## Chapiter 9 DOM

1. **document.body.innerHTML** get the value(string) inside the body element, we can replace the content inside it.

**Method** is a function saved inside an object.

1. **Document.querySelector(‘’)** get any element in page and put it into JS

**Document.querySelector(‘element or .class’).innerHTML** get the content inside the chosen element or a chosen JS Class element. It gives JS full control of the page.

***Example:***

Update the content in some element:

Click a button the text change: **Subscribe** to **Subscribed (use innerText instead of innerHTML can eliminate the space impact)**

Create a function for the update content, and use this function on where the content was changed.

***Input*** *doesn’t have* ***innerHTML*** *we use* ***.value*** *to get the content*

If want to convert a string to num, we need to use the function **Number()**

1. **Onkeydown to operate the function for some special enter item such as Enter**



Eventlsit

Window object is content of the browser.

## Chapitre 10 CSS

To add or remove a class for an element, in order to get the different CSS in the same element with different content:

Element.**classList**.remove

Element.**classList**.add

Load an image by using **src=” ”**

Load JS file

Link to load the CSS file ***rel***=stylesheet ***href***=”CSS path”

We seperate the different language in different files.

## Chapiter 11 Arrays and Loops

1. **Arrays** is A list of values, the type of array is **object**

**Example** MyArray = [a,b,c] ;

**Array.isArryay** to check if a object is an array

**MyArray.length** to calculate the length

**Myarray.push()** add a value to the array

**Myarray.splice(index,number)** to remove the value from index and the number we want to remove

1. Loops run soc e code over and over

**While loop(non-standard loop) *eg:*** *get a random number at least 0.5*

While(**loop** **condition**){

**Loop body**

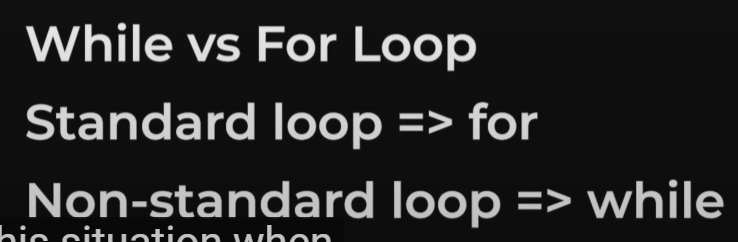
}

**For loop(standard loop)**

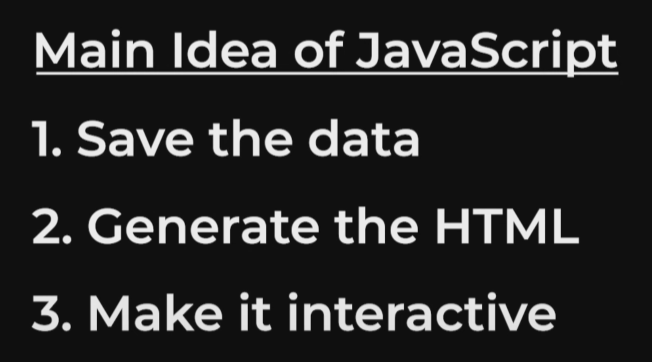
for(loop variable ; loop condition ;increment){

Loop body

}



Looping through an Array : go through each value of Array



Input date selector

Array is a reference, we can use **array.slice** to create an array, slice only copy the value

**Break** exit the loop earlier

**Continue** skip one iteration for the loop attention for while loop in increment step

**Return** statement will end the function immediately

## Chapiter 12 Advanced Features of functions

Function Name(**variable**){

Function code

}

Hoisting feature, we don’t need to worry the order of code, you can call the function before the function code.

We need to call the function,

**Name(variable1) ;**

function is also a value, we can define a function

**const** func1 = function name(variable){

funciton code

}

**Anonymous** function is a function without a name,

Const function1 = function(){

Function code

}

Buildin functions

**setTimeout**(function,time=how long we need to wait to call this function in ms)

**setInterval**(fun want to run in the future , a number in ms and keeping running each interval) we can use this function to **autoplay**

**clearInterval** to stop an interval function

**forEach() method**

[ ].foreach(function(value,index){

Function code

}) ;

The value is the value in the array, we can look throught the array

Foreach doesnot have **continue**

**ArrowFunction**

Const Functionname = (variable) =>{

Function code

} we save the function a const, if only has one paras, we can **remove the ()**

Const name = para =>{

code

}

**One line function**

Const oneline = () => value (donot need (), {} and return)

**AddEvent listener(event,function want to run)**

Let us run solme code when we interact with the element, it does the same thing as **Onclick (the event is click)**

We can add multiple eventlistener to one event

We can also add **removeEventListener**(event,function name want to remove, can’t use the code must be the function name, so we need to save the function in a const or variable first)

**Keydown : Event.key** to get the key we press

**QuerySelectorAll** select all the elements have the same class

Filter((value,index)=>{

Return boonlean true or false if true add the value in the new array if flase do not add the value in the new array

})

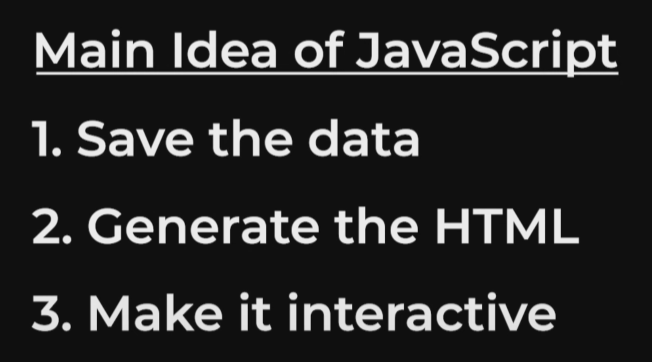
Map ((value,index)=>{

Return the value into a new array

)

## Chapiter 13 Amazon Project

**Basic Logic**



1. **Save the data, create the database** in array, in the array, each value use the object to save more parameters.
2. **Generate HTML**, **get the value from the database**, we want to use a loop, such as **foreach**, in the foreach function we generate the HTML, the same parameter can have the same **class** in order to have the same **CSS** and the same **JS** actions, so we need to use the **${object.value}** to get the different value for the different object saved in the **Array**
3. After generating the HTML code, **save the value to HTML** we need to **combine** this HTML together(create a variable to combine together by using **accumulator pattern**)
4. Put it on the web page by using **DOM,show the data on page,** find the container **Class** and **documlent.querySelector(‘.js’).innerHTML= accumulator pattern**
5. When we update the database, for each can update the HTML automatically
6. We can use **mutliple JS** document to save data, one for save the data, one for loop the date to generate HTML, and the order of scripts is matters, they run in order. It is important to separate the code in different file, each file just focus on one thing, such as product, cart, checkout etc
7. Make the page interactive. **Add event listener to the button**

