ORT אורט ARGENTINA

Educando para la vida

Programación en Nuevas Tecnologías 2

Repaso, Javascript

Javascript es interpretado

- Cada navegador tiene su propio motor de Javascript, los cuales interpretan el código o usan algun tipo de compilación provisionada
  - V8: Chrome and Node.js
  - SpiderMonkey: Firefox (hasta la version 25)
  - JavaScriptCore: Safari
  - Chakra: Microsoft Edge/IE
- Cada intérprete implementa el estándar EcmaScript. Sin embargo algunos lo implementan en mayor o menos medida.

Sintaxis

```
const nombre = "daniel";
const apellido = guzman;
const arr = ['Instituto ORT', 42, true, function() {
console.log('Hola mundo')

}];

// Hola soy un comentario

for (let i = 0; i < arr.length; i++) {

console.log(arr[i]);

}
```

Tipos de datos

- Tipeado Dinámico.
- Tipeado Primitivo:
  - undefined
  - bool
  - number
  - string
  - (symbol)
- Objetos.

Coerción de datos

- Explícito vs. Implícito
  - `const x = 17;`
  - `const explicito = String(x); // explicito === "17"`
  - `const implicito = x + ""; // implicito === "17"`
- == vs. ===
  - == Castea el tipo de dato para comparación
  - === Require tipo de datos iguales para comparación

Coerción de datos

Para más referencia de casteo de datos y como es la coerción, pueden visualizar este github page:

- https://dorey.github.io/JavaScript-Equality-Table/

Coerción de datos.

- Valores falsos
  - undefined
  - null
  - false
  - +0, -0, NaN
  - ""
- Valores positivos
  - {}
  - []
  - etc

Objetos, Arrays, Funciones, Objetos

- En Javascript, todo lo que no sea tipos primario de datos representa un **Objeto**

- Prototype. Herencias de objetos.

Datos primitivos vs Objetos

- Tipos de datos primitivos inmutables.

- Objetos. Son mutables y almacenados por referencia.

Herencias Prototype

- Tipos de datos No-Primitivos, tienen algunas propiedades y métodos asociados:
  - `Array.prototype.push()`
  - `String.prototype.toUpperCase()`
- Cada objeto almacena una referencia a su prototipo base.
- Las propiedades y/o métodos definidos más cerca de la instancia prevalecen sobre el prototipo.

Herencias Prototype

- La mayoría de los tipos de datos primarios pueden ser definidos por objetos.
  - String()
  - Number()
  - Boolean()
  - Object()
  - (Symbol())

Herencias Prototype

- Javascript automáticamente encapsula valores primitivos, así se puede tener acceso a los métodos.

```
42.toString() // Errors

const x = 42;

x.toString() // "42"

x.__proto__ // [Number: 0]

x instanceof Number // false
```

Scope

- Duración de Variables
  - Lexical scoping (var): Desde donde es declarado hasta que la función o método termine.
  - Block scoping (const, let): Declaraciones dentro de bloques de control. Por ejemplo: for, if, entre otros. Se encuentra disponible hasta que se alcance el próximo *}*
- Hoisting
  - Aplica para definición de funciones y variables. Mecanismo que mueve la declaración de funciones y variables al inicio de su ámbito. Así siempre estará disponible

El motor de Javascript

- Antes de ejecutar el código, el motor lee el archivo entero y lanza una error de sintaxis si se encuentra uno.
  - Cualquier función definida será almacenada en memoria.
  - La inicialización de variables no será ejecutada, pero los nombres de variables dentro del scope global serán declaradas y habilitadas en el entorno.

Objeto Global

- Todas las variables y funciones definidas son en realidad parametros y metodos pertenecientes al objeto global.
  - Navegadores: El objeto global es `window`
  - Nodejs: El objeto global es `global`