutamiento, TCP garantiza la fiabilidad de la transmisión, y HTTP/HTTPS permiten la transferencia de contenido web y la interacción entre clientes y servidores web.

2. Modelo de comunicaciones cliente-servidor:

El modelo de comunicaciones cliente-servidor es un paradigma arquitectónico fundamental en la computación que se aplica en una amplia variedad de aplicaciones, incluyendo las aplicaciones web. En este modelo, dos entidades cooperan para realizar una tarea: un cliente y un servidor. Aquí se explica la relación entre el modelo de comunicaciones cliente-servidor y las aplicaciones web:

1. **Cliente**:
    - El cliente es una entidad que solicita servicios o recursos a un servidor.
    - En el contexto de las aplicaciones web, el cliente suele ser un navegador web (como Chrome, Firefox o Safari) que se ejecuta en el dispositivo del usuario.
    - El cliente puede ser una aplicación móvil o de escritorio que hace solicitudes a través de Internet a un servidor web.
2. **Servidor**:
    - El servidor es una entidad que responde a las solicitudes del cliente proporcionando los recursos o servicios solicitados.
    - En el contexto de las aplicaciones web, el servidor suele ser una computadora o un conjunto de computadoras que aloja el sitio web o la aplicación web.
    - El servidor web está configurado para procesar las solicitudes entrantes, interactuar con bases de datos, realizar lógica de aplicación y devolver respuestas al cliente.
3. **Comunicación**:
    - La comunicación entre el cliente y el servidor se realiza a través de protocolos de comunicación estándar en la web, como HTTP (Hypertext Transfer Protocol) o su versión segura, HTTPS.
    - Cuando un usuario ingresa una URL en un navegador y presiona "Enter", el navegador actúa como cliente y envía una solicitud HTTP al servidor que aloja el sitio web correspondiente.

- El servidor procesa la solicitud y envía una respuesta, generalmente en forma de HTML, CSS, JavaScript y otros recursos, al cliente para que el navegador pueda renderizar la página web.

4. **Aplicaciones Web**:
    - Las aplicaciones web son programas informáticos que se ejecutan en un navegador web y brindan funcionalidad interactiva al usuario.
    - En el modelo cliente-servidor, el cliente (navegador) solicita la aplicación web al servidor, que luego envía el código y los recursos necesarios para que la aplicación se ejecute en el navegador del usuario.
    - La interacción y la lógica de la aplicación se realizan en gran medida en el lado del cliente utilizando tecnologías web como JavaScript, mientras que el servidor puede manejar tareas como el procesamiento de formularios, acceso a bases de datos y autenticación.

En resumen, el modelo de comunicaciones cliente-servidor es esencial para el funcionamiento de las aplicaciones web, ya que permite que los clientes (navegadores) soliciten recursos y servicios a servidores web. Esto habilita la interacción y la entrega de contenido dinámico en la web, lo que es fundamental para la mayoría de las experiencias en línea que utilizamos a diario.

3. Métodos de petición HTTP/HTTPS más utilizados:

Las peticiones HTTP/HTTPS son fundamentales en el funcionamiento de la web moderna, ya que permiten la comunicación entre clientes (navegadores, aplicaciones) y servidores web. Aquí se presentan los métodos de petición HTTP/HTTPS más utilizados:

1. **GET**:
    - Uso: Solicita un recurso del servidor.
    - Características: Los datos se envían en la URL como parte de la cadena de consulta (query string).
    - Uso común: Obtener páginas web, imágenes y recursos estáticos.
2. **POST**:
    - Uso: Envía datos al servidor para ser procesados.
    - Características: Los datos se envían en el cuerpo de la solicitud.
    - Uso común: Enviar formularios, subir archivos, realizar actualizaciones en la base de datos.

    Estos son los métodos de petición HTTP/HTTPS más utilizados, y cada uno tiene su propósito específico en la interacción entre clientes y servidores web. La elección del método adecuado depende de la acción que se quiera realizar y de las convenciones de diseño de la aplicación. Además, es importante tener en cuenta

que HTTPS es la versión segura de HTTP, que cifra la comunicación entre el cliente y el servidor para proteger la confidencialidad y la integridad de los datos.

4. URI/URL/URN:

Los términos URI (Uniform Resource Identifier), URL (Uniform Resource Locator) y URN (Uniform Resource Name) son conceptos relacionados pero distintos que se utilizan para identificar recursos en la web. Aquí te proporciono información sobre cada uno de ellos y su relación con los protocolos HTTP y HTTPS:

1. **URI (Uniform Resource Identifier):**
    - Una URI es una cadena de caracteres que identifica de manera única un recurso en la web o en cualquier otro lugar. Puede ser un URL o un URN.
    - Su utilidad principal es proporcionar un identificador único y universal para recursos, lo que facilita su localización y acceso en una red.
    - Ejemplos de URIs incluyen URLs para ubicar recursos web, como "https://www.ejemplo.com/pagina" y URNs para nombrar recursos de manera persistente, como "urn:isbn:0451450523".
2. **URL (Uniform Resource Locator):**
    - Una URL es un tipo de URI que se utiliza para especificar la ubicación y el medio de acceso a un recurso en la web. Es la forma más común de URI.
    - La estructura de una URL típica incluye un protocolo (como HTTP o HTTPS), un nombre de dominio (como "www.ejemplo.com"), una ruta (como "/pagina"), y posiblemente otros componentes como puertos y fragmentos.
    - Las URLs son fundamentales en la navegación web, ya que se utilizan para acceder a páginas web, imágenes, archivos y otros recursos.
3. **URN (Uniform Resource Name):**
    - Un URN es otro tipo de URI que se utiliza para proporcionar un nombre único y persistente para un recurso, independientemente de su ubicación o método de acceso.
    - A diferencia de las URLs, que indican cómo acceder a un recurso, los URNs se centran en su identidad.
    - Ejemplos de URNs incluyen identificadores de ISBN para libros, DOI (Digital Object Identifier) para objetos digitales, etc.
4. **Relación con HTTP/HTTPS:**
    - HTTP (Protocolo de Transferencia de Hipertexto) y HTTPS (HTTP Seguro) son protocolos de la capa de aplicación que se utilizan para acceder y transferir recursos web identificados por URIs.

- Las URLs se utilizan comúnmente en las solicitudes HTTP/HTTPS para especificar la ubicación de los recursos que se desean obtener de un servidor web.
- Por ejemplo, cuando escribes "https://www.ejemplo.com/pagina" en tu navegador, estás utilizando una URL con el protocolo HTTPS para acceder a una página web identificada por esa URI.

En resumen, las URIs, que incluyen URLs y URNs, son fundamentales para identificar y acceder a recursos en la web y en otros contextos. Las URLs son la forma más común de URI y se utilizan en las solicitudes HTTP/HTTPS para acceder a recursos web, mientras que los URNs se utilizan para proporcionar nombres únicos y persistentes para recursos, independientemente de su ubicación o método de acceso.

5. Modelo de desarrollo de aplicaciones multicapa:

- El modelo de desarrollo multicapa divide una aplicación en capas lógicas, como la capa de presentación, la capa de lógica de negocio y la capa de acceso a datos. Cada capa tiene una función específica.

6. Modelo de división funcional front-end/back-end:

- El front-end se refiere a la parte de la aplicación que interactúa directamente con todos los usuarios en el navegador. El back-end se encarga de los usuarios especiales como administradores.

7. Página web estática/dinámica/aplicación web/mashup:

- Una página web estática muestra contenido fijo. Una página web dinámica utiliza datos de bases de datos u otros recursos para generar contenido en tiempo real. Una aplicación web es un software interactivo accesible a través de un navegador. Un mashup combina datos o funcionalidades de múltiples fuentes en una sola aplicación (Flickr).

8. Componentes de una aplicación web:

- Los componentes incluyen la interfaz de usuario (HTML, CSS), la lógica de aplicación (JavaScript), el servidor (back-end), sistema gestor de base de datos, navegador.

9. Programas en el lado del cliente/lado del servidor:

- En el lado del cliente, se ejecutan programas en el navegador, generalmente en JavaScript. En el lado del servidor, se ejecutan programas en el servidor web, utilizando lenguajes como PHP, Python, Ruby, etc.

10. Lenguajes de programación en el lado del servidor:

- Ejemplos incluyen PHP, Python (con frameworks como Django), Ruby (con Ruby on Rails), Java (con Servlets o JSP), Node.js (usando JavaScript), entre otros.

11. XAMPP:

- XAMPP es un paquete que incluye Apache (servidor web), MySQL (base de datos), PHP y Perl. Se utiliza comúnmente para crear entornos de desarrollo web locales.

12. Java Virtual Machine (JVM) y JDK:

- JVM se instala en el entorno de explotación y se necesita para ejecutar aplicaciones Java. JDK (Java Development Kit) se utiliza en el entorno de desarrollo para compilar y depurar código Java.

13. IDE más utilizados:

- Ejemplos incluyen Visual Studio Code, Eclipse, IntelliJ IDEA, NetBeans, y más.

14. Servidores HTTP/HTTPS más utilizados:

- Apache

  Ha sido uno de los servidores web más populares y ampliamente utilizados durante muchos años.

  Ha tenido una base de usuarios sólida debido a su estabilidad, flexibilidad y capacidad de personalización.

  Aunque su participación de mercado ha disminuido ligeramente en comparación con años anteriores, todavía se usa ampliamente, especialmente en servidores compartidos y en la comunidad de desarrollo de código abierto.

- Nginx

  Ha experimentado un crecimiento significativo en popularidad y es conocido por su rendimiento y eficiencia.

  Es especialmente popular como servidor proxy inverso y se utiliza para equilibrar la carga del tráfico web y para servir contenido estático de manera eficiente.

  Su cuota de mercado ha aumentado considerablemente, especialmente en servidores web de alto tráfico y aplicaciones web modernas.

- Microsoft IIS(Internet Information Service)

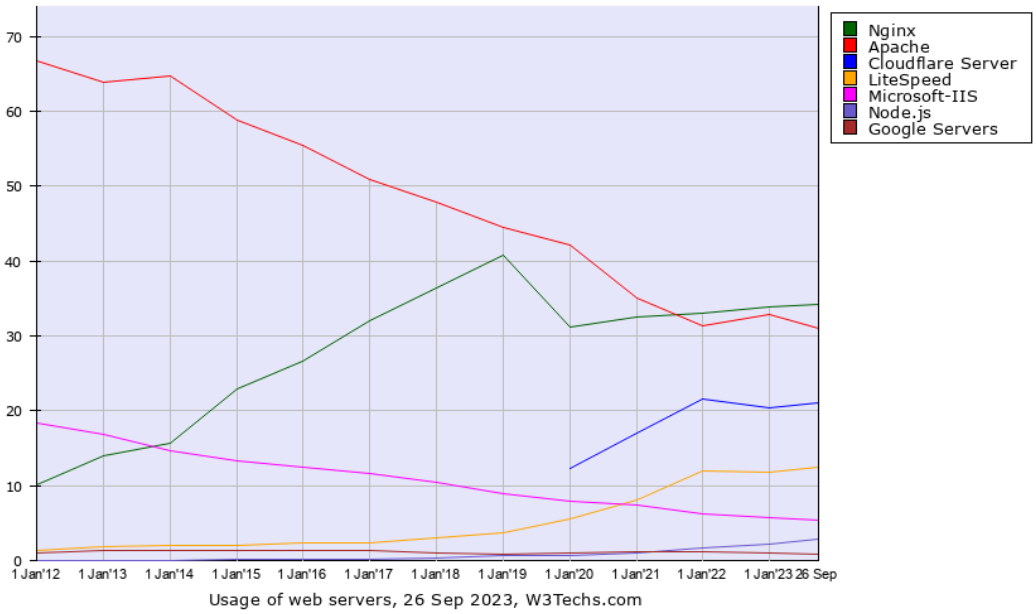
  Es el servidor web de Microsoft y es comúnmente utilizado en entornos que ejecutan sistemas operativos Windows Server.

  Tiene una base de usuarios sólida en el mundo empresarial y en organizaciones que utilizan tecnologías de Microsoft.

  Ha mantenido una participación significativa en el mercado de servidores web, especialmente en entornos corporativos.

- LiteSpeed

Es conocido por su alto rendimiento velocidad incluso bajo cargas de trabajo intensivas.



Usage of web servers, 26 Sep 2023, W3Techs.com

15. Apache HTTP vs Apache Tomcat:

- Apache HTTP Server es un servidor web HTTP, mientras que Apache Tomcat es un contenedor de servlets y JSP para aplicaciones Java.

16. Navegadores HTTP/HTTPS más utilizados:

- Google Chrome

  Rápido y eficiente, integra todos los servicios de Google.

  Líder en la implementación de estándares web y tecnologías emergentes.

  Es el navegador mas popular a fecha de hoy y desde hace años.

- Mozilla Firefox

  Enfocado en la privacidad y personalización.

  Es de código abierto con amplia comunidad activa.

  Ha perdido cuota frente a Chrome sigue siendo importante

- Microsoft Edge

  Desde que Microsoft adopto Chromium como motor de renderizado ha ganado popularidad gracias a la mejora de compatibilidad y rendimiento

- Safari

  Navegador por defecto de Apple su cuota de mercado esta centrada mayoritariamente en los dispositivos de ese ecosistema.

17. Generadores de documentación HTML (PHPDoc):

- PHPDocumentor y ApiGen son herramientas que generan documentación automática para código PHP.

18. Repositorios de software/sistemas de control de versiones:

- GIT, CVS y Subversion son sistemas de control de versiones que permiten rastrear cambios en el código fuente.

19. Configuración del entorno de desarrollo:

- La configuración puede variar según las preferencias y tecnologías utilizadas, incluyendo versiones de software y sistemas operativos.

20. Configuración del entorno de explotación:

- La configuración para la explotación debe garantizar la seguridad y el rendimiento de la aplicación en un entorno de producción.

Ten en cuenta que la elección de tecnologías y configuraciones específicas puede depender de los requisitos del proyecto y las preferencias individuales.

21. Realizar un estudio sobre los siguientes conceptos y su relación con el desarrollo de aplicaciones web:

- **CMS (Sistema de Gestión de Contenidos)**:
  - Facilita la creación y gestión de contenido web.
  - Permite la personalización del diseño y la colaboración.
  - Ayuda con SEO y escalabilidad.
  - Esencial para sitios web dinámicos y aplicaciones con contenido constante.
- **ERP (Sistema de Planificación de Recursos Empresariales)**:
  - Automatiza procesos empresariales.
  - Integra datos y funciones empresariales.
  - Permite análisis de datos y toma de decisiones.
  - Puede conectarse a aplicaciones web para mejorar la eficiencia y la gestión en línea.

Ambos sistemas, aunque diferentes, pueden complementarse en el desarrollo de aplicaciones web empresariales para mejorar la gestión de contenido y los procesos empresariales.