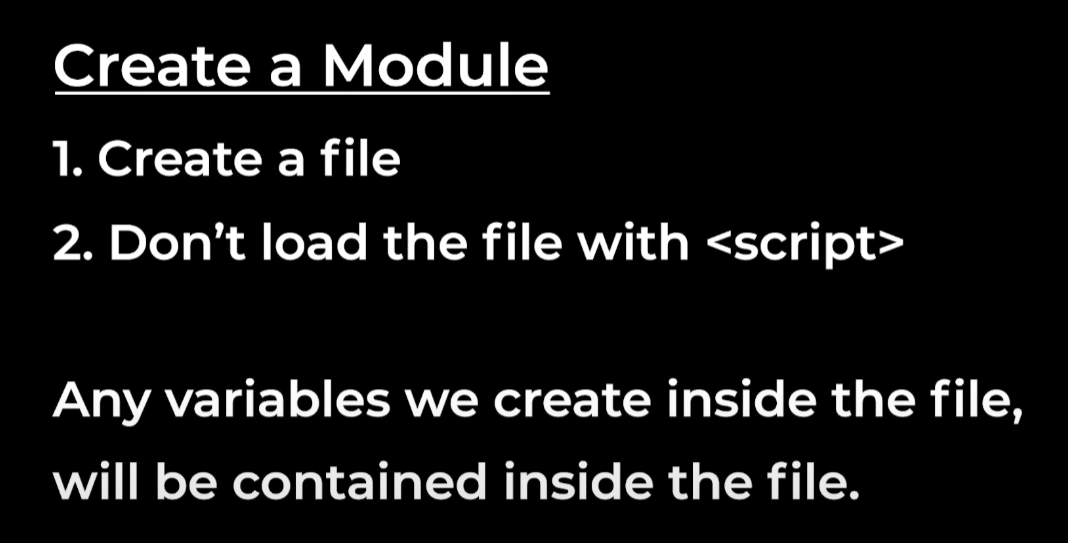
**Data Attribute,** allow us to attach any info to an element, is a HTML attribute, **data-*anyname*(kebab-case)**

Then use **dataset.*Anyname*** to get the data attribute value, the ***anyname*** here will change from the kebab case to the kamel case.

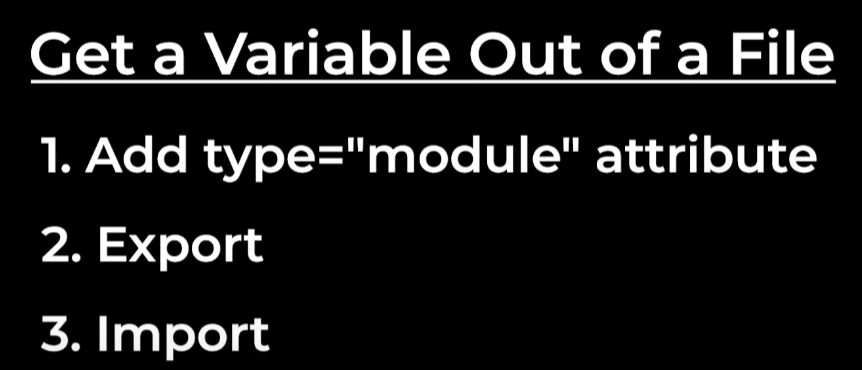
1. We add the product name to the button by using **data attribute**
2. We get the product name when we click the button by using **dataset**
3. We push the product name we get to the cart array by using **cart.push()**

## Chapiter 14 Module

To avoid the **naming** conflict. Modules contain the variable inside the file,



How to use module to get the variable out of the file,



1. Add the type= module at script where you want to get the variable
2. Add export before the variable
3. Add import in the JS and the variable name: **import {variable as Variable name in this file} from ‘file path**
4. We do not need to worry the order of the file, as we import the variable into the JS file

A link **herf** to open a new page when we click the button

1. **Dom.remove** to remvoe the container

Save the date in **local storage** can only save **strings**(we need to convert the array to string)

localStorage.setItem(‘name’,**JSON.stringify**(‘name’));

get the date from local storage(we need to convert the string back to array)

**JSON.parse**(localStorage.getItem(‘name’))

## Chapiter 15 External Library

In script, we give the **external url** in src,

1. we can share code, save time
2. avoid duplicating work.

DayJS in external library, then search **dayjs documentation** to check how to use dayjs

<https://day.js.org/docs/en/manipulate/add>

1. a = dayjs() to get the date
2. The expected days will be b= **a.add(**number of time want to add**, ‘days’)**
3. Display **b .format(‘dddd’)** can find in the table the expression

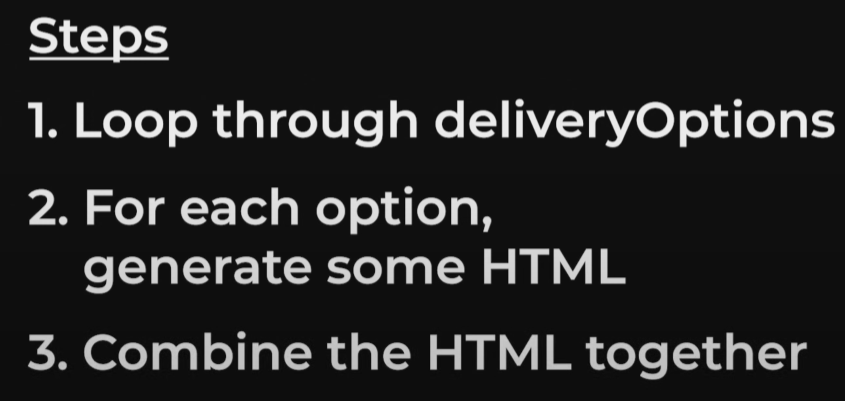
Use module to avoid the name conflict, **ESM Version = EcmaScript Module**, when import the function use the **url,** don’t need to add in script

**Default Export**

1. Another way of exporting
2. We can only export 1 thing from the file, we don’t need to add { }, each file can only have 1 default export

For delivery part

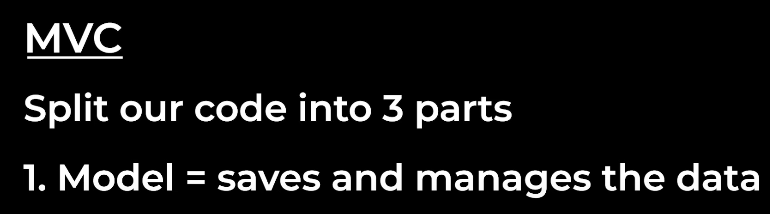
1. **Save the data**
   1. **DeliveryOptions array** save all the data as an **object**, include **id**, **deliveryDays** and **priceCents,** at the same time, add a **deliveryoptionID** in cart to connext the data.
2. Generate HTML



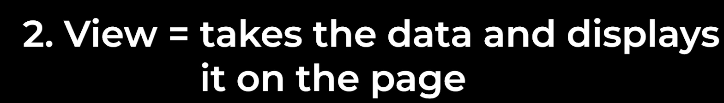
* 1. Import the **deliveryOptions** data to the checkout file to **get the data**
  2. Use foreach to loop the **deliveryOptions**, and generate some HTML, use dayjs to get the deliveryDate, also add price use if statement to verify the free shipping
  3. Create a blank HTML and add the content to HTML, insert the HTML into the page(we don’t use DOM as there is no event listener)
  4. In the input element, add an attribute **Checked,** find the cart **deliveryOptionId** and checked it

When we chose the option, we need to update the database, include the cart.deliveryoption; the radio check display and the cart item display

