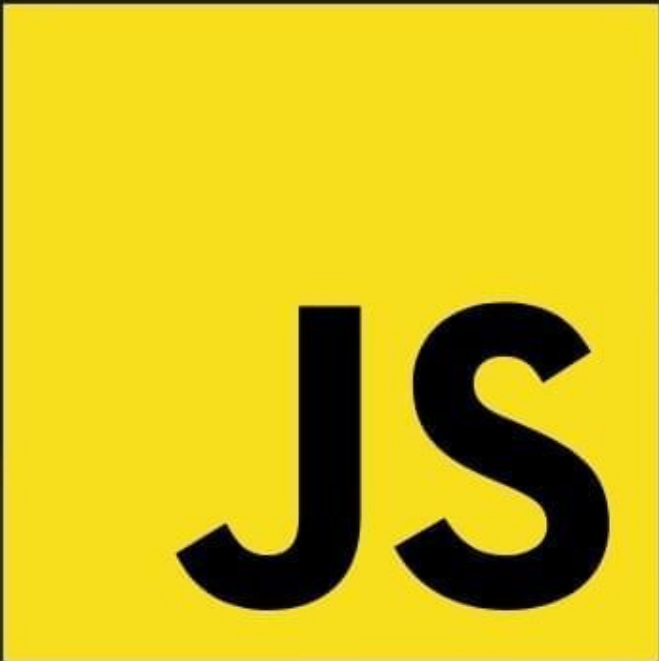


# Types Of Functions

6 Ways

The JavaScript logo, consisting of the letters 'JS' in a bold, black, sans-serif font, centered within a bright yellow square.

