JavaScript • Unicode

@mathias · #wildcard13

@mathias


NaNtyNaN problems

www.phpwtf.org

Default arguments and type hinting

By chx – Posted Mon, 05/06/2013 - 18:10

Run these three:

```
php -r 'function a(array $a){}; a(NULL);'
php -r 'function a(array $a = NULL){}; a(NULL);'
php -r 'function a(array $a = FALSE){}; a(FALSE);'
```

» Read more

The poor parser is easily confused

PHP can just cast numbers to strings, right?

```
<?php
print "a"."2";
?>
```

```
<?php
print "a".2;
?>
```

Results

```
a2
```

```
PHP Parse error:  syntax error, unexpected '.2' (T_DNUMBER) in Command line code on line 1
```

array comparison

jan 1 , 2013

Did you know that JavaScript can compare arrays using lexicographical ordering?

```
[1, 2, 4] < [1, 2, 5]  // true
[1, 3, 4] < [1, 2, 5]  // false
```

Just don't expect trichotomy to hold.

```
[1, 2, 3] === [1, 2, 3]   // false
[1, 2, 3] <   [1, 2, 3]   // false
[1, 2, 3] ==  [1, 2, 3]   // false
[1, 2, 3] >   [1, 2, 3]   // false
```

Oh, and just in case you're wondering, it knows it's messing with you.

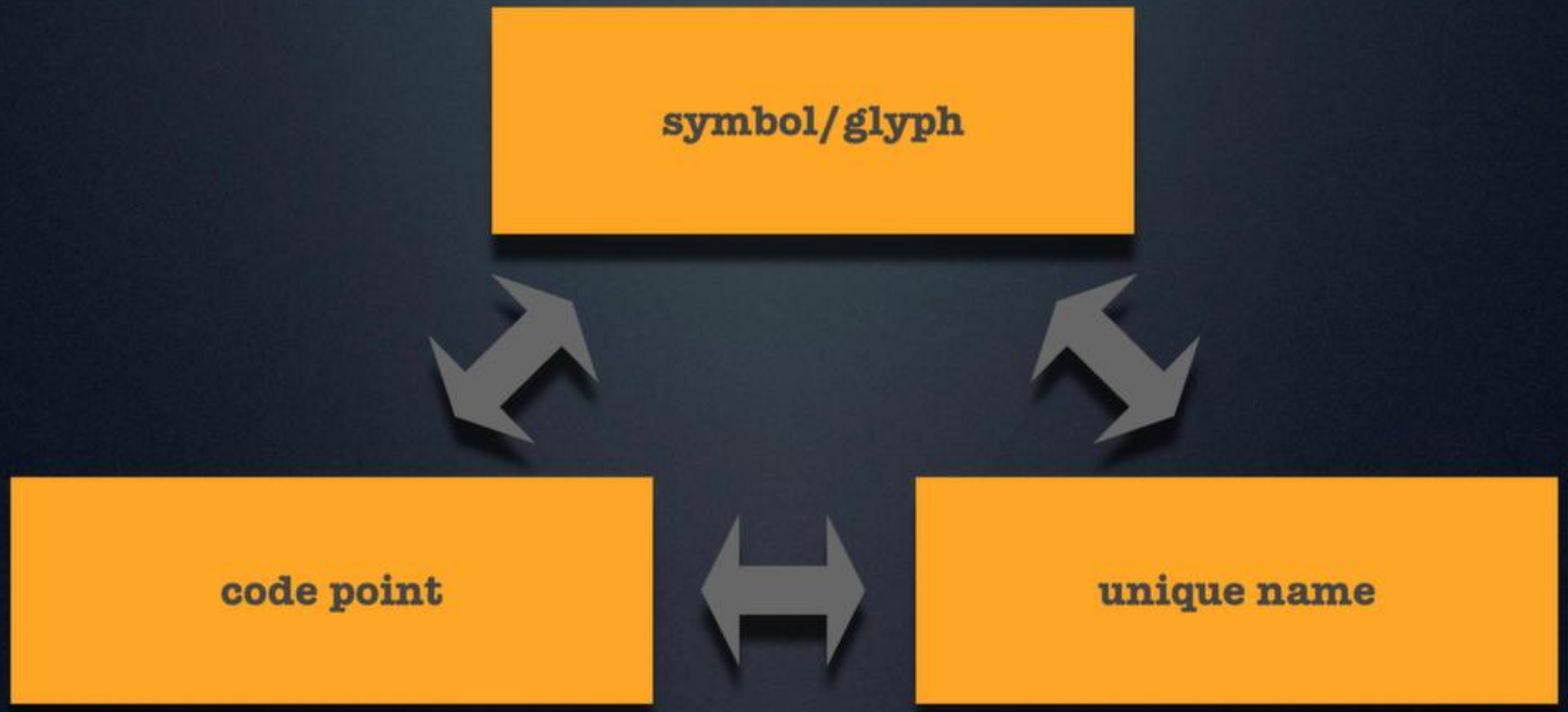
```
[1, 2, 3] <= [1, 2, 3]   // true
[1, 2, 3] >= [1, 2, 3]   // true
```

