

WEEK 1 – KICK OFF JAVASCRIPT PROGRAMMING

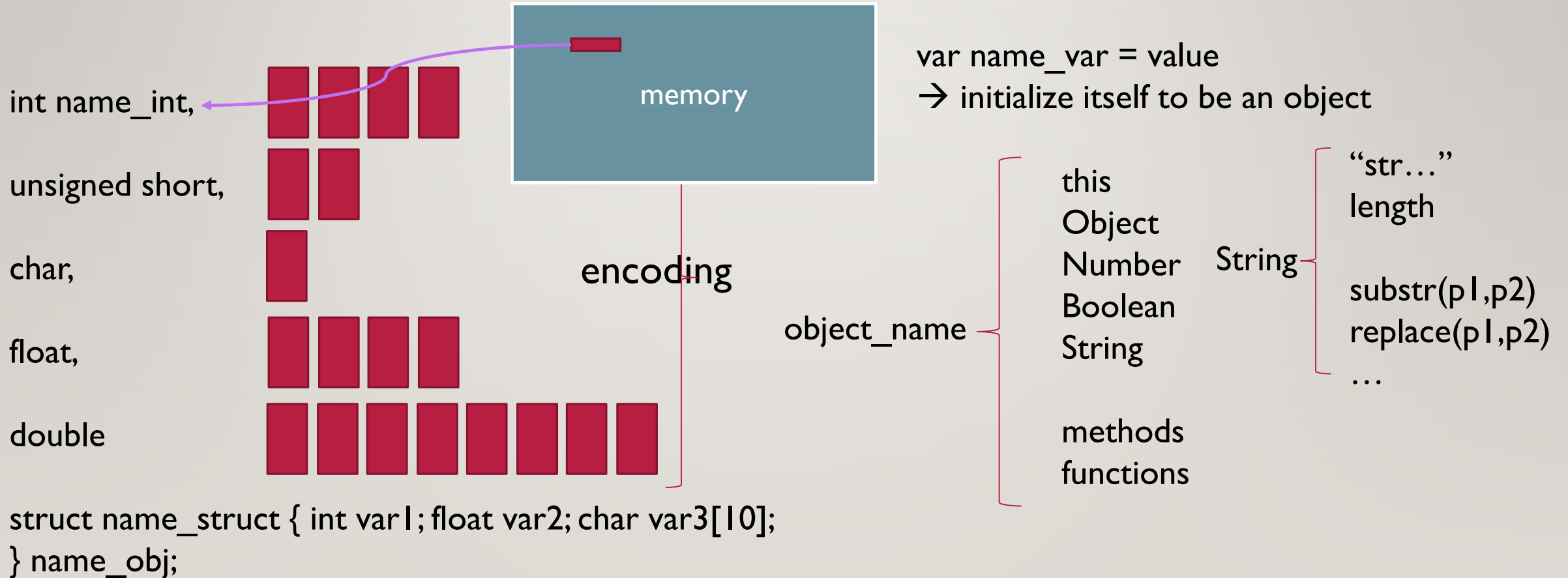
WEN-BIN JIAN

DEPARTMENT OF ELECTROPHYSICS, NATIONAL CHIAO TUNG UNIVERSITY

OUTLINE

- 1. The Number Type – Integer & Float**
- 2. Number Conversion**
- 3. The String & The Object**
- 4. Operators, Bitwise Operators, Arithmetic Operators**
- 5. Assignment & Comma Operators**
6. Relational, Equality, Logic Operators
7. If, Do-While, and While Statements
8. Switch & With Statements
9. While Statements
10. For and For-In Statements

