**Preguntas sobre HTTP**

1. Sobre que protocolo binario está montado el protocolo HTTP?

Se monta sobre el protocolo TCP

1. Cuales son los clientes http y los servidores http mas utilizados?

Sevidores http mas utilizados:

Microsoft IIS

Apache

Nginx

Google GWS

1. Que verbos admite un comando en el requerimiento HTTP

* GET: Obtiene el recurso indicado. Es el método que se utiliza cuando se pide el contenido de una página web, por ejemplo.
* HEAD: Similar a GET, pero no se obtiene el cuerpo de respuesta, únicamente los metadatos de la cabecera.
* POST: añade datos al servidor. Siempre es un método de creación.
* PUT: es una solicitud para almacenar la entidad suministrada en el URI indicado. Si la entidad no existe, se crea. Si la entidad existe, se actualiza.
* DELETE: elimina el recurso indicado.
* TRACE: devolverá la misma información que se ha enviado en la solicitud. Es una especie de eco. Sirve para comprobar si la solicitud se ha visto modificada por servidores intermedios.
* OPTIONS: Devuelve los métodos HTTP soportados por el servidor para la URL especificada.
* CONNECT: Convierte la solicitud en un tunel TCP/IP. Normalmente se usa para crear comunicaciones HTTPS a través de proxys HTTP sin encriptación.
* PATCH; Aplica modificaciones parciales al recurso especificado.

1. Que contenido lleva el body de un requerimiento HTTP?

El body puede contener datos en cualquier formato como por ejemplo:

Datos

ingresados en campos de un formulario.

Files enviados en un proceso de

upload

1. Que diferencia existe entre un URL y un URI?

La diferencia entre URL y URI es que el primero solo funciona como un localizador, indicando donde encontrar el elemento. Mientras que la URI permite tanto identificar como localizar cada recurso ya que dentro de esta se encuentra a su vez la URL, por lo que puede cumplir ambas funciones.

1. En que variables quedan almacenados datos de las respuestas http?

En las cookies

1. Que significa Virtual hosting?

El virtual hosting permite que un mismo servidor web pueda manejar múltiples nombres de hots distintos sobre una misma IP, es capaz de conmutar contenido por nombre y no por IP

1. Porque el HTTP se considera STATE LESS?

Se considera STATE LESS (sin estado) porque es muy difícil mantener el estado de una aplicación cuando luego de una sucesiva cantidad de mensajes de requerimientos y respuesta se termina cortando la conexión en un determinado momento

1. Como va a mejorar la velocidad de la WEB en poco tiempo?

Con HTTP/1.x para cargar cualquier contenido web es necesario el uso de múltiples conexiones TCP simultáneas para poder descargar todos los elementos de dicha web. En cambio, HTTP 2.0 utiliza una única conexión para ofrecer múltiples solicitudes y respuestas en paralelo.

El servicio “server push”5​ también conocido como “cache push”, se basa en estimaciones para que el servidor sea capaz de enviar información al usuario antes de que éste la solicite para que la información esté disponible de forma inmediata.

La forma de actuar del servidor es enviar varias respuestas a una única solicitud del cliente, es decir, además de la respuesta a la solicitud original, el servidor puede enviar recursos adicionales. Esto es así porque una página web está formada por decenas de archivos referenciados que gracias al servicio “server push” el servidor envía tras recibir una única solicitud ahorrando mensajes innecesarios.

HTTP 2.0 contiene un campo denominado ‘Ajustes’ con el que el cliente puede indicar si desea o no obtener los recursos que proporciona el servicio ‘server push’.

El uso del protocolo binario, la ventaja que tiene el esto es la facilidad para encontrar el comienzo y el final de cada frame, que es algo realmente complicado en cualquier protocolo de texto. Además, los protocolos binarios son mucho más simples y por lo tanto son menos propensos a tener errores que los protocolos de texto utilizados por las versiones anteriores a HTTP 2.0.

1. Como va a mejorar el inicio de las aplicaciones WEB su comportamiento en ambientes ruidosos o de debilidad de conexión?

A través de QUIC que multiplexa sub conexiones el cual permite recibir paquetes de origen que corresponden a sub conexiones distintas y si llega a aparecer un error solamente ese error va a afectar a la conexión donde se produjo evitando que no se tenga que volver a repetir todo el envío