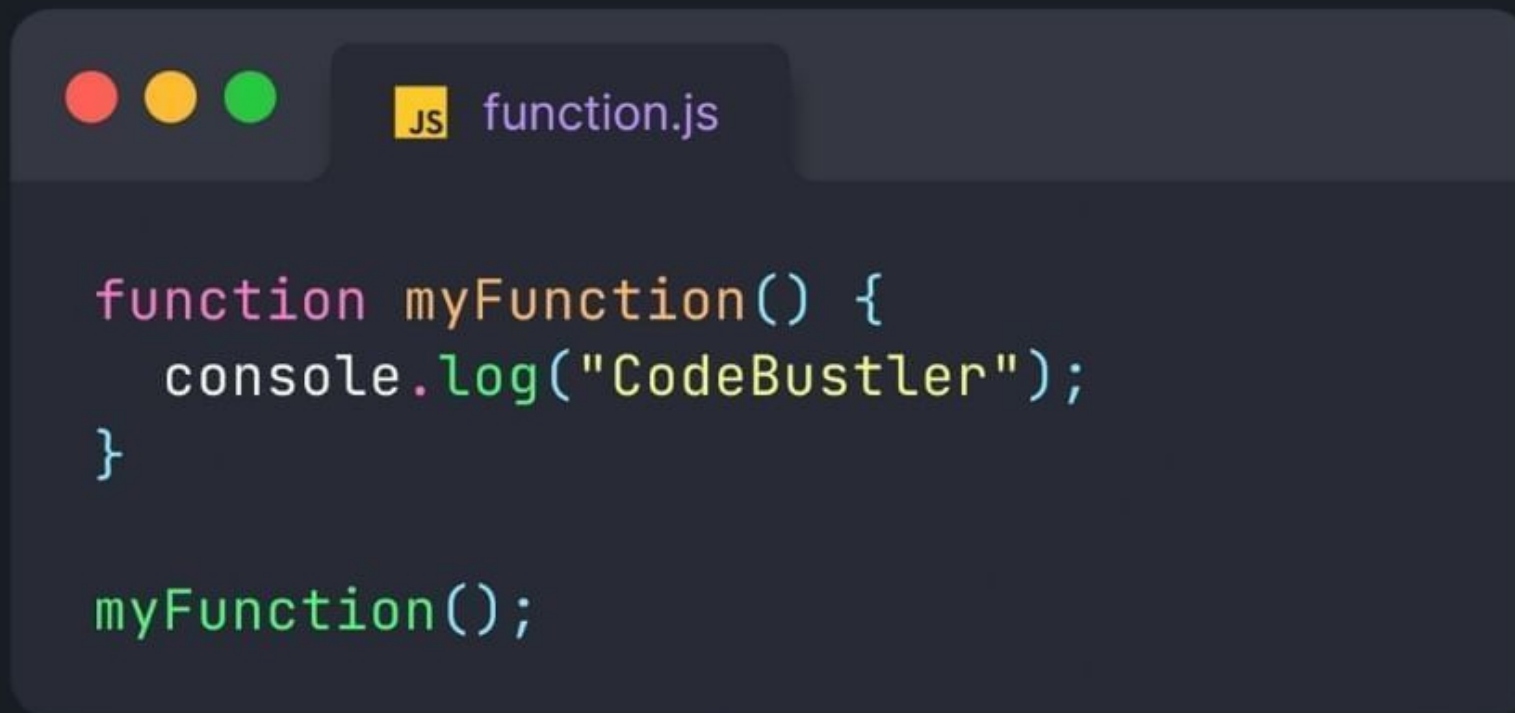
Mallikarjun | @CodeBustler

# JS **Functions** | 6 Ways

- **Named Functions**
- **Anonymous Functions**
- **Arrow Functions**
- **Immediately Invoked Function Expressions (IIFE)**
- **Higher Order Functions**
- **Constructor Functions**

# 1. Named Functions

Lacacy / Traditional way



```
function myFunction() {  
  console.log("CodeBustler");  
}  
  
myFunction();
```

The image shows a code editor window with a dark background. The title bar at the top has three colored circles (red, yellow, green) on the left and a tab labeled 'JS function.js' in the center. The code inside the editor is written in a light green monospace font. It defines a function named 'myFunction' that takes no arguments and logs 'CodeBustler' to the console. Below the function definition, the function is called with 'myFunction();'.