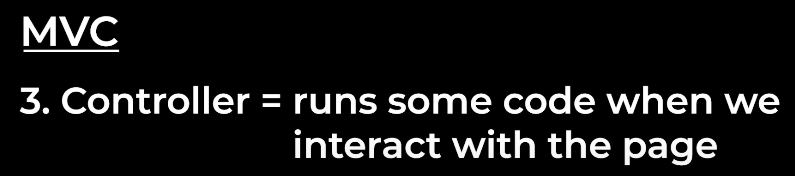
1. First, update the delivery option in the cart, we need to create a function to help us find which product have changed(use **productId**) and change the **deliveryoptionId** for this product.
2. Run the function by using the **addEventListener**, by using the dataset attribute to get the **productid** and checked **deliveryoptionId**
3. Update the page, we can regenerate all the HTML- **MVC Model View Controller**

****

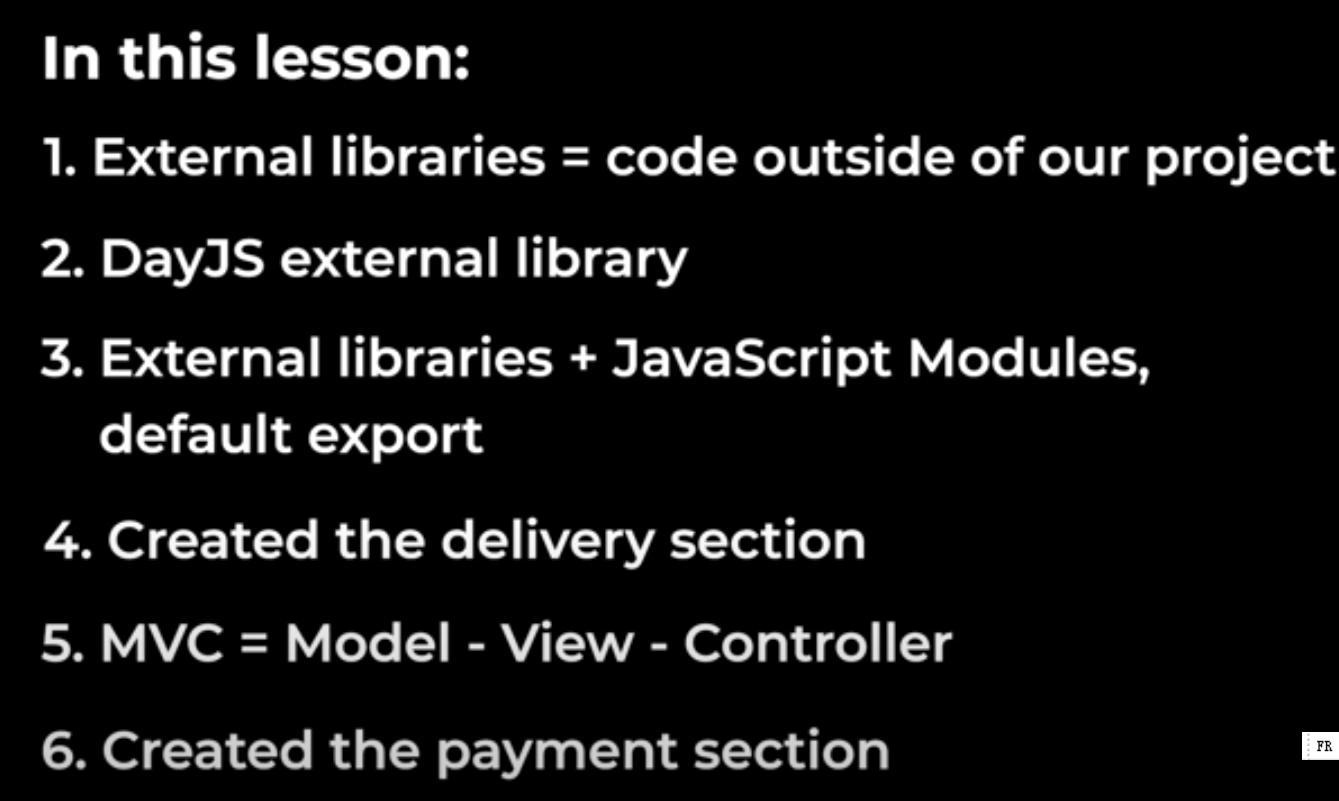
**The cart data**

****

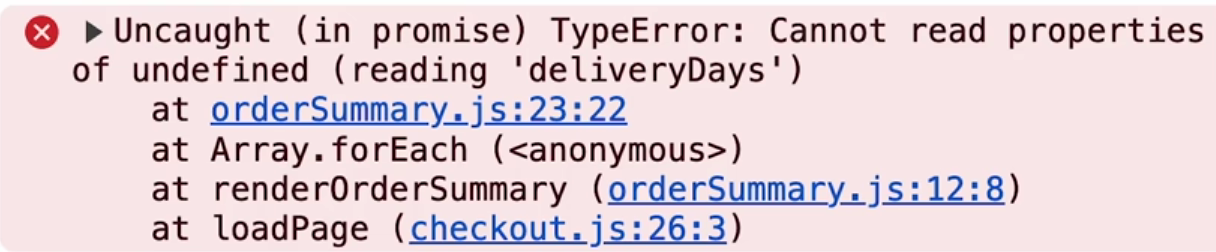
**The HTML**

****

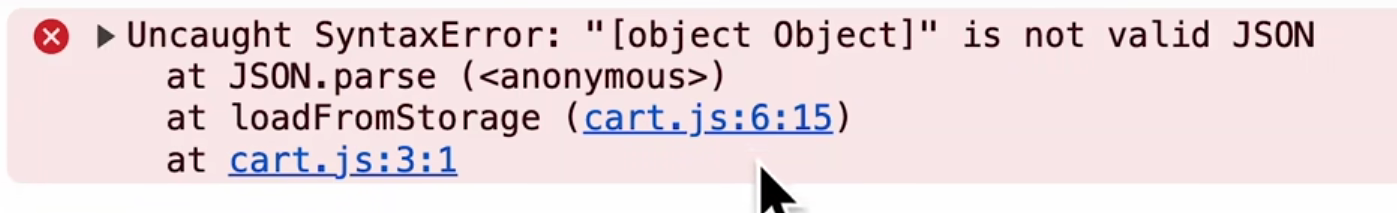
**The Event Listener, update the date**

****

## Chapiter 16 Test

****

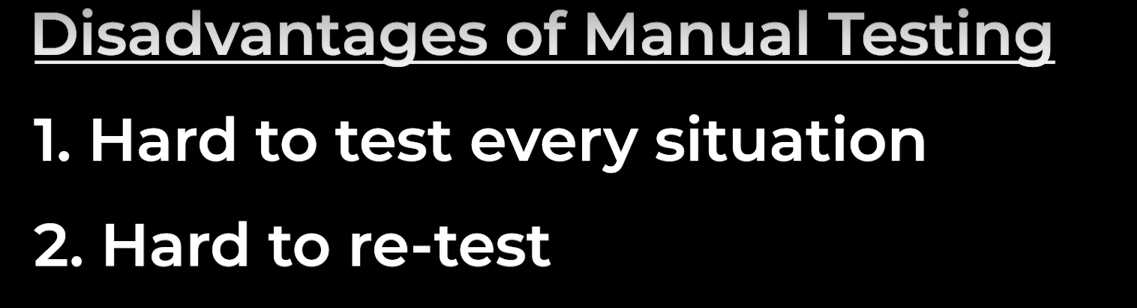
**localstorage.clear()** to clear the local storage.