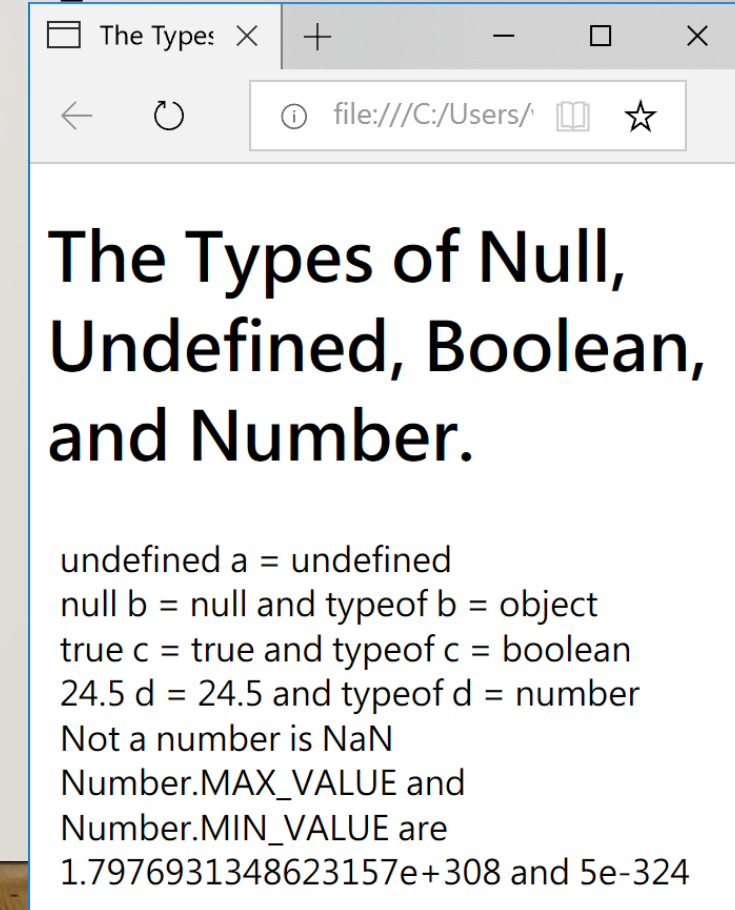
CONVENTIONAL VARIABLES & JAVASCRIPT VARIABLES



TYPE OF VARIABLES – NUMBER

- The **null** type variable (an object) is different from the **undefined** one.
- The boolean type is claimed when you give values of **true** or **false**.
- The number type is claimed by a number:
 - 55 – decimal, **055** – octal, **0x55** – hexadecimal
 - **2.1** – float, **2.0** – automatically converted to integer 2
 - A built-in object Number, the maximum number is stored as **Number.MAX_VALUE**
 - NaN is a special numeric value called Not a Number.

IC_WI001.html



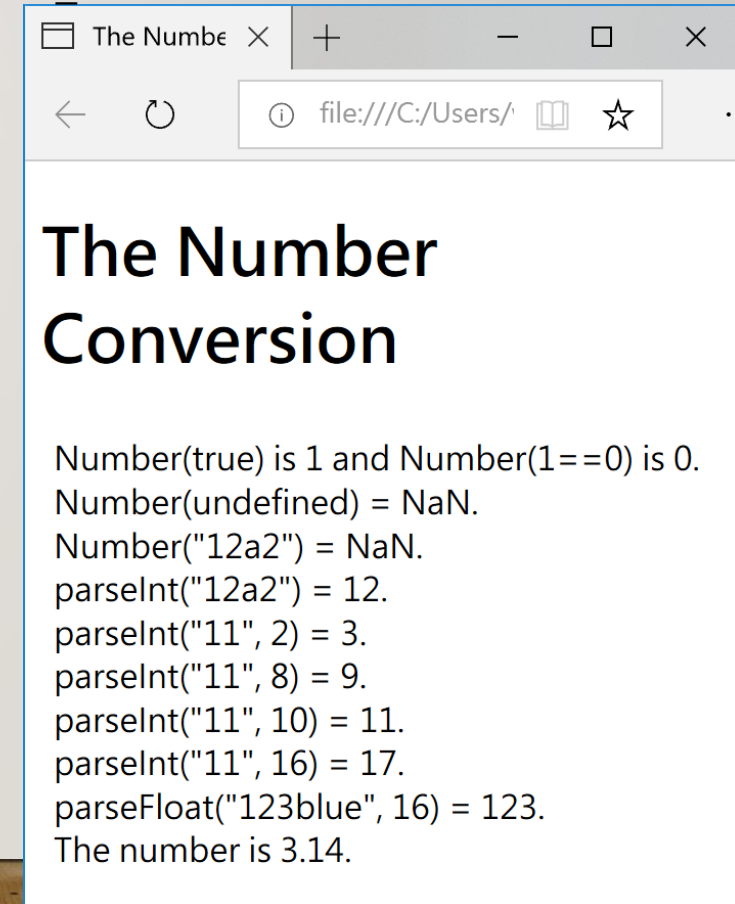
The Types of Null, Undefined, Boolean, and Number.

```
undefined a = undefined
null b = null and typeof b = object
true c = true and typeof c = boolean
24.5 d = 24.5 and typeof d = number
Not a number is NaN
Number.MAX_VALUE and
Number.MIN_VALUE are
1.7976931348623157e+308 and 5e-324
```

