

# JavaScript Essentials

## *Strings*



# Table of Contents

1. Overview
2. Strings basics
3. Template strings

# Lesson Objectives

- Understand the power of text in JavaScript and Front-end Development
- Able to create/concatenate strings
- Understand the difference of strings and numbers
- Able to use template strings concatenate strings

## Section 1

# Overview

# Overview – Power of text

```
<nav style="background-color: #dcdcdc;">
  <span class="container">
    <span class="navbar-group">
      <ul class="navbar-list">
        <li class="nav-item"><a href="/Home">Home</a></li>
        <li class="nav-item"><a href="/Products">Products</a></li>
        <li class="nav-item"><a href="/Services">Services</a></li>
      </ul>
    </span>
    <span class="navbar-group">
      <ul class="navbar-list">
        <li class="nav-item"><a href="/About">About Us</a></li>
        <li class="nav-item"><a href="/Privacy">Privacy</a></li>
      </ul>
    </span>
  </span>
</nav>
```

```
h1 { color: white;
background: orange;
border: 1px solid black
padding: 0 0 0 0;
font-weight: bold;
}
/* begin: seaside-theme */

body {
background-color:white;
color:black;
font-family:Arial,sans-serif;
margin: 0 4px 0 0;
border: 12px solid;
}
```

CSS

Text everywhere

- Pretty much all of the programs we've shown you so far in the course have involved some string manipulation.
- Understand strings give us a huge advantage in Programming


## Section 2

# Strings basics

- Syntax (correct):

Single Quote


```
var name1 = 'Nguyen Van A'
```



The diagram shows the code `var name1 = 'Nguyen Van A'` enclosed in a black rectangular box. The opening single quote and the closing single quote are each circled with a dashed blue line. Two blue arrows point from the text 'Single Quote' above to these two circles. Below the box is a green circle containing a white checkmark.

Double Quote

```
var name1 = "Nguyen Van A";
```



The diagram shows the code `var name1 = "Nguyen Van A";` enclosed in a black rectangular box. The opening double quote and the closing double quote are each circled with a dashed blue line. Two blue arrows point from the text 'Double Quote' above to these two circles. Below the box is a green circle containing a white checkmark.

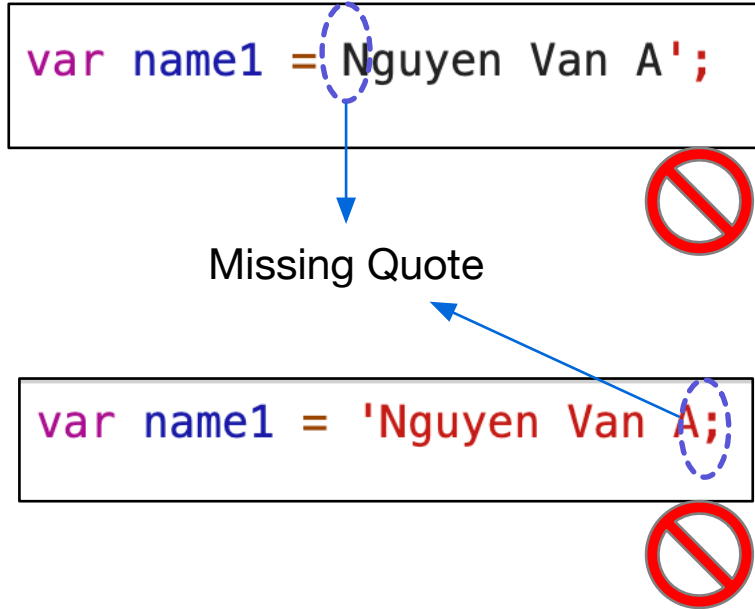


- Syntax (wrong):

