In this lesson we are going to learn the difference in syntax between Arrays and Objects in JavaScript. An Array is used to store a list of data, and when we store items in an array there is a very specific order, every item is bound to an index that we refer to it by. The first item is bound with index 0, the next item is bound to index 1 and so on.

An Object on the other hand is not a list, there is no order, its just like a gelatinous blob with a bunch of things floating around inside. The key-value pairs are an important aspect of an Object, and in some languages, objects are called dictionaries.

The idea of dictionary is important here, if you think of a dictionary it has a bunch of key-value pairs, it has words and then those words have their own corresponding definitions.

Let's demonstrate their respective syntaxes. First, we will see how we declare a variable in an Array format and in an Object format.

Array:

var dogs = ["Rusty", "Lucky", "Bubba"];

dogs

➔ *["Rusty", "Lucky", "Bubba"]*

Objects:

```
var dog = {
    name: "Bubba",
    breed: "Lab"
    }
```

dog

➔ *Object {name: "Bubba", breed: "Lab"}*

Lets now compare the syntax of how to access the data from an Array and an Object

Array:

We want to retrieve the second item; we want to get Lucky out of the array.

dogs[1];

➔ *"Lucky"*

Object:

Now we want to get the name value from the Object. Let's see how we can get it using the square bracket notation.

dog["name"]

➔ *Bubba*

Now we will see how we can get the same item using the dot notation.

dog.name

➔ *"Bubba"*

Both notations do the exact same thing, and you should use either one.

One important comparison we need to note here is that an Array is a very special type of an object where the keys are always numbers. That is why Array also follows the key-value pairs logic between 0 and Rusty, 1 and Lucky, 2 and Bubba.

On the Object we have the key-value pairs where name is the key and Bubba is the value, breed is the key and Lab is the value.

That is why Array is a special version of an Object and behind the scene it is an Object which we will see later.

Now we will see the syntax difference in adding new data to these two data structures.

Array:

We can use methods like push or unshift; or we can do it manually by figuring out which index we want to add the data to.

dogs.push("Wyatt")

➔ *4*

dogs

➔ *["Rusty", "Lucky", "Bubba", "Wyatt"]*

We can see that there is a new element in our dogs' array.

Object:

To add a new element to our dog object, if we want to add the property "age" we do not need a special method like "push" or "unshift", because the whole point of those Array methods are to work with the order of the Array, to add something to the end or beginning; but as there are no beginning or end in an object, all we have to do is as below,

dog["age"] = 6; *or* *dog.age = 6;*

dog

➔ *Object {name: "Bubba", breed: "Lab", age: 6}*

We can see that there is a new key-value pair that we added as age:6.

This is a big difference, to add something in an Object we can add it arbitrarily by assigning any key we want and any value we want and it will be automatically added in for us; whereas in an Array we specify exactly where we want our new element to be added in, at the end of the Array, or at the beginning, or index 1000, its all up to us.

Lastly, lets talk about Updating items. Let's change the name of "Lucky" to "Lucy" in our array.

Array:

dogs

➔ *["Rusty", "Lucky", "Bubba", "Wyatt"]*

dogs[1]

➔ *"Lucky"*

dogs[1] = "Lucy"

➔ *"Lucy"*

dogs

➔ *["Rusty", "Lucy", "Bubba", "Wyatt"]*

Object:

In an Object it works very similarly. Let's change the breed from "Lab" to "Black Lab".

dog

➔ *Object {name: "Bubba", breed: "Lab", age: 6}*

dog.breed = "Black Lab"

➔ *"Black Lab"*

dog

➔ *Object {name: "Bubba", breed: "Black Lab", age: 6}*

Now we can see that breed has a value of "Black Lab".

Lets sum up the big differences here, both of them, Arrays and Objects use key-value pairs but Arrays are special subset of key-value pairs, where the keys are always numbers and they are always in order; but in an Object the key can be anything, it does not matter if it is a number, or the key is a name or a breed, its arbitrary and there is no special order in an Object.