NUMBER CONVERSIONS

- Convert nonnumeric values to numbers: **Number()**.
 - true → 1, false → 0, null → 0
 - undefined → NaN
 - "123" → 123, "0xff" → 255, "12a2" → NaN, "a2" → 0
- Convert **string** to numbers: **parseInt()** & **parseFloat()**.
 - "12a2" → 12, **parseInt("10", 2);**
 - **parseFloat("1234blue");** → an integer of 1234
- Automatic conversions:
 - 10 + 2.1 → float
 - 3.2 * (21 + 3.0), (21+3.0) → integer, 3.2 * (21+3.0) → float
 - **var a = 3.14; var str = "The number is " + a;** → a is converted to string by **a.toString()**

IC_WI002.html



The Number Conversion

Number(true) is 1 and Number(1==0) is 0.
Number(undefined) = NaN.
Number("12a2") = NaN.
parseInt("12a2") = 12.
parseInt("11", 2) = 3.
parseInt("11", 8) = 9.
parseInt("11", 10) = 11.
parseInt("11", 16) = 17.
parseFloat("123blue", 16) = 123.
The number is 3.14.

THE INTEGER AND FLOAT IN BINARY FORM

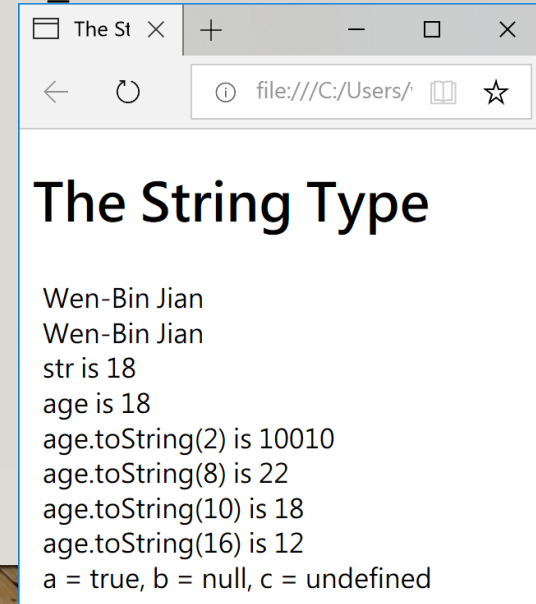
- In JavaScript, a **number** is stored using 64-bit float format.
- When doing bit operation, the number will be converted to **32-bit signed integer**. After the bitwise operation, the number will be converted back to 64-bit float format.
- In JavaScript, a number format follows the IEEE 754 specs (double precision).
- The float number format specifies 1 bit for sign, 11 bits for exponent, and 52 bits for mantissa (value).

0 011111111 0001 = $+2^{(1023-1023)} * (1 + 2^{(-52)})$
0 1000000000 100 = $+2^{(1024-1023)} * (1 + 2^{(-1)}) = 3$
 $6 = 4+2=110 \rightarrow 2^2 * (1+2^{-1})$

TYPES OF VARIABLES - STRING

- The string type is specified when you initialize the variable value by “a string”.
 - `var givenName = “Wen-Bin”;`
 - `var lastName = “Jian”;`
- Several special characters are used to control line format (conventions inherited from the DOS commands, cannot be used in html, use `window.alert()` function to show it)
 - `\n`: new line, `\t`: tab, `\b`: backspace, `\r`: carriage return, `\f`: form feed, `\\`: `\`, `\'`: `'`, `\"`: `“`, `\xnn`: hexadecimal, `\unnnn`: unicode
- The string operator: `var fullname = givenName + lastName;`
- Converting to a string: `var age = 18; var age_string = age.toString();`
- `age.toString(2)`, `age.toString(8)`, `age.toString(10)`, `age.toString(16)`
- `var a = true, b = null, c; String(a)` is “true”, `String(b)` is “null”, `String(c)` is “undefined”

IC_WI003.html



```
Wen-Bin Jian
Wen-Bin Jian
str is 18
age is 18
age.toString(2) is 10010
age.toString(8) is 22
age.toString(10) is 18
age.toString(16) is 12
a = true, b = null, c = undefined
```

TYPES OF VARIABLES – OBJECT, NULL

- The object type is initialized by a new operator.
 - `var obj = new Object();`
- The Object type is the base form from which all other objects are derived.
- The basic form of the Object type has the following properties and methods:
 - Constructor
 - `hasOwnProperty(propertyName)`, `isPrototypeOf(object)`, `propertyIsEnumerable(propertyName)`
 - `toLocaleString()`, `toString()`
 - `valueOf()` – returns a string, number, or Boolean value of the object

