2 Important ways the prototype is used in JS:

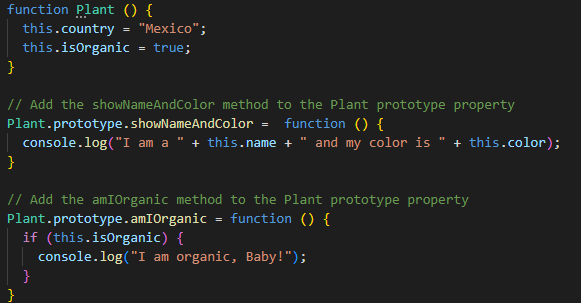
1. Prototype Property: Prototype-based Inheritance

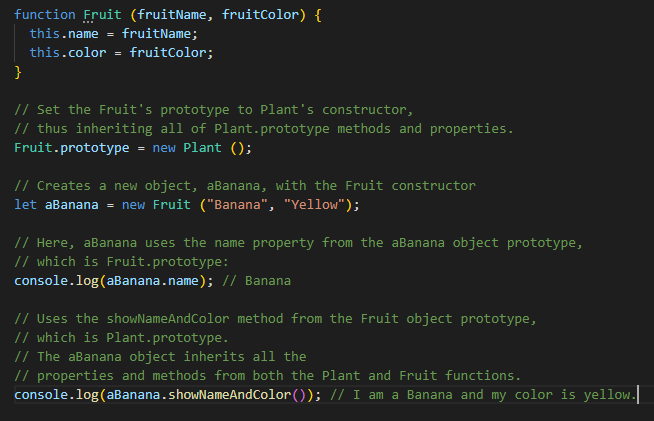
* JS does not have CLASSICAL inheritance (based on Classes)
* All inheritance is made possible by prototype property

Inheritance

* + Programming paradigm where objects (or Classes in some languages) can inherit properties and methods from other objects (or Classes)

Demo of Inheritance in JavaScript





You can make a prototype(blueprint) inherit a certain prototype. (Like merging of blueprints)

Any object that uses the Fruit() constructor will now INHERIT ALL the Fruit.prototype properties and the Plant.prototype properties. But, any object that uses the Plant() constructor, will only inherit Plant.prototype and not Fruit.prototype properties

1. Prototype Attribute: Accessing Properties and Methods on Objects

Analogy:

This is loosely analogous to the way you might inherit your surname from your father—he is your “prototype parent.” If we wanted to find out where your surname came from, we would first check to see if you created it yourself; if not, the search will move to your prototype parent to see if you inherited it from him. If it was not created by him, the search continues to his father (your father’s prototype parent).

First look at the object’s prototype, then move up to the prototype of the object’s prototype and so on… this is called the PROTOTYPE CHAIN (upwards to the inherited from)

If property or method does not exist in the prototype chain, the return value is *undefined*.

Example code:

