

# Java Script Essentials



- JS History
- JS on webpage
- Operators
- Decision Making (If, if-else and if-else-if)
- Loop Controls (for, while and do-while)
- Functions
- Arrays & String
- Events

# Functions in JS



- JavaScript function is a block of code designed to perform a particular task.
- JavaScript function is executed when "something" invokes it (calls it).

```
// Function to compute the product  
of p1 and p2  
function myFunction(p1, p2) {  
    return p1 * p2;  
}
```

# Arrow Functions in JS

- ES6, introduced Arrow functions.
- Arrow functions is used to write shorter function syntax:

## Before Arrow:

```
function square(params) {  
    return params * 2;  
}  
  
square(4);
```

## After Arrow:

```
var square = (params) => {  
    return params * 2;  
}  
  
square(4);
```

**Note:** It is important to note that arrow function is anonymous, which means that it is not named.

# Arrow Functions in JS



If the function has only one statement and the statement returns a value, you can remove the brackets and the *return* keyword.

Arrow Functions Return Value by Default:

```
hello = () => "Hello World!";
```

Arrow Function With Parameters:

```
hello = (val) => "Hello " + val;
```

Arrow Function Without Parentheses:

```
hello = val => "Hello " + val;
```

# Class and Object in JS



- ES6, introduced JavaScript Classes.
- JavaScript Classes are templates for JavaScript Objects.

Use the keyword **class** to create a class.

Always add a method named **constructor()**

## Class Declaration:

```
class ClassName {  
  constructor() { ... }  
  method_1() { ... }  
  method_2() { ... }  
}
```

## Object Creation:

```
let objectName = new ClassName ();
```

# Class and Object in JS

## Constructor with no parameter:

```
class Person {  
  constructor() {  
    this.name = "Ram";  
    this.age = 20;  
  }  
  show() {  
    return (this.name + this.age);  
  }  
}
```

```
let person1 = new Person();
```

# Class and Object in JS

## Constructor with parameter:

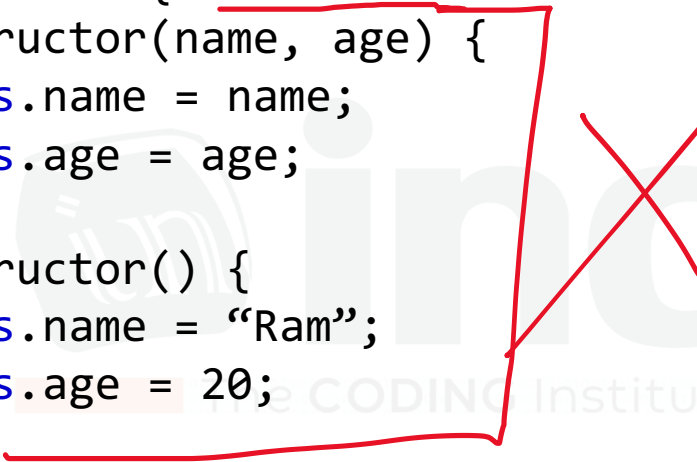
```
class Person {  
  constructor(name, age) {  
    this.name = name;  
    this.age = age;  
  }  
  show() {  
    return (this.name + this.age);  
  }  
}
```

```
let person1 = new Person("Rohan", 23);  
let person2 = new Person("Himesh", 21);
```

# Class and Object in JS

Constructor overloading is NOT allowed.

```
class Person {  
  constructor(name, age) {  
    this.name = name;  
    this.age = age;  
  }  
  constructor() {  
    this.name = "Ram";  
    this.age = 20;  
  }  
  show() {  
    return (this.name + this.age);  
  }  
}
```

A red rectangular box is drawn around the two constructor methods in the code. A large red 'X' is drawn over the box, indicating that having two constructors is invalid.

**Note:** Only one constructor is allowed at a time.