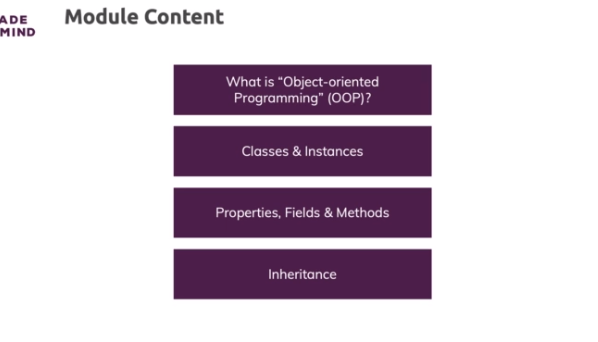
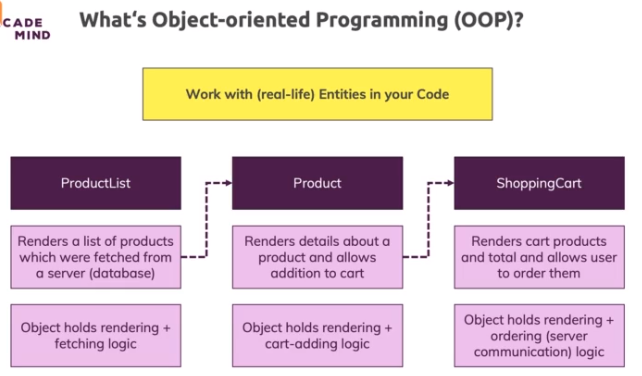
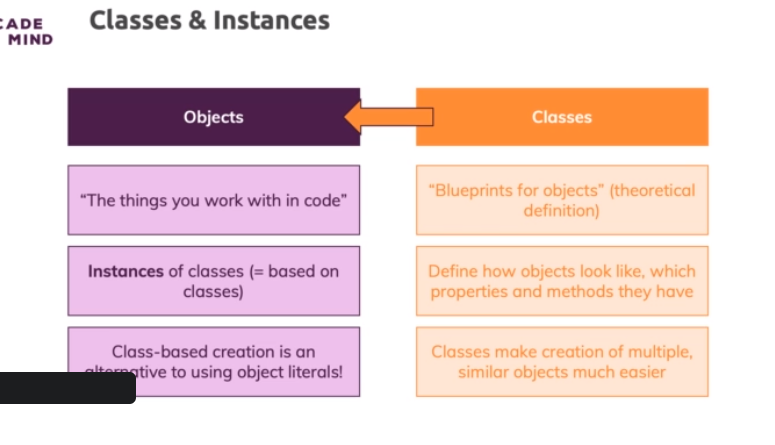
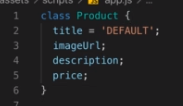
**Classes & Object-oriented Programming**

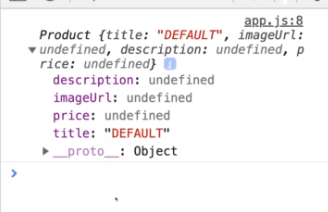




**Defining & Using a First Class**

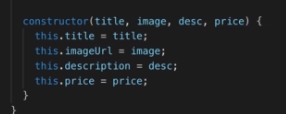






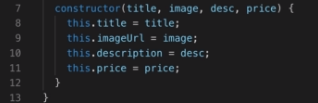
**Working with Constructor Methods**

* Now a method is added just by adding some name and this can be any name of your choice of course, then parentheses and then curly braces

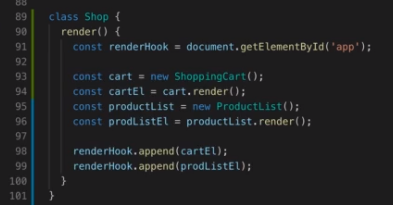


**Fields vs Properties**

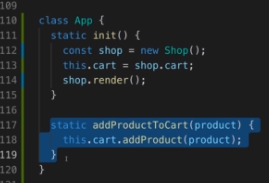




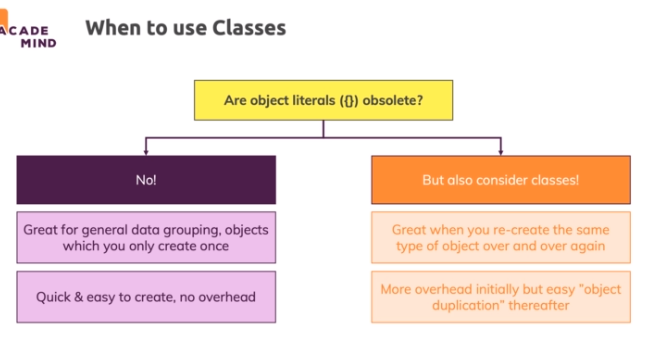
**Adding a Cart and Shop Class**



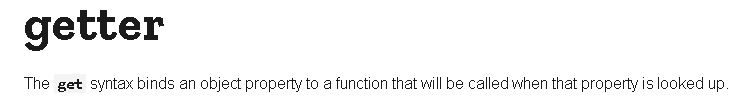
**Static Methods & Properties**

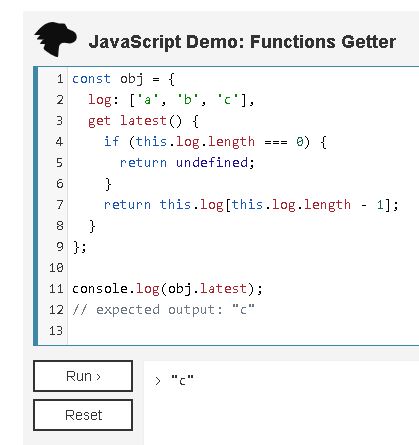


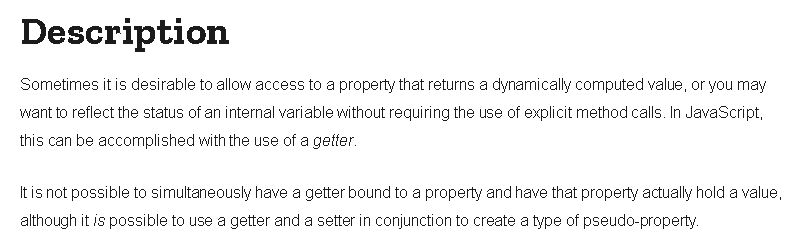
**First Summary & Classes vs Object Literals**

