Every single “object” is built by a constructor function

Each time a constructor is called, a new object is created

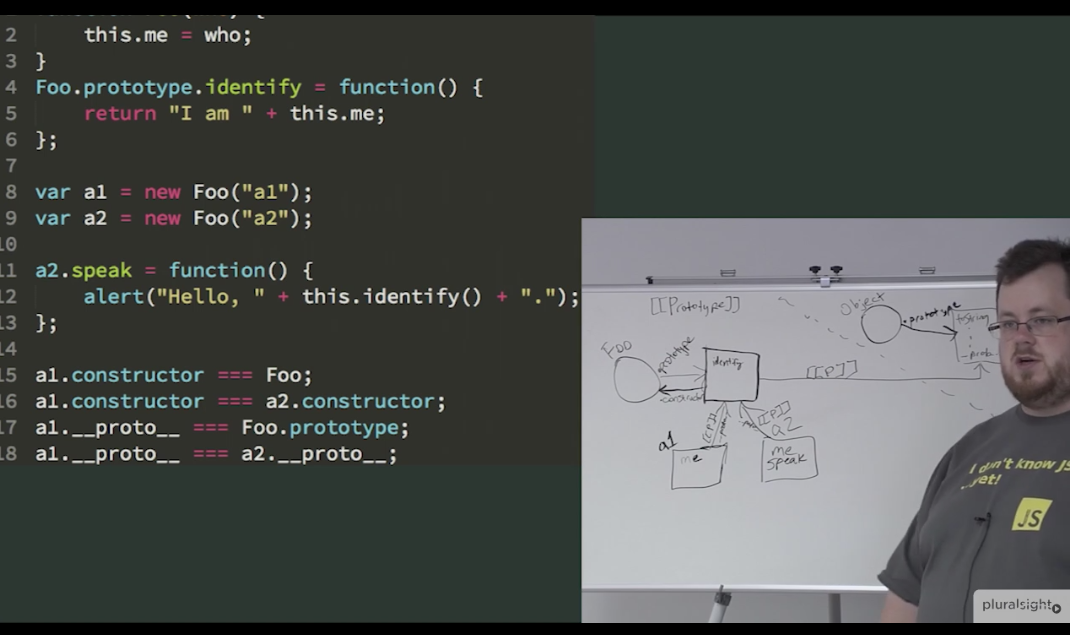
A constructor makes an object “~~based on~~” (**linked to**) its own prototype

Function -> circle

Object -> square

New functionality

1. Brand new object is created
2. Object gets linked
3. Context will be set to **this**
4. Assign label to object



In JavaScript, objects have a special hidden property **[[Prototype]]** (internal linkage to another object).

The property [[Prototype]] is internal and hidden, but there are many ways to set it.

One of them is to use \_\_proto\_\_.

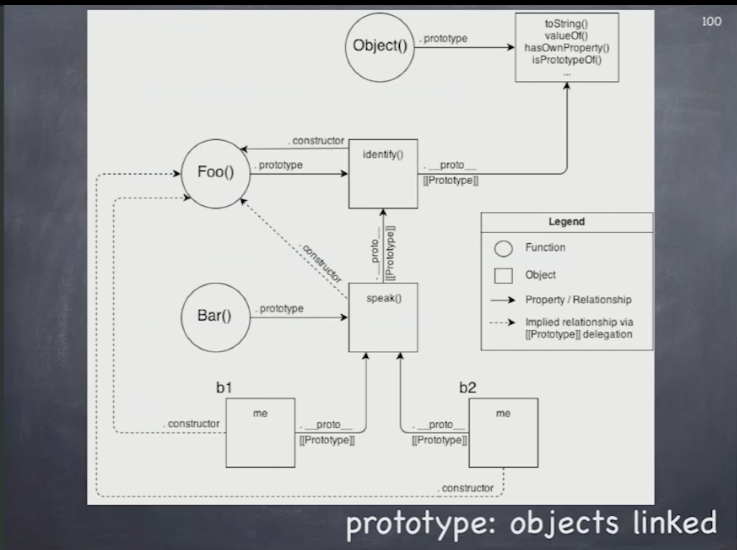
Prototype linkage -> delegate to different to handle method call/ property call

Bottom to top

Inheritance -> copy behavior from top to bottom

Object.create()

1. Brand new object is created
2. Object gets linked



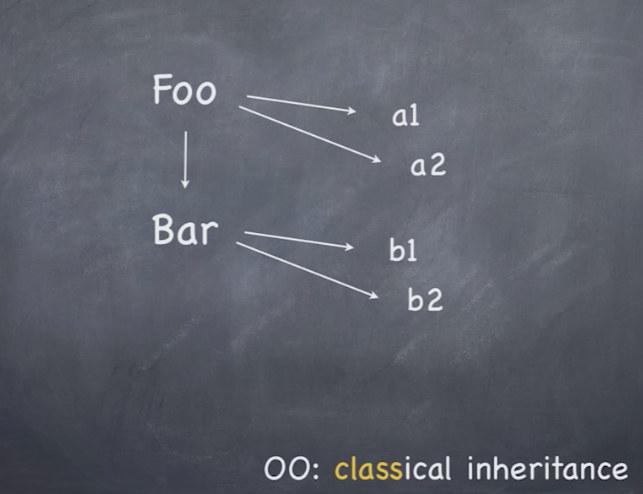
**Object literal vs constructor function**

The regular {...} syntax allows to create one object. But often we need to create many similar objects.

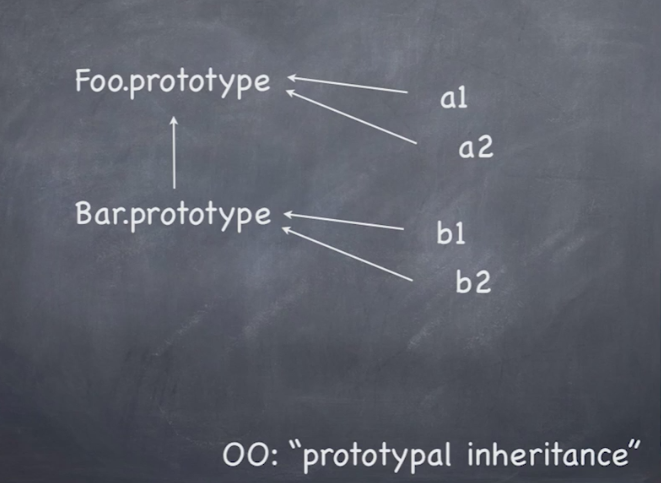
That can be done using constructor functions and the "new" operator.

With **new** operator

Constructor functions should only be called using new



Copy down



Delegation up to the chain

