MyWebServer class:

User Arguments: *port, path*

Accepts the arguments for the user for port number and path, if either argument is missing, throw an error and exit the program with a detailed error message.

Establish the listen socket

Open socket and wait for TCP connection request – do this in infinite loop

Construct an object to process the HTTP request message, this is done in the HTTPRequest class

Create a thread to process the request, and start the thread

HTTPRequest class:

Takes the port and root path as an argument. Variable *Socket* used to store a reference to the connection socket, which is passed to the constructor of this class.

Methods:

**parseDate:** Takes a string argument extracted from the substring of an If-Modified-Since: date request. Method to parse the input after Get-Modified-Since, takes three different formats, returns 0 if the date is not in correct format. If this returns 0 will set a Boolean value to true to indicate a bad request for improper format and a 400 Bad Request error will be returned.

**processRequest:** Most of the program is done within this method.

We Get a reference to the socket's input and output streams. Setup input stream filters. Gets the request line of the HTTP request message. Create a tokenizer to parse out the request and filename – check if GET message has included an If-modified-since: Next it checks to see if the path includes the root path, if not concatenates the root path to the beginning of the file. Then we create a file object and check if the file exists, and if it is a directory. If the file does not exist create a Boolean value that tells us we will response with a 404 File not found error. Create a Boolean variable to mark if the fileExists, if it is a Directory figure out if there is a “/” at the end and append an index.html to this filename. Then try to open this file to check if the index.html exists in that directory. If it does not, again create a Boolean value so that we know to return a 404 File not found error. We then check if the method is GET or HEAD – if not, then we raise a flag for method not implemented and return a 501 error.

If-Modified-Since: Get the time the file was last modified represented in seconds GMT. Now compare that to the time the file has been last modified, if it has not been modified since then – raise a flag and we return a 304 Not modified message.

Finally, all output messages are printed:

Create a string/date object to represent the current time in GMT correctly formatted for the output

1. 200 OK: If we have a get or head request and the file has been found, or a get with an if-modified and the file has been modified. Setup the statusLine, contentLengthLine (use file.available() to return the # of bytes in the file), the dateLine to return the date, serverLine to return the server name, I named mine: "Server: Nadia Aly's SNadiaimple Java Http Server.” If the file has been modified return a line with Last-modified: + date last modified. Then finally, if it is a get message we return the file/print it’s contents to the screen for the user.
2. 304 Not Modified: Return a 304 error if the Boolean value check\_modified (indicating the user has provided this in the request) and a Boolean indicating the file has not been modified since that date.
3. 404 Not Found: If the user has a get or head request and we have been unable to find the file, return this value.
4. 400 Bad Request: If the date was not pars-able or other not pars-able input from the user, this bad request will return
5. 501 Method not implemented: Returned if the Boolean method\_unavailable is set to true after we checked if the method was not GET or HEAD (case sensitive).

\*\* All output written with os.writeBytes(line);

Finally close the streams and socket if !null