INDIAN INSTITUTE OF TECHNOLOGY MADRAS • सिद्धिर्भवति कर्मजा •

IIT Madras

ONLINE DEGREE

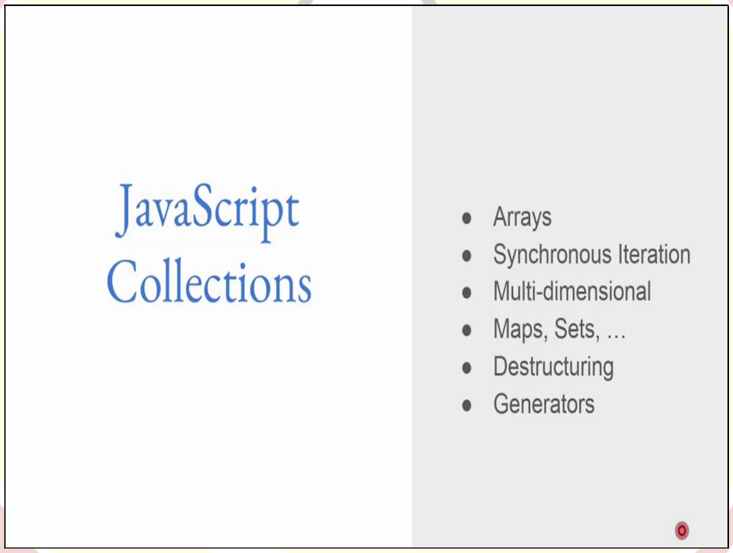
Modern Application Development II
Online Degree Programme
B. Sc in Programming and Data Science
Diploma Level

Prof. Nitin Chandrachoodan
Department of Electrical Engineering
Indian Institute of Technology- Madras

Introduction to Javascript Collections

Hello everyone welcome to modern application development part 2. Now we are going to continue on our journey through Javascript and in this set of videos we are going to be looking at some slightly more advanced topics in the context of Javascript. So, we will start with collections.

(Refer Slide Time: 00:27)



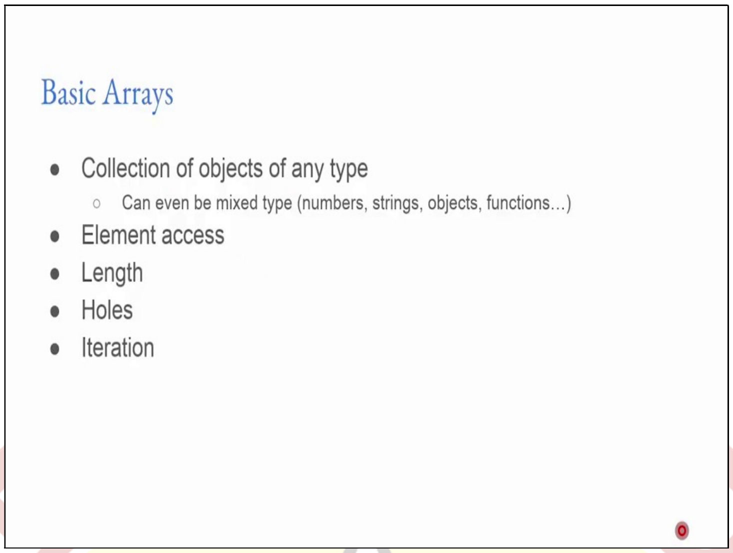
JavaScript Collections

- Arrays
- Synchronous Iteration
- Multi-dimensional
- Maps, Sets, ...
- Destructuring
- Generators

And what do I mean by Javascript collections the most basic form of a collection is an array and what do we do with arrays we basically want to be able to look at the items one at a time right which is basically a cons called iteration which we will look at there are also something. So, like multi-dimensional arrays that can be implemented and a few other concepts that we need to look at.

What I am going to do is first cover some of these concepts at a slide level where I just sort of talk about them and then we will switch over and look at some examples using replit.

