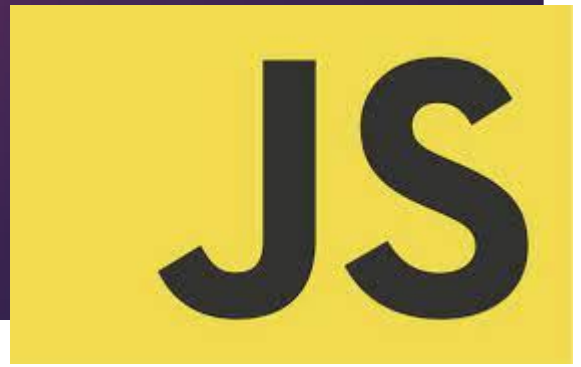
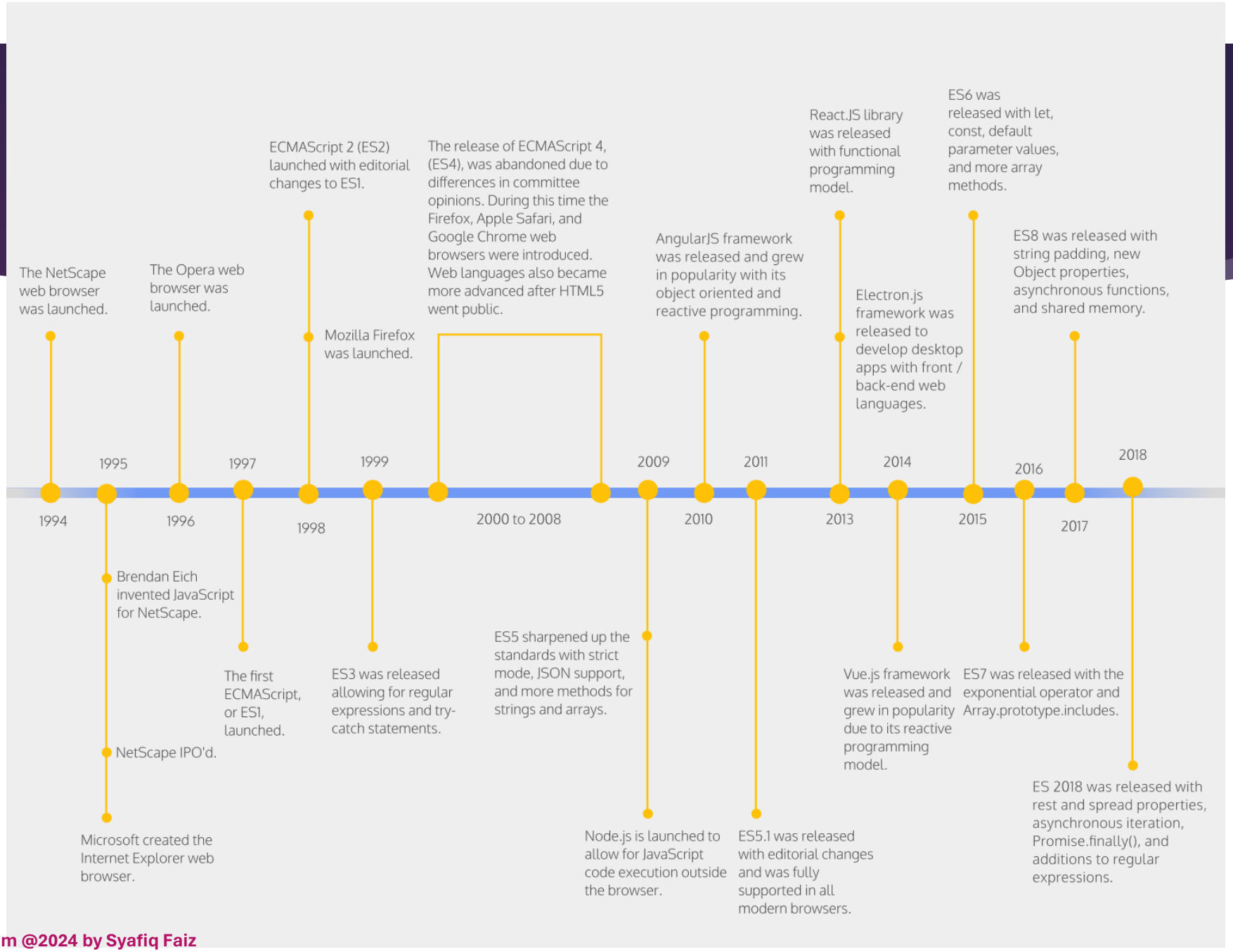
JAVASCRIPT

ASAS PENGATURACARAAN JAVASCRIPT



Asal Usul Javascript

- Dibangunkan pada 1995 oleh Breindan Eich
- Mula digunakan pada browser Netscape2
- Tak ada kaitan dengan JAVA melainkan nama
- ECMA(european computer manufacturer association) mengambil alih Pembangunan Javascript pada 1997
- 2009, kelahiran NODE js. Menjadikan pengunaan JS yg lebih meluas


