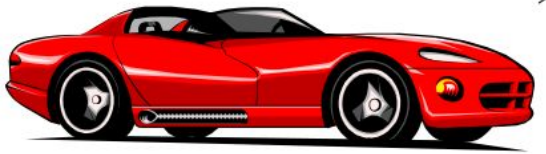
Objetos

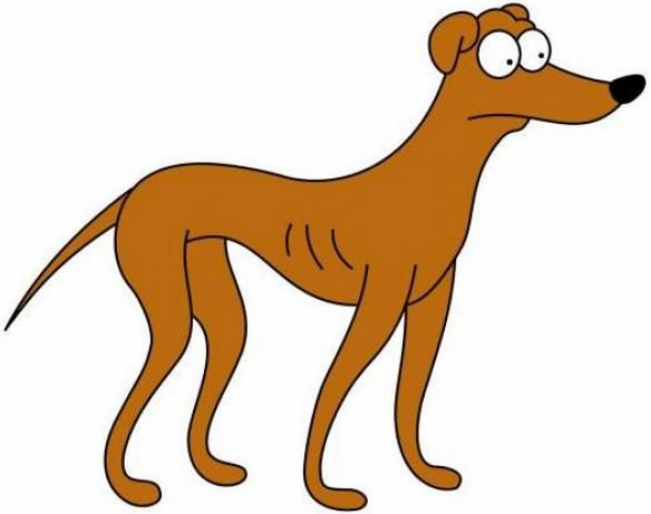
dev.f

desarrollamos(personas);

JavaScript está diseñado en un paradigma basado en objetos. Un objeto es una colección de atributos y métodos



- Atributos:
  - color
  - velocidad
  - ruedas
  - motor
- Métodos:
  - arranca()
  - frena()
  - dobla()



Metodos.

Atributos.

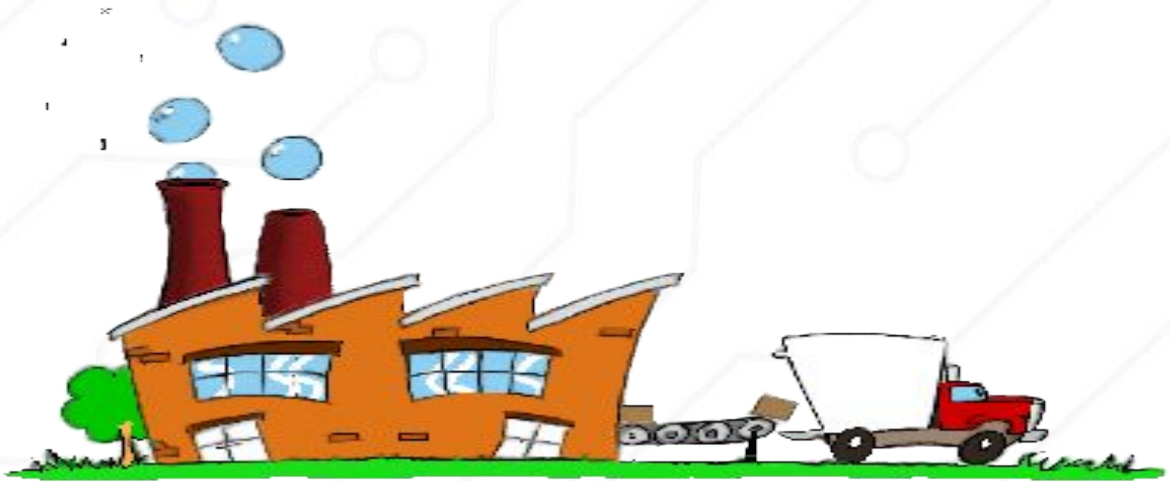
Todo como un objeto

En JavaScript, casi todo es un objeto. Todos los tipos primitivos excepto null y undefined se tratan como objetos. Pueden asignar propiedades, y tienen todas las características de los objetos.

Constructor

El constructor es una método especial que se ejecuta automáticamente cuando se crea una instancia de esa clase.

Su función es inicializar el objeto y sirve para asegurarnos que los objetos siempre contengan valores iniciales válidos.



