**Section 13 – Quiz**

**Prototype - Quiz**

**Hot diggity dog!**You have really come a long way.

Prototypes are one of the most important concepts that every JavaScript developer should understand. Its important because in programming, we often want to take an existing object we’ve created, and extend it. And in order to do this, we have prototypes.

Remember, all JavaScript objects inherit properties and methods from a **prototype**. Let me say it another way … every object in JavaScript has a private property which holds a link to another object. This other object is called its **prototype**. This means that the Array object inherits from a prototype (the Array.prototype object to be exact), the Date object inherits from the Date.prototype, and so on.

As I’m sure you have already seen, prototypes can be difficult to grasp, but you've got this.

Enjoy these few questions.

Top of Form

Question 1:

What is a prototype?

* 

**Prototypes are the mechanism by which JavaScript objects inherit properties and methods from one another.**

* 

**It is just a fancy word for ‘execution’**

* 

**It is a datatype of JavaScript that allows you to store multiple key:value pairs**

Bottom of Form

Answer: A

Top of Form

Question 2:

What is the difference between prototype and \_\_proto\_\_?

* 

**Prototypes are only available on objects, whereas \_*proto*\_\_ is available on everything**

* 

**Prototype is available everywhere, whereas \_\_proto\_\_ is only available on constructor functions**

* 

**Prototype is not available on the instances themselves (or other objects), but only on constructor functions. It is used to build \_\_proto\_\_ when the function happens to be used as a constructor with the new keyword**

Bottom of Form

Answer: C - \_\_proto\_\_ is an internal property of an object, pointing to its prototype. Current standards provide an equivalent Object.getPrototypeOf(O) method, though typing \_\_proto\_\_ is quicker. According to ECMA specifications it is supposed to be an internal property, however most vendors (such as Google Chrome) allow it to be accessed and modified.

Top of Form

Question 3:

How do you access an objects [[prototype]]?

* 

**You’ve always been able to access an objects prototype by Object.getPrototype(obj)**

* 

**Since ES 2015 you *can* access an object's prototype indirectly via Object.getPrototypeOf(obj)**

* 

**There isn't a way to access an object's [[prototype]] object**

Bottom of Form

Answer: B