www.scribbler.live

What is this ?

- this refers to the context within which a function is executed. It dynamically binds to different values depending on how a function is called.
- In the global context, this refers to the global object in Node.js.

JS

```
console.log(this === window);
// Outputs: true (in the browser)
```

Function Context

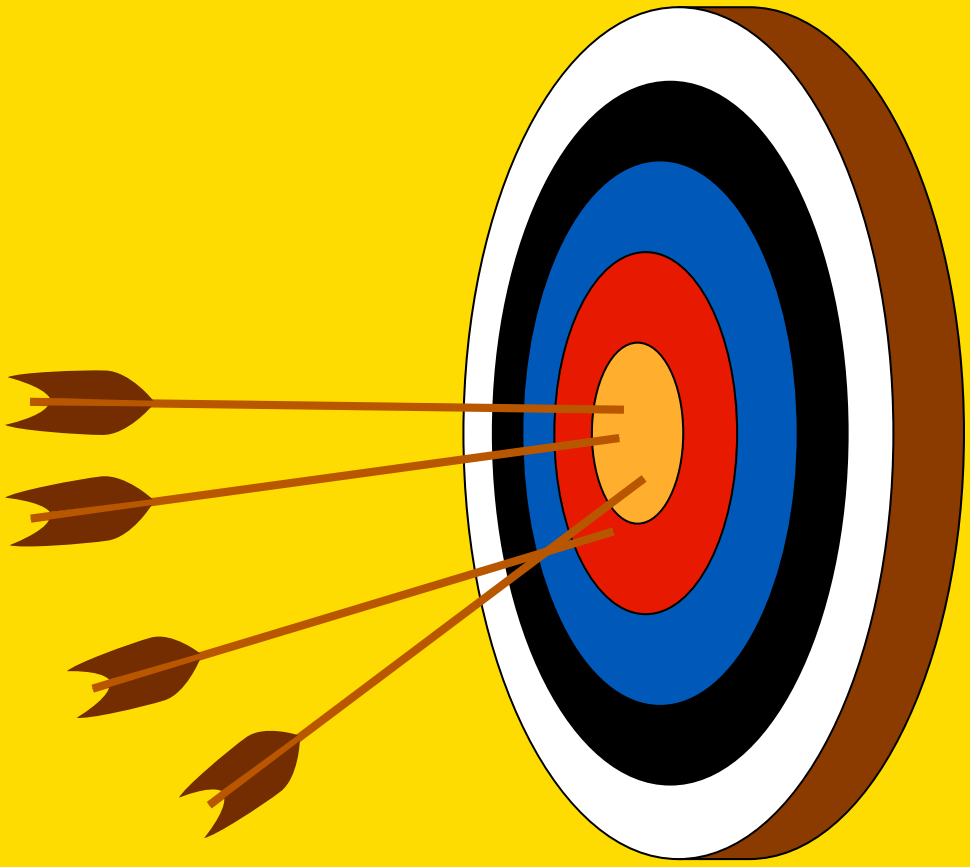
- In regular functions, this refers to the object that calls the function.
- If no explicit context is provided, it defaults to the global object.

```js
function greet() {
  console.log("Hello, " + this.name);
}

const person = { name: "Alice", greet: greet };
person.greet(); // Outputs: Hello, Alice
```

Arrow Functions

- Unlike regular functions, arrow functions do not have their own this context. Instead, they inherit this from the surrounding lexical scope.

```JS
const printName = () ⇒ {
  console.log("My name is " + this.name);
};

const user = { name: "Bob", printName: printName };
user.printName(); // Outputs: My name is
```



Event Handlers

- In event handlers, this typically refers to the element that triggered the event. However, it may vary depending on how the handler is attached.

JS

```
<button onclick="console.log(this)">Click me</button>
 // Outputs: {}
```

Constructor Functions

- Inside constructor functions, this refers to the newly created instance of the object being constructed.

JS

```
function Person(name) {
  this.name = name;
}

const alice = new Person("Alice");
console.log(alice.name);
// Outputs: Alice
```

Keep Exploring Javascript with us!

Share this with a friend who needs it and make sure to practice these in scribbler.

Scribbler.live

Free and Open Interface to experiment JavaScript