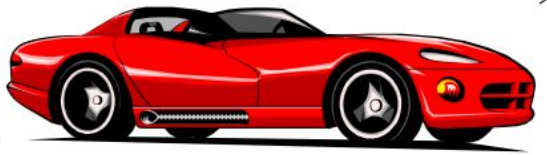
dev.f

desarrollamos(personas);

Programación orientada a objetos

Clases

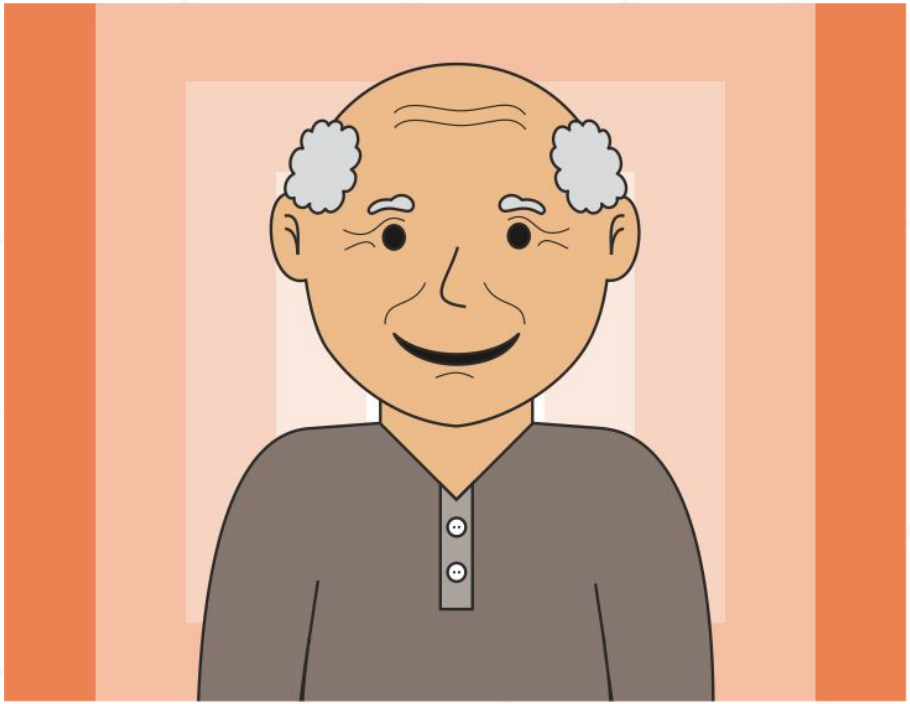
Definiciones de las propiedades y comportamiento de un tipo de objeto concreto.

