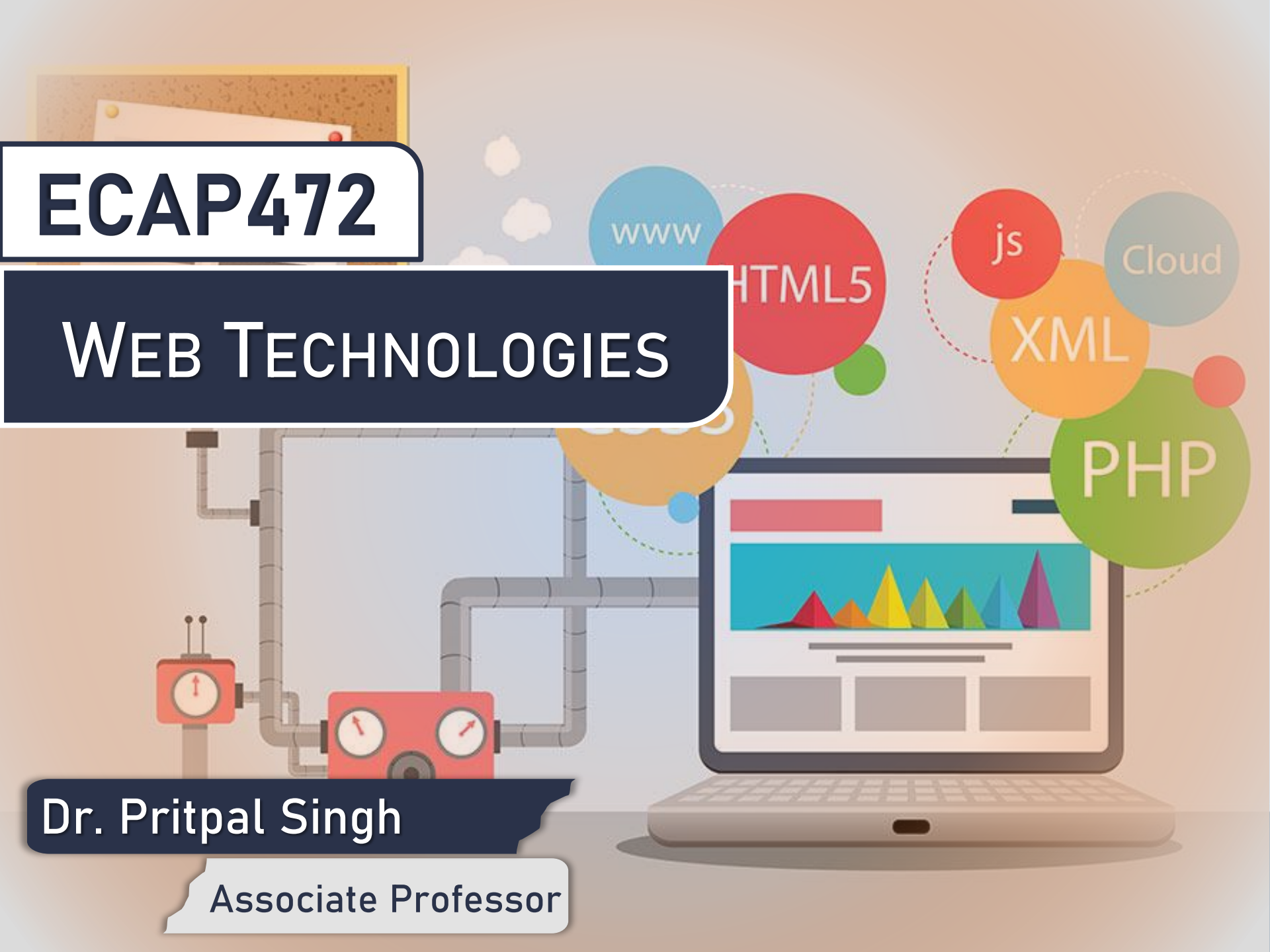
ECAP472

Web Technologies



Dr. Pritpal Singh

Associate Professor

Learning Outcomes

After this lecture, you will be able to

- understand JavaScript String and numbers .
- go over Arrays in JavaScript.