(Refer Slide Time: 01:05)



Basic Arrays

- Collection of objects of any type
  - Can even be mixed type (numbers, strings, objects, functions…)
- Element access
- Length
- Holes
- Iteration

So, the basic idea of an array is that it is a collection of objects right and the interesting thing about Javascript arrays unlike let us say a C array is that the objects can be of any type. Now in C when I declare an array I usually declare it as an array of int or an array of short or an array of char or in some case it could be that I have defined a string data type and therefore I could have an array of strings right but in Javascript I can basically have a mixed set of values right.

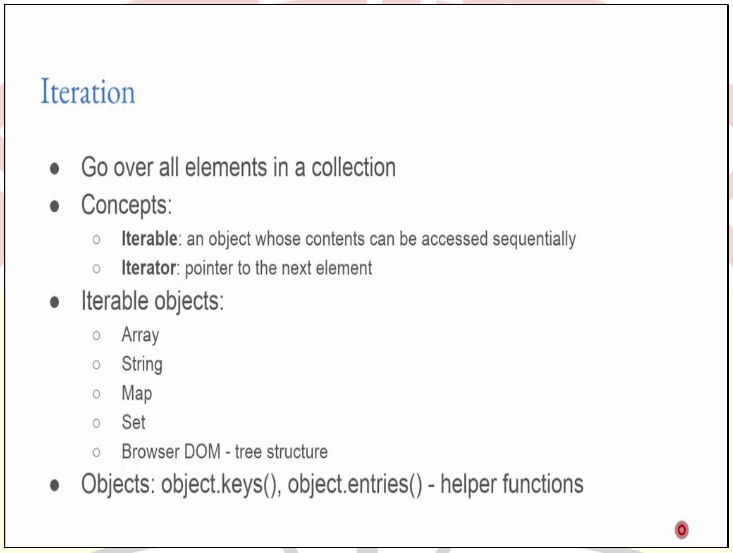
So, in this sense it is very close to what a list in python is like lists in python are also mixed type right whereas a numeric python array has a very specific data type that is applied to all the elements in the array. So, even though in Javascript these are called arrays they are closer to the lists in python and are more sort of generic collections. Now what are the things that we would want to do with arrays.

We would need we want to be able to access any given element in the array we want to find the length of the array that is the number of elements in the array and over here there is also one interesting thing which comes up in the context of Javascript which is not really there in other languages. So, much which is that you can create an array with holes in it. What is that? We will get to that soon.

And of course one of the most important things you want to do with an array is after all you have put a number of different items together into one collection presumably because they have something in common right otherwise I mean why even bother to put them into the same array or collection. Which means that you might also want to sort of iterate over them that is go over each one of those elements and do something right.

Call a function or print them out or do something else but for doing that we need to be able to sort of iterate over all the elements in the array sequentially. So, all of these things are what we are going to look at.

(Refer Slide Time: 03:01)



Iteration

- Go over all elements in a collection
- Concepts:
  - **Iterable**: an object whose contents can be accessed sequentially
  - **Iterator**: pointer to the next element
- Iterable objects:
  - Array
  - String
  - Map
  - Set
  - Browser DOM - tree structure
- Objects: object.keys(), object.entries() - helper functions

Now the idea of iteration or iteration right both pronunciations seem to be common enough is to go over all the elements in a collection right and there are a couple of concepts over here in the language of Javascript. They refer to something called an iterable or an iterable is an object whose contents can be accessed sequentially . So, you can think about this I mean if you after all in Javascript in principle everything is an object right.

So, you could just have an object which is like a number one right and one is not really an object whose contents can be accessed sequentially there is only one piece of content over there right. So, there is no sort of concept of sequential axis. And similarly if I just have a single character a then same thing right you do not really have any concept of accessing the character after there is only one of them sequentially.

But pretty much anything else can be iterated over and what is it that does the iteration you need to have a pointer to where you are right now. And you need some way of sort of moving forward in that iteration right. And the iterator is basically the pointer to the next element and how we implement that iterator function will finally decide how iteration happens on a collection.