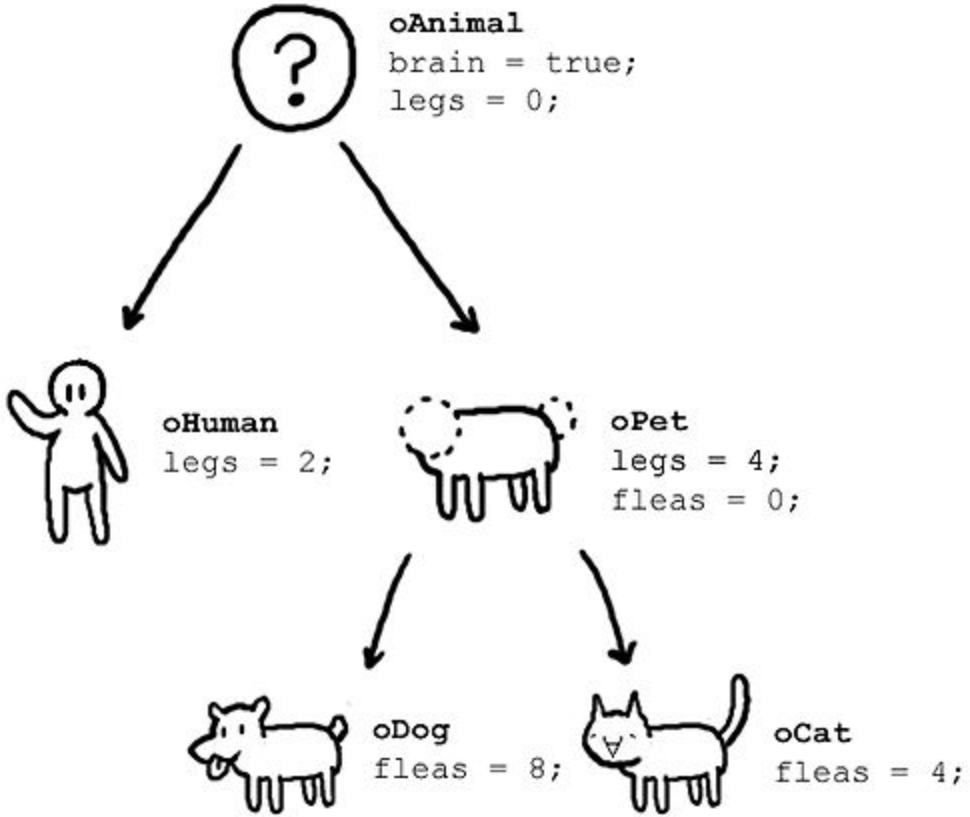


- Atributos:
  - color
  - velocidad
  - ruedas
  - motor
- Métodos:
  - arranca()
  - frena()
  - dobla()

Herencia

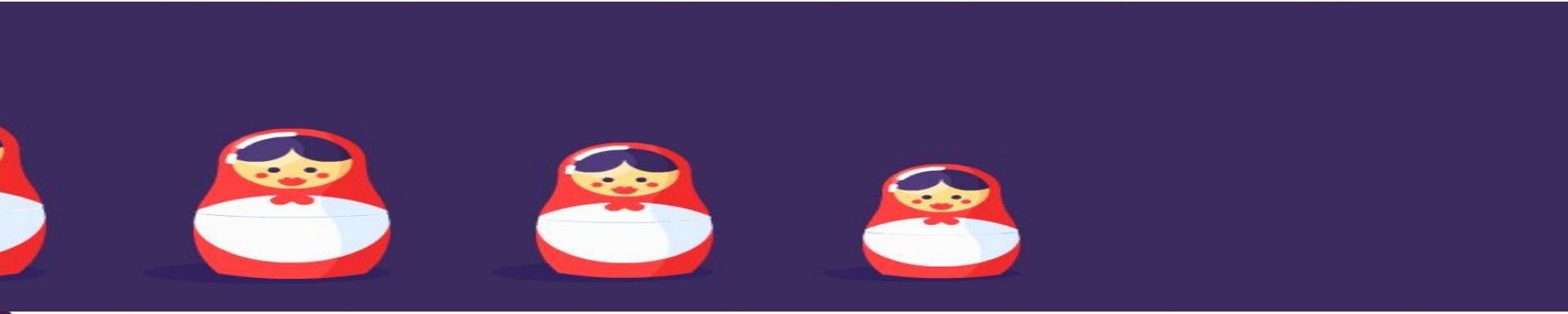
Es una clase nueva se crea a partir de una clase existente, esta clase comparte los atributos y comportamientos a la principal otra ventaja de la herencia es la capacidad para definir atributos y métodos nuevos para la subclase.