TYPES OF VARIABLES

- Basic Types of Variables:
 - undefined – Undefined
 - null – Object
 - boolean, true or false – Boolean
 - number, int or double – Number
 - string, “content of string” - String

UNARY OPERATORS

- Unary operators of ++ and --:
 - `var age = 18; ++age; // age is 19, age++; // age is 20`
 - `var age = 18; --age; // age is 17`
 - `var age = 18; var parentAge = ++age + 18; // age is 19, parentAge is 37`
 - `var age = 18; var parentAge = age++ + 18; // age is 19, parentAge is 36`
- Unary operators of + and -:
 - `var age = "2.1"; age = +age; // change to number 2.1`
 - `var fnc = {valueOf: function(){ return -2.1; } }; fnc = +fnc; // change to number -2.1`
 - `var age = "MyAge"; age = -age; // change to NaN`

IC_WI004.html

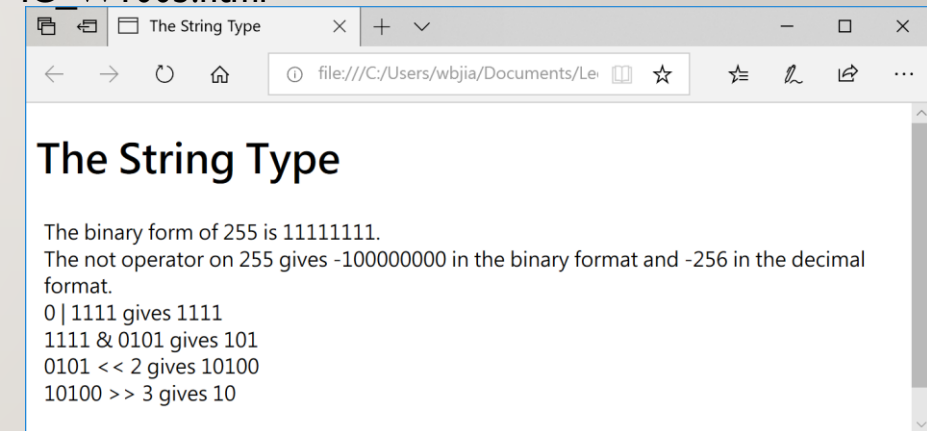
The String Type

age is 18 and age++ is 18 and ++age is 20
age is 21.5 and age++ is 21.5 and ++age is 23.5
change object function to value: fnc = +fnc + 3:
0.8999999999999999
a = 'MyAge', -a = NaN

BITWISE OPERATORS (\neq LOGICAL OPERATORS)

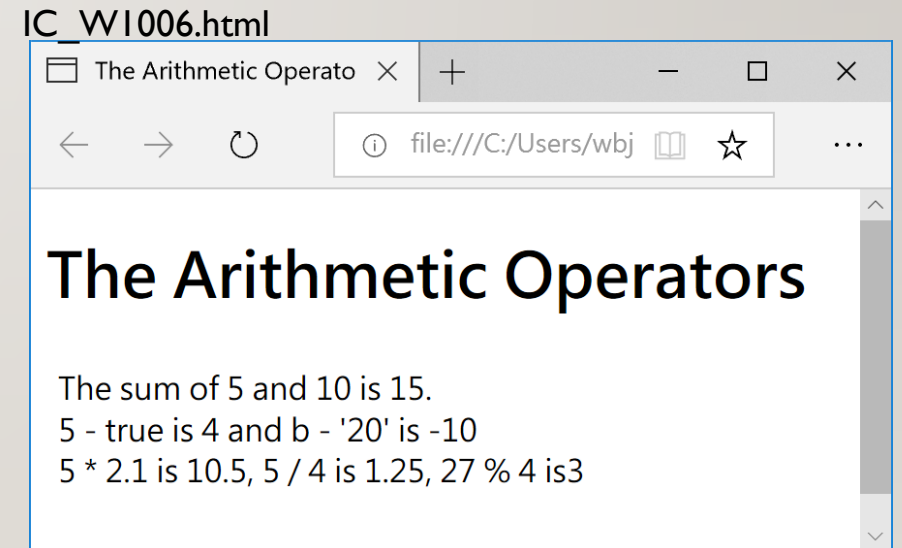
- All numbers in JavaScript are stored in double-precision float format.
- When you ask for bitwise operation, the numbers are converted to 32-bit integer.
- Do you remember the binary form of an signed integer -18?
 - `1111 1111 1111 1111 1111 1111 1110 1110`
- Bitwise NOT: `var num = 255; num = ~num;`
- Bitwise AND: `num = num & 3;`
- Bitwise OR: `num = num | 3;` Bitwise XOR: `num = num ^ 3;`
- Left Shift: `var num = 2; num = num << 5;`
- Signed Right Shift: `num = num >> 5;`
- Unsigned Right Shift: `num = num >>> 5;`