— @pwnall

undefined props on numbers

JavaScript has a Unicode problem

Unicode


symbol/glyph
code point
unique name

U+0041

A

LATIN CAPITAL LETTER A

U+0061

a

LATIN SMALL LETTER A

U+00A9

©

COPYRIGHT SIGN

U+2603

SNOWMAN

U+1F4A9

PILE OF POO

U+000000 ⊥ U+10FFFF

(0x10FFFF + 1) code points

∴

17 planes
(0xFFFF + 1) code points each

Unicode plane #1

U+0000 ⊥ U+FFFF

Basic Multilingual Plane

Unicode planes #2-17

U+010000 ⊥ U+10FFFF

supplementary planes
astral planes

JavaScript

Hexadecimal escape sequences

```
>> '\x41\x42\x43'
'ABC'
>> '\x61\x62\x63'
'abc'
```

can be used for U+0000 ⊥ U+00FF

Unicode escape sequences

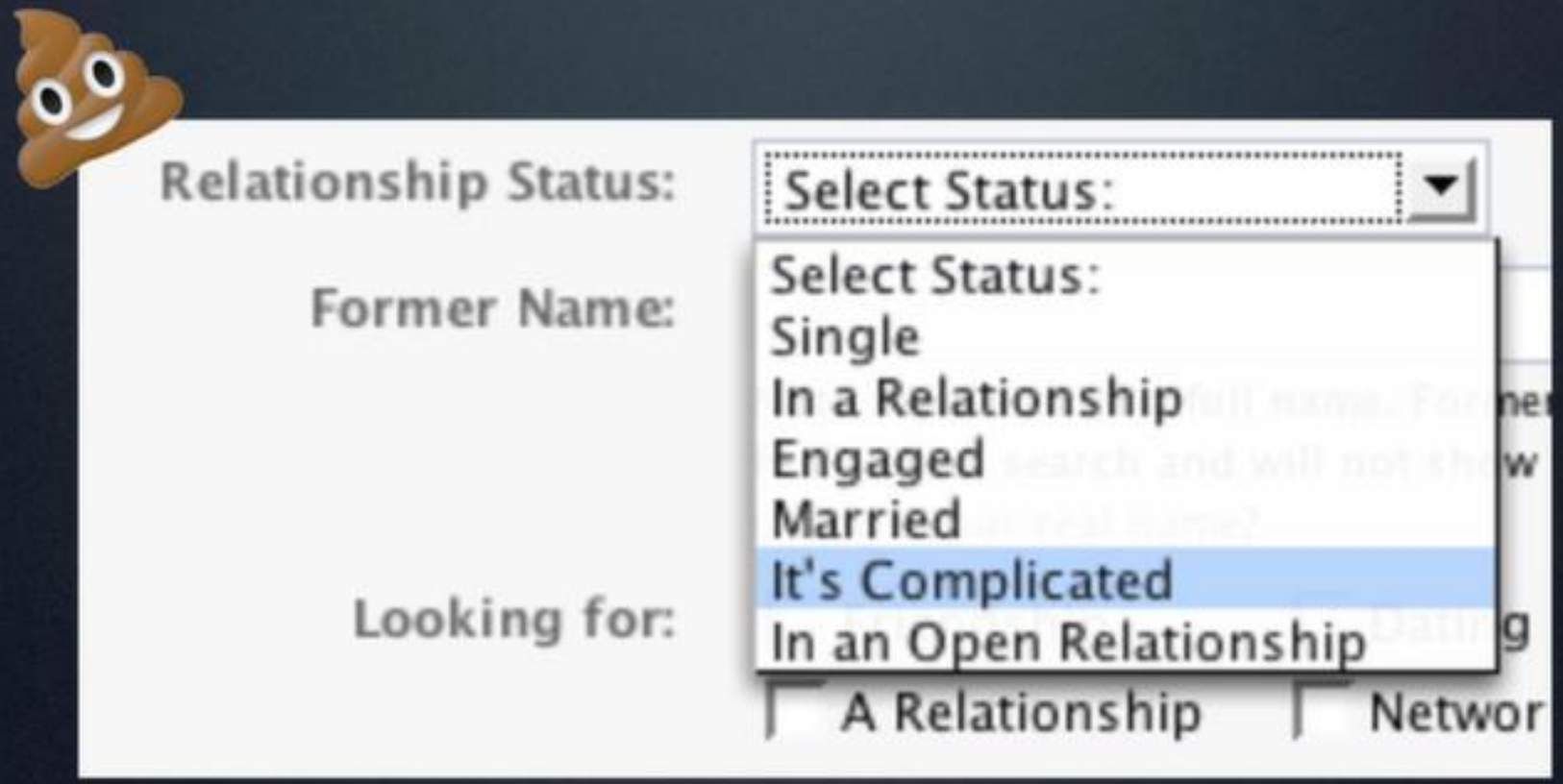
```
>> '\u0041\u0042\u0043'
'ABC'
>> 'I \u2661 JavaScript!'
'I ♡ JavaScript!'
```

