
```
function add(x, y) {
    return x+y
}
```

JavaScript Functions: Exploring Types and Features

Wasif Alam
@wasif alam

1 Function Declarations

```
1  function greet(name) {
2    return `Hello, ${name}!`;
3  }
4
5  // Call the function
6  const message = greet('Alice');
7  console.log(message); // Output: Hello, Alice!
```

Features:

- Clear and straightforward syntax.
- Hoisting: Can be called before they are defined.
- Suitable for defining functions globally and locally.

Wasif Alam
@wasif alam

2 Function Expressions

```
1  const greet = function (name) {
2    return `Hello, ${name}!`;
3  };
4
5  // Call the function
6  const message = greet('Bob');
7  console.log(message); // Output: Hello, Bob!
```

Features:

- Define functions as expressions within other expressions.
- Can be anonymous (unnamed) or named.
- Use functions as arguments or store them in variables.

Wasif Alam
@wasif alam

3 Arrow Functions:

```
1  const greet = (name) => `Hello, ${name}!`;
2
3  // Call the function
4  const message = greet("Charlie");
5  console.log(message); // Output: Hello, Charlie!
```

Features:

- Concise syntax with implicit return.
- No binding of 'this': Lexical scoping.
- Perfect for short, one-liner functions and callback functions.

Wasif Alam
@wasif alam

4 Immediately Invoked Function Expressions (IIFE)

```
(function () {
  const greetMessage = 'Hello, John!';
  console.log(greetMessage); // Output: Hello, John!
})();
```

Features:

- Self-invoking: Executes automatically upon declaration.
- Encapsulates code: Provides a private scope.
- Ideal for isolation: Commonly used in modular code.

Wasif Alam
@wasif alam

🚀 Ready to master JavaScript functions? Dive in, follow for more coding insights, and share your thoughts in the comments below. Let's learn and grow together! 💬👇

Wasif Alam
@wasif alam