IC_WI005.html



ARITHMETIC OPERATORS

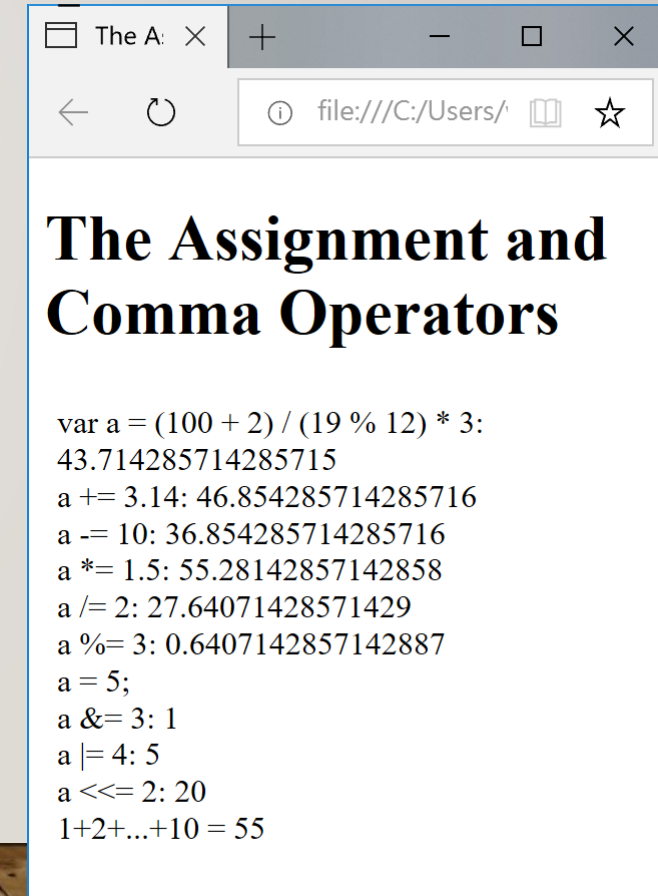
- Add: `var a = 5, b = 10; var outtxt = "The sum of "+a+" and "+b+" is +(a+b);`
- Subtract: `var a = 5-true; a = 5-"20";`
- Multiply: `var r = 5 * 2.1;`
- Divide: `var r = 5 / 4; // here the number is converted to a float number of 1.25`
- Modulus: `var r = 27 % 4;`



ASSIGNMENT & COMMA OPERATORS

- The assignment operator is `=`. It is putting the evaluated result of the right part (on the right of the operator) to the variable on the left.
 - `var num = 10; var num = 10 * (2 + 3) - 7;`
- The compound-assignment operators:
 - `+=`: `num += 10;` \rightarrow `num = num + 10;` `txt += "another sentence";`
 - `+=`, `-=`, `*=`, `/=`, `%=`
 - `&=`, `|=`, `>>=`, `<<=`
- The comma operator: `var a = 1;`
 - `var a = function(n) { let val = 0;`
 - `for (let i = 1; i <= n; i++) val += i;`
 - `return val; };`

IC_WI007.html



```
var a = (100 + 2) / (19 % 12) * 3:
43.714285714285715
a += 3.14: 46.854285714285716
a -= 10: 36.854285714285716
a *= 1.5: 55.28142857142858
a /= 2: 27.64071428571429
a %= 3: 0.6407142857142887
a = 5;
a &= 3: 1
a |= 4: 5
a <<= 2: 20
1+2+...+10 = 55
```

