JavaScript (.js)

5 Primitive Datatypes: Numbers, String, boolean, null and undefined.

1. Numbers: 1/5 = 0.2 (get float number)
2. Strings: single or double quotes ok.
   concatenation with "+".
   Escape characters start with "\"
   String's length: str.length.
   Access individual characters using [] and an index: "hello"[0] → "h"
   str.indexOf("") : -1, not found

To define an variable: var name = value

3. Boolean: true / false.
4. undefined: variables that are declared but not initialized are undefined.
5. null: explicitly nothing.

Comment: //, (same as Java).

JavaScript Built-in methods

1. clear().
2. alert( ): pop up a message.
3. console.log( ): print message out in console.
4. prompt( ): ask for input from users.

Connect html with JavaScript:

Add javascript in the head/body using <script type="text/javascript" src=".js"></script>.

Comparison Operators

>=, <=, <, >.

==, !=: performs type coercion, while "===" does not. (type coercion ⇓ just compare values.)

===: Equal value and type.

!==: not equal value or equal type.

NaN == NaN (false, NaN is not comparable)

values that are not actually true or false are still inherently "truthy" or "falsey" when evaluated in a boolean context.

Falsy values: false, 0, "", null, undefined, NaN.

Everything else is truthy.

Condition = if, else if, else.

Cast type: Number("4") → 4.

while ( ) { }. same as java.

for (init; condition; step) { }.

JavaScript function.

```
function doSomething() {
    return ...;
}
```

function is a keyword to define a function.

To add argument, simply add names to function: function doSomething(arg1, arg2,

To replace substring in a string: str.replace(target, newValue).
or str.replace(/target/g, newValue).
or str.replace(/target/gi, newValue).
↓ case insensitive.

↑ just function name: doSomething

We can pass functions to other functions.
i.e: setInterval(doSomething, 1000);

JavaScript Array

Array are indexed starting at 0. arr[index].

var arr = [1, 2, 3, 4];

We can initialize an empty array two ways: { var arr = []; / var arr = new Array(); } equal.

The same array can hold any type of data: arr = [49, "hi", true, null];

length: arr.length

Arrays method:

① arr.push(...): add to the end of an array.
② arr.pop(): remove the last item in an array and return it.
③ arr.unshift(...): add to the front of an array.
④ arr.shift(): remove the first item in an array.

⑥. arr.slice(id1 , id2); to copy parts of an array.
(id1 → inclusive, id2 → exclusive)

arr.slice( ) // copy the whole array.

⑦ arr.splice( ) : add/Removes items to/from an array, and returns the removed items.
arr.splice(index, how_many, items1, ..., items n)
(index → start Index)

Array Iteration:

1) for( )
2) arr.forEach(someFunction) → someFunction(arg1), someFunction(arg1, id)
(arg1 → element; id → index)

→ will pass each element in arr to the function.

→ this is callback function, which is invoked with at most three arguments:
someFunction(element, idx, arr).

Arr.forEach( ).

We can define our own methods of array by using:

Array.prototype.myMethod = function (func){ }

(myMethod ⇓ can be any name, we use keyword "this" to indicate current array.)

After that, we can use myMethod directly: arr.myMethod( ) ...

JavaScript : Object.

```
var person = {
    attr1 : "Name",
    attr2 : num,
    attr3 : "name"
};
```

To retrieve information, we can do: person.attr1 or person["attr1"]
We can update info by the same way.

Creating Objects: ① var person = { };
then : person.attr1 = "...", ...

② var person = {
attr1 : "...",
...
}

③ var person = new Object( );
then person.attr1 = "...";

Nested objects and arrays : an array of objects or an object contains an arra

We can also add a function to an object :

i.e : var obj = {
add : function (x,y) {
return x+y;
}
}

or : var obj = { };
obj.add = function ( ) { ... };

keyword : "this" can be used to indicate specific object