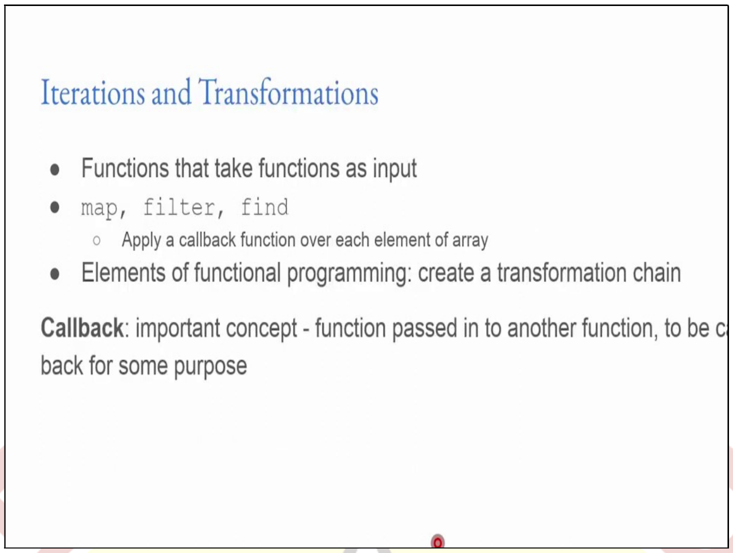
Now the reason for having the sort of abstract description is that you are no longer just limited to very single for loops or while loops you can also do something more complicated where you can create an object and make it iterable and define new kinds of functionality on it. Now the most basic iterable objects in Javascript are of course arrays which we are looking at.

But apart from that even a string right a Javascript string can be iterated over. Similarly there are a couple of other objects called maps sets and there is also something called a weak map right which are slightly more advanced topics we are not really going to get into those. All of those can be iterated over meaning that there is some notion of multiple entities being present inside such an object and you can go through those in sequence.

And from the point of view of Javascript and its application to the web the browser DOM which is basically some kind of a tree structure right the document object model itself can be iterated over. So, it is possible to sort of traverse the entire document object model right by saying next next next and going through all the elements one by one. That becomes useful when you are basically going through the different processing of the DOM.

Now there are some helper functions that are defined in terms of objects right things like object.keys, object.entries and so on we will look at these a little bit later when we get to the examples of code.

(Refer Slide Time: 05:55)



Iterations and Transformations

- Functions that take functions as input
- `map, filter, find`
  - Apply a callback function over each element of array
- Elements of functional programming: create a transformation chain

Callback: important concept - function passed in to another function, to be c back for some purpose

Now in addition to this and this is something that we will be looking at more when we get to the examples we have functions that can take functions as input. So, this whole idea of a function being passed as an argument to another function is a very powerful construct right it can be done even in a language like C you do have this concept of function pointers and how you can pass function pointers as inputs to other functions and so on.

But in a language like C it is cumbersome at best right I mean you have to sort of be very clear that this is a function this is a pointer to a function and I am therefore sort of you know passing around pointers to the functions and so on. Whereas languages like python and Javascript make it a lot easier they almost treat functions as sort of first class objects right which means that a function can itself be treated as an object in some way.

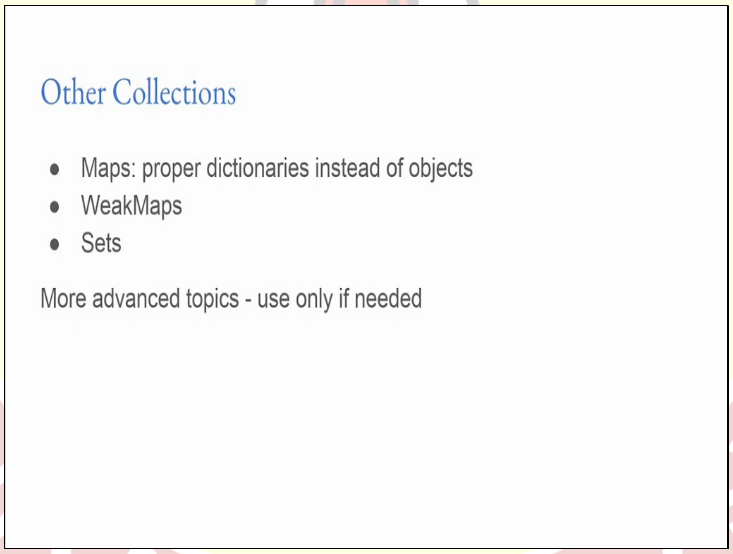
Now where this really becomes powerful is that there are certain classes of functions this map filter find and so on are some examples of those and effectively what they do is they apply something called a callback function right. So, there is an additional function that is passed into let us say map right. And this idea of a callback right is a very important concept it basically says there is a function which is parsed into another function and depending on something that happens internally in the top level function.

