**Objects & Functions:**

**Topic: Objects and the dot**

In summary, Objects are collections of name value pair and those values can be a collection other name value pairs.

An object can consist primitive property, object property & function “methods”.

**Example:**

var person = new Object();

person[“firstname”] = “Ali”;

person[“lastname”] = “Khan”;

In the above example, “firstname” is a property and “Ali” is a string (primitive). Javascript will create a property inside the person object and add it into the memory space. Then it knows there is something in the memory called firstname, which is inside person object and it should be assigned the value = “Ali”;

In short, it will look for the property or method inside that object, which in this case is person object.

Var firstNameProperty = “firstname”;

Console.log(person);  
// this will display you the object in console and if you expand it then you will be able to see its properties i.e. firstname & lastname.

console.log (person[firstNameProperty]);

// this will display the only the value defined in firstname because we mentioned in our code that a variable called firstNameProperty is equivalent to the property called firstname of person object.

The above practice is kind of an old way of working with objects specially working computed Member Access [] (square brackets). The better & easier way is access any property of an object is to use a dot (.), which has second highest precedence.

**For example:**

In the following example you will see both ways of retrieving the properties of an object.

console.log(person["firstName"]); // old method of working

console.log(person.firstName); // new & easier method of working

The new way works easily because you can directly look at the object then use a dot then the name of the property. Also, you don’t have to use the quotes used in the old method of working.

An object can consist primitive property, object property & function “methods”.

Remember Person is an object and address is it’s property but we will define another object as its property.

person.address = new Object();  
person.address.steet = “111 Main Street”;

person.address.city = ”London”;

or

console.log(person[“address”] [“state”]);

The dot (.) operator has left to right associativity which means it will work from left to right

Example:

person.address.street

the first dot between person & address will be read then it will understand that I need to look into the property or method called address but when it reaches address it will find out from the memory that it is also an object (sub-object) because it is already an object (person).

The preferred approach is to use dot (.) instead of computer member access [ ] .

For Reference:

console.log(person.address.street);

console.log(person.address.city);

console.log(person[“address”][“state”]);

Working Example – Documents/udemy/javascript – the weird part/objects/ objects.js