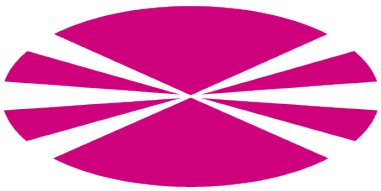

UNIVERSIDADE DA CORUÑA

LABORATORIO DE REDES

Práctica 1: Implementación de un Servidor Web

Implementación de un Servidor Web en Java

El objetivo final será obtener un servidor web capaz de interactuar con un cliente para navegar a través de un sitio web.

Requisitos

Los requisitos para el servidor web son:

- Usar el protocolo HTTP versión 1.0, en donde se generan solicitudes HTTP independientes para cada recurso (por ejemplo, ficheros HTML, imágenes, etc.).
- Hacer que el servidor sea multihilo, para poder procesar múltiples peticiones en paralelo.
- Procesar los métodos GET y HEAD de una petición HTTP.
- Procesar correctamente la cabecera If-Modified-Since de la petición, en caso de ser recibida.

El servidor web recibirá y procesará las peticiones HTTP, preparará y enviará un respuesta HTTP. Con respecto a la línea de estado de las respuestas HTTP, el servidor web implementará los códigos siguientes:

- 200 OK
- 304 Not Modified: se enviará como respuesta a una petición que incluya la cabecera If-Modified-Since, siempre que sea necesario.
- 400 Bad Request: la petición no ha sido comprendida por el servidor.
- 404 Not Found: el recurso solicitado no existe en el servidor.

Respecto a las líneas de cabecera de la respuesta HTTP, el servidor deberá implementar, al menos, las siguientes:

- **Date:** fecha y hora en la que se creó y envió la respuesta HTTP.
- **Server:** especifica el tipo de servidor web que ha atendido la petición. La cabecera Server tendrá como contenido un nombre para el servidor web a elección del estudiante.
- **Content-Length:** indica el número de bytes del recurso enviado.
- **Content-Type:** indica el tipo de recurso incluido en el cuerpo de entidad. Este campo es necesario, ya que la extensión del archivo no especifica (formalmente) el tipo de recurso asociado. Los tipos más comúnmente utilizados son:
  - text/html: indica que la respuesta está en formato HTML.

- text/plain: indica que la respuesta está en texto plano.
- image/gif: indica que se trata de una imagen en formato gif.
- image/png: indica que se trata de una imagen en formato png.
- application/octet-stream: utilizado cuando no se identifica el formato del archivo.

- **Last-Modified:** indica la fecha y hora en que el recurso fue creado o modificado por última vez.

Para procesar correctamente una petición GET con la cabecera If-Modified-Since, si la fecha de modificación del recurso solicitado es posterior a la especificada en la cabecera, el recurso deberá ser enviado normalmente (código de estado 200 OK). En otro caso, el servidor responderá con un código 304 Not Modified en la línea de estado y no se enviará el recurso solicitado.

Para manejar las fechas en Java es aconsejable leer el siguiente apartado: https://docs.oracle.com/javase/tutorial/datetime/iso/format.html. Se recomienda emplear RFC_1123_DATE_TIME como formato de las fechas (https://docs.oracle.com/javase/8/docs/api/java/time/format/DateTimeFormatter.html#patterns).

Implementación

Para la realización de esta práctica se empleará como base el proyecto facilitado en **https://github.com/GEI-Red-614G010172324/java-labs-<team-name>**. En concreto, se modificarán clases disponibles en el paquete **es.udc.redes.webserver**. Además, el proyecto incluye un fichero **README.md** con los distintos pasos para la configuración del entorno (https://github.com/GEI-Red-614G010172324/java-labs/blob/master/README.md), en caso de que algún estudiante no lo tenga ya creado.

El proyecto incluye el directorio **p1-files**, que contiene ficheros útiles para el desarrollo de la práctica, como, por ejemplo, ficheros de ejemplo en los formatos requeridos: index.html, LICENSE.txt, fic.png, udc.gif, etc. Este directorio debe ser el directorio base para recuperar los recursos solicitados por los clientes, es decir, se considerará que las rutas de los recursos solicitados son relativas a este directorio. Por su parte, el *working directory* configurado para el servidor web en IntelliJ debe mantener su valor por defecto (directorio java-labs-<team-name>*).*

El servidor web debería ser ejecutado indicando el número de puerto en el que escuchará:

java es.udc.redes.webserver.WebServer <port>

Para la implementación de la práctica se recomienda una aproximación en dos etapas. En una primera etapa, el servidor multihilo únicamente mostrará por pantalla el contenido de las solicitudes HTTP recibidas. Esto puede ser probado

iniciando el servidor en un puerto no reservado (por encima del puerto 1024, como el 5000, por ejemplo) y usando el navegador para enviar una petición al servidor http://localhost:5000/index.html.

En el segundo paso el servidor debería ser capaz de generar la respuesta HTTP apropiada a la consulta recibida, y enviar el recurso solicitado en caso de ser necesario. Si se produce algún error (por ejemplo, el fichero solicitado no se encuentra), el servidor deberá responder con el código HTTP adecuado para cada situación errónea y una página HTML para el error (en el directorio p1-files/ se proporcionan algunos ficheros HTML de error).

Comprobaciones

En los primeros pasos del desarrollo del servidor web, recomendamos el empleo del **comando nc** para comprobar la correcta ejecución del servidor.

Siguiendo los siguientes puntos se pueden evaluar diferentes aspectos del servidor web:

1. Arrancar un nc al puerto del servidor y dejarlo conectado (para probar que es multithread).
2. Abrir un navegador y cargar una página HTML que incluya texto plano, imágenes gif y png.
3. Si se muestra la página => Multithread OK, GET OK, formatos básicos OK
4. Volver al nc y enviar una petición HEAD (p.e. HEAD /index.html HTTP/1.0).
5. Comprobar que devuelve la línea de estado con las líneas de cabecera => HEAD OK.
6. Comprobar que se envían correctamente las cabeceras Date, Server, Content-Type, Content-Length y Last-Modified => Parámetros cabecera HTTP OK.
7. Arrancar otro nc al puerto del servidor y enviar una petición incorrecta (p.e. GETO /index.html HTTP/1.0): Comprobar que devuelve un error 400 Bad Request + una página de error (con las líneas de cabecera correctas) => Bad Request OK.
8. Enviar una petición a un recurso que no exista (p.e. GET /indeeex.html HTTP/1.0): Comprobar que devuelve un error 404 Not Found + una página de error (con las líneas de cabecera correctas) => Not Found OK.
9. Usando el navegador web, recargar la petición a un recurso (por ejemplo, http://localhost:5000/index.html) y comprobar que el código de respuesta es 304 Not Modified, usando las herramientas de desarrollador disponibles en el navegador.

Se proporciona una aplicación Java (p1-files/httptester.jar) para verificar que el servidor web funciona correctamente:

java -jar httptester.jar <host> <puerto> [<0-9>]

Durante la evaluación de la práctica, los tests de httptester.jar se usarán para verificar que el servidor funciona correctamente. Por tanto, recomendamos probar el servidor web con este *tester* antes de su entrega definitiva.

Evaluación

Las clases implementadas se subirán al repositorio en Github: https://github.com/GEI-Red-614G010172324/java-labs-<team-name>.

Para indicar cuál es la versión de la práctica que debe ser evaluada, tras realizar los últimos cambios para la entrega se deberá etiquetar el commit y posteriormente publicar el tag ejecutando los siguientes comandos:

1. **git tag -a p1 -m "p1"**
2. **git push origin p1**

La entrega de la práctica se realizará a través de Github. La fecha límite de entrega será el 15 de marzo de 2024 a las 23:59. No se tendrán en cuenta commits posteriores a dicha fecha.

En caso de no entregar la práctica en plazo o de que no exista en el repositorio el tag p1, se considerará que la práctica NO ha sido presentada.

La programación de sockets en Java supondrá hasta 1,25 puntos en la nota final de la asignatura. Esta calificación se calculará sumando los dos apartados siguientes:

1. Nota del script httptester.jar (0,5 puntos máximo). El profesor evaluará el funcionamiento de p1 con el script proporcionado. La nota obtenida será directamente ponderada hasta un máximo de 0,5 puntos, siempre y cuando se alcance un 5 (sobre 10) en el primer examen de prácticas (y será de 0 en otro caso).
2. Primer examen de prácticas (0,75 puntos máximo). Los contenidos de p0 y p1 serán evaluados en un examen escrito en el grupo de teoría que corresponda. Cualquier cambio de grupo deberá solicitarse y justificarse con anterioridad a la fecha del examen. **Para poder presentarse a este examen, el estudiante debe haber entregado la práctica p1**. La nota (sobre 10) será ponderada hasta un máximo de 0,75 (se alcance o no el umbral de 5 puntos). **La fecha del examen será el** 21 **de marzo de 2024.**