The function that was passed in as an argument would be called back. Now why is that useful there are a number of different types of functionality that can be realized in this way right. And all of these are sort of related to some concepts of what's called functional programming right it allows us to sort of create chains of transformations and also pass functions around and have different kinds of behaviour depending on different scenarios that are present.

But we will not be getting into any details on functional programming why it is useful or how it is done what is useful to know is that Javascript sort of allows you to do some of those things and more than anything else it makes the syntax relatively easy right. Remember like I said even in C you can do this kind of you know call backs right except that it is somewhat cumbersome to do.

Whereas in Javascript or python it becomes a lot easier because the language syntax allows it in a more sort of fluid manner and that is primarily what we are looking for over here.

(Refer Slide Time: 08:26)



Other Collections

- Maps: proper dictionaries instead of objects
- WeakMaps
- Sets

More advanced topics - use only if needed

There are some other collections in Javascript in particular there is a concept of a map right which is essentially the equivalent of a python dictionary but as we will see you can actually even just use regular Javascript objects almost like dictionaries. So, do you really need a map what are the benefits of using maps. We will not be getting into too many details on those in fact there is something else called a weak map and the set and the weak set all of those are things that were brought into Javascript relatively recently.

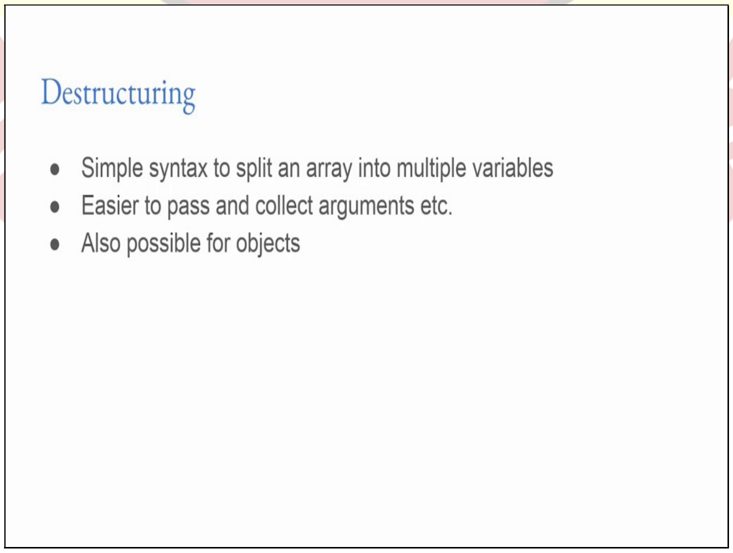
With the aim of sort of you know making certain kinds of coding more well structured right. Now you have probably understood that you know Javascript the real power lies in the fact that it had a very simple sort of definition to start with right it was very flexible which meant that even structures that did not exist in the language. So, for example proper block level scoping right or classes even though they did not exist in the language to start with people were able to sort of hack around it.

And come up with ways of implementing proper scoping using IIFEs right immediately invoked function expressions or they could for example have object orientation using some notion of something called prototypes right. The point is those are somewhat difficult to use and more importantly they are difficult to read and understand. So, a lot of these newer changes including the introduction of data types like maps and so on has been done with the aim of making Javascript a little easier to read and understand for others.

