Angel Jhoan Diaz Vanegas - 1152242 Juan David Afanador Verjel - 1152247 Andres Felipe Beltran Assaf- 1152262

/JavaScript

Ingeniería de Sistemas
Programación Orientada
a Objetos

INDEX.HTML

/Historia de JavaScript

Start

1. JavaScript se introdujo en 1995 como una forma
2. de agregar programas a páginas web en el navegador
3. Netscape Navigator.
4. //
5. A principios de los años 90, la mayoría de
6. usuarios que se conectaban a Internet lo hacían
7. con módems a una **velocidad máxima de 28.8 kbps**.
8. //
9. JavaScript fue desarrollado originalmente por
10. Brendan Eich de Netscape con el nombre de *Mocha*.

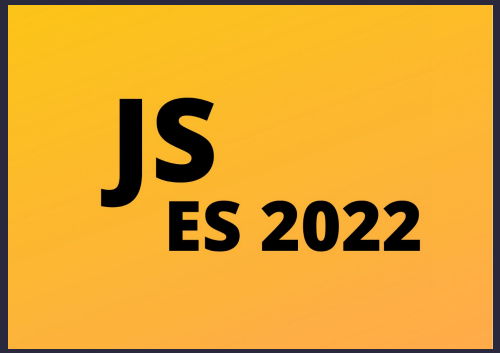
```
<html>
  <head>
    <div>
          <div>

            <form method="post" action="#" id="formvalue" onkeyup="
            drawChart()" />

            </form>
          </div>
    </div>



    <script type="text/javascript" src="https://www.google.com/jsapi"></
      script>
    <script type="text/javascript">

    var bid = 43;
    var ask = 21;


      google.load("visualization", "1", {packages:["corechart"]});
      google.setOnLoadCallback(drawChart);
      function drawChart() {
        var data = google.visualization.arrayToDataTable([
          ['Price', 'Quantity'],
          ['Value #1', bid],
          ['Value #2', ask],
        ]);
```

//Fin de historia JavaScript

/Release Actual

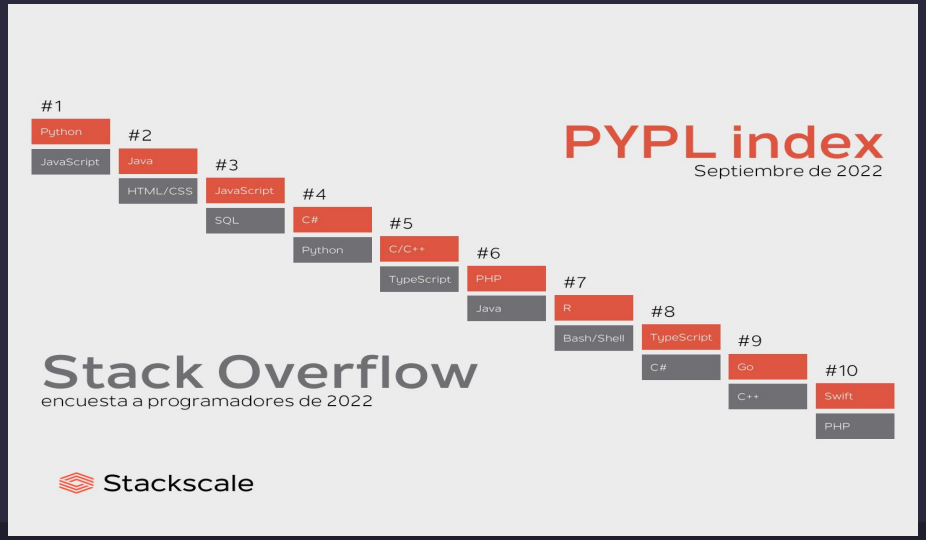
```
Start
```

1. La última actualización de JavaScript salió en julio de 2022.
2. //
3. Su nombre de desarrollo es ES2022.
4. //
5. Es la decimotercera edición desde que JavaScript salió, en 1997.
6. //




```
//Fin de releaseActual
```

<Ranking>



PopularitY of Programming Language Index

INDEX.HTML

/Utilidad

1. Páginas web más interactivas ya que permite los navegadores responder a la interacción de los usuarios y cambiar la distribución del contenido en la página web.