can be used for U+0000 ⊥ U+FFFF

...what about astral code points?

...what about 💩?*

*...and other, equally important astral symbols


Relationship Status:
Select Status:
Former Name:
Select Status:
Single
In a Relationship
Engaged
Married
It's Complicated
In an Open Relationship
Looking for:
A Relationship
Networ

Unicode code point escapes

ES6

```
>> '\u{41}\u{42}\u{43}'
'ABC'
>> '\u{1F4A9}'
'💩'
```

can be used for U+000000 ⊥ U+10FFFF

Surrogate pairs

```
>> '\uD83D\uDCA9'
'💩' // U+1F4A9
```

can be used for U+010000 ⊥ U+10FFFF

Surrogate pairs

```
// for astral code points (> 0xFFFF)
function getSurrogates(codePoint) {
  var high = Math.floor((codePoint - 0x10000) / 0x400) + 0xD800;
  var low = (codePoint - 0x10000) % 0x400 + 0xDC00;
  return [ high, low ];
}

function getCodePoint(high, low) {
  var codePoint = (high - 0xD800) * 0x400 + low - 0xDC00 + 0x10000;
  return codePoint;
}

>> getSurrogates(0x1F4A9); // U+1F4A9 is 💩
[ 0xD83D, 0xDCA9 ]
>> getCodePoint(0xD83D, 0xDCA9);
0x1F4A9
```

http://mths.be/bed

JavaScript string length

```
>> 'A'.length // U+0041
1
>> 'A' == '\u0041'
true
>> 'B'.length // U+0042
1
>> 'B' == '\u0042'
true
```