`var name1 = Nguyen Van A';`

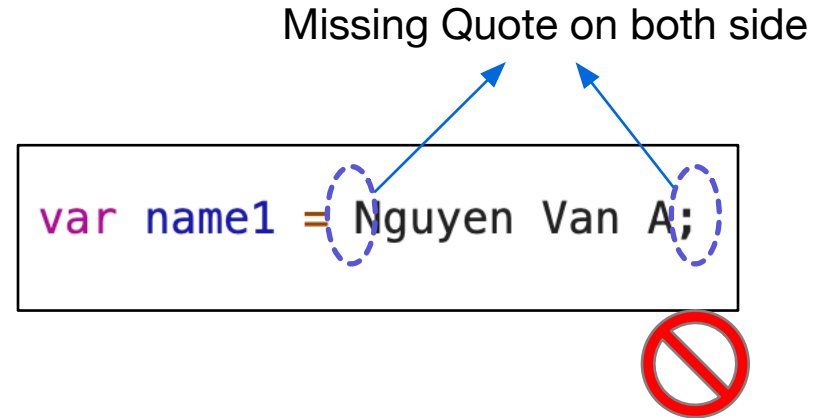
Missing Quote

`var name1 = 'Nguyen Van A;`




Missing Quote on both side

`var name1 = Nguyen Van A;`



- What if we have single quote or double quote in our string ?  
Like this: I'm studying
- Use Escape notation syntax:

Add \ before quote to escape it



```
var name1 = "I\"m studing ";  
name1;  
"I'm studing "
```

- Escape notation usage:

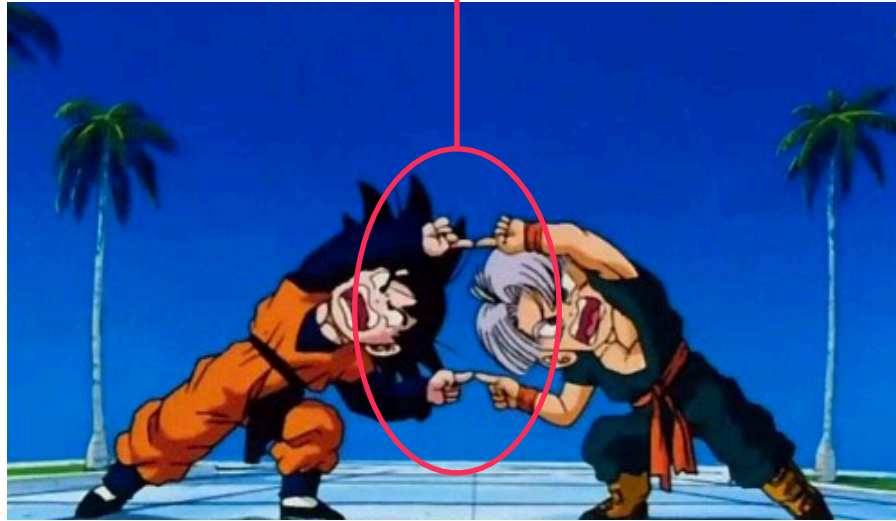
```
var multiline = "I\"m studing\nIm working";  
multiline;  
"I\"m studing  
Im working"
```

new line

```
var tab = "I\"m studing\tIm working";  
tab;  
"I\"m studing   Im working"
```

tab

Concatenate (fancy way to call  
"join" together)



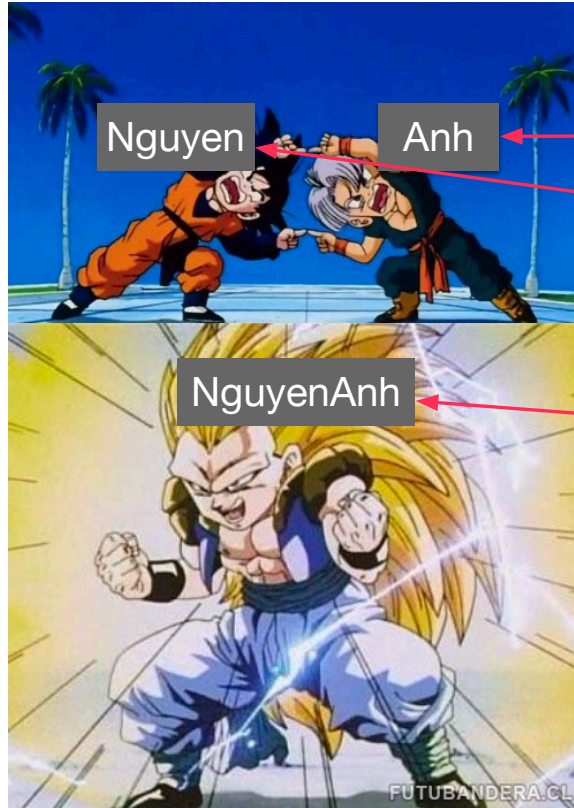
- Joining together strings in JavaScript uses the plus (+) operator, the same one we use to add numbers together, but in this context it does something different

```
var firstName = "Nguyen";  
var lastName = 'Anh';  
  
firstName + lastName;  
"NguyenAnh"
```