Usos especificos:

1. Interactividad de interfaz o front-end: el desarrollo web mejora por el aumento de la interactividad y funciones
2. Juegos de navegador: los navegadores web actuales han cambiado mucho; los desarrolladores pueden crear juegos robustos que funcionan en ellos.
3. Desarrollo web dorsal o back-end: el desarrollo web se ha transformado tanto, que JavaScript puede utilizarse para gestionar el back-end de sitios y aplicaciones.

/Clases y objetos

Como crear un objeto persona

```
//clase, instancia de la clase, objeto , propiedades publicas y privadas(cedula)
function persona(nombre,edad){                          //
  this.nombre=nombre;                                     //     este
  this.edad=edad;                                           //   seria nuestro
  let cedula=12323; //private en este caso                    //  constructor

  this.getCedula= function(){
      return cedula;
  }
    this.identificarse= function(){
      console.log('hola soy ' + nombre + ' y tengo '  + edad + ' años' +  ' mi
          cedula es ' + this.getCedula()) //this haciedno referencia que ese objeto
  }
}
/*persona.prototype.identificarse = function(){

    console.log('hola soy ' + this.nombre + '  tengo '  + this.edad + ' años' +  ' y
       mi cedula es ' + this.getCedula())
}*/
let objetoPersona= new persona("felipe Beltran",18); //instancia de la clase persona
let objetoPersona2= new persona("milton Vera",30); //instancia de la clase persona
objetoPersona.identificarse(); // imprime la persona
objetoPersona2.identificarse();
```

```
node /tmp/l5CvkhVEfP.js
hola soy felipe Beltran y tengo 18 años mi cedula es 12323
hola soy milton Vera y tengo 30 años mi cedula es 12323
```

Usando prototype o de una forma muy nativa

/Clases y objetos

```
1
2  class persona{
3
4      constructor(nombre,edad){
5          this.nombre=nombre;
6          this.edad=edad;
7          let cedula; //privada
8      }
9      set numeroDeCedula(cedula){
10         this.cedula= cedula;
11     }
12     get obtenerNumeroDeCedula(){
13         return this.cedula;
14     }
15
16     identificarse(){
17         console.log('hola soy ' + this.nombre + ' y tengo '  + this.edad + '
               años' +  ' mi cedula es ' + this.cedula); }
18 }
19
20 let objetoPersona = new persona("Felipe Beltran",18);
21 objetoPersona.numeroDeCedula= "109035281";
22 objetoPersona.identificarse();
```

```
node /tmp/kt4PoMbKSe.js
hola soy Felipe Beltran y tengo 18 años mi cedula es 109035281
```

INDEX.HTML

/AGREGACIÓN

Un objeto hace parte de otro, pero éste puede existir sin necesidad del objeto contenedor.

```
class Persona {
    constructor(nombre, cedula, fechaDeNacimiento){
        this.nombre = nombre;
        this.cedula = new Cedula(cedula)
        this.fechaDeNacimiento = new Fecha(fechaDeNacimiento);
    }
}
```

El objeto Fecha puede ser usado por otros objetos diferentes a Persona, y en caso de eliminar Persona, Fecha podría seguir existiendo.

/COMPOSICIÓN

Un objeto hace parte de otro, pero éste no puede existir sin su objeto contenedor.

```
class Persona {
    constructor(nombre, cedula, fechaDeNacimiento){
        this.nombre = nombre;
        this.cedula = new Cedula(cedula)
        this.fechaDeNacimiento = new Fecha(fechaDeNacimiento);
    }
}
```

El objeto Cedula no puede ser usado por otros objetos diferentes a Persona, y en caso de eliminar Persona, Cedula no podría existir.

/HERENCIA

Un objeto hereda los métodos y atributos de otro

```
class Empleado extends Persona {
    constructor(nombre, cedula, fechaDeNacimiento, salario){
        super(nombre, cedula, fechaDeNacimiento);
        this.salario = salario;
    }
}
```

El objeto Empleado hereda las propiedades de Persona, y se define el atributo salario, el cual es exclusivo de éste.

/GUI

/Web

HTML y CSS

/Móvil

React Native

/CLI

Node Package Manager

/Desktop

Electron.js

/EJEMPLO

INDEX.HTML

<GRACIAS!>