String length ≠ char count

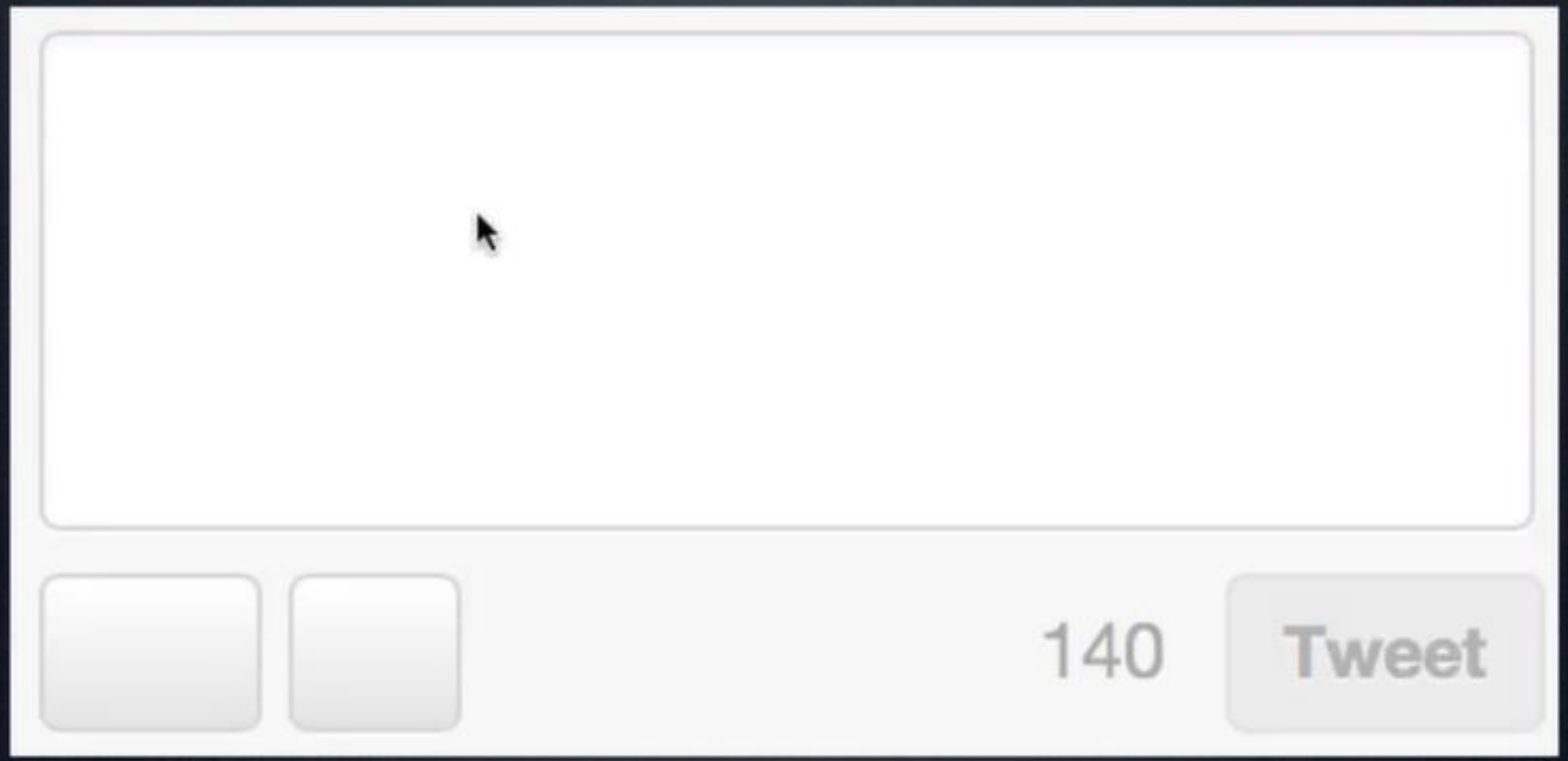
```
>> 'A'.length // U+1D400
2
>> 'A' == '\uD835\uDC00'
true
>> 'B'.length // U+1D401
2
>> 'B' == '\uD835\uDC01'
true
```

String length ≠ char count

```
>> '💩'.length // U+1F4A9
2
>> '💩' == '\uD83D\uDCA9'
true
```

insert obligatory
“number two” joke here

Real-world example



COUNTABLE.JS

Countable is a JavaScript function to add **live paragraph-, word- and character-counting** to an HTML element. Countable does not rely on any libraries and is very small in size.

Download on GitHub

Start entering some text here

Paragraphs: 0

Words: 0

Characters: 0

Characters (with spaces): 0

COUNTABLE.JS

Countable is a JavaScript function to add **live paragraph-, word- and character-counting** to an HTML element. Countable does not rely on any libraries and is very small in size.

