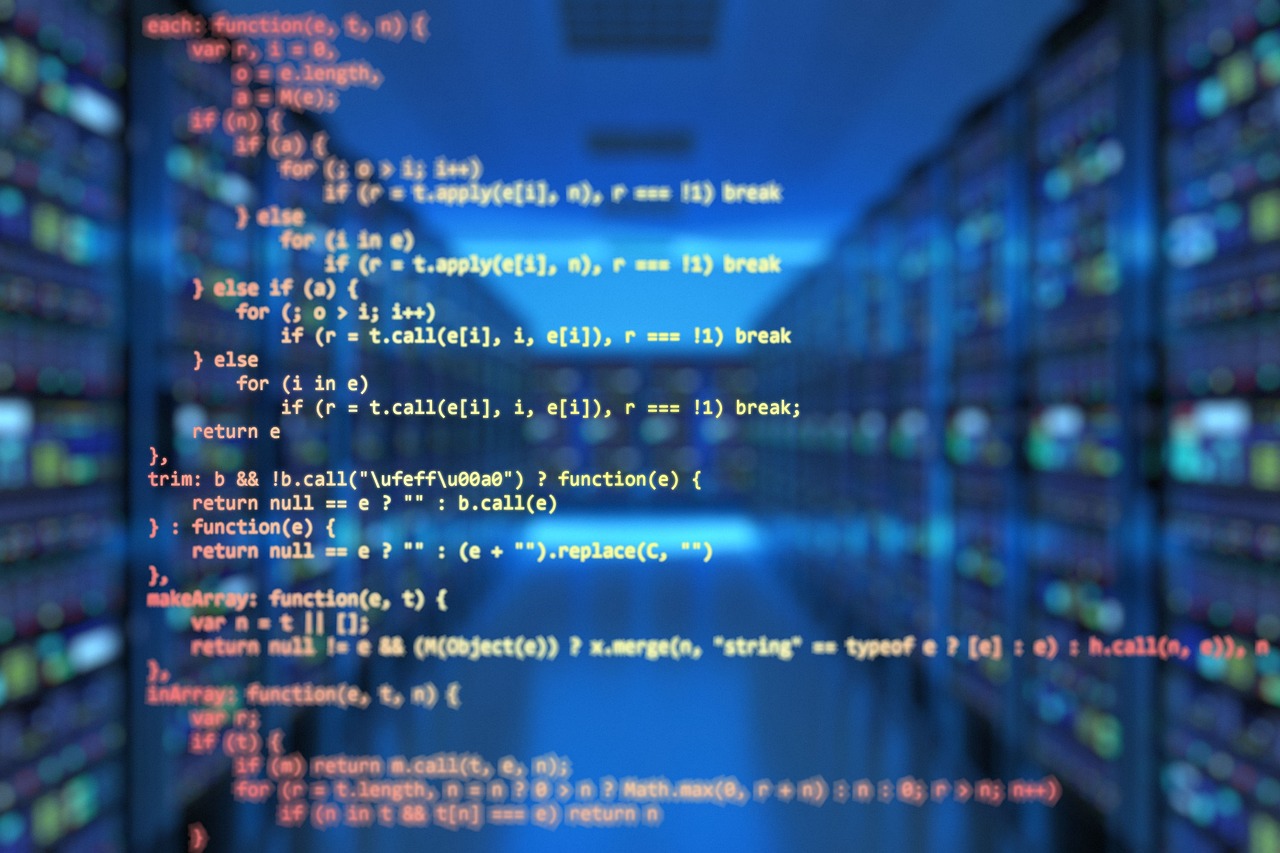
Erklärung / mini Dokumentation WebServer-js-http-run

<https://github.com/dziubanoah/WebServer-js-http-run>



!WICHTIG!: Die Browserkonsole ist in dem Projekt und bei der Implementierung dein bester freund 😊

Einführung und Funktion des Scripts:

Das script WebServer-js-http-run ist für das ausführen eine „Überweisungsbestätigung“ zuständig. Dh: auf Client 1 wird eine Überweisung durchgeführt. Diese Überweisung sollte durch einen Button „bestätigt“ werden wodurch eine JS Funktion ausgeführt wird. Diese JS Funktion wird an einen Web Server (idealerweise xampp ) geschickt und von einem PHP Scripta angenommen. Durch das PHP Scrip wird zudem ein GUI geöffnet, der die Überreizung bestätigt. Optional wird noch eine log.txt mit dem String „Team“ und dem Datum sowie uhrzeit beschrieben.

Die Umsetzung in eine reale Umgebung.

Der komplette Quellcode ist auf meine ursprüngliche Testumgebung angepasst. Somit müssen IP Adressen und Zielordner sowie Dateinamen angepasst werden. Weitere Schritte folgen .

Erklärung code:

Index.html:



Der code von index.html dient als Testumgebung , um einen Button zu klicken, der nachher eine Funktion ausführt. Der Button muss eine ID haben, damit JavaScript auch etwas mit dem Button click anfangen kann. In der Realität muss der Quellcode einer vorhandenen.html angepasst werden, damit der korrekte Buttons angesprochen wird.

Main.js

Ein Bild, das Text, Screenshot, Software, Multimedia-Software enthält.

Automatisch generierte Beschreibung  
Die main.js ist für die Kommunikation mit dem Web Server zuständig. In Zeile 1 wird der Button mit der ID (siehe index.html) aufgerufen und mit der click function wird eine Aktion bei einem klick ausgeführt. Zum testen ob der link zwischen html und js funktioniert wird ein einfacher alert gestartet. In Zeile 3-5 wird dafür gesorgt, dass der Server 192.168.1.156 mit der Methode GET angesprochen wird, und die Seite / das Script script\_mit\_log.php aufruft. Alle Zeilen danach sind nicht funktional, sondern verraten nur, ob es fehler bei der Übertragung gab (Browser / JS konsole).

FakeBank.html:

Ist nur das Frontend für die .jar Dateien und kann ggf auch als Browser „Bestätigung“ auf dem Web Server verwendet werden (Wenn CORS deaktiviert ist (nicht empfohlen)).

Andere Scripts im Hauptorder:

Werden zum aktuellen Zeitpunkt nicht gebraucht, weil das php Server Skript viel Arbeit abnimmt.

\Server scripts\

In diesem Hauptverzeichniss befindet sich der Quellcode für .jar Sachen. Sowie Klassen und Bilder. Dieses verzeichniss sollte zur Überarbeitung de Java codes dienen. Die Unterordner Ready to open .jar und Ready Jar function (main) sollen nicht verändert werden. Falls eine neue Version von dem Java code verwendet werden soll, dann bitte einen Neuen Ordner erstellen, und dort die Aktuelle Version Kopieren. Diese soll dann ebenfalls nicht mehr verändert werden. Falls ein neuer Ordner mit überarbeiten Java code der Main Ordner werden soll, dann mit v.1.1, v.1.2 etc. Arbeiten, damit nicht verändert und gelöscht wird.

\Server scripts\Ready Jar function (main)  
Dieser Ordner beinhaltet aktuell den main Ordner (bei einem anderen Ordner mit einem v. hinter main ist dies der neue Main Ordner).

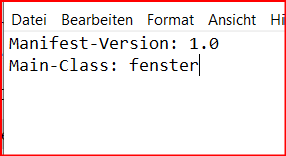
Ein Bild, das Text, Screenshot, Software, Betriebssystem enthält.

