UD5 - ESQUEMA VISUAL – Mapa Conceptual Tema 5: Programación Orientada a Objetos en JavaScript

🔵 1. Introducción

- En JS **todo es un objeto** (arrays, funciones, fechas...).
- Un **objeto** tiene:
  - **Propiedades** (atributos)
  - **Métodos** (comportamientos)
- La **POO** permite modelar entidades del mundo real.
- JS está basado en **prototipos**, no en clases tradicionales.

🔵 2. JSON

✔️ ¿Qué es JSON?

- Formato de intercambio de datos (texto).
- Ligero, legible, estándar en web.
- No es un lenguaje, es una notación.

✔️ Estructuras JSON:

- **Objeto:** `{ clave: valor }`
- **Array:** `[ valor1, valor2 ]`

✔️ Tipos permitidos:

- string
- number
- boolean

- null
- array
- objeto

✔️ Reglas:

- Claves y strings → **comillas dobles**
- Sin comentarios
- Sin coma final

✔️ Conversión desde/hacia JS:

- `JSON.parse()` → cadena → objeto JS
- `JSON.stringify()` → objeto JS → cadena JSON

🔵 3. Gestión de Objetos en JS

- Acceso:
  - `obj.prop`
  - `obj["prop"]`
- Ver tipo:
  - `typeof objeto`
- Comprobar prototipo:
  - `obj instanceof Tipo`
- Crear objeto:
  - `new Object()`
  - `{ ... }` (notación JSON)
- Métodos dentro del objeto.
- `this`:
  - Referencia al objeto actual.
  - Depende del contexto.

🔵 4. Constructores

- Funciones especiales para crear objetos.
- Se usan con `new`.
- Inicializan propiedades.
- Permiten definir métodos propios.
- En clases: `constructor()`

🔵 5. Herencia y Polimorfismo

✔️ Herencia

- Una clase **hereda** propiedades/métodos de otra.
- Palabra clave: `extends`
- Llamar al constructor padre: `super()`

✔️ Polimorfismo

- Métodos con el mismo nombre → comportamientos distintos.
- Depende de la clase hija.

🔵 6. Recorrer un Objeto

- `for...in`
- `for...of` (con Object.values)
- Evita recorrer prototipos con:
  - `Object.getOwnPropertyNames(obj)`

🔵 7. Borrado de Propiedades

```
delete obj.propiedad;
```

🔵 8. Prototipos

- Todos los objetos tienen un **prototype**.
- JS usa herencia prototípica.
- Se puede extender el prototipo dinámicamente:
  - `Constructor.prototype.nuevoMetodo = function() {}`

🔵 9. Objetos Predefinidos

🔷 String

- Métodos: `length`, `charAt`, `indexOf`, `slice`, `trim`, `toUpperCase`, etc.
- Métodos estáticos: `String.fromCharCode()`, etc.

🔷 Date

- Representa un momento en tiempo.
- Constructor flexible:
  - `new Date()`
  - `new Date(año, mes, día...)`
- Métodos: `getFullYear`, `getMonth`, `getTime`, etc.

🔷 Math

- No se instancia: `Math.METHOD()`
- Constantes: `Math.PI`
- Métodos: `sqrt`, `pow`, `floor`, `ceil`, `random`, etc.

🔷 Boolean

- No confundir objeto Boolean con valor booleano.
- Consejo: **no usar** `new Boolean()`.

🔷 Expresiones Regulares

- Sintaxis:
  - Literal: `/patron/flags`
  - Constructor: `new RegExp("patron")`

- Métodos:
  - `test()` → true/false
  - `exec()` → más información
- Modificadores:
  - `i`, `^`, `$`, `.`, `[]`, `[^]`, `()`, `|`, `+`, , `{}`...
- Modificadores abreviados:
  - `\d`, `\w`, `\s`, `\D`, etc.