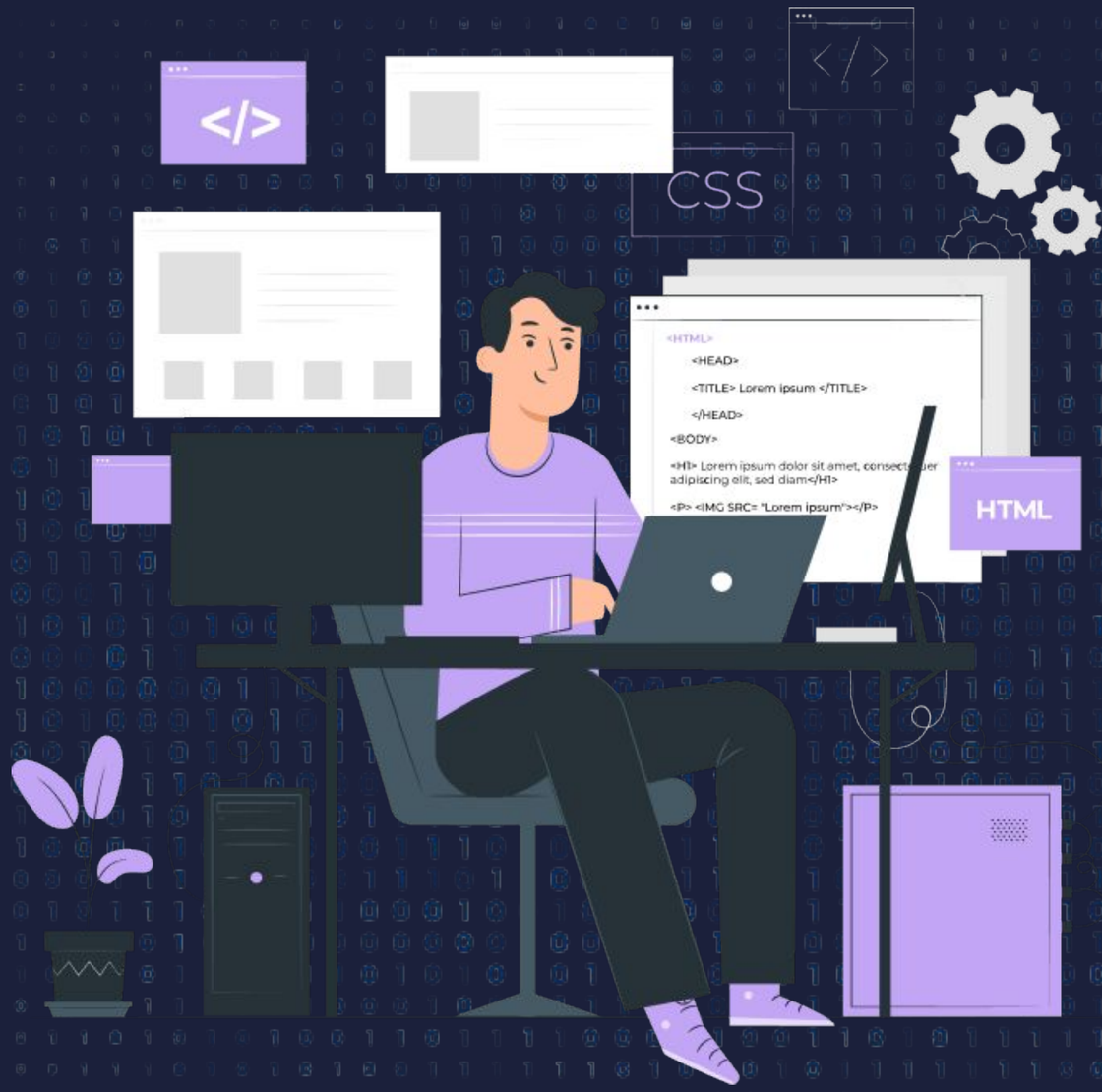
PW SKILLS

More on functions – call, apply, bind



Topics Covered

- Revisit Functions
- Function method - call()
- Function method - apply()
- Function method - bind()

Revisit Functions

```
const newFunction = new Function();

// EQUIVALENT TO

function newFunction(){}
```

It means, functions are also objects, then it can have methods too.

In this lecture will talk about some important function methods.

call()
apply()
bind()

Function method - call()

The call() is a method present in every function, which is used to invoke the function, with a given object reference (**thisValue**) and a set of **arguments**.

```
func.call(thisValue, arg1, arg2, arg3, /* ..., */ argN)
```

It calls the function **func** upon an object (**thisValue**) with a set of arguments (arg1, arg2, arg3 .... argN).

Example - call()

```
const sides = {
  a:10,
  b:12
}

function area(shapeName)  {
   console.log(`Area of ${shapeName} is ${this.a*this.b}`)
}
```

```
func.area.call(sides, "rectangle") // 120
```

Function method - apply()

The **apply()** method is similar to call(), with one difference that arguments are passed as a single array.

```
func.apply(thisValue, argsArray);
```

It calls the func with **thisValue** and arguments as a single array **argsArray**.

Example - apply()

```
const arr = [1,2,3,4];
const newValues = ["x", "y", "z"];

arr.push.apply(arr, newValues);
console.log(arr); // (7) [1, 2, 3, 4, 'x', 'y', 'z']
```

Function method - bind()

The **bind()** method instead of calling a function, creates a new function attached with an object **(thisValue)** and set of **arguments**.

```
newFunc = func.bind(thisValue, arg1, arg2, /* ..., */ argN)
```

It attaches **thisValue** and **arguments** to func, and returns a new function **newFunc.**

Example - bind() Creating a Bound Function

```
const newObject = {
  value: 2048,
  getValue() {
    return this.value;
  }
}

// Calling 'getValue' method of 'newObject'
newObject.getValue() // 2048

const newGetValue = newObject.getValue;

// Create a new function 'boundedGetValue' with the 'this'
parameter bound to 'newObject'
const boundedGetValue = newGetValue.bind(newObject);

// Calling 'boundedGetValue' function
boundedGetValue(); // 2048
```


SKILLS
THANK YOU