## 2. Anonymous Functions

Anonymous Functions | without name, used as function expression or arguments

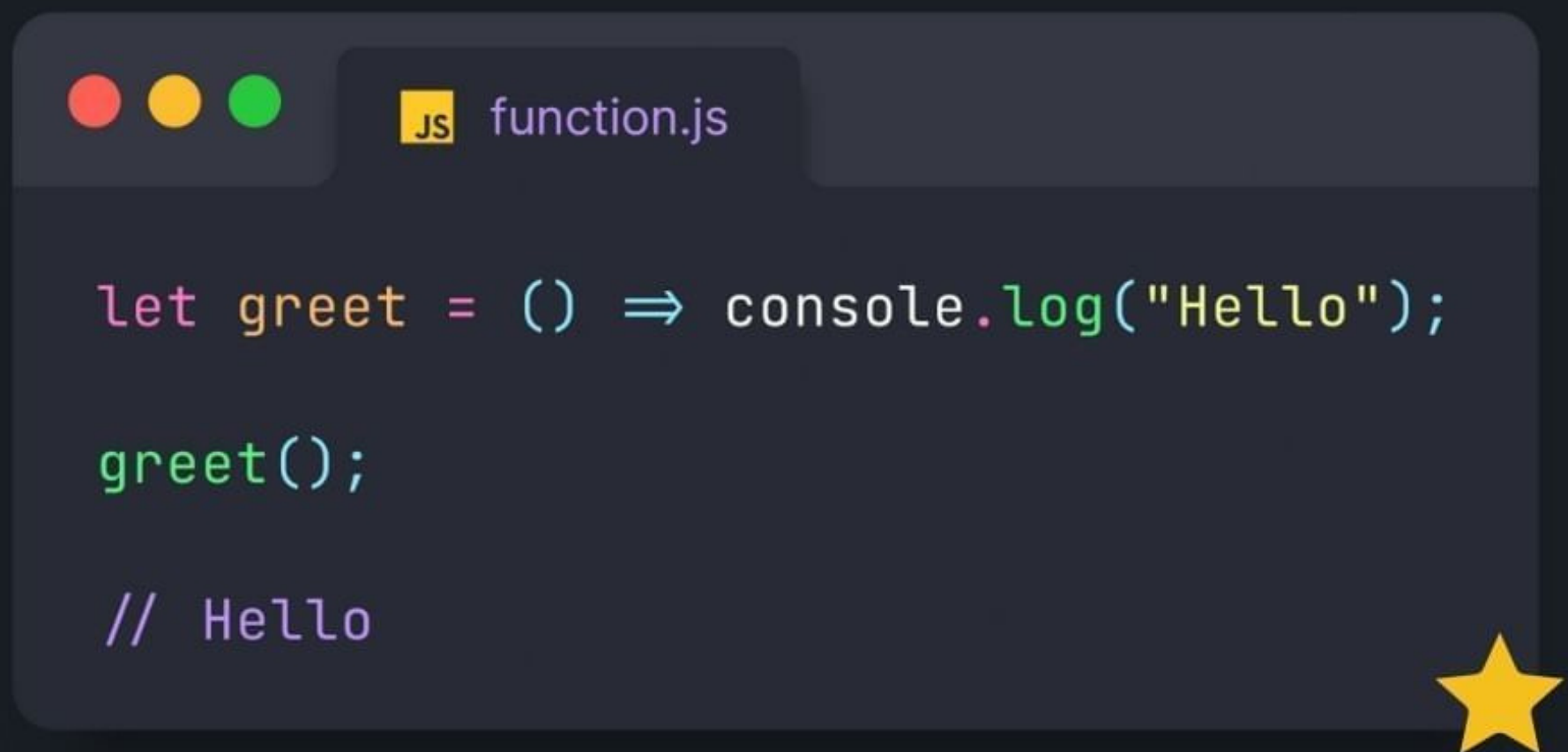


JS function.js


```
let greet = function (name) {  
  console.log(`Hello ${name}`);  
};  
  
greet("Sai");  
  
// Hello Sai
```

### 3. **Arrow** Functions

Arrow Functions Introduced in **ES6**,  
Shorter Syntax & one-lined functions



```
let greet = () => console.log("Hello");  
  
greet();  
  
// Hello
```





## 4. **IIFE** Functions

**Immediately Invoked Function Expressions (IIFE)** Executed immediately after their creation. Used to create private scopes and avoid polluting the global namespace.



A code editor window with a dark theme. The title bar shows three colored circles (red, yellow, green) and a tab labeled 'function.js' with a 'JS' icon. The code is as follows:

```
(function () {  
    let str = "Good Evening";  
    console.log(str);  
})();  
  
// Good Morning (Immediate Invoked)
```

## 5. Higher Order Func

Functions that take one or more **functions as arguments** or return a function.

eg: `map()`, `filter()`, `reduce()`



JS

function.js

```
let ary = [10, 20, 23];

let newArray = ary.map((element) => element + 10);
console.log(newArray);

// [20, 30, 33]
```

## 6. Constructor Function

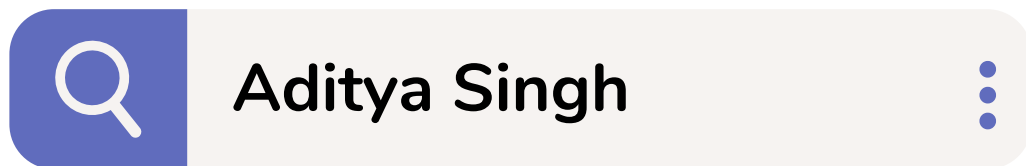
Used as blueprints for **creating objects with similar properties** and methods. They are invoked using the **new** keyword to create instances of objects.



```
function Person(name, place) {  
  this.name = name;  
  this.place = place;  
}  
  
let user1 = new Person("CodeBustler", "India");  
  
console.log(`Hello everyone this is ${user1.name},  
            and i am from ${user1.place}`);  
  
// Hello everyone this is CodeBustler, and i am from India
```



# Follow **Aditya**



For more amazing content related

- ✓ web development
- ✓ Programming
- ✓ Javascript



LinkedIn:

<https://www.linkedin.com/in/aditya-singh621>