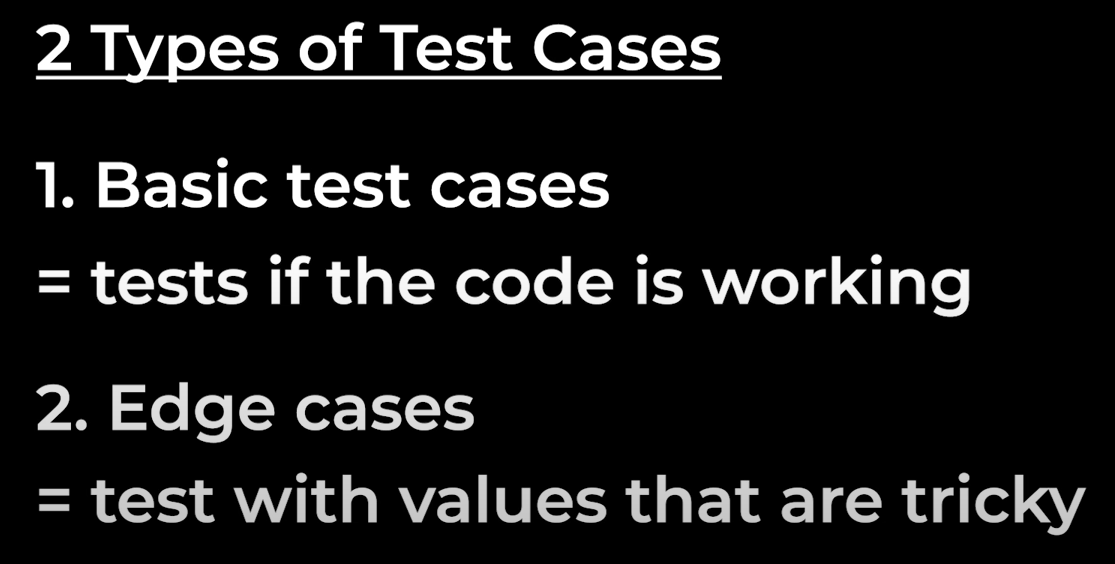
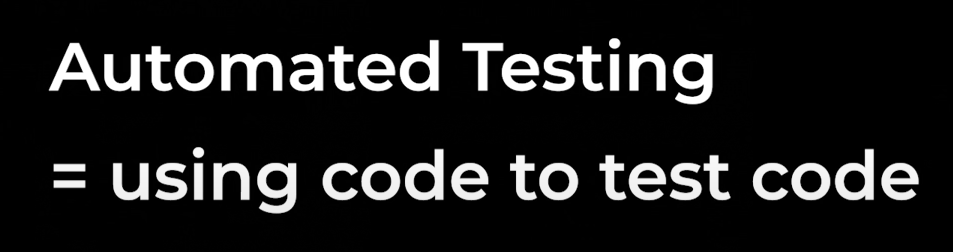
Need to do the **JSON.stringify()** before saving to local storage. Then clear the local storage and refresh the page.

**Test :**

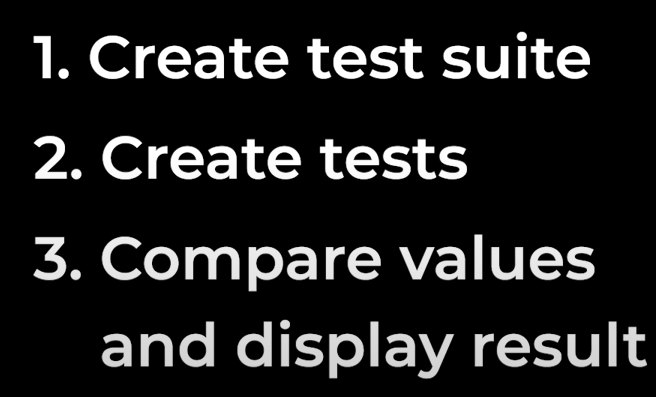
Manual testing try the code, difficult to try all the situation.

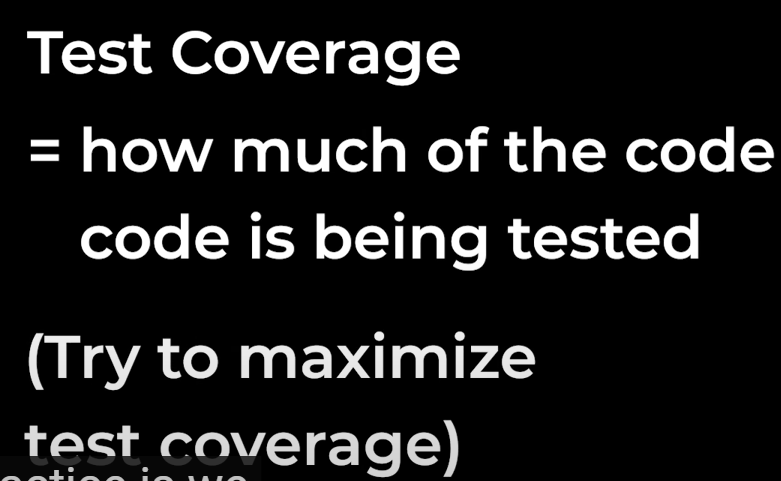


Automated Testing : using code to test code.