Download on GitHub

Paragraphs: 1

Words: 1

Characters: 2

Characters (with spaces): 2

JS string character count

```
function countSymbols(string) {
  return punycode.ucs2.decode(string).length;
}

>> countSymbols('A') // U+0041
1
>> countSymbols('𝐀') // U+1D400
1
>> countSymbols('💩') // U+1F4A9
1
```

http://mths.be/punycode

JS escape sequences

JavaScript escapes

Here are some special characters: © ☃ 💩

☑ only escape non-ASCII and unprintable ASCII characters
☑ treat as JavaScript string body

permalink

```
'Here are some special characters: \xA9 \u2603 \uD83D\uDCA9 \u200B'
```

Learn all about JavaScript character escapes and do it manually, or just use this tool.

http://mths.be/bmf

If we're being pedantic...

```
// it's actually even more complicated:

>> 'mañana' == 'mañana'
false
```

If we're being pedantic...

```
// it's actually even more complicated:

>> 'mañana' == 'mañana'
false
>> 'ma\xF1ana' == 'man\u0303ana'
false
>> 'ma\xF1ana'.length
6
>> 'man\u0303ana'.length
7
```

Unicode normalization

```
function countSymbolsPedantically(string) {
  // Unicode Normalization, NFC form:
  var normalized = unorm.nfc(string);
  // Account for astral symbols / surrogates:
  return punycode.ucs2.decode(normalized).length;
}

>> countSymbolsPedantically('mañana') // U+00F1
6
>> countSymbolsPedantically('mañana') // U+006E + U+0303
6
```

http://mths.be/punycode & http://git.io/unorm

Unicode normalization

ES6

```
function countSymbolsPedantically(string) {
  // Unicode Normalization, NFC form:
  var normalized = string.normalize('NFC');
  // Account for astral symbols / surrogates:
  return punycode.ucs2.decode(normalized).length;
}

>> countSymbolsPedantically('mañana') // U+00F1
6
>> countSymbolsPedantically('mañana') // U+006E + U+0303
6
```

Reversing a string in JavaScript

```
// naive solution
function reverse(string) {
  return string.split('').reverse().join('');
}
```

Reversing a string in JavaScript

```
function reverse(string) {
  return string.split('').reverse().join(''); // naive solution
}

>> reverse('abc')
'cba'
```

Reversing a string in JavaScript

```
function reverse(string) {
  return string.split('').reverse().join(''); // naive solution
}

>> reverse('abc')
'cba'
>> reverse('mañana') // U+00F1
'anañam'
```

Reversing a string in JavaScript

```
function reverse(string) {
  return string.split('').reverse().join(''); // naive solution
}

>> reverse('abc')
'cba'
>> reverse('mañana') // U+00F1
'anañam'
>> reverse('mañana') // U+006E + U+0303
'añanam'
```

Reversing a string in JavaScript

```
function reverse(string) {
  return string.split('').reverse().join(''); // naive solution
}

>> reverse('abc')
'cba'
>> reverse('mañana') // U+00F1
'anañam'
>> reverse('mañana') // U+006E + U+0303
'anãnam'
>> reverse('💩') // U+1F4A9
'⍰⍰'
'\uDCA9\uD83D' // the surrogate pair for 💩, in the wrong order
```


"I put my thang down,
flip it, and reverse it"
– Missy 'Misdemeanor' Elliot, 2002

Reversing a string in JavaScript

```
// Using the Esrever library
var reverse = esrever.reverse;

>> reverse('abc')
'cba'
>> reverse('mañana') // U+00F1
'anañam'
>> reverse('mañana') // U+006E + U+0303
'anañam'
>> reverse('💩') // U+1F4A9
'💩'
```

http://mths.be/esrever

This affects other string methods, too.

String.fromCharCode()

```
>> String.fromCharCode(0x0041) // U+0041
'A' // U+0041
>> String.fromCharCode(0x1F4A9) // U+1F4A9
'[illegible]' // U+F4A9
```

only works as you'd expect for
U+0000 ⊥ U+FFFF

String.fromCharCode()

⊥ use surrogate pairs for astral symbols:

```
>> String.fromCharCode(0xD83D, 0xDCA9)
'💩' // U+1F4A9
```

String.fromCharCode()

⊥ use surrogate pairs for astral symbols:

```
>> String.fromCharCode(0xD83D, 0xDCA9)
'💩' // U+1F4A9
```

