

# **Web Application Development for Industrial Engineers**

**การพัฒนาแอปพลิเคชันสำหรับวิศวกรอุตสาหกรรม**

# What is a variable?

- A variable is a *container* for a value
- Things we can store.
  - Actual value: 1 / "Text" / true , ...
  - Reference (pointer) to an object

# Declaring a variable

```
let myName;  
let myAge;
```

- All code instructions should end with a semi-colon.

# Initializing a variable

```
let myName;  
myName = 'Chris';
```

or just

```
let myName = 'Chris';
```

## Note about `var`

```
var myName;  
var myAge;
```

- Old way of declaring variable
- Error prone
  - [Hoisting](#)
  - Allowing re-declarations
- *Not recommended*

# Re-declaration

- (Use quokka)

```
let x = 5;  
let x = 10;  
console.log(x); \\SyntaxError: Identifier 'x' has already been declared
```

# Updating a variable

```
let myName = 'Chris';  
myName = 'Bob'; //Updating
```

# Good variable naming

Good

```
age;  
myAge;  
init;  
initialColor;  
finalOutputValue;  
audio1;  
audio2;
```



# Bad variable naming

```
1;  
a;  
_12;  
myage;  
MYAGE;  
var Document;  
skjfnbskjfnbskjfb;  
thisisareallylongstupidvariablename;
```

# Variable types

- Numbers
- Strings
- Booleans
- Arrays
- Objects

# Numbers

```
let myAge = 17;  
console.log(typeof myAge); //number
```

# Number operations

- `+`, `-`, `*`, `/`
- Modulo `%`
- Exponent `**`

# Operator precedence

```
const num1 = 10;  
const num2 = 50;  
num2 + num1 / 8 + 2; //Get 53.25  
(num2 + num1) / (8 + 2); //Get 6
```

# Increment and decrement operators

```
let a = 1;  
a++;  
console.log(a); // 2  
++a;  
console.log(a); // 3  
a += 1;  
console.log(a); // 4
```

# Strings

- Strings are pieces of text.
- When you give a variable a string value, you need to wrap it in single or double quote marks.

```
let dolphinGoodbye = 'So long and thanks for all the fish';  
typeof dolphinGoodbye; //string
```

# Use quotes in string

```
const bigmouth = 'Using \' is okay. Also is "...';  
console.log(bigmouth);  
// Using ' is okay. Also is "...
```



# Concatenating strings

- `+` operator

```
const str1 = 'Hello';  
const str2 = 'World';  
console.log(str1 + ' ' + str2 + ' !');
```

- Template literal

```
console.log(`${str1} ${str2} !`);
```

- <https://codepen.io/nnnpoooh/pen/QWqNwzo>

# String / Number transformation

- A problem with a number stored as a string type

```
let myNumber = '74';  
myNumber += 3; // Get 743
```

- To fix this

```
Number(myNumber) + 3;
```

- <https://codepen.io/nnnpoooh/pen/OJxNPKB>

# Multiline strings

- Break characters `\n`

```
const output = 'I like the song.\nI gave it a score of 90%.';  
console.log(output);
```

```
// I like the song.  
// I gave it a score of 90%.
```

## Multiline strings (cont)

- Template literals respect the line breaks in the source code.

```
const output = `I like the song.  
I gave it a score of 90%.`;  
console.log(output);  
  
// I like the song.  
// I gave it a score of 90%.
```

# Strings as objects

- Most things are objects in JavaScript.

```
const string = 'This is my string';
```

- The variable becomes a `String` object instance.
  - Contains [properties and methods](#).

# String length

```
const browserType = 'mozilla';  
browserType.length; // 7
```

# Accessing string characters

```
browserType[0]; // "m"  
browserType[browserType.length - 1]; // "a"
```

- Remember: computers count from 0, not 1!

# Testing if a string contains a substring

```
browserType.includes('zilla'); // true
```



# Extracting a substring from a string

```
browserType.slice(1, 4); // "ozi"  
browserType.slice(2); // "zilla"  
browserType.slice(0, -1); // "mozill
```

# Changing case

```
const radData = 'My NaMe Is MuD';  
console.log(radData.toLowerCase()); //my name is mud  
console.log(radData.toUpperCase()); //MY NAME IS MUD
```

# Updating parts of a string

```
const browserType = 'mozilla';  
const updated = browserType.replace('moz', 'van');  
  
console.log(updated); // "vanilla"  
console.log(browserType); // "mozilla"
```

- Note that `replace()` doesn't change the string it was called on.

# Booleans

- Booleans are `true` / `false` values.
- These are generally used to test a condition, after which code is run as appropriate.

```
let iAmAlive = true;  
let test = 6 < 3;  
typeof test; //boolean
```

# Calculator App

- [https://ie-software-dev.netlify.app/codes/t07\\_js/t01\\_calculator/](https://ie-software-dev.netlify.app/codes/t07_js/t01_calculator/)