Concatenate

# Strings basics – Concatenating strings

- Result:



```
var firstName = "Nguyen";  
var lastName = "Anh";  
  
firstName + lastName;  
"NguyenAnh"
```

- In the last instance, we joined only two strings, but you can join as many as you like, as long as you include a + between each pair. Try this:

```
var greeting = 'Hello';  
var firstName = 'Nguyen';  
var lastName = 'Anh';  
var space = ' ';  
  
greeting + space + greeting + space + firstName + space + lastName;  
"Hello Hello Nguyen Anh"
```

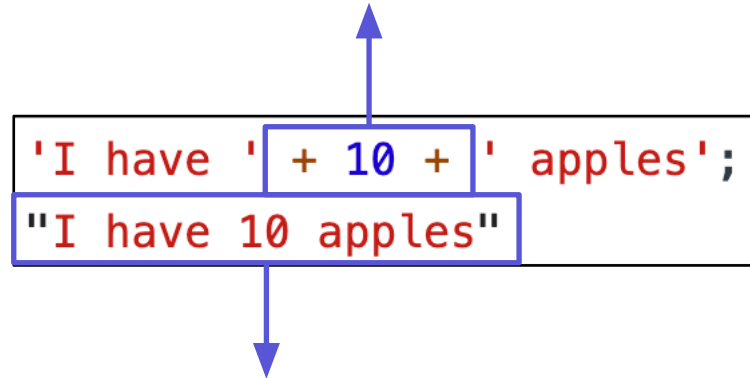
- You can also use a mix of variables and actual strings. Try this:

```
var greeting = 'Hello';  
var firstName = 'Nguyen';  
var lastName = 'Anh';  
  
greeting + ' ' + firstName + ' ' + lastName + ' from FA';  
"Hello Nguyen Anh from FA"
```



- So what happens when we try to add (or concatenate) a string and a number? Let's try it in our console:

You might expect this to return an error ?

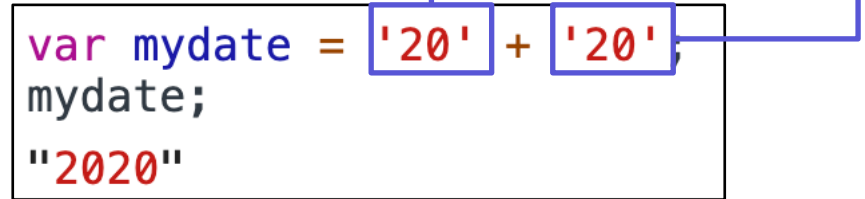


The diagram illustrates the concatenation of a string and a number. It consists of a large rectangular box containing two lines of text. The top line is `'I have ' + 10 + ' apples';` and the bottom line is `"I have 10 apples"`. A blue arrow points from the `10` in the top line down to the `10` in the bottom line. Another blue arrow points from the top of the box upwards towards the text "You might expect this to return an error ?".

But it works fine!

- You can even do this with two numbers — you can force a number to become a string by wrapping it in quote marks. Try the following:

Notice the quote between 20



```
var mydate = '20' + '20';  
mydate;  
"2020"
```

- To convert between number and string:

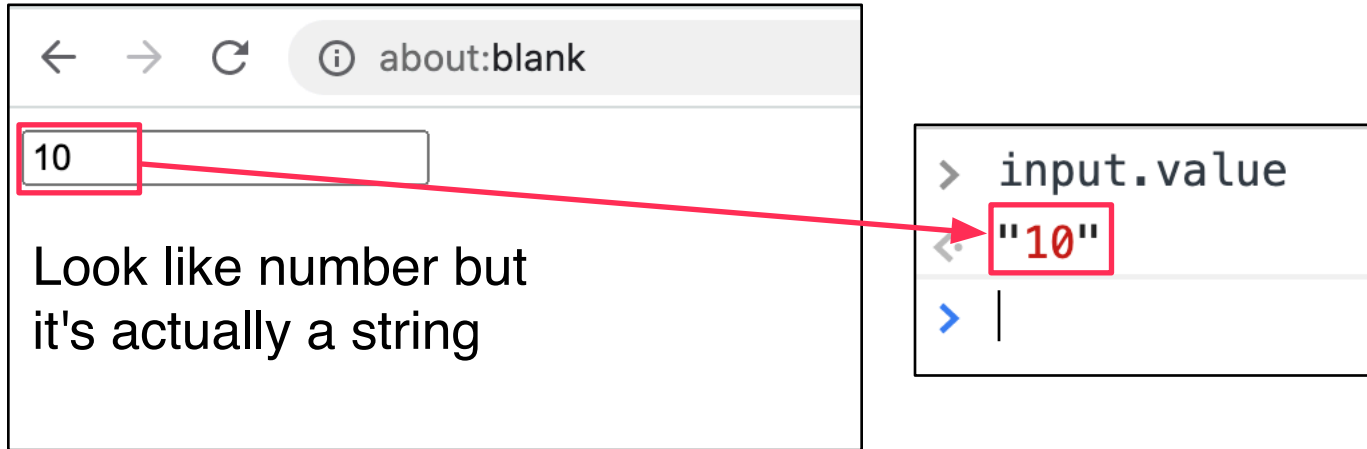
Convert from string to number

```
var s = '20';  
Number(s);  
20
```

