# Presentation :This keyword

**UROOJ KHALID**
**ALIZA M.ZAHID**
**BATCH:09**

## Introduction to the 'this' Keyword

What is the 'this' Keyword?

In JavaScript, the 'this' keyword refers to the context in which a function is executed.

It allows access to the current object or the object that the function belongs to.

The Importance of 'this‘:

Understanding 'this' is crucial for object-oriented programming in JavaScript.

It provides a way to access object properties and methods dynamically.

'this' plays a significant role in event handling, callbacks, and method chaining.

## THIS IN METHOD:

- ‘**this'** keyword is often used within methods to refer to the current object on which the method is being called.

EXAMPLE:

```
const person = {
  name: 'John',
  age: 30,
  greet() {
    console.log(`Hello, my name is ${this.name}. I am ${this.age} years old.`);
  }
};
```

person.greet(); // **Output:** Hello, my name is John. I am 30 years old.

## EXPLICITLY USING THIS KEYWORD METHODS

**USING CALL()**:The call() method invokes a function and explicitly set the value of this with in the function.

**EXAMPLE:**

```
const person = {
  name: 'John',
  greet: function () {
    console.log(`Hello, ${this.name}!`);
  }
};

const anotherPerson = {
  name: 'Alice'
};

person.greet.call(anotherPerson); // Output: Hello, Alice!
```

**USING APPLY():** it takes ab array-like object as a second Argument,which allows us to pass arguments to the function
Being invoked.

**EXAMPLE:**

```
const person = {
  name: 'John',
  greet: function (greeting) {
    console.log(`${greeting}, ${this.name}!`);
  }
};

const anotherPerson = {
  name: 'Alice'
};

person.greet.apply(anotherPerson, ['Hey']); // Output: Hey, Alice!
```

**USING BIND():**creates a new function with a permanently bound 'this' value. It returns a new function that can be invoked later

**EXAMPLE:**

```
const person = {
  name: 'John',
  greet: function () {
    console.log(`Hello, ${this.name}!`);
  }
};

const anotherPerson = {
  name: 'Alice'
};

const boundGreet = person.greet.bind(anotherPerson);
boundGreet(); // Output: Hello, Alice!
```

## this in Event Handlers:

In HTML event handlers, `this` refers to the HTML element that received the event:
E.g:

```
<button onclick="this.style.display='none'">
  Click to Remove Me!
</button>
```

THANKYOU!!!!