ECMASCRIPT

SOPORTE ACTUAL, DESARROLLOS CROSS-BROWSER Y RENDIMIENTO

By Fran Linde Blázquez

¿QUIÉN SOY?

Fran Linde Blázquez

Ingeniero de Software

Desarrollador Webs/Apps desde 2013

Desarrollador Front-End en Minsait desde 2016

¿QUÉ VAMOS A VER?

1. ECMAScript
2. Algunas novedades de ES6/ES7/ES8
3. ECMAScript & Cross Browser
4. Herramientas de Benchmarking
5. Comparativas de código
6. Conclusiones

1. ECMASCRIPT

¿Qué es ECMAScript?

Estándar de ECMA International (algo así como ISO)

TC39: comité que lo regula

Formado por Mozilla, Apple, Google, Facebook...

Siguen un proceso

Decimos que es el estándar de JavaScript

También de ActionScript (Flash) y JScript (Netscape)

¿ES6? ¿ES2015?

Edition	Official name	Date published
ES9	ES2018	June 2018
ES8	ES2017	June 2017
ES7	ES2016	June 2016
ES6	ES2015	June 2015
ES5.1	ES5.1	June 2011
ES5	ES5	December 2009
ES4	ES4	Abandoned
ES3	ES3	December 1999
ES2	ES2	June 1998
ES1	ES1	June 1997

ESNext es la próxima versión: ES2019 (ES10)

2.1 ALGUNAS NOVEDADES DE ES6

CLASS

Ejemplo:

```
class Poligono {
  constructor(height, width) {
    this.height = height;
    this.width = width;
  }

  get area() {
    return this.calcArea();
  }

  calcArea() {
    return this.height * this.width;
  }
}
```

TEMPLATE STRINGS / LITERALS

Cadenas multi-línea:

```
console.log(`text line 1
text line 2
and the last text line`);
```

Interpolación de valores:

```
let a = 5;
let b = 10;
console.log(`Fifteen is ${a + b}.`);
```

DESTRUCTURING

Ejemplo:

```
let miArray = ["uno", "dos", "tres"];

// sin destructuring
let var1 = miArray[0];
let var2 = miArray[1];
let var3 = miArray[2];

// con destructuring
let [var1, var2, tres] = miArray;
```

PARÁMETROS POR DEFECTO

Ejemplo:

```
function multiplicar(a, b = 1) {
  return a*b;
}

multiplicar(5);
```

SPREAD

Ejemplo:

```
var partes = ['hombros', 'rodillas'];
var persona = ['cabeza', ...partes, 'pies'];
```

Ejemplo 2:

```
let arr1 = [0, 1, 2];
let arr2 = [3, 4, 5];
arr1.push(...arr2);
```

ARROW FUNCTIONS

Ejemplo:

```
let operacion = (x, y) => {
  let aux = x + y;
  aux = aux*8.45;
  return aux;
}
```

Ejemplo 2:

```
let duplica = (x) => 2*x;
```

Ejemplo 3:

```
setTimeout(() => console.log('Hola'), 1000));
```

OBJECT.ASSIGN

Ejemplo:

```
var o1 = { a: 1 };
var o2 = { b: 2 };
var o3 = { c: 3 };

var obj = Object.assign(o1, o2, o3);
```

Ejemplo 2:

```
var o1 = { a: 1, b: 1, c: 1 };
var o2 = { b: 2, c: 2 };
var o3 = { c: 3 };

var obj = Object.assign({}, o1, o2, o3);
```

PROMISE

Ejemplo:

```
var promise1 = new Promise(function(resolve, reject) {
  setTimeout(function() {
    resolve('foo');
  }, 300);
});

promise1.then(function(value) {
  console.log(value);
  // expected output: "foo"
});
```

2.2 ALGUNAS NOVEDADES DE ES7

ARRAY.PROTOTYPE.INCLUDES()

Ejemplo:

```
let numbers = [1, 2, 3, 4];

if(numbers.includes(2)) {
  console.log('Array contains value');
}
```

POTENCIAS

Ejemplo:

```
let base = 3;
let exponent = 4;
let result = base**exponent;
console.log(result); //81
```

2.3 ALGUNAS NOVEDADES DE ES8

ASYNC / AWAIT

Ejemplo:

```
async function fetchData(url) {
  try {
   let request = await fetch(url);
   let text = await request.text();
   return JSON.parse(text);
  }
  catch (err) {
    console.log(`Error: ${err.stack}`);
  }
}
```

TRAILING COMMAS

Ejemplo:

```
const trailCommaFn = function(
  param1,
  param2,
  param3,
  param4,
) {
  // do something in function body
  console.log(param1 + param2);
}
```

OBJECT.VALUES / OBJECT.ENTRIES

Ejemplo:

```
Object.entries({ foo: 1, bar: 2 })
// [['foo', 1], ['bar', 2]]
```

Ejemplo 2:

```
Object.values({ foo: 1, bar: 2 })
// [1, 2]
```

3. ECMASCRIPT & CROSS BROWSER

Algunos problemillas:

Llega poco a poco a los navegadores

Estado actual en formato Tabla

¿Cómo lidiamos con esto?

Babel Web Oficial

Transpilador. Reescribe código JavaScript usando estándares anteriores.

Paso 1: Transpilación

Paso 2: Polyfills (basado en core-js)

Otro Polyfill interesante: polyfill.io

¿Qué es transpilable y qué no?

Feature	Transpiled	Polyfilled	Nope
class	✓		
const and let	✓		
Object rest/spread	✓		
Arrow functions () => {}	✓		
Map		✓	
Promise		✓	
Fetch		✓	
String.prototype.padEnd()		✓	
Array.prototype.includes()		✓	
URLSearchParams		✓	
requestIdleCallback		✓	
Service Worker			✓
Push API			✓
Bluetooth API			✓
Proxy			✓

Una buena regla:

Si es nueva sintáxis: transpilar

Si es nuevo objecto/método: poylfill

Si es nueva API: estás muerto :)

3. HERRAMIENTAS DE BENCHMARKING

Chrome DevTools. Chrome only ¯_(ツ)_/¯

Librerías como benchmarkjs usada en jsPerf.com

performance.now()

```
var t0 = performance.now();
hacerAlgo();
var t1 = performance.now();
console.log("hacerAlgo ha tardado " + (t1 - t0) + " ms.");
```

console.time() y console.timeEnd()

```
console.time('test');
let acum = 0;
for(let i=0; i<9999999; i++){
  acum++;
}
console.timeEnd('test');
```

4. COMPARATIVAS DE CÓDIGO

Se enviará enlace al repositorio

5. CONCLUSIONES

CONCLUSIONES:

No todos los navegadores implementan todo ES

Cada navegador usa un motor JS diferente

Babel transpila SIEMPRE lo que no es 'polyficable'

Los polyfills son func. grandes pero eficientes

CONSEJOS:

Debemos preguntar SIEMPRE navegadores soportados

Usa JS moderno y aplica transpilador/polyfills

Cuidado con los polyfills que elegimos

¿DUDAS?

GRACIAS :)