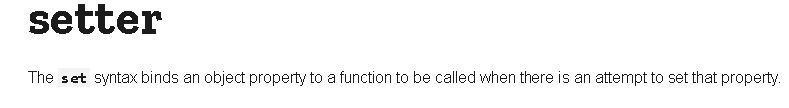


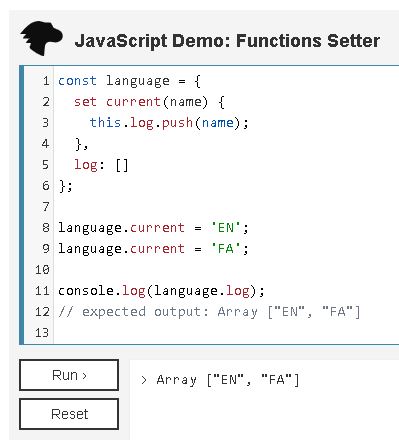
**Getters & Setters**

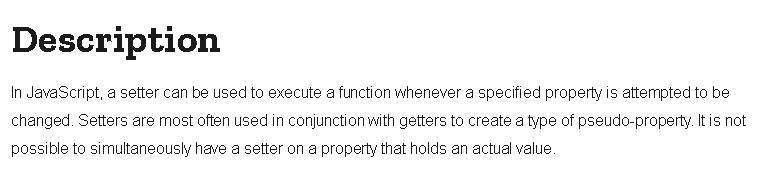




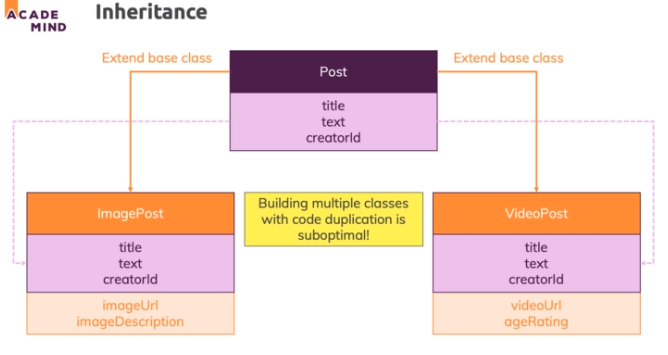




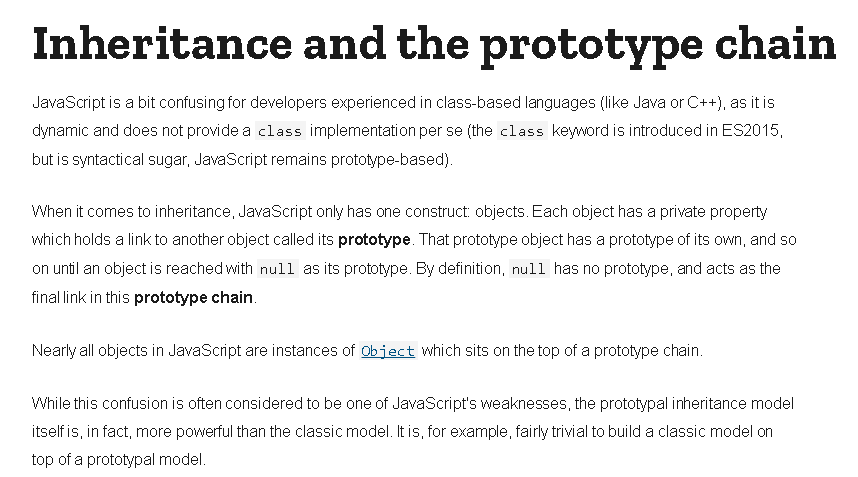


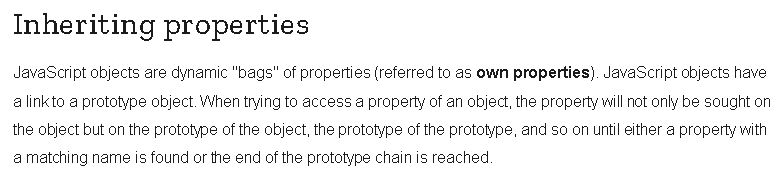


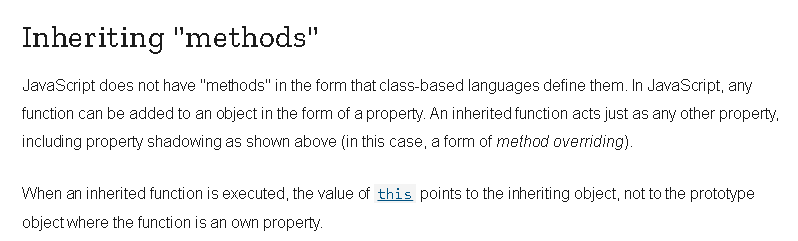
**Introducing Inheritance**



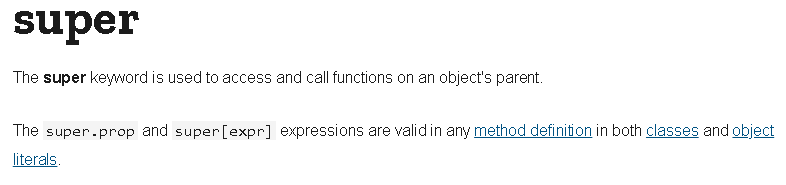
**Implementing Inheritance**

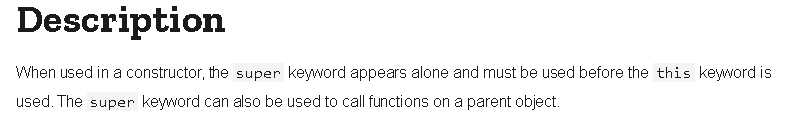




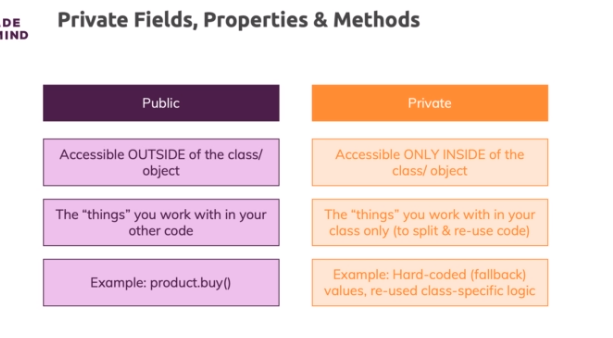


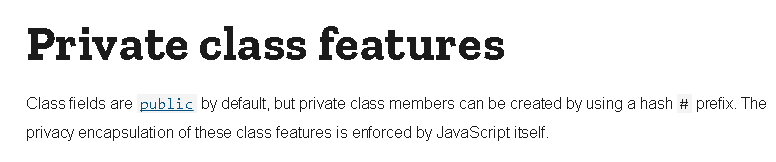
**Overriding Methods and the super() Constructor**

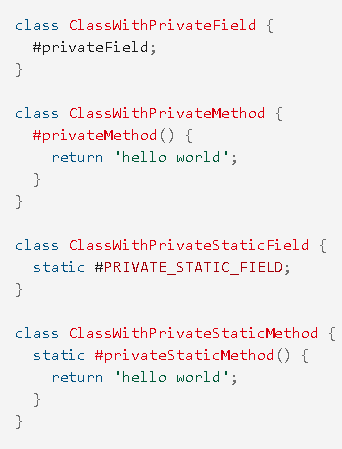




**Private Properties**







**"Pseudo-Private" Properties**

The addition of private fields and properties is **relatively new** - in the past, such a feature was not part of JavaScript.

Hence you might find many scripts that use a concept which you could describe as **"pseudo-private" properties**.

**It would look like this:**

    1. class User {

    2.     constructor() {

    3.         this.\_role = 'admin';

    4.     }

    5. }

    6.

    7. // or directly in an object

    8.

    9. const product = {

    10.     \_internalId: 'abc1'

    11. };

**What's that?**

It's a quite common convention to prefix private properties with an underscore (\_) to signal that they should not be accessed from outside of the object.

**Important:** It's **just a convention** that should signal something! It does **NOT technically prevent** access. You **CAN** run this code without errors for example:

    1. const product = {

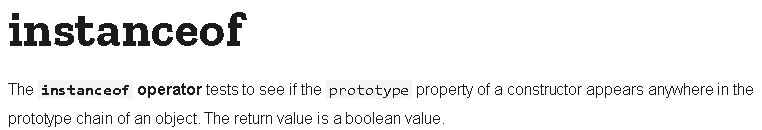
    2.     \_internalId: 'abc1'

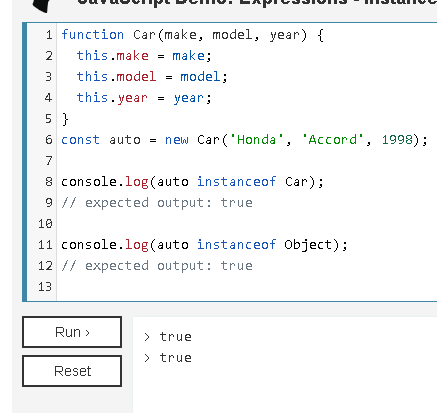
    3. };

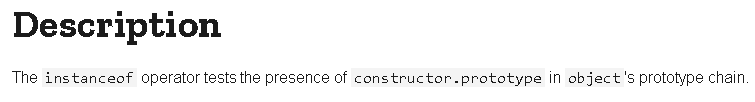
    4. console.log(product.\_internalId); // works!

It's really just a hint that developers should respect. It's not as strict as the "real" private properties introduced recently (#propertyName).

**The "instanceof" Operator**







**Built-in Classes**

