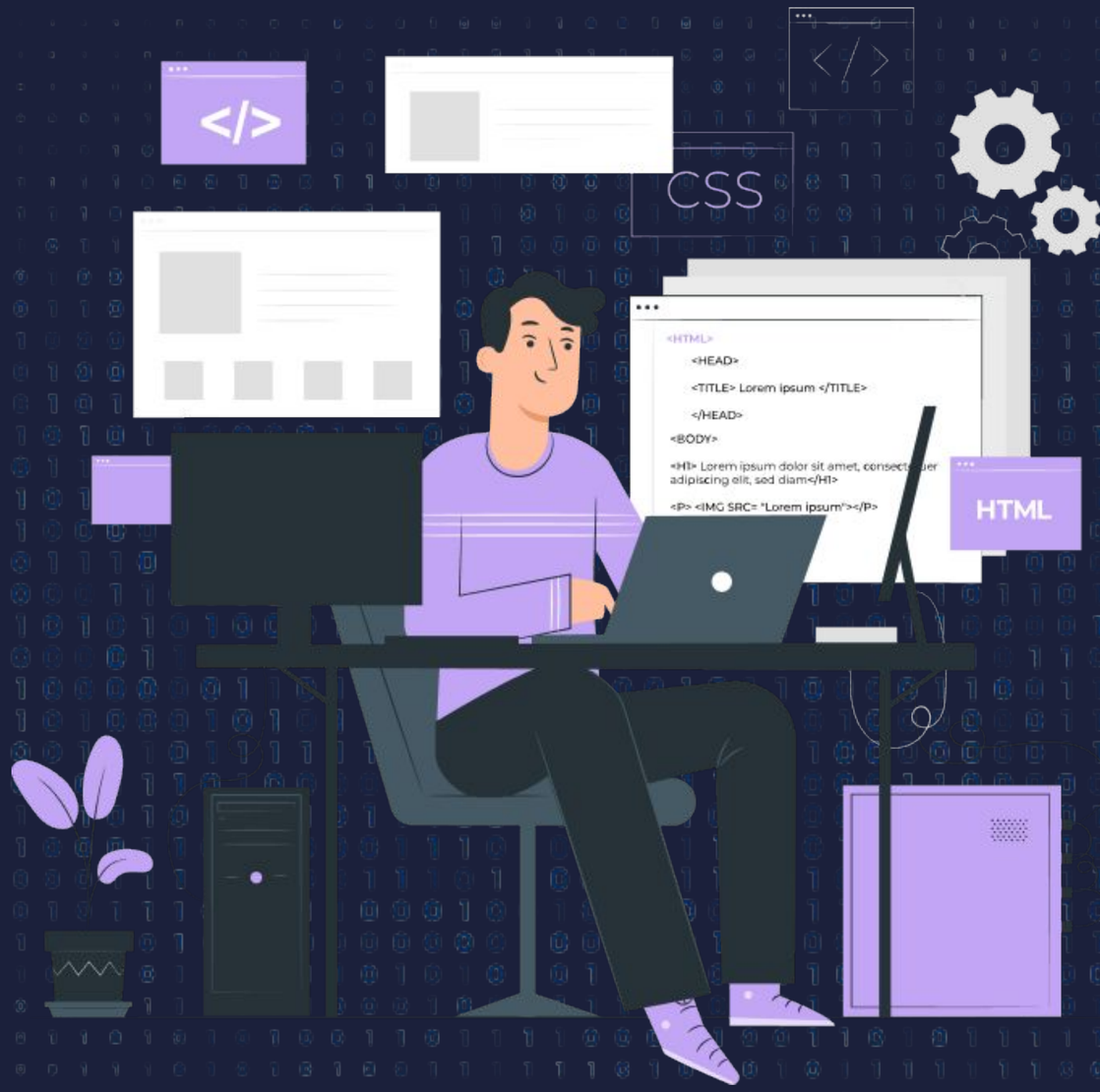
PW SKILLS

IIFE, pure function, and function currying



Topics:

- IIFE functions with examples
- Pure functions with examples
- function currying.