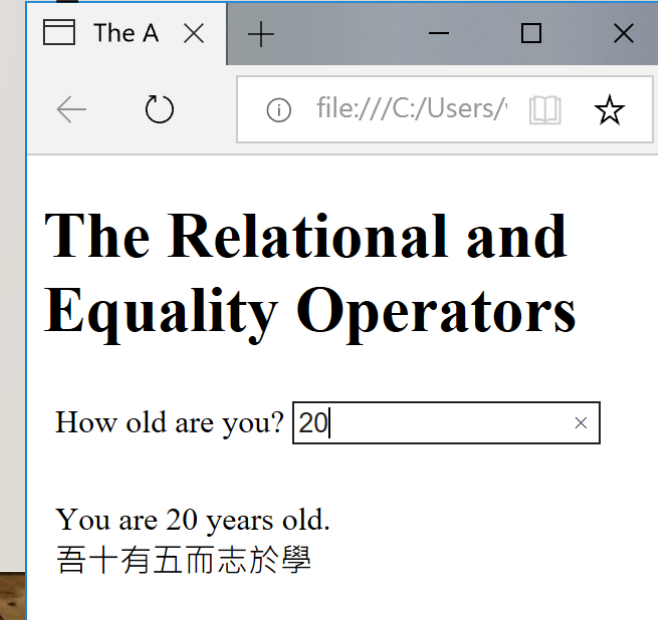

OUTLINE

1. The Number Type – Integer & Float
2. Number Conversion
3. The String & The Object
4. Operators, Bitwise Operators, Arithmetic Operators
5. Assignment & Comma Operators
- 6. Relational, Equality, Logic Operators**
- 7. If, Do-While, and While Statements**
- 8. Switch & With Statements**
- 9. While Statements**
- 10. For and For-In Statements**

RELATIONAL AND EQUALITY OPERATORS

- The relational operators give you “true” or “false” for your flow control.
- Greater than, >: 20>10, “Box” > “apple”; No less than, >=: 30>=30
- Less than: <; No greater than, <=: “30” < 3
- Equal operator: ==; Not equal operator: !=; true == 1, false != 1, ...
- The identically equal operator: ===; “55” == 55, “55” === 55
- The identically not equal operator: !==

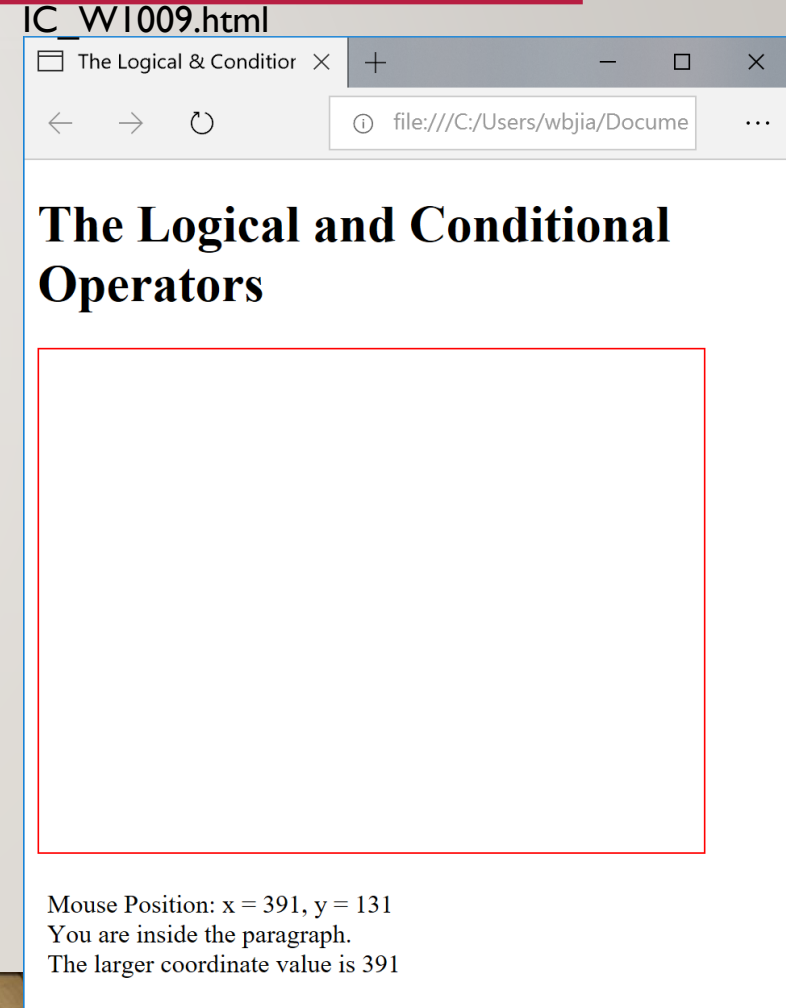
IC_WI008.html



The screenshot shows a web browser window with the title "IC_WI008.html". The browser's address bar shows the file path "file:///C:/Users/". The page content includes a form with the label "How old are you?" and a text input field containing the value "20". Below the form, the output text reads "You are 20 years old." followed by the Chinese translation "吾十有五而志於學".