The NetScape web browser was launched.
1994
1995
Brendan Eich invented JavaScript for NetScape.
NetScape IPO'd.
Microsoft created the Internet Explorer web browser.
The Opera web browser was launched.
1996
1997
The first ECMAScript, or ES1, launched.
ECMAScript 2 (ES2) launched with editorial changes to ES1.
1998
Mozilla Firefox was launched.
1999
ES3 was released allowing for regular expressions and try-catch statements.
The release of ECMAScript 4, (ES4), was abandoned due to differences in committee opinions. During this time the Firefox, Apple Safari, and Google Chrome web browsers were introduced. Web languages also became more advanced after HTML5 went public.
2000 to 2008
2009
ES5 sharpened up the standards with strict mode, JSON support, and more methods for strings and arrays.
Node.js is launched to allow for JavaScript code execution outside the browser.
AngularJS framework was released and grew in popularity with its object oriented and reactive programming.
2010
2011
ES5.1 was released with editorial changes and was fully supported in all modern browsers.
React.JS library was released with functional programming model.
2013
Electron.js framework was released to develop desktop apps with front / back-end web languages.
2014
Vue.js framework was released and grew in popularity due to its reactive programming model.
ES6 was released with let, const, default parameter values, and more array methods.
2015
2016
ES7 was released with the exponential operator and Array.prototype.includes.
ES8 was released with string padding, new Object properties, asynchronous functions, and shared memory.
2017
2018
ES 2018 was released with rest and spread properties, asynchronous iteration, Promise.finally(), and additions to regular expressions.

Atwood's Law

"Any application that can be written in JavaScript, will eventually be written in JavaScript."

JEFF ATWOOD

FOUNDER OF STACK OVERFLOW

4 Perkara Wajib Tahu

- Variable
- Loops
- Control flow
- Function
- OOP

Variable

- Tempat simpan maklumat sementara
- var, let, const
- Declaration and assignment

```
4
5   var maklumat = 'abc'
6   var maklumat2 = 'def'
7
```

Nilai boleh diubah

```
9
10  let result = maklumat + maklumat2
11  console.log(result) // abcdef
12  result = maklumat + ' ' + maklumat2
13  console.log(result) // abc def
14
```

Nilai boleh diubah

```
15
16  const maklumat3 = 'ghi'
17
```

Nilai tak boleh diubah

Javascript Data Types

- Primitive data types
  - String
  - Number
  - Boolean
  - Null
  - Undefined
- Object data types
- Special data types
  - Function
  - Array

```
3   // string
4   const perkataan = 'ini adalan jenis string'
5
6   // number
7   const nombor = 1234
8   const nomborPi = 3.14
9   const nomborNegatif = -1234
10
11  // boolean
12  const betulAtauSalah = true
13  const salah = false
14
15  // null
16  const kosong = null
17  console.log(kosong) // null;
18
19  // undefined
20  let belumDitentukan;
21  console.log(belumDitentukan) // undefined
22
23  // object
24  const objek = {
25    nama: 'John',
26    umur: 30
27  }
28
29  // array
30  const senarai = [1, 2, 3, 4, 5]
31
32  // function
33  const fungsi = () => {
34    return 'ini adalah fungsi'
35  }
36  console.log(fungsi()) // ini adalah fungsi
37
```

Expression

- Gabungan variable dan operation yang di evaluate

```
3   const a = 10
4   const b = 20
5   const c = 5 + a + b
6   console.log(c); // 35
7
8
9   const nama = 'faiz'
10  const umur = 18
11  const greeting = 'hai ' + nama + 'umur: ' + umur // hai faiz umur: 18
12
13  const umurSekarang = 18
14  const hadUmurTua = 20
15  const masihMuda = umurSekarang < hadUmurTua
16  console.log(masihMuda); // true
17
18  const fungsiGreeting = (nama, umur) => {
19    return 'hai ' + nama + ' umur: ' + umur
20  }
21  console.log(fungsiGreeting('faiz', 18)); // hai faiz umur: 18
22  console.log(fungsiGreeting('ali', 20)); // hai ali umur: 20
23
24
```

Operators

- Arithmetic
- Comparison
- Logical
- Unary
- Tenary
- String

Arithmetic Operators

```
3   // addition
4   var a = 10;
5   var b = 20;
6   var c = a + b;
7   console.log(c); // 30
8
9   // subtraction
10  var d = 20;
11  var e = 10;
12  var f = d - e;
13  console.log(f); // 10
14
15  // multiplication
16  var g = 10;
17  var h = 20;
18  var i = g * h;
19  console.log(i); // 200
20
21  // division
22  var j = 20;
23  var k = 10;
24  var l = j / k;
25  console.log(l); // 2
```

```
27  // modulus
28  var m = 20;
29  var n = 10;
30  var o = m % n;
31  console.log(o); // 0
32
33  // increment
34  var p = 10;
35  p++;
36  console.log(p); // 11
37
38  // decrement
39  var q = 20;
40  q--;
41  console.log(q); // 19
42
```