IIFE functions with examples

An IIFE (immediately Invoked Function Expression) or Self-invoking function is a javascript function that runs as soon as it is defined.
Syntax -

```
(function() {
  // Code to be executed immediately
})();
//or
(() => {
})()
// or
( async () => {
  // code here ....
})()
// or
( async (parameters) => {
  // code here ....
})(arguments)
```

Example of Self-invoking function

```
// simple self-invoking function
(() => {
  console.log("This is a self-invoking function");
})();
// output - This is a self-invoking function


// self-invoking function with parameter
((a, b) => {
  console.log(a + " " + b);
})("hello", "world");
// output -- hello world
```

The self-invoking function or IIFE can also take parameters and return values just like any regular function.

```
const result = ((a, b) => {
  return a + " " + b;
})("hello", "world");

console.log(result);
//output - hello world
```

Some of the uses of self-invoking functions include

1. Avoid polluting the global namespace
2. Data Privacy and Creating Private Variables
3. Module Pattern

Avoid polluting the global namespace example -

```
const greet = "global variable";
const greeting = () => {
  return "hello global variable";};
(() => {
  const greet = "IIFE variable";
  const greeting = () => {
    return " hello IIFE function"; };
  console.log(greet);
  console.log(greeting());})();
console.log(greet);
console.log(greeting());
/**** output -
IIFE variable
hello IIFE function
global variable
hello global variable
 */
```

Data Privacy and Creating Private Variables Example

```
(() => {
  let password = "431251";
  const showPassword = () => {
    console.log("Password - ", password);
  };
  showPassword();
})();
/**
 * output
 * Password -  43125
 */
showPassword(); // Reference error - showPassword is
not defined
console.log(password); // Reference error - password
is not defined
```

Continue...

Module pattern example -

```
const point = (() => {
  let count = 0;
  return {
    balance: () => {
      return count;
    },
    increment: () => {
      count++;
    },
    reset: () => {
      count = 0;
    },};})();
console.log(point.balance()); // 0
point.increment();
console.log(point.balance()); // 1
point.increment(); // 2
point.increment(); //2
console.log(point.balance());
point.reset();
console.log(point.balance()); // 0
```

Pure functions with examples

A pure function in JavaScript is a function that always returns with the same output for the same input. It does not have any side effects, meaning that it does not change any global state or the state of any objects that are passed to it

Example - pure function

```
function add(x, y) {
  return x + y;
}
console.log(add(2, 5));
// output - 7
```

Example - impure function

```
let sum = 1;
function SumUp(price) {
  return (sum += price);
}
console.log(SumUp(44));
// output - 45
```

Some of the benefits of using the Pure function include -

1. Predictability - Pure functions are easier to reason about since they don't have hidden dependencies or side effects. Given the same inputs, you can expect the same outputs consistently.
2. Testability - Pure functions are easy to test because they don't rely on external states. You can test them by providing different inputs and asserting the expected outputs.
3. Modularity - Pure functions are self-contained units of code. They can be composed and combined without worrying about unexpected interactions or unintended consequences.

Function Currying

In JavaScript, Currying is a functional programming technique that involves transforming a function with multiple arguments into a sequence of functions, each taking a single argument. This allows you to partially apply arguments to a function and create new specialized functions.

Example of currying functions

```
function volume(length) {
  return function (width) {
    return function (height) {
      return height * width * length;
    };
  };
}
const result = volume(2)(6)(3);
console.log(result); // 36
```

Another example -

```
const posts = [{userId: 1,id: 1,title: "The shooting
star",body: "quia et suscipit\nsuscipit....",},
  {userId: 1, id: 2,title: "The snow white",body: "est
rerum tempore.....",
  },
  {userId: 1,id: 3,title: "The evolution of earth",body:
"et iusto sed quo iure\nvo....",},
];
let getPlaceHolderData = (posts) => {
 // return another function with one argument
  return (num) => {
    // return another function with one argument
    return posts.slice(0, num).map((item) => {
      // get any of the available field.
      let label = item.title;
      // return data
      return `posts title :: ${label}`;}); };};
let posts1 = getPlaceHolderData(posts);
console.log(posts1(2));
```


SKILLS
THANK YOU