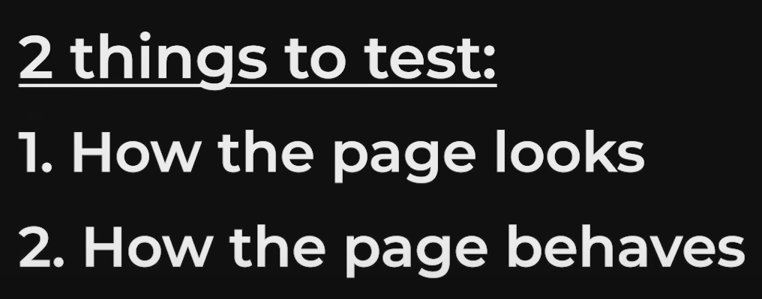
Testing framework

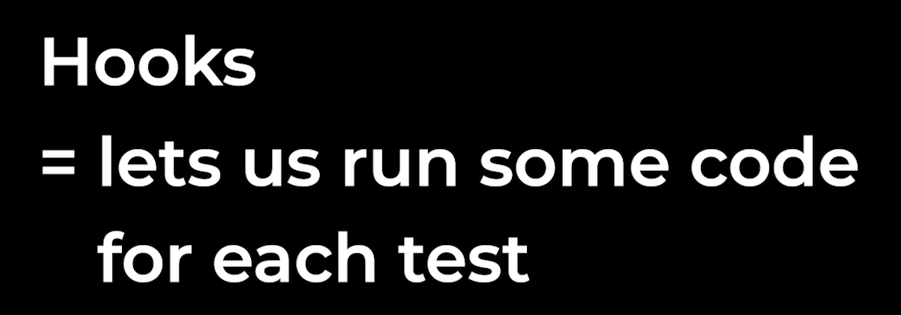




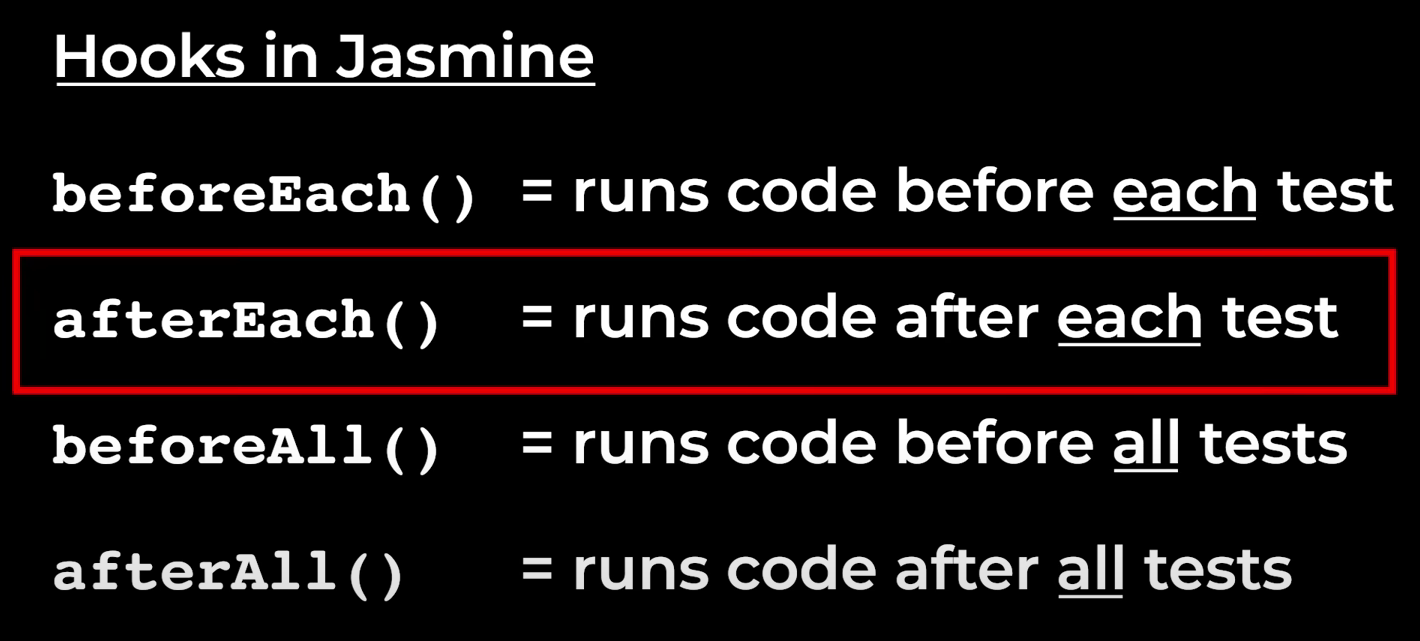
Test the add to cart function; local storage will impact the results.