Automatisch generierte Beschreibung  
mit diesem Java code wird ein JFrame Fenster erstellt, Zeile 17-17 sind für das Fenster Design. Zeile19 & 20 geben dem Fenster ein anderes Icon, als das default java icon. Zeile 22.24 gibt dem JFrame ein Frontend in form eines .JPG. dh. Der GUI ist nicht funktional, sondern zeigt nur ein Bild. Am ende wird das Fenster visible gesetzt. Alles was das JFrame beeinflusst muss über dem setVisible stehen.

Um eine .jar zu erstellen muss zudem eine leere MANIFEST.MF erstellt werden. Durchs Kompilieren werden zudem Klassen erstellt. !WICHTIG! alle Bilder etc. auf das die .jar zugreifen soll, MUSS im selben Verzeichnis sein, wenn kein spezieller Pfard angegeben wurde. Um den code ausführbar zu machen muss jetzt der befehl: javac Dateiname.java eingegeben werden (zum Kompilieren). Dannach muss folgender Befehl eingegeben werden:   
  

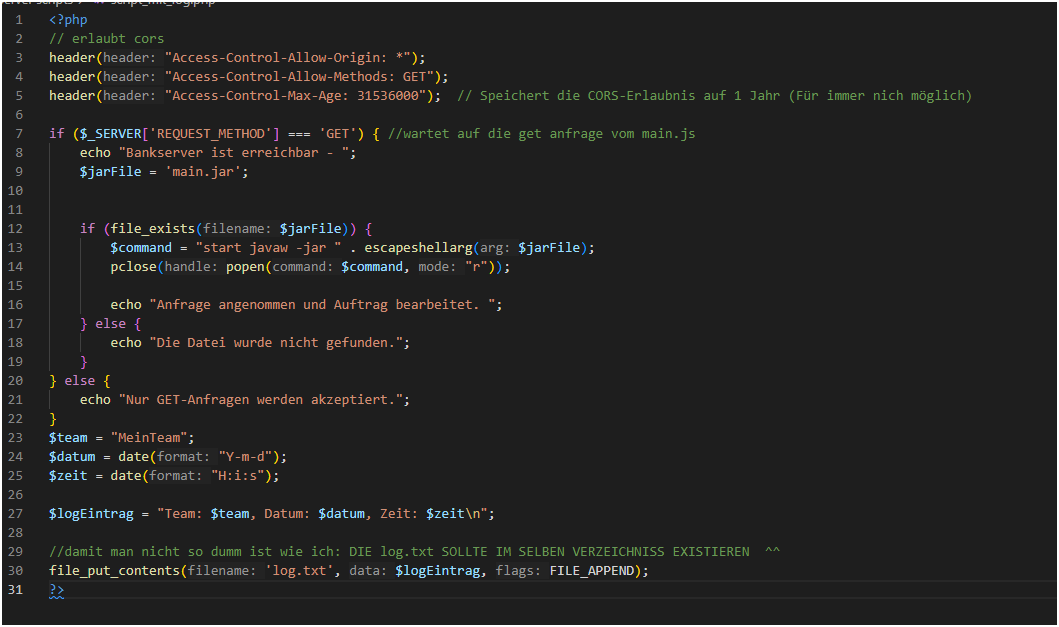

Hier wird eine Ausführbare Datei, mit allen Klassen erstellt. Trotzdem ist Die Datei nicht ausführbar. Das Liegt an der MANIFEST.MF, welche dies nicht zulässt. Dazu muss der erstellte Jar file mit einem Zip tool erstmal geöffnet werden (nicht entpackt).