JavaScript Strings

- JavaScript strings are for storing and manipulating text.
- A JavaScript string is zero or more characters written inside quotes.
- Example
- let carName1 = "Volvo XC60"; // Double quotes

  let carName2 = 'Volvo XC60'; // Single quotes

JavaScript String Methods

- String methods help you to work with strings.
- But with JavaScript, methods and properties are also available to primitive values, because JavaScript treats primitive values as objects when executing methods and properties

Extracting String Parts

There are 3 methods for extracting a part of a string:

slice(start, end)

substring(start, end)

substr(start, length)

JavaScript String slice()

- slice() extracts a part of a string and returns the extracted part in a new string.
- The method takes 2 parameters: the start position, and the end position (end not included).

Example

- let str = "Apple, Banana, Kiwi";
- let part = str.slice(7, 13);

NOTE

- JavaScript counts positions from zero.
- First position is 0.
- Second position is 1.
- If a parameter is negative, the position is counted from the end of the string.

JavaScript String substring()

- substring() is similar to slice().
- The difference is that substring() cannot accept negative indexes.

Example

- let str = "Apple, Banana, Kiwi";

  let part = str.substring(7, 13);

Converting to Upper and Lower Case

- A string is converted to upper case with toUpperCase():
- A string is converted to lower case with toLowerCase():

JavaScript String toUpperCase()

- JavaScript String toUpperCase()

Example

- let text1 = "Hello World!";
- let text2 = text1.toUpperCase();

JavaScript Numbers

- JavaScript has only one type of number. Numbers can be written with or without decimals.

Example

- let x = 3.14; // A number with decimals
- let y = 3; // A number without decimals

JavaScript Numbers are Always 64-bit Floating Point

- Unlike many other programming languages, JavaScript does not define different types of numbers, like integers, short, long, floating-point etc.
- JavaScript numbers are always stored as double precision floating point numbers, following the international IEEE 754 standard.
- This format stores numbers in 64 bits, where the number (the fraction) is stored in bits 0 to 51, the exponent in bits 52 to 62, and the sign in bit 63

Array

An array is a collection of items stored at contiguous memory locations. The idea is to store multiple items of the same type together. This makes it easier to calculate the position of each element by simply adding an offset to a base value, i.e., the memory location of the first element of the array (generally denoted by the name of the array). The base value is index 0 and the difference between the two indexes is the offset.

Array example

- For simplicity, we can think of an array as a fleet of stairs where on each step is placed a value (let's say one of your friends). Here, you can identify the location of any of your friends by simply knowing the count of the step they are on.
- Remember: "Location of next index depends on the data type we use".

JavaScript Arrays

An array is a special variable, which can hold more than one value:

```
const cars = ["Saab", "Volvo", "BMW"];
```

Why Using an Array?

- If you have a list of items (a list of car names, for example), storing the cars in single variables could look like this:
  - let car1 = "Saab";
  - let car2 = "Volvo";
  - let car3 = "BMW";
- However, what if you want to loop through the cars and find a specific one? And what if you had not 3 cars, but 300?
- The solution is an array!
- An array can hold many values under a single name, and you can access the values by referring to an index number.

Creating an Array

- Using an array literal is the easiest way to create a JavaScript Array.

Syntax:

- const array_name = [item1, item2, ...];
- It is a common practice to declare arrays with the const keyword

Example

- const cars = ["Saab", "Volvo", "BMW"];

Accessing Array Elements

You access an array element by referring to the index number:

- const cars = ["Saab", "Volvo", "BMW"];
- let car = cars[0];

Access the Full Array

With JavaScript, the full array can be accessed by referring to the array name:

Example

- const cars = ["Saab", "Volvo", "BMW"];
- document.getElementById("demo").innerHTML=cars;

Accessing the First Array Element
Example

Example

- const fruits = ["Banana", "Orange", "Apple", "Mango"];
- let fruit = fruits[0];
- Accessing the Last Array Element

Example

- const fruits = ["Banana", "Orange", "Apple", "Mango"];

  let fruit = fruits[fruits.length - 1];

The Difference Between Arrays and Objects

- In JavaScript, arrays use numbered indexes.
- In JavaScript, objects use named indexes.

When to Use Arrays. When to use Objects.

- JavaScript does not support associative arrays.
- You should use objects when you want the element names to be strings (text).
- You should use arrays when you want the element names to be numbers.


Practical

That’s all for now…