Functions in JavaScript

In JavaScript, a function can be declared using the "function" keyword, followed by the function name, a set of parentheses (which may include parameters), and a block of code to be executed when the function is called. For example:

```
function Function1(para1, param2) {
    // code to be executed
}
```

This function can be invoked by calling its name followed by parentheses containing any necessary arguments, like so:

```
Function1(arg1, arg2);
```

Function Expression Syntax in JavaScript

In JavaScript, a function expression is a way to create a function and assign it to a variable. It can be used in place of a function declaration, and has some important differences in behavior.

```
const funcVar = function func() {
    console.log("Hello World!");
};
```

One of the main differences between function expressions and function declarations is that function expressions are not hoisted, which means that they cannot be called before they are defined in the code. In contrast, function declarations are hoisted, meaning that they can be called before they are defined.

Argument Object of a Function in JavaScript

You can use the arguments keyword inside the function to refer to all the arguments passed to the function, even if you didn't specify the number of arguments in the function definition.

```
function myFunc1() {
    console.log(arguments); // logs [1, 2, 3]

    console.log(arguments.length); // logs 3

    console.log(arguments[0], arguments[1], arguments[2], arguments[3]); // logs
[1, 2, 3]
}
myFunc1(1, 2, 3);
```

Anonymous Functions

In JavaScript, an anonymous function is a function without a name. Anonymous functions are often used in situations where a function is only needed once or for a very specific purpose, and giving it a name is not necessary.

```
const myFunc1 = function () {
    console.log("Hello World!");
};
```