In dem Ordner META-INF muss dann die MANIFEST.MF dann so gestaltet werden:

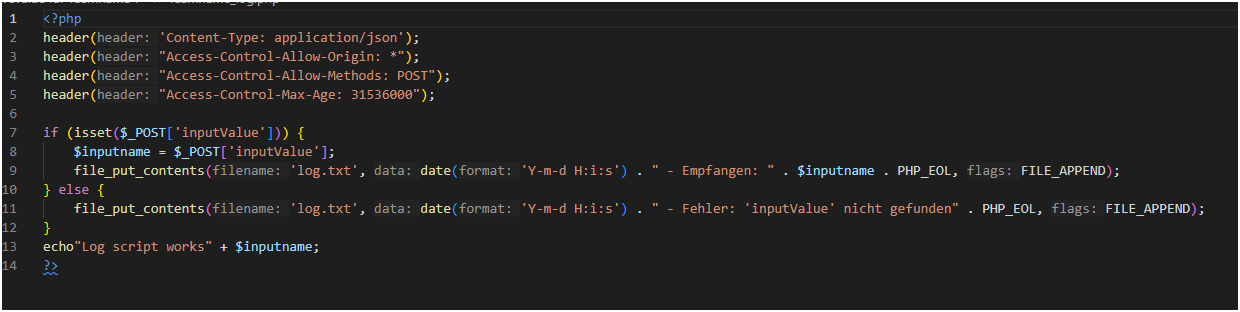


Die Main-Class muss der tatsächlichen klasse auch angepasst werden (im Hauptverzeichniss). Danach ist die Datei auch per Doppelklick ausführbar.

PHP Script für die Banküberweisung / Bestätigung script\_mit\_log.php im Verzeichnis Server scripts (Lauschen).

  
im Header werden CORS Eigenschaften verändert / aktiviert.In Zeile 7-9 wird auf eine CORS anfrage von main.js gelauscht. Im If auf zeile 12 wird ein echo zur überprüfung (sowie in den beiden else) und die Aktion $comand ausgeführt, um die Datei $JarFile (main.jar) auszuführen !WICHTIG! wenn der code nicht angepasst werden soll, muss die main.jar ebenfalls im Verzeichnis htdocs auf dem xampp sein. Im letzten drittel des codes wird das schreiben in die log.txt vorbereitet und durchgeführt.

PHP Script zum empfangen und Protokollieren von givevalue / Teamnamen Teamname\_log.php (Lauschen)

  
  
Im header werden wieder Bedingungen die zur Datenübertragung Notwendig sind aufgerufen.

Die IF Anweisung schreibt den wert bei IF in die Datei log.txt, welcher vorher in dem Textfeld zu Beginn des Escape Rooms eingetragen wurde (Hacker\_1.html & givevalue.js). Bei dem else wird eine Standard Nachricht ohne Variablen wert festgelegt (Escape Room Durchgang kann dadurch trotzdem gestartet werden, weil zumindest die Zeit protokolliert wurde).

Givevalue.js ist essentieller Bestandteil von Teamname\_log.php und arbeitet im Frontend.

  
in dem code wird bei den Klick auf den Button submit (in der Hacker\_1.html) eine POST anfrage an den Webserver geschickt, auf dem Webserver wird dann die Datei Teamname\_log.php ausgeführt. Die .then Anweisungen sind nur zur Überprüfung (nicht unbedingt funktional). Die Werte bekommt das Backend (Teamname\_log.php) über einen URL code (Zeile 12-14).

Keks.js prüft ob der Browser das Input Feld aus Hacker\_1.html anzeigen darf (Über die Cookie ID).



In dem code wird eine Cookie ID für 1 Jahr festgelegt. Aus Sicherheitsgründen kann die cookie id nicht Dauerhaft gesetzt werden. Danach kommt eine IF anfrage um zu prüfen ob die Client id stimmt um die Display Eigenschaft in Hacker\_1.html auf block zu setzen, andernfalls bleibt die Eigenschaft auf none. Ab ende kommt ein Popup damit man sehen kann, dass das Script durchgelaufen ist. In der else Anweisung wird in der Konsole ausgegeben, dass die cookie id nicht stimmt. Falls so ein error kommt wird ebenfalls die Client ID ausgegeben, diese muss dann in die IF anfrage Zeile 18 ersetzt oder eingefügt werden.

* Output für die Cookie ID aus der Firefoxkonsole  
  