Test the page

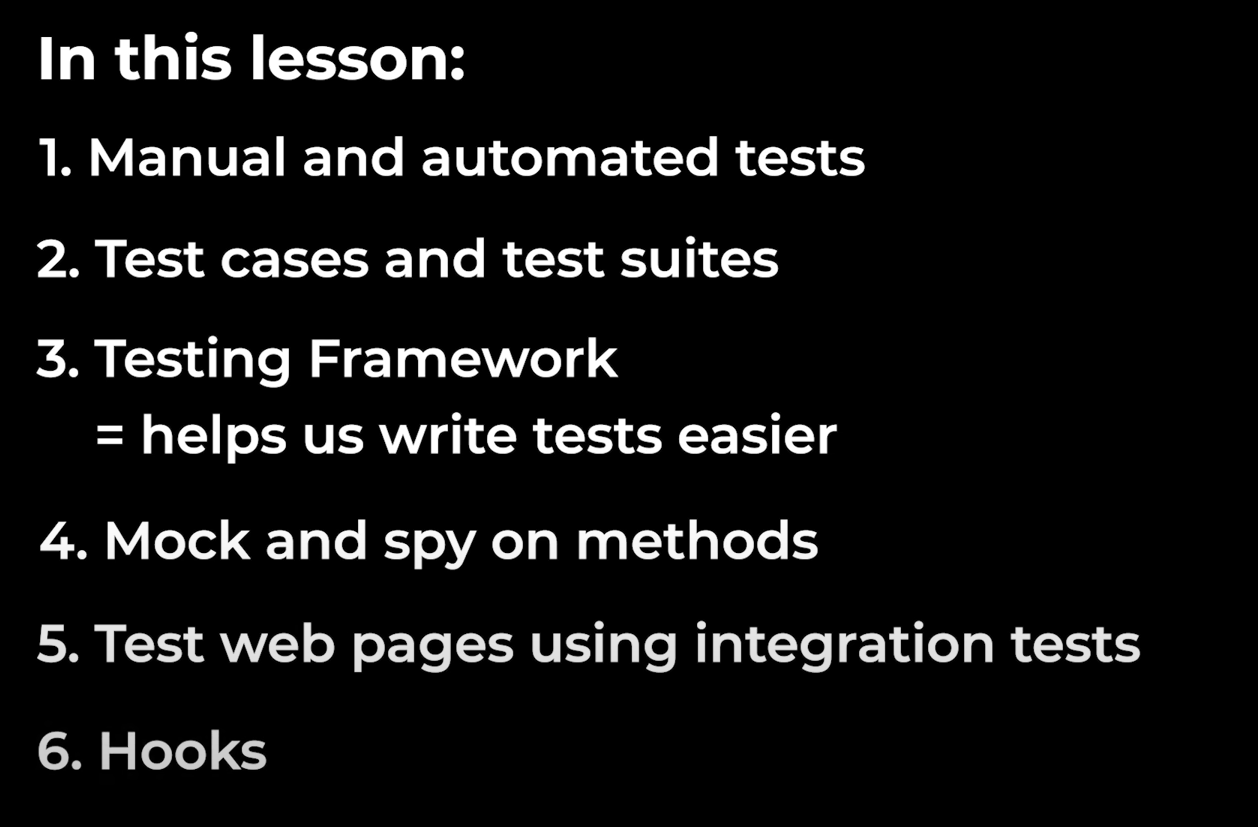




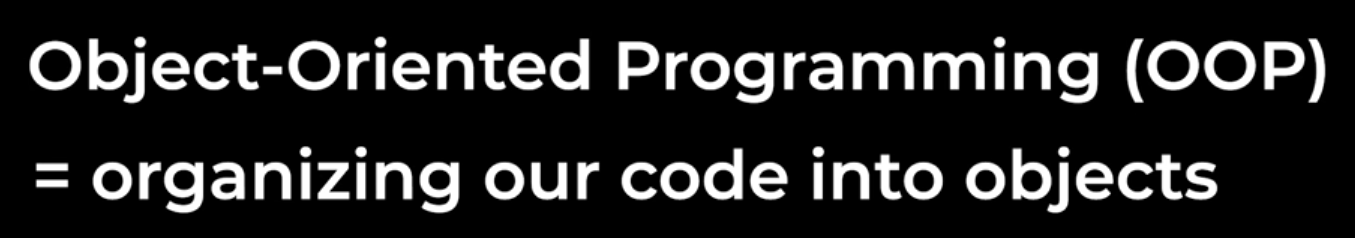
Share the code between the test





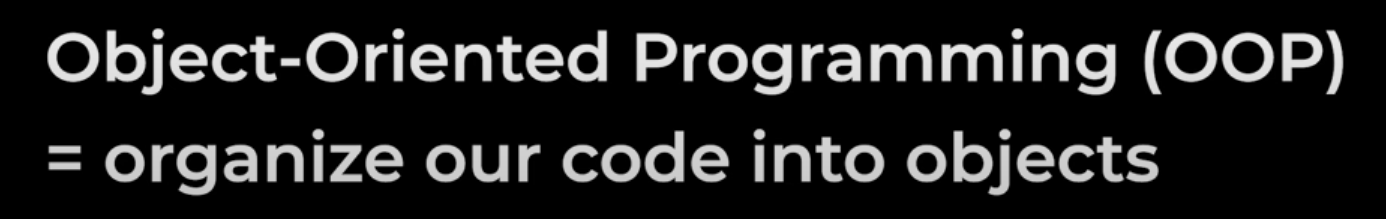


