Basic Obfuscation

Code obfuscation is usually not done manually, as there are many tools for various languages that do automated code obfuscation. Many online tools can be found to do so, though many malicious actors and professional developers develop their own obfuscation tools to make it more difficult to deobfuscate.

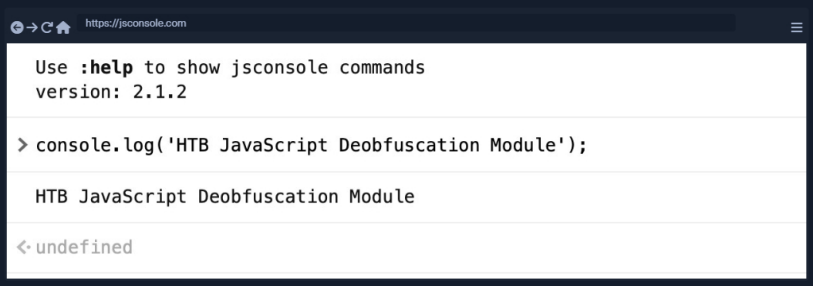
Running JavaScript code

Let us take the following line of code as an example and attempt to obfuscate it:

Code: javascript

```
console.log('HTB JavaScript Deobfuscation Module');
```

First, let us test running this code in cleartext, to see it work in action. We can go to JSConsole, paste the code and hit enter, and see its output:

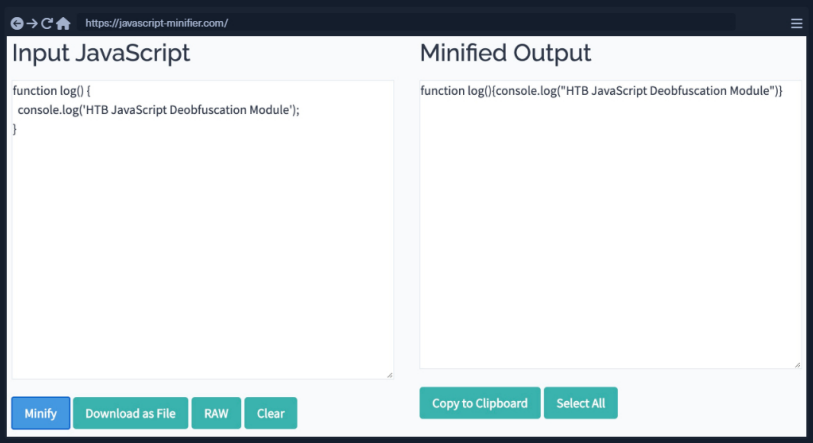


We see that this line of code prints `HTB JavaScript Deobfuscation Module`, which is done using the `console.log()` function.

Minifying JavaScript code

A common way of reducing the readability of a snippet of JavaScript code while keeping it fully functional is JavaScript minification. `Code minification` means having the entire code in a single (often very long) line. `Code minification` is more useful for longer code, as if our code only consisted of a single line, it would not look much different when minified.

Many tools can help us minify JavaScript code, like javascript-minifier. We simply copy our code, and click `Minify`, and we get the minified output on the right:

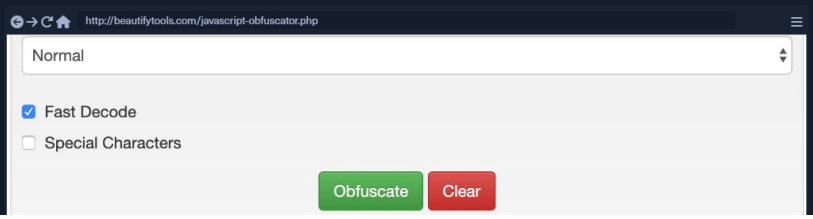


Once again, we can copy the minified code to JSConsole, and run it, and we see that it runs as expected. Usually, minified JavaScript code is saved with the extension `.min.js`.

Note: Code minification is not exclusive to JavaScript, and can be applied to many other languages, as can be seen on javascript-minifier.

Packing JavaScript code

Now, let us obfuscate our line of code to make it more obscure and difficult to read. First, we will try BeautifyTools to obfuscate our code:



Cheat Sheet

Table of Contents

Introduction
Introduction
Source Code

Obfuscation
Code Obfuscation
Basic Obfuscation
Advanced Obfuscation
Deobfuscation

Deobfuscation Examples
Code Analysis
HTTP Requests
Decoding

Skills Assessment
Skills Assessment
Summary

My Workstation

OFFLINE

Start Instance

∞ / 1 spawns left

```
eval(function(p,a,c,k,e,d){e=function(c){return c};if(!''.replace(/^/,String)){while(c--){d[c]=k[c]||c}k=[function(e){return d[e]}];e=function(){return'\\w+'};c=1};while(c--){if(k[c]){p=p.replace(new RegExp('\\b'+e(c)+'\\b','g'),k[c])}}return p}('5.4(\'3 2 1 0\');',6,6,'Module|Deobfuscation|JavaScript|HTB|log|console'.split('|'),0,{}))
```

Code: javascript

```
eval(function(p,a,c,k,e,d){e=function(c){return c};if(!''.replace(/^/,String)){while(c--){d[c]=k[c]||c}k=[function(e){
```

We see that our code became much more obfuscated and difficult to read. We can copy this code into https://jsconsole.com, to verify that it still does its main function:

We see that we get the same output.

> Note: The above type of obfuscation is known as "packing", which is usually recognizable from the six function arguments used in the initial function "function(p,a,c,k,e,d)".

A `packer` obfuscation tool usually attempts to convert all words and symbols of the code into a list or a dictionary and then refer to them using the `(p,a,c,k,e,d)` function to re-build the original code during execution. The `(p,a,c,k,e,d)` can be different from one packer to another. However, it usually contains a certain order in which the words and symbols of the original code were packed to know how to order them during execution.

While a packer does a great job reducing the code's readability, we can still see its main strings written in cleartext, which may reveal some of its functionality. This is why we may want to look for better ways to obfuscate our code.