Comparison Operators

```
50  // equal to
51  var r = 10;
52  var s = 20;
53  console.log(r == s); // false
54
55  // not equal to
56  var t = 10;
57  var u = 20;
58  console.log(t != u); // true
59
60  // greater than
61  var v = 10;
62  var w = 20;
63  console.log(v > w); // false
64
```

```
65  // less than
66  var x = 10;
67  var y = 20;
68  console.log(x < y); // true
69
70  // greater than or equal to
71  var z = 10;
72  var aa = 20;
73  console.log(z >= aa); // false
74
75  // less than or equal to
76  var ab = 10;
77  var ac = 20;
78  console.log(ab <= ac); // true
79
```

Logical Operators

```
82  // and
83  var ad = 10;
84  var ae = 20;
85  console.log(ad < 20 && ae > 10); // true
86
87  // or
88  var af = 10;
89  var ag = 20;
90  console.log(af < 20 || ag > 10); // true
91
92  // not
93  var ah = 10;
94  var ai = 20;
95  console.log(!(ah < 20 && ai > 10)); // false
96
```

Ternary Operator

```
118  // conditional
119  var ao = 10;
120  var ap = 20;
121  var aq = ao > ap ? "true" : "false";
122  console.log(aq); // false
123
```

String Operator

```
100   // string concatenation
101   var aj = "Hello";
102   var ak = "World";
103   var al = aj + ak;
104   console.log(al); // HelloWorld
105
```

Unary Operator

```
109  // typeof
110  var am = 10;
111  console.log(typeof am); // number
112
113  // negation
114  var an = true
115  console.log(!an); // false
116  var ao = true;
117  console.log(!!ao); // true
```

Array

- Kita simpan senarai data

```
const senaraiNombor = [1,2,3,4,5,6]
const abu = 'abu'
const senaraNama = ['ali', 'ahmad', abu]
console.log(senaraiNama) // ['ali', 'ahmad', 'abu']
const senaraiCampur = ['ali', true, 18, senaraiNama, ()=>{console.log('hello world')}]
console.log(senaraiCampur[0]) // ali
```

```
const senarai2D = [
  [5, 3, 0, 0, 7, 0, 0, 0, 0],
  [6, 0, 0, 1, 9, 5, 0, 0, 0],
  [0, 9, 8, 0, 0, 0, 0, 6, 0],
  [8, 0, 0, 0, 6, 0, 0, 0, 3],
  [4, 0, 0, 8, 0, 3, 0, 0, 1],
  [7, 0, 0, 0, 2, 0, 0, 0, 6],
  [0, 6, 0, 0, 0, 0, 2, 8, 0],
  [0, 0, 0, 4, 1, 9, 0, 0, 5],
  [0, 0, 0, 0, 8, 0, 0, 7, 9]
]
```

Array methods(sebahagian)

```
150  const senaraiBuah = ['apple', 'banana', 'cherry']
151  senaraiBuah.push('date')
152  console.log(senaraiBuah) // ['apple', 'banana', 'cherry', 'date']
153
```

Push
- Menambah data kedalam array

```
154  const senaraiKereta = ['proton', 'perodua', 'honda']
155  senaraiKereta.pop()
156  console.log(senaraiKereta) // ['proton', 'perodua']
157
```

Pop
- Membuang data terakhir dalam array

```
158  const senaraiNombor = [1,2,3,4,5,6]
159  senaraiNombor.shift()
160  console.log(senaraiNombor) // [2,3,4,5,6]
```

Shift
- Membuang data terawal dalam array

```
162  const senaraiNombor2 = [1,2,3,4,5,6]
163  senaraiNombor2.unshift(0)
164  console.log(senaraiNombor2) // [0,1,2,3,4,5,6]
165
```

Unshift
- Menambah data pada permulaan array

Loop

- Loop adalah satu operasi dalam array yang membolehkan kita mengakses setiap data dalam array secara satu persatu mengikut urutan
  - for
  - while
  - forEach
  - map
  - filter

```
168  const senaraiBuah = ['apple', 'banana', 'cherry']
169  senaraiBuah.forEach((buah, index) => {
170    console.log(buah, index)
171  }) // apple 0, banana 1, cherry 2
172
173  const senaraiNombor = [1,2,3,4,5,6]
174  const hasil = senaraiNombor.map(nombor => nombor * 2)
175  console.log(hasil) // [2,4,6,8,10,12]
176
177  const senaraiNombor2 = [1,2,3,4,5,6]
178  const hasil2 = senaraiNombor2.filter(nombor => nombor > 3)
179  console.log(hasil2) // [4,5,6]
180
181  const senaraiNombor3 = [1,2,3,4,5,6]
182  for (let i = 0; i < senaraiNombor3.length; i++) {
183    console.log(senaraiNombor3[i])
184  } // 1, 2, 3, 4, 5
185
```

Terima Kasih. Soalan?