**ASSIGNMENT WEEK 9**

**Part 1: Introduction**

**Part 2: Operator, String Templates and Type Conversion and Coercion**

**2.1, What is Operator ?**

- an operator basically allows us to transform value or combine multiple values and do all kinds of work with values.

- many types of Operator: mathematical operators, logical operators, assignment operators, etc.

**2.2, What are the different types of operators that you have learnt in video 14 ?**

a. Arithmetic operators (mathematical operators)

- plus operator: +

- minus operator: -

- multiplication operator: \*

- division operator: /

b. exponentiation operator:

Ex: 2 to the power of 3: 2 \*\* 3

c. Use plus operator to join strings:

- create 2 different variables a, b

- then we can concatenate these 2 strings using plus

Ex:

const a = ‘linh’;

const b = ‘xinh’;

console.log(a + ‘ ‘ + b)

=> linh xinh

d. typeof operator:

- to define the type of a specific variable

Ex:

const age = 21;

console.log(typeof age);

=>  number

e. Assignment operator:

- the most straightforward operator is the equal sign: “=”

Ex:

let x = 10 + 5;

In the above line of code, we have 2 operators: the plus and the equal. In this case, x will be assigned 15 because the plus operator is executed before the assignment operator.

-         “+=”

Ex: x += 10 ( means x=x+10)

-         “\*=”

-         “-=”

-         “/=”

-         “++”

-         “- -“

Ex: x++ (means x=x+1)

f. Comparison operators: >, <, =>, =<

- to produce Boolean value

**2.3, What is Type Conversion ? How does it work ?**

- Definition: Type Conversion is the process when we manually convert data from one to another. (explicit)

- How it works:

Ex:

const year = “1999”;

console.log(year + 18);

=> 199918 (because of the type of year is string, so we need to convert data year from string to number so that we can concatenate 1999 and 18 in a mathematical way)

SOLVE:

const year = “1999”;

console.log(Number(year + 18));

=> 2017

(!) Cannot convert sth to null or undefined

**2.4, What is Coercion:**

* Coercion happens whenever an operator is dealing with 2 values that have different value=> then, JS will automatically converts types behind the scene for us (happens implicitly, completely hidden from us)

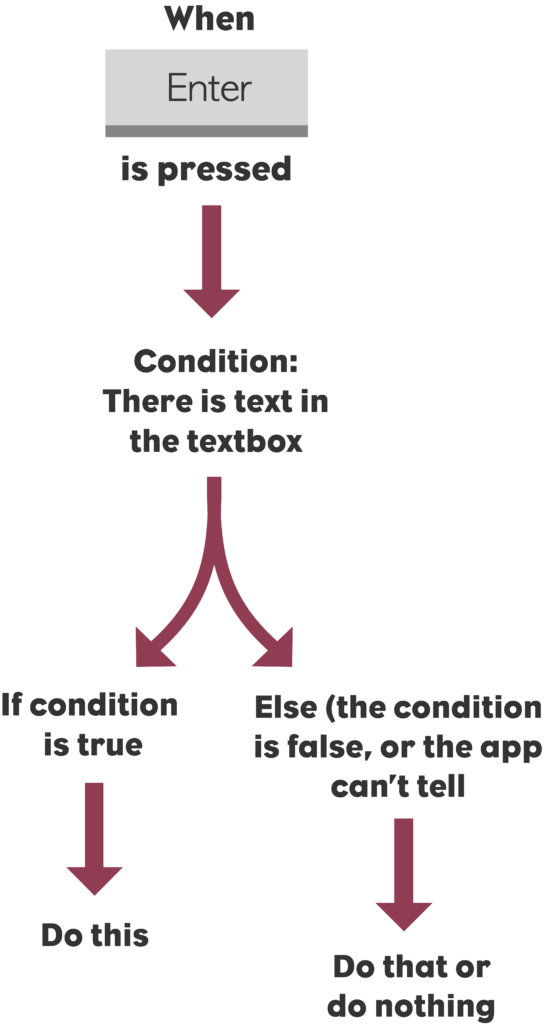
**3. Part 3: Conditional Statements**

**\*, Personal Research:**

**3.0, What is a Conditional Statement?**

**Conditionals Statements -**  A way for computers to make decisions based on conditions.

Conditional statements always have an if part, which tells the app what to do when the condition is true. Conditional statements also usually have an else part, which tells the app what to do when the condition is false. If you leave out the else part then your app will do nothing when the condition is false. Now your code for your enter button would look something like this:



**\*, Knowledge Review:**

**3.1, What are the differences between “==” and “===”?**

The ‘==’ operator tests for abstract equality. It **does** the necessary type conversions before doing the equality comparison. (loose equality)

The ‘===’ operator tests for strict equality. It will **not** do the type conversion hence if the two values are not of the same type, when compared, it will return false. (strict equality)

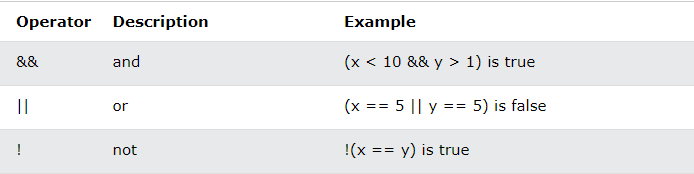
(!) As a general rule for clean code, avoid the loose equality operator “==” as much as you can

(!!) When comparing values, always use strict equality “===”

**3.2, Explain Logical Operators:**

Logical operators are used to determine the logic between variables or values.

There are 3 operators:



**3.3, What are the 3 ways that you have learnt to perform conditional statements in JavaScript?**

C1: If/else statement

* if: to specify a block of code to be executed, if a specified condition is true
* else: to specify a block of code to be executed, if the same condition is false
* else if: to specify a new condition to test, if the first condition is false

Ex:

const day = 'monday';

if (day === 'monday') {

   console.log('Plan course structure');

   console.log('Go to coding meetup');

} else if (day === 'tuesday') {

   console.log('Prepare theory videos');

} else if (day === 'wednesday' || day ==='thurdays') {

   console.log('Write code examples');

} else if (day === 'friday') {

   console.log('Record videos');

} else if (day === 'saturday' || day === 'sunday') {

   console.log('Enjoy the weekend :D');

} else {

   console.log('Not a valid day!');

}

C2: The Switch Statement:

* switch: to select one of many blocks of code to be executed

Ex:

const day = 'monday';

switch (day) {

   case 'monday':

       console.log('Plan course structure');

       console.log('Go to coding meetup');

       break;

   case 'tuesday':

       console.log('Prepare theory videos');

       break

   case 'wednesday':

   case 'thursday':

       console.log('Write code examples');

       break;

   case 'friday':

       console.log('Record videos');

       break;

   case 'saturday':

   case 'sunday':

       console.log('Enjoy the weekend :D');

       break

   default:

       console.log('Not a valid day!');

}

C3: The conditional (ternary) operator:

* the conditional (ternary) operator has 3 parts: a condition followed by a question mark (?), then an expression to execute if the condition is truthy followed by a colon (:); and finally the expression to execute if the condition is falsy.
* This operator is frequently used as an alternative to an if/ else statement.
* Ex:

console.log(`I like to drink ${age >= 18 ? 'wine 🍷' : 'water 💧'}`)

**Part 4: Personal research:**

1. **Differences between null and undefined (data types):**

**\*reference: https://flexiple.com/undefined-vs-null-javascript/**

* In Javascript, undefined means a variable **has been declared** but **has not yet been assigned a value**
* Null is an assignment value. It can be assigned to a variable as a presentation of no value
* So undefined and null are two distinct types: undefined is a type itself(undefined) while null is an object

**Diffences:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **NO** | **Name of differences** | **Explain** | | **Note** |
| **Undefined** | **Null** |
| 1 | Data types | Undefined | Object |  |
| 2 | In arithmetic operations | Will result in NaN (not a number) | Will be converted to 0 behind the screens | VD:  undefined + 1;  // NaN  Null + 1;  // 1 |
| 3 | In conditional logic | Will return false | Will return false |  |
| 4 | Comparing using equality operators | // comparing undefined and null  **undefined** == **null**;  //true  **undefined** === **null**;  //false | | Case 1: using “==”  Because both undefined and null are falsy by default so “==” return True  Case 2: using “===”  When we use strict quality “===” which checks both type and value, since undefined and null are of different types, the strict equality operator returns false |

1. **What is the differences between linking JS file at the head and at the body in a HTML file?**

– the main difference is the time that it runs

The </script> tag inside the <head> tag usually makes the html page loading slower. This is because the html parser is loaded first and as soon as it encounters </script> it has to load the JavaScript interpreter first and then after the script is loaded it has to return back to the html parsing. This wastes some time.

Instead putting it after the <body> tag first lets the parser to load the html content first and then start loading the script. Thus here there is only one switching b/w the parser and interpreter. Thus, this approach is better.