You could have a lot of the core functionality without requiring these things as well. So, in our case what we are going to do is we will only be looking at some of these things as in when needed we are not going to be sort of getting deep into them and trying to understand how these work straight away.

Now one last thing with regard to collections is that there is one interesting idea that can be used over here which is called destructuring.

(Refer Slide Time: 10:28)



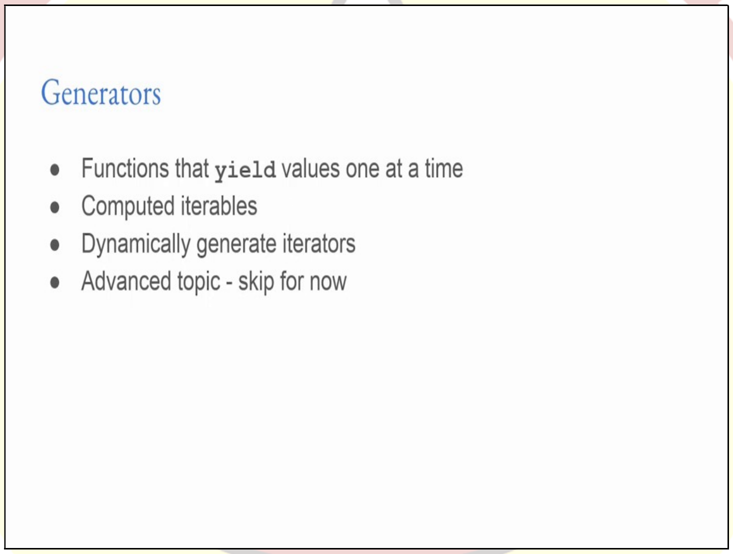
Destructuring

- Simple syntax to split an array into multiple variables
- Easier to pass and collect arguments etc.
- Also possible for objects

Once again this is just part of the language syntax it is not a sort of new concept in the language as such what it provides is a simple syntax that allows us to split an array into multiple values. And there are a number of uses that have been found for this in Javascript right including the ability to sort of pass the so-called function arguments can be passed by name rather than passed by position.

You can have a concept of basically using named arguments which is there in python of course but which is not really there in C by default. So, that can be powerful in a lot of cases it allows you to sort of set default argument values and various other things and Javascript even though it does not have it as part of the syntax is able to achieve the same kind of functionality using some very simple syntactic constructs. So, we will briefly look at that as well at some point.

(Refer Slide Time: 11:26)



Generators

- Functions that `yield` values one at a time
- Computed iterables
- Dynamically generate iterators
- Advanced topic - skip for now

Now finally there is this notion of something called a generator right and a generator and this is a slightly advanced topic. So, we are not really going to get too deep into this. Having said that the reason I am introducing it over here is because especially for those of you who are interested in learning more about Javascript and understanding in particular some concepts like concurrent execution.

The idea of a generator is important to know and a generator is basically a function that yields values. So, what this keyword yield is in fact used inside the generator and the idea behind that yield is to say that whenever it is sort of the equivalent of a return statement except that it is not exiting the function completely. It is just saying that at this point in time the function is giving a value and saving its present state and waiting to be restarted from the point where it left off.

Why is this useful it basically allows us to think in terms of functions that are sort of cooperating with each other right one function yields a value to another one the other one does something with it exits and we go back to the original function and continue from where we left off right. So, in principle both the functions are active and functioning at the same time right that is a notion called concurrency which is very powerful and is sort of the basis of parallelism concurrency does not mean the same as parallelism but is in some sense required if you want to be able to implement parallelism.

So, generators are functions that sort of take you in that direction right they allow you to sort of yield values and have multiple functions operational at the same time it sets up the sales stage for what's called cooperative concurrency right. And there are a number of sort of advanced topics over here computer iterables dynamically generated iterators and so on which we are not going to get into any details at this point all right.
(Video Start: 13:24)

So, now that we have looked at some of the as we can say theory behind collections it is not exactly theory it is sort of just a moral description of a few of the concepts that we can look at. Let us look at code right because ultimately in order to learn a language the best way is to actually look at the code and understand what it is doing. So, once again you know I am going back to the replit example that I had I have this index.html right which is a very basic code.

