Web server project

(Finalsolution)

Oscar Rénier & Nicolas Antoine

Link to github repository

https://github.com/sfsu-csc-667-spring-2022-roberts/web-server-finalsolution

Rubric

Category	Description		
Code quality	Code is clean, well formatted (appropriate white space and indentation)	5	
	Classes, methods, and variables are meaningfully named (no comments exist to explain functionality - the identifiers serve that purpose)	5	
	Methods are small and serve a single purpose	3	
	Code is well organized into a meaningful structure	2	15
Documentation	A PDF is submitted that contains:	3	
	Full names of team members	3	
	A link to github repository	3	
	A copy of this rubric with each item checked off that was completed (feel free to provide a suggested total you deserve based on completion)	1	
	Brief description of architecture (pictures are handy here, but do not re-submit the pictures I provided)	5	
	Problems you encountered during implementation, and how you solved them	5	
	A discussion of what was difficult, and why	5	
	A thorough description of your test plan (if you can't prove that it works, you shouldn't get 100%)	5	30
Functionality - Server	Starts up and listens on correct port	3	
	Logs in the common log format to stdout and log file	2	
	Multithreading	5	10
Functionality - Responses	200	2	
	201	2	
	204	2	
	400	2	
	401	2	
	403	2	
	404	2	
	500	2	
	Required headers present (Server, Date)	1	
	Response specific headers present as needed (Content- Length, Content-Type)	2	
	Simple caching (HEAD with If-Modified-Since results in 304 with Last-Modified header, Last-Modified header sent)	1	
	Response body correctly sent	3	23

Category	Description		
Functionality - Mime Types	Appropriate mime type returned based on file extension (defaults to text/text if not found in mime.types)	2	2
Functionality - Config	Correct index file used (defaults to index.html)	1	
	Correct htaccess file used	1	
	Correct document root used	1	
	Aliases working (will be mutually exclusive)	3	
	Script Aliases working (will be mutually exclusive)	3	
	Correct port used (defaults to 8080)	1	
	Correct log file used	1	11
CGI	Correctly executes and responds	4	
	Receives correct environment variables	3	
	Connects request body to standard input of cgi process	2	9

Description of architecture

Problems we encountered during implementation, and how we solved them

A discussion of what was difficult

Sticking to OOP principles, writing abstract classes from which our classes inherit if they have code or methods in common.

Test plan