⊥ or just use Punycode.js:

```
>> punycode.ucs2.encode([ 0x1F4A9 ])
'💩' // U+1F4A9
```

String.fromCodePoint()

ES6

```
>> String.fromCodePoint(0x1F4A9)
```

💩

can be used for U+000000 ⊥ U+10FFFF

String#char{Code}At()

```
>> '💩'.charAt(0) // U+1F4A9
'\uD83D' // U+D83D
>> '💩'.charCodeAt(0)
0xD83D
```

String#codePointAt()

ES6

```
>> '💩'.codePointAt(0)
0x1F4A9
```

Iterate over all symbols in a string

```
function getSymbols(string) {
  var length = string.length;
  var index = -1;
  var output = [];
  var character;
  var charCode;
  while (++index < length) {
    character = string.charAt(index);
    charCode = character.charCodeAt(0);
    if (charCode >= 0xD800 && charCode <= 0xDBFF) {
      output.push(character + string.charAt(++index));
    } else {
      output.push(character);
    }
  }
  return output;
}

var symbols = getSymbols('💩');
symbols.forEach(function(symbol) {
  assert(symbol == '💩');
});
```

Iterate over all symbols in a string

ES6

```
for (let symbol of 💩 ') {
  assert(symbol == '💩');
}
```

More string madness

- String#substring
- String#slice
- ...anything that involves strings

Regular expressions

```
>> /foo.bar/.test('foo💩bar')
false
```

Match any Unicode symbol

Match any Unicode symbol

```
>> /^.$/.test('💩')
false // doesn't match line breaks, either
```

Match any Unicode symbol

```
>> /^.$/.test('💩')
false // doesn't match line breaks, either

>> /^[\s\S]$/.test('💩')
false // matches line breaks, but still doesn't match whole astral symbols
```

Match any Unicode symbol

```
>> /^.$/.test('💩')
false // doesn't match line breaks, either

>> /^[\s\S]$/.test('💩')
false // matches line breaks, but still doesn't match whole astral symbols

>> /^[\0-\uD7FF\uDC00-\uFFFF]|[\uD800-\uDBFF][\uDC00-
\uDFFF]|[\uD800-\uDBFF]$/.test('💩')
true // wtf
```

Create Unicode-aware regular expressions

```
>> regenerate.fromCodePointRange(0x0, 0x10FFFF)
'[\0-\uD7FF\uDC00-\uFFFF]|[\uD800-\uDBFF][\uDC00-\uDFFF]|[\uD800-
\uDBFF]'
```

http://mths.be/regenerate

Create Unicode-aware regular expressions

```
>> regenerate.fromCodePointRange(0x0, 0x10FFFF)
'[\0-\uD7FF\uDC00-\uFFFF]|[\uD800-\uDBFF][\uDC00-\uDFFF]|[\uD800-
\uDBFF]'

>> regenerate()
……    .addRange(0x000000, 0x10FFFF) // add all Unicode code points
……    .removeRange('A', 'z') // remove all symbols from `A` to `z`
```

http://mths.be/regenerate

Create Unicode-aware regular expressions

```
>> regenerate.fromCodePointRange(0x0, 0x10FFFF)
'[\0-\uD7FF\uDC00-\uFFFF]|[\uD800-\uDBFF][\uDC00-\uDFFF]|[\uD800-
\uDBFF]'

>> regenerate()
……    .addRange(0x000000, 0x10FFFF) // add all Unicode code points
……    .removeRange('A', 'z') // remove all symbols from `A` to `z`
……    .remove('💩') // remove U+1F4A9 PILE OF POO
……    .toString();
'[\0-\x1F\x21-\x40\x7B-\uD7FF\uDC00-\uFFFF]|[\uD800-\uDBFF][\uDC00-
\uDFFF]|[\uD800-\uDBFF]'
```

http://mths.be/regenerate

Regular expressions

ES6

```
>> /foo.bar/.test('foo💩bar')
false

>> /foo.bar/u.test('foo💩bar')
true
```

JavaScript has a Unicode problem

Thanks!

Questions? ⊥ @mathias

U+23F0 ALARM CLOCK

U+1F37A BEER MUG

U+1F37B CLINKING BEER MUGS