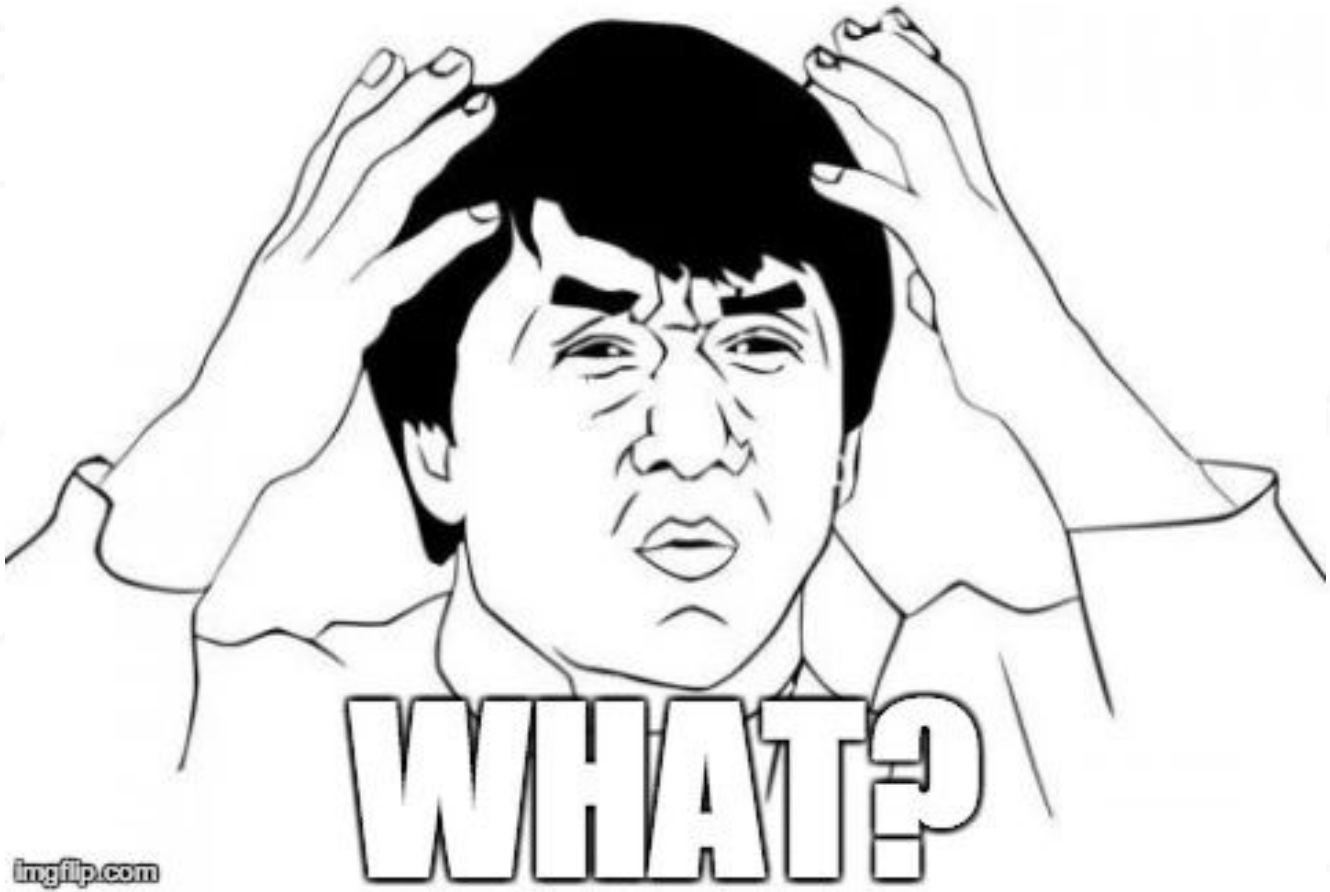



oAnimal
brain = true;
legs = 0;
oHuman
legs = 2;
oPet
legs = 4;
fleas = 0;
oDog
fleas = 8;
oCat
fleas = 4;

Liskov substitution principle

En lenguaje más formal: si S es un subtipo de T, entonces los objetos de tipo T en un programa de computadora pueden ser sustituidos por objetos de tipo S (es decir, los objetos de tipo S pueden sustituir objetos de tipo T), sin alterar ninguna de las propiedades deseables de ese programa (la corrección, la tarea que realiza, etc.).




WHAT?
imgflip.com

Ejemplo

En un cine se reproducen largometrajes. Puedes, no obstante, tener varios tipos de largometrajes, como películas o documentales, etc. Quizás las películas y documentales tienen diferentes características, distintos horarios de audiencia, distintos precios para los espectadores y por ello has decidido que tu clase "Largometraje" tenga clases hijas o derivadas como "Película" y "Documental".

Imagina que en tu clase "Cine" creas un método que se llama "reproducir()". Este método podrá recibir como parámetro aquello que quieres emitir en una sala de cine y podrán llegarte a veces objetos de la clase "Película" y otras veces objetos de la clase "Documental".

Si quisiera reproducir una película tendría los siguiente:

reproducir(Pelicula peliculaParaReproducir){... }

Pero si luego tienes que reproducir documentales, tendrás que declarar:

reproducir(Documental documentaParaReproducir){... }

Es realmente necesario hacer 2 métodos?