# Chapiter 17 Object Oriented Programming (OOP)

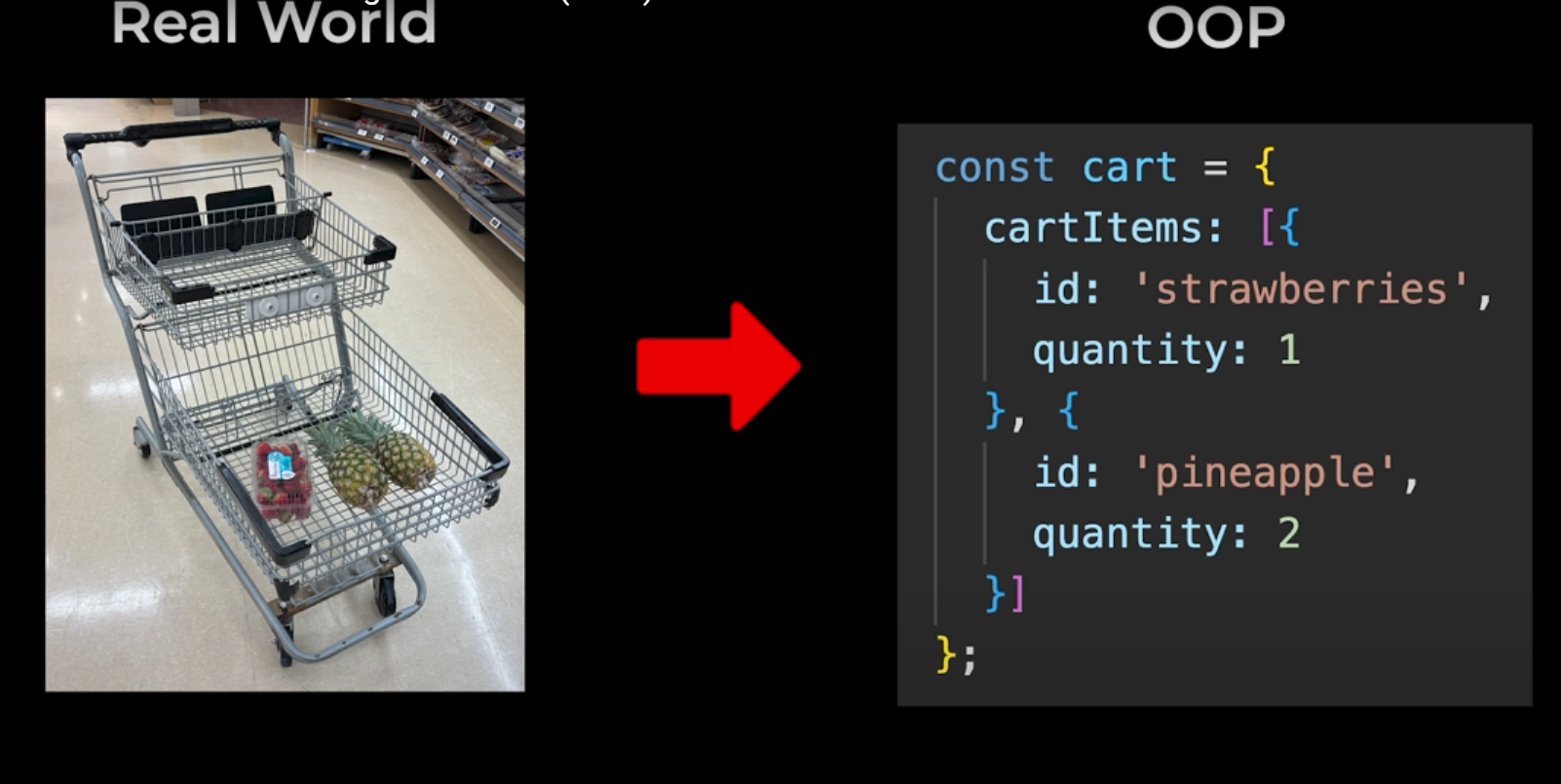


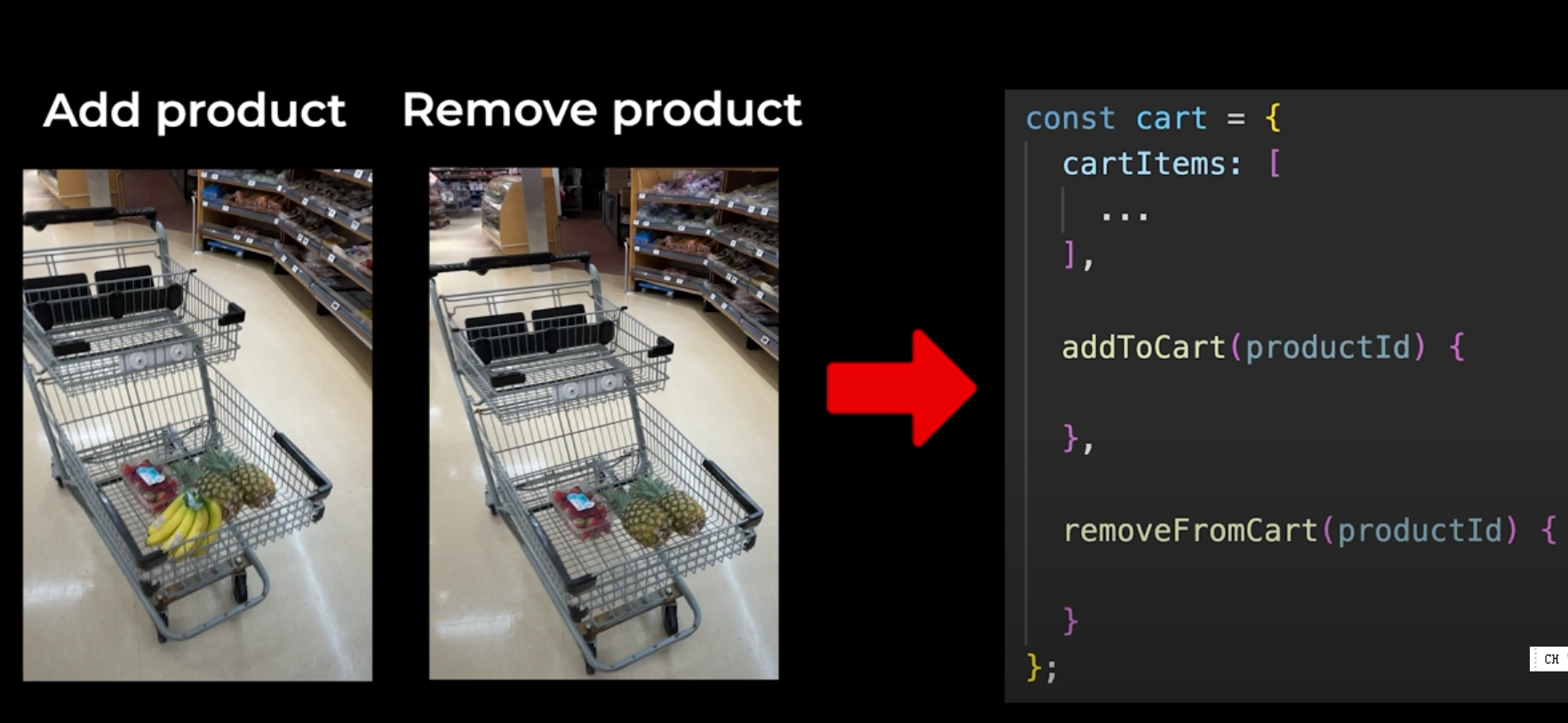


Separate the data and the function organize the code into separate functions.