LOGICAL & CONDITIONAL OPERATORS

- Logical not: ! (unlike bitwise not operator of ~)
- Logical and: &&; Logical or: ||
- The logical operators are commonly used in the statements like:
 - `if ((a >= window.left) && (a <= window.right)) do something; while (!(a == "r") || (a == "R"))`
- Conditional operator: `var max = (num1 > num2)? num1: num2;`



IC_WI009.html

The Logical & Conditor

file:///C:/Users/wbjia/Docume

The Logical and Conditional Operators

Mouse Position: x = 391, y = 131
You are inside the paragraph.
The larger coordinate value is 391

THE IF STATEMENT

- **if** (condition) statement1; else statement2;
- **If** (condition1) statement1; **else if** (condition2) statement2; else statement3;
- **If** (condition) {statement1; statement2; ...} **else** {statementA; statementB; ...}
 - **var flag = 15;**
 - **if (flag & 1) ...; if (flag & 2) ...; if (flag & 4) ...; ...**
 - **var x = 4;**
 - **If (x > 0) ...; else if (x == 0) ...; else ...;**

THE SWITCH AND WITH STATEMENTS

- The switch statement:
 - `switch(variable){`
 - `case value 1: statement 1; break;`
 - `case value 2: statement 2; break;`
 - `...}`
- The with statement sets the codes under the usage of an object.
 - `with (window.location){`
 - `window.alert(hostname); // the same as`
`window.alert(window.location.hostname) outside the scope;}`

IC_WI010.html



The screenshot shows a web browser window with the title "IC_WI010.html". The address bar contains "file:///C:/Users/". The main content area features a large heading "The With and Switch Statements". Below the heading is a form with a label "How many roses do you want to send to your fr" and a text input field containing the number "2". At the bottom of the page, there is Chinese text: "你儂我儂、心心相印、世界只有我和你 (妳) 中世界只有我倆".

THE DO-WHILE AND WHILE STATEMENTS

- The do-while statement is a post-check loop. It will do at least one time the commands in the do-while statement.
- The while statement will check the condition before it carry out the commands in the while statement.
 - `do {statement1; statement2; ...} while(condition);`
 - `while (condition) {statement1; statement2; ...}`

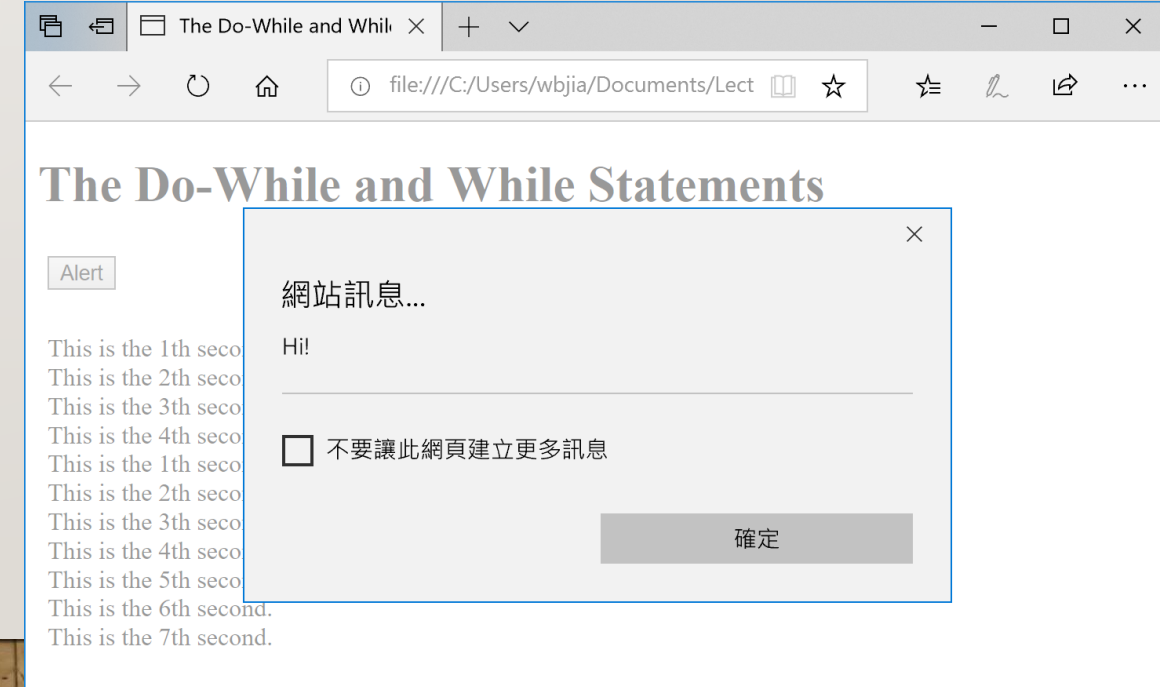
window object functions:

```
setTimeout(func_name, time in ms);
```

```
setInterval(func_name, time in ms);
```

