

# Webgoat Google Chrome Developer Tools

Twitter: @BlackSheepSpicy

Twitch: <https://twitch.tv/BlackSheepSpicy>

## Note:

So despite Google Chrome being explicitly referenced in the name of this section, both of these challenges can be done just as well in Firefox. As that is the browser I use because open source is rad and Google makes me nervous. Even though im using Google Docs to do this writeup but that's a different conversation.

4. After clicking through information that may or may not be helpful to us we come across the first challenge:

## Try It! Using the console

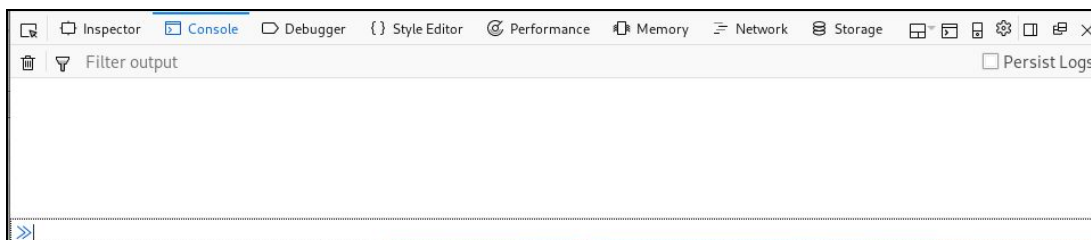
Let us try it. Use the console in the dev tools and call the javascript function **webgoat.customjs.phoneHome()**.

You should get a response in the console. Your result should look something like:

```
phone home said {"lessonCompleted:true, ... ,"output":"phone home response is..."}
```

Paste the random number, after that, in the text field below. (Make sure you got the most recent number, since it is randomly generated each time you call the function)

Pretty straightforward challenge, we can just call upon the skills we used to impress our middle school friends in computing class by opening up the dev console by hitting F12 and navigating over to the console tab:



Now from here we simply type in the function they specify (or copy and paste if you're lazy like yours truly) and hit enter:





```
Array [ "http://localhost:8080/WebGoat/CrossSiteScripting/dom-follow-up" ]
Array []
webgoat.customjs.phoneHome()
phoneHome invoked
undefined
phone home said {"lessonCompleted":true,"feedback":"Congratulations. You have successfully completed the assignment.", "output":"phoneHome Response is 1746749864"}
```

Note: if you type in the function call notice how the console will attempt to autocomplete your input (shown in light grey text. To accept this autocompletion hit the right arrow key.)

Odd thing about this challenge: The functions states we have completed the challenge though in order to fully complete the challenge we must copy and paste the response (the massive number on the end) into the text input on the page to actually complete the challenge. Also, if you take too long to submit the number the number will regenerate and it will say the number is incorrect, that threw me for a loop for a hot second, just call the function again and you'll be fine.

## 6. 🙌 NEXT 🙌 MEME

### Try It! Working with the Network tab

In this assignment you need to find a specific HTTP request and read a randomized number from it. To start click the first button, this will generate an HTTP request. Try to find the specific HTTP request. The request should contain a field: **networkNum**: Copy the number which is displayed afterwards, into the input field below and click on the check button.

Click this button to make a request:

Go!

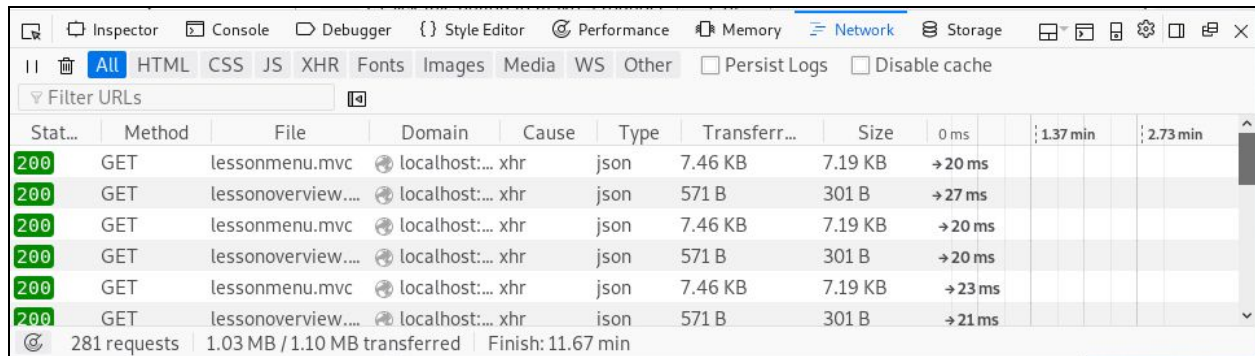
What is the number  
you found:

check

I couldn't think of another way to introduce this challenge spare me please



Before we send the request, let's go ahead and navigate over to the network tab, found near the end of the menu where we found our console:



The screenshot shows the Chrome Developer Tools Network tab. The top bar includes tabs for Inspector, Console, Debugger, Style Editor, Performance, Memory, Network (selected), and Storage. Below the tabs is a filter bar with 'All' selected and options for HTML, CSS, JS, XHR, Fonts, Images, Media, WS, and Other. There are checkboxes for 'Persist Logs' and 'Disable cache'. A 'Filter URLs' input field is present. The main area displays a table of network requests. The table has columns: Stat..., Method, File, Domain, Cause, Type, Transferr..., Size, 0 ms, 1.37 min, and 2.73 min. The first six rows show GET requests to lessonmenu.mvc and lessonoverview.... The bottom status bar indicates 281 requests, 1.03 MB / 1.10 MB transferred, and a finish time of 11.67 min.

Stat...	Method	File	Domain	Cause	Type	Transferr...	Size	0 ms	1.37 min	2.73 min
200	GET	lessonmenu.mvc	localhost:...	xhr	json	7.46 KB	7.19 KB	→ 20 ms		
200	GET	lessonoverview....	localhost:...	xhr	json	571 B	301 B	→ 27 ms		
200	GET	lessonmenu.mvc	localhost:...	xhr	json	7.46 KB	7.19 KB	→ 20 ms		
200	GET	lessonoverview....	localhost:...	xhr	json	571 B	301 B	→ 20 ms		
200	GET	lessonmenu.mvc	localhost:...	xhr	json	7.46 KB	7.19 KB	→ 23 ms		
200	GET	lessonoverview....	localhost:...	xhr	json	571 B	301 B	→ 21 ms		

281 requests | 1.03 MB / 1.10 MB transferred | Finish: 11.67 min

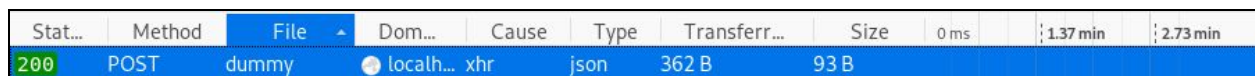
So now that we can see all the requests that our client is sending and receiving let's go ahead and fire off that request then spend the next 10 minutes getting the word "lesson" burned into our brain.

Just messing with you, we can sort the requests to make them easier to look through by clicking on the column name. As we do that, notice a little arrow to the right of the column name:

Method ▲

- When the arrow is pointing down it means the requests are being sorted in ascending order (A to Z or highest number to lowest number depending on the tab)
- In contrast when the arrow is pointing down the requests are being sorted in descending order (Z to A or lowest number to highest number depending on the tab)
- If this is confusing think of the list as just being flipped around with each column click, as in the example when the arrow is pointing up you start at the bottom of the list and work your way up.

So let's go ahead and sort through our list of requests until we find... oh...



The screenshot shows the Network tab with the 'Method' column sorted. A single request is visible: a POST request to 'dummy' on 'localhost:...' with a status of 200. The request is of type 'json' and has a size of 362 B. The response size is 93 B.

Stat...	Method	File	Dom...	Cause	Type	Transferr...	Size	0 ms	1.37 min	2.73 min
200	POST	dummy	localhost:...	xhr	json	362 B	93 B			

Who's idea was it to name a request after me?



Well... that's pretty sus... let's take a closer look at this request by clicking on it and checking out all the information on the right side, specifically under the **Params** tab:

Headers	Cookies	Params	Response	Timings	Stack Trace
▼ Filter request parameters					
▼ Form data					
networkNum: 94.69254618911347					

Welp... that looks like our payday, lets go ahead and copy/paste this into the challenge to complete it!

I'm not sure how much the solutions to the challenges will differ if you use Chrome rather than firefox, though I suspect that there will not be much of a difference. Regardless I hope I made this write up entertaining! If you want to see me do these challenges live be sure to drop by my Twitch when I'm live and also follow my Twitter for some fresh memes!