Convert from number to string

```
var n = 20;  
'' + n; // or n.toString();  
"20"
```

# Strings basics – Numbers and strings



10

Look like number but  
it's actually a string

```
> input.value  
< "10"  
> |
```

- Sometimes, it not work the way you wanted

```
var nbFresherAngular = 14;  
var nbFresherJava = 10;  
  
'FA have total of ' + nbFresherAngular + nbFresherJava;  
"FA have total of 1410"
```



Wrong numbers (must by 24)

- To fix it:

```
var nbFresherAngular = 14;  
var nbFresherJava = 10;  
var totalFresher = nbFresherAngular + nbFresherJava;  
  
'FA have total of ' + totalFresher;  
"FA have total of 24"
```

Correct now

## Practice Strings manipulation

- Create a string using single quote ('), double quote (“)
- When your string has single quote or double quote, escape it with \', \"
- Concatenating strings using + operators
- You can concatenate strings and number with + operators
- Take care when you concatenate strings and number it may give wrong answer



## Section 3

# Template strings

- Another type of string syntax that you may come across is **template literals** (sometimes referred to as template strings). This is a newer syntax that provides more flexible, easier to read strings.

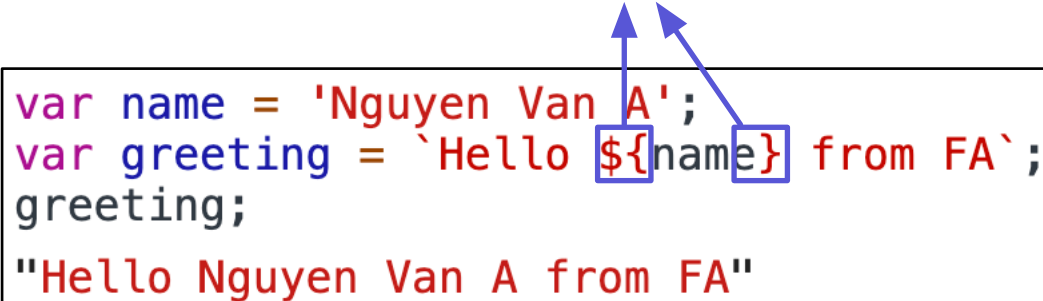
Backquote

```
var greeting = `Hello from FA`;  
greeting  
"Hello from FA"
```

A diagram illustrating the equivalence between backquote and double quote string literals. It shows a code snippet where a variable 'greeting' is assigned the value of a backquote-enclosed string 'Hello from FA'. Below this, the same string is shown enclosed in double quotes. Blue boxes highlight the backquote characters in the first line and the double quote characters in the second line. Two blue arrows point from the word 'Backquote' above to these highlighted characters, indicating that the backquote syntax is being compared to the standard double quote syntax.

- Template literals simplify concatenating strings:

Syntax to add variables, values



```
var name = 'Nguyen Van A';  
var greeting = `Hello ${name} from FA`;  
greeting;  
"Hello Nguyen Van A from FA"
```

- Template literals work great with multiline:

```
var name = 'Nguyen Van A';  
var greeting = `Hello ${name} from FA.  
How are you ?`;  
greeting;  
"Hello Nguyen Van A from FA.  
How are you ?"
```

→ No \n

- Template literals work great with numbers:

```
var nbFresherAngular = 14;  
var nbFresherJava = 10;  
  
`FA have total of ${nbFresherAngular + nbFresherJava}`;  
"FA have total of 24"
```

## Practice Template Strings

- Assignment operators provide useful shortcuts to keep our code cleaner and more efficient
- Can use other variables on the right hand side of each expression as well

# Thank you

Q&A