The func_name will be called after the specified time in ms

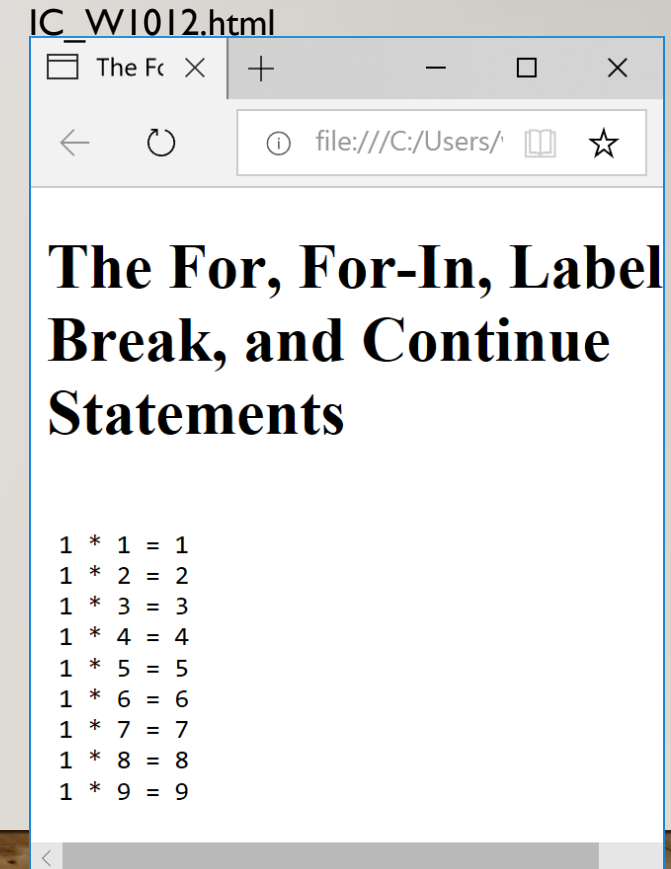
IC_WI011.html



The screenshot shows a web browser window with the title 'IC_WI011.html'. The address bar shows the file path: file:///C:/Users/wbjia/Documents/Lect. The page content is titled 'The Do-While and While Statements' and displays a list of messages: 'This is the 1th seco', 'This is the 2th seco', 'This is the 3th seco', 'This is the 4th seco', 'This is the 1th seco', 'This is the 2th seco', 'This is the 3th seco', 'This is the 4th seco', 'This is the 5th seco', 'This is the 6th second.', and 'This is the 7th second.'. An 'Alert' dialog box is overlaid on the page, containing the text '網站訊息...' and 'Hi!'. Below the text is a checkbox labeled '不要讓此網頁建立更多訊息' and a '確定' button.

THE FOR, FOR-IN, LABEL, BREAK AND CONTINUE STATEMENTS

- The for statement: for (initialization; condition; do something in each loop) statement;
- for (var i = 0; i < 10; i++) do something;
- for (;;) do something; // an infinite loop
- for (var propertyName in window) do something; // like enumerate, used for an object
- Label, break, and continue
 - var str = "";
 - loop1:
 - for (var i = 0; ;){ i++; if (i == 3) continue loop1;
 - str += i; if (i == 5) break;}



EXERCISE

1. Please prepare two input fields to get two integer numbers from users . Please give one action button for the users. When the users put two decimal numbers and press the action button, please show your result in the html using the MathJax about the addition, the subtraction, the multiplication, and the division of the two numbers. In addition, please show in normal html about the binary form of the two numbers, the binary form of the results of the 'and' operation and the 'or' operation, and the binary form of the bitwise not operation on the two numbers.
2. Please write an XO game by using the buttons. Please determine who wins the game.
3. Please use 7X7 characters of "o" to draw numbers from 0 to 9.
4. Please write a program to draw open squares by using any alphabet characters. Let user use a range input to determine the size of your squares from 2 to 10.

EXERCISE

1. Please put a button inside a rectangular area in an html view and put another button outside the area for users to start the moving of the button in the rectangular. The button can move either horizontally or vertically until it hits the rectangular boundary. You can use `setTimeout` function to move the button one step per second.
2. Please demonstrate the walking green man by using several images and the `img` element.
3. Please demonstrate a traffic light with green, yellow, and red light. The green light turn on for 60 s. It follows with a flashing yellow light for 10 s at a frequency of 1 Hz. The red light will then be turned on for 60 s. After the red light, the green light turns on again.