All that it does is pretty much just puts hello world on the screen. In the previous videos we had seen you know how the div's could be declared and we could actually sort of insert data into it right modify it and so on. And essentially showing how the Javascript API's can interact with the DOM. Now I am not going to be using a lot of that right. Now what I do have is there are a bunch of scripts but most of them are commented out there is only one that is active currently which is collections.

So, let us go see what is in collections. Now this first part is in some sense the most basic form of declaring an array right what we have is let x equals 1, 2, 3 and what we can see over here is I am going to basically log something saying dollar remember this back tick method that we use right which basically is sort of a templating string right.

So, once I have that back tick back tick over here inside that dollar and the curly brackets anything inside of that will actually get evaluated.

So, dollar type of x should actually run the type of function on x which in this case is array right and print something out the actual array itself and also tell you what the length of the array is. And the second statement console dot log x of 0 should pick out the first element because these are again like python zero indexed. So, what happens when I run this I find that the first thing it says is that this is an object.

So, keep in mind I mean even though I have declared it using these square brackets in this way as far as Javascript is concerned this is an object it does not really think of it as a array data type it is ultimately an object that is pretty much what it is there are other ways of sort of finding out whether it is an instance of an array and so on. But for the time being we are not really getting into that.

All right let us go a bit further and what happens if I let say have another array that I declare but now look at this second array y right I have got one over here the second element is b and the third element is actually well take a moment and think about it right what is this actually doing remember the arrow notation right what it is saying is it takes an input a and converts it right into a + 1.

So, replit has this nice behaviour where when you hover over a particular syntactic construct in the code it actually shows the small tooltip right hanging above this that part and clearly it shows that this is basically a function right. And what kind of a function is it it takes a as input where a is of any type right it is not necessarily an integer it could be anything and it returns any type.

Now why is that being written over here because we are not specified the type of a Javascript by itself does not have a way by which it specifies types unlike C for example right. Now there is a variant of Javascript not exactly a variant it is actually almost a language of its own called typescript where you actually have to explicitly specify types right. And that leads to a lot of very useful optimizations that the compiler can do which lead to sort of better functioning Javascript.

And in general you know better debugging capabilities at least right in some cases even performance optimizations can be performed that could not have been done in pure Javascript but basic Javascript does not have that notion of types. So, this notation here function a colon any colon any is more a sort of shorthand that people understand rather than being part of Javascript itself all right.

So, you know long story short let us run it with y over here right and what we find is that the second one once again comes out to be an object right y is also an object the contents of y are 1, b, a and a + 1 right and with length equal to 3. Now this we could go one step further over here right and yeah in fact we will get to that later. So, you know for the time being let us just leave this as right.

The important point over here was that I could have an object with mixed data types I could have a number I could have a character and I could have a function all of them as being part of the same array right. And the length of both x as well as y are essentially the same they were determined just by the fact that the number of elements in each over there. Now I can do one thing further which is basically to say that I can essentially null out an entire collection or an array right.

What happens when I do that I find that x has now become it continues to be an object but now its length is equal to zero. So, just setting it to null like this effectively deleted everything that was there inside x. Now this next stage here is a little bit more interesting right what I am going to do is y dot length equal to 10. Now normally in python for example the length of a function is not something that you can change a length of an array or a list is not something that you directly change right it is a computed parameter but in Javascript it looks like you can actually assign directly to the length.

So, what happens when I do this I find that now right maybe it is easier if we comment out all the other console lines and run this what we find is that y has. Now become an object with length equal to 10 and if you look at it look at what it is printed out it first the first element was one then the character b then the function a to a + 1 and then just commas one after the other right with nothing in them right.

So, basically saying in other words that yes there are there is data there but none of it basically has any value. They are all undefined or null what exactly is it that depends on how you iterate through it and what you are trying to get the value out of it. Basically if you try directly accessing let us say y of 4 you will find that it is undefined because it is not being initialized.

(Video End: 20:14)