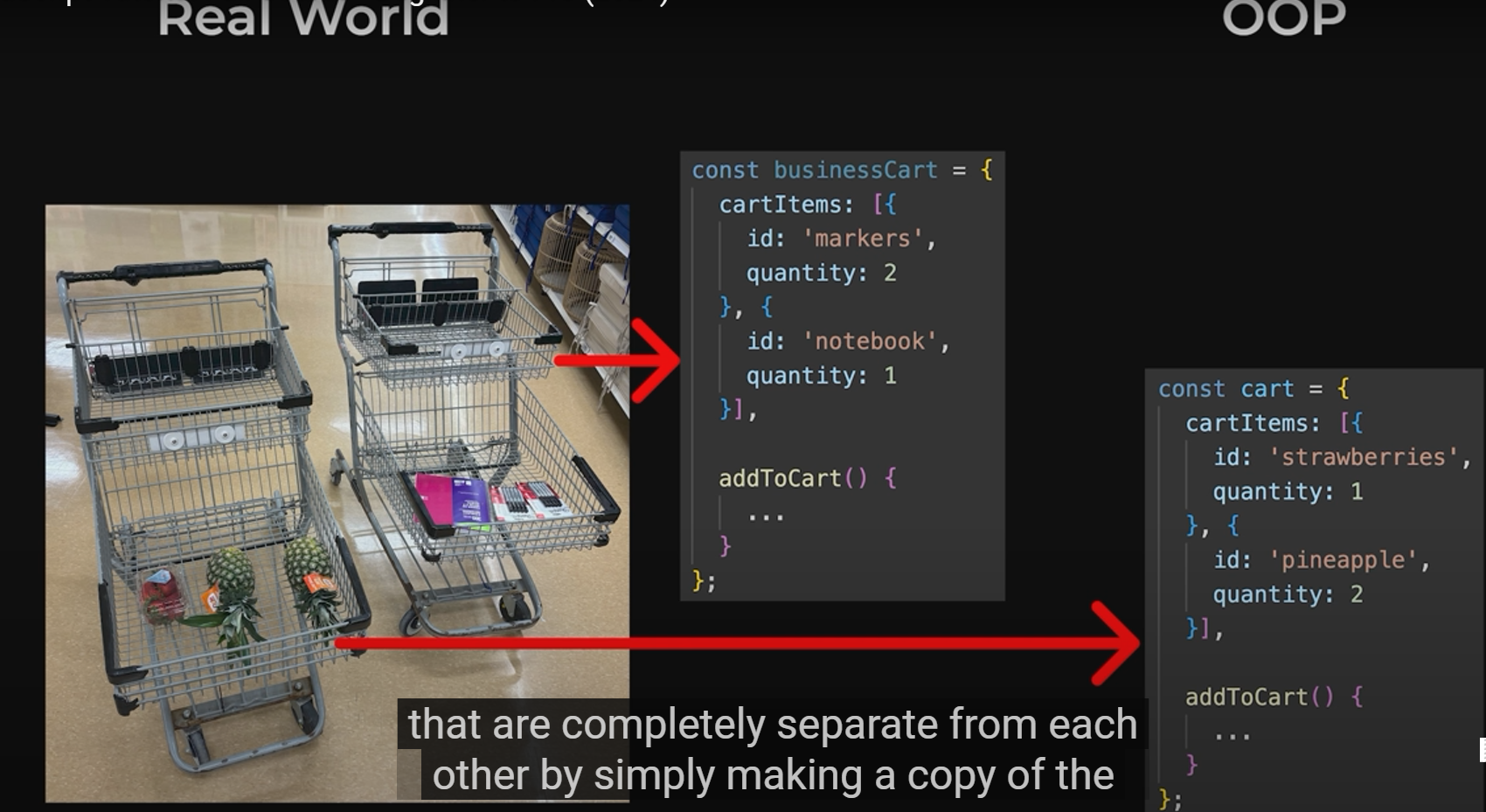


We put everything together into an object. Tries to represent the real world

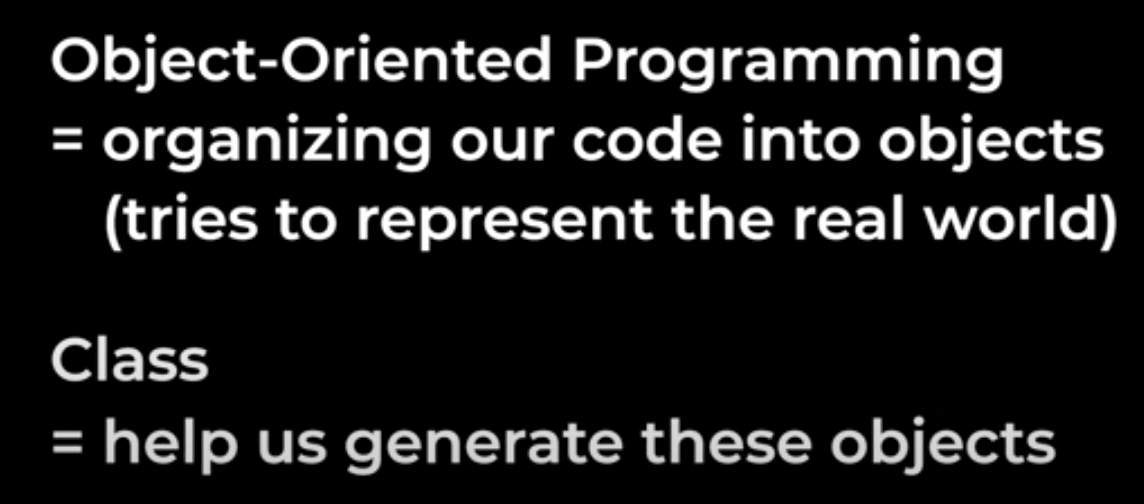


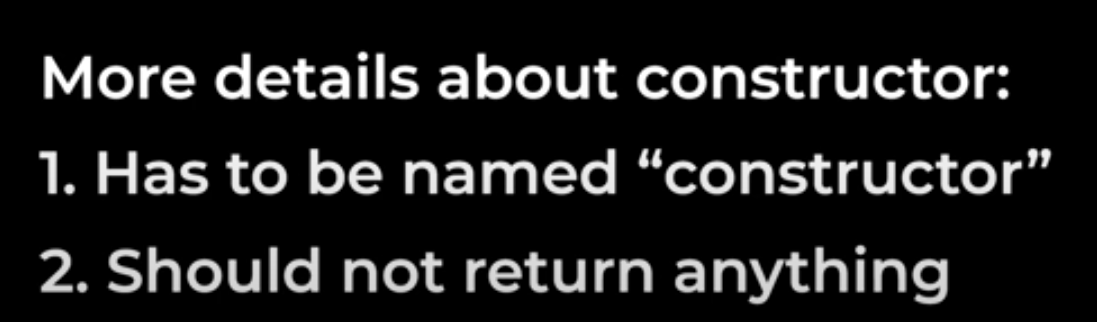


Easy to create multiple object



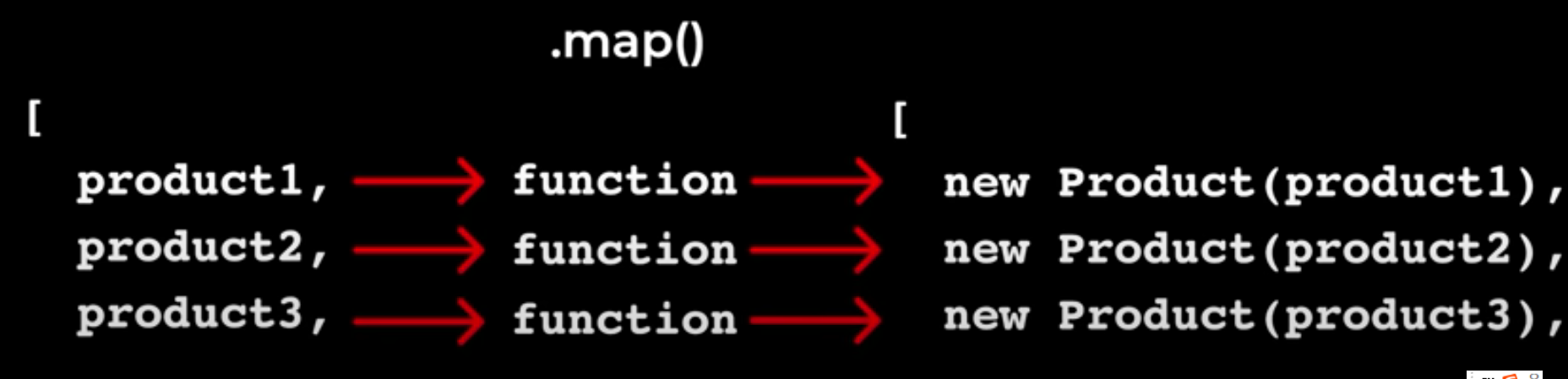
Class is a basic object generator,

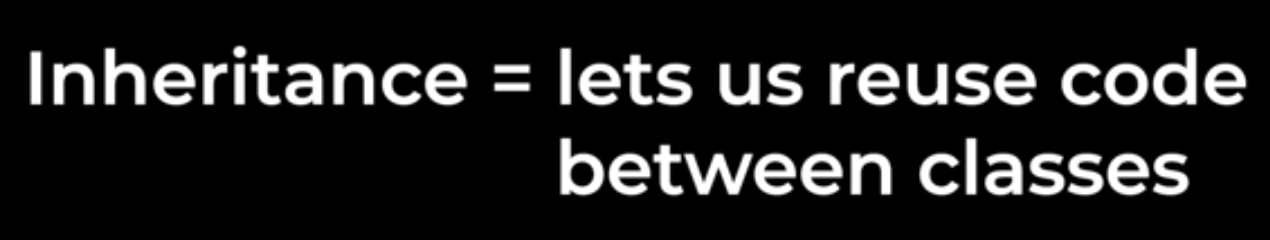




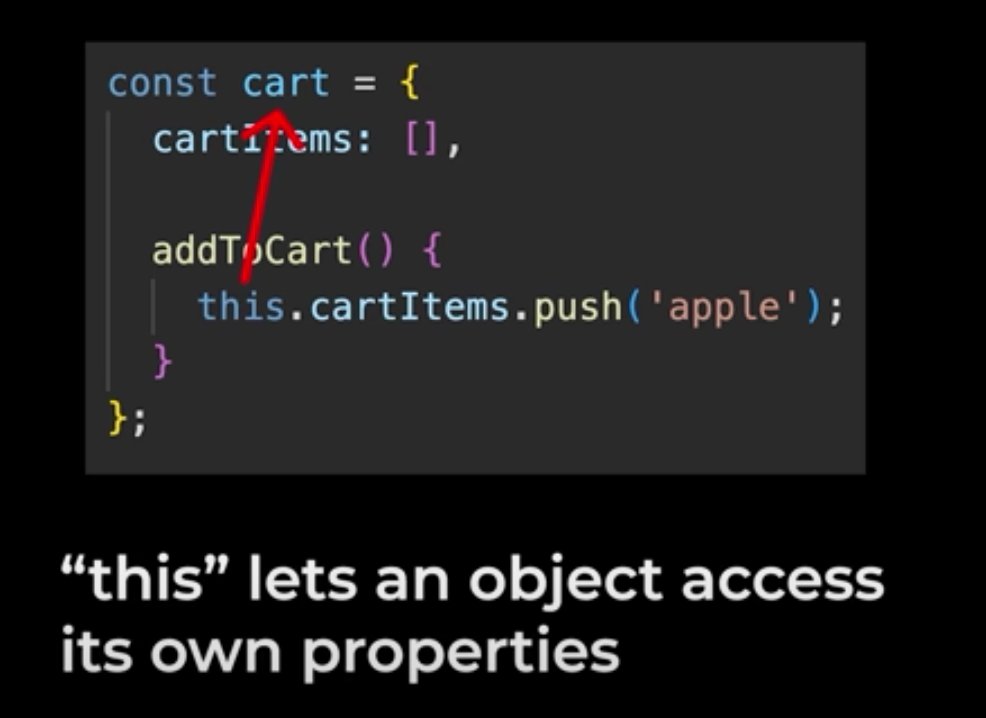
Private property, **add # before the parameter**, the para will be only used inside the class.

**Converting an object into a class,**

**Inheritance, get all the methods and the property of another class,**

****

Build in class , Date(),generate the current date. But in Dayjs it has more external features



This appoints an object.



