Secao 1 – Construindo o WebServer

Explicar a primeira parte do trabalho

Secao 2 – Tratando as requisicoes

No momento temos um Webserver recebendo chamadas de forma multithread. Entretando o nosso servidor ainda nao consegue reconhecer as chamadas que esta recebendo para dar ao usuario uma resposta de acordo.

Para comecarmos a entender as requisicoes do usuario precisamos primeiramente enteder qual o tipo de requisicao HTTP o usuario esta realizando (GET, POST, PUT, DELETE e etc) e apos isso conseguimos verificar onde o usuario esta realizando a requisicao.

Dividindo esse processo em 2 etapas vamos primeiramente reconhecer em nosso servidor o tipo de requisicao do usuario. Para isso vamos pegar o nosso leitor de Buffer que declaramos em

*BufferedReader br = new BufferedReader(new InputStreamReader(socket.getInputStream()));*

Vamos realizer a leitura desse buffer e atribui-la em uma string

*String requestLine = br.readLine();*

Agora na variavel requestLine ja temos as informacoes da requisicao realizada pelo usuario que precisamos, entretando precisamos dividir essas informacoes entre o tipo de requisicao do usuario e o caminho de requisicao do usuario, como por exemplo no caso em que o usuario realizar uma requisicao GET em “localhost:6789/about” precisamos reconhecer que essa foi uma requisicao GET no diretorio “about”. Para realizarmos essa operacao vamos utilizar dividir a String em tokens da seguinte forma.

*StringTokenizer tokens = new StringTokenizer(requestLine);*

Dentro dos nossos tokens temos uma string no seguinte formato:

“Tipo de requisicao” “diretorio” “HTTP/1.1”

Logo no primeiro token da nossa variavel tokens vamos ter o tipo de requisicao e no segundo token o diretorio que o usuario realizou a requisicao. E podemos dividir essas duas informacoes dessa forma

*String requestType = tokens.nextToken();*

*String fileName = tokens.nextToken();*

Para finalizer essa secao precisamos apenas tratar o arquivo requisitado apontando para o diretorio atual

*fileName = "." + fileName;*

Pronto com isso temos as nossas requisicoes tratadas e podemos continuar para a nossa proxima secao onde vamos entender as requisicoes do usuario e dar respostas de acordo com o que foi pedido.

Seção 3 – Respondendo o usuário

Nesse momento temos um Webserver que recebe e reconhece as requisições do usuário, entretanto o nosso servidor ainda não da nenhuma resposta independente da requisição que o usuário faça. Nesse seção nós vamos mudar isso.

Por conveniência nós não vamos dar diferentes respostas dependendo do tipo de requisição do usuário. Vamos apenas reconhecer o diretório que o usuário acessou e dar a resposta sobre o diretório.

Para começarmos essa estrutura primeiramente precisamos avaliar se o diretório que o usuário solicitou existe no nosso servidor. Vamos realizar essa operação da seguinte maneira

*FileInputStream fis1 = null;*

*boolean fileExists = true;*

*try {*

*fis1 = new FileInputStream(fileName);*

*} catch (FileNotFoundException e) {*

*fileExists = false;*

*}*

Primeiramente criamos um FileInputStream que, caso o arquivo exista, irá guardar as informações da página nele. Caso o servidor tenta pegar as informações do diretório requisitado e ocorra um erro de arquivo não encontrado vamos atribuir a uma variável auxiliar o valor de false para que dessa forma possamos devolver uma resposta de 404 Not found adequada.

Após ler o nosso arquivo precisamos devolver o arquivo para o usuário caso ele existe. Para isso primeiro vamos criar uma função que lê o nosso arquivo constrói os buffers e utiliza o nosso DataOutputStream para dar a resposta para o usuário.

*private static void sendBytes(FileInputStream fis, OutputStream os)*

*throws Exception {*

*byte[] buffer = new byte[1024];*

*int bytes = 0;*

*while((bytes = fis.read(buffer)) != -1 ) {*

*os.write(buffer, 0, bytes);*

*}*

*}*

A nossa função recebe o arquivo lido em fis. Cria um buffer de 1k para comportar os bytes no caminho para o socket e depois realiza um loop no nosso arquivo lendo os bytes e utilizando o nosso OutputStream para enviar para o usuário.

Com isso já conseguimos ler arquivos no nosso diretório e enviar para o usuário, para finalizar a construção desse seção precisamos tratar os casos em que o usuário tentou requisitar algo que não existe em nosso servidor. Para realizar essa tarefa vamos utilizar o seguinte código.

*String entityBody = null;*

*if (fileExists) {*

*// Envia o conteudo do arquivo requisitado*

*sendBytes(fis1, os);*

*// Enviar uma linha em branco para indicar o fim das linhas de cabeçalho.*

*os.writeBytes(CRLF);*

*} else {*

*entityBody = "<HTML>" +*

*"<HEAD><TITLE>404 Not Found</TITLE></HEAD>" +*

*"<BODY>404 Not Found</BODY></HTML>";*

*// Enviar a linha de conteudo.*

*os.writeBytes(entityBody);*

*// Enviar uma linha em branco para indicar o fim das linhas de cabeçalho.*

*os.writeBytes(CRLF);*

*}*

Dividimos a resposta em 2 cenários. O primeiro onde na nossa variável boolean auxiliar detectamos que o arquivo existe e dessa forma chamamos a função criada anteriormente para ler e devolve para o usuário o conteúdo. No segundo o arquivo solicitado não existe e então construímos uma resposta 404 no momento e devolvemos para o usuário. Não criamos um html para esse cenário por conveniência, entretanto o correto seria termos um html especial para tratar esse evento.

Com isso feito já conseguimos devolver para o usuário qualquer as respostas para os diretórios que ele acessar com base nos arquivos que possuímos nos nossos diretórios, temos então um Webserver que já poderia ser aplicado em diversos cenários do mundo real. Vamos continuar na próxima seção a armazenar os Logs de requisição do usuário.