ThePower



this es una palabra clave en JavaScript que se refiere al objeto en el contexto actual. El valor de this puede cambiar dependiendo del contexto en el que se utiliza.

En el contexto de una función global o en el nivel principal del código, this se refiere al objeto global, que es window en el navegador y global en Node.js.

```
console.log(this === window); // true
```

En el contexto de un método de un objeto, **this** se refiere al objeto al que pertenece el método.

```
let obj = {
  name: 'John',
  greet: function() {
    console.log(`Hello, my name is ${this.name}`);
  };

obj.greet(); // 'Hello, my name is John'
```

También puedes usar this en una función anónima que se pasa como un argumento a otra función. En este caso, this se refiere al objeto que se ha pasado como argumento.

```
let obj = {
  name: 'John'
};

setTimeout(function() {
  console.log(`Hello, my name is ${this.name}`);
}, 1000);
```

```
setTimeout(function() {
  console.log(`Hello, my name is ${this.name}`);
}.bind(obj), 1000);
```

El primer ejemplo imprimirá undefined, ya que this se refiere al objeto global en ese contexto. El segundo ejemplo imprimirá 'Hello, my name is John', ya que se ha usado el método bind() para fijar el valor de this al objeto obj.

## bind || call || apply

bind(), call() y apply() son métodos que se pueden usar con la palabra clave this en JavaScript para cambiar el contexto en el que se ejecuta una función.

El método bind() se utiliza para crear una nueva función a partir de una función existente, fijando el valor de this al objeto especificado. La nueva función se puede utilizar en cualquier momento, incluso en un contexto diferente al que se creó.

```
let obj = {
  name: 'John'
};

let greet = function() {
  console.log(`Hello, my name is ${this.name}`);
}.bind(obj);

greet(); // 'Hello, my name is John'
```

El método call() se utiliza para llamar a una función con un valor específico para this, y puede pasar argumentos adicionales a la función a través de los parámetros.

```
let obj = {
  name: 'John'
};
let greet = function(greeting) {
```

```
console.log(`${greeting}, my name is ${this.name}`);
};
greet.call(obj, 'Hello'); // 'Hello, my name is John'
```

El método apply() es similar a call(), pero se usa para pasar los argumentos a la función como un array en lugar de como una lista separada por comas.

```
let obj = {
  name: 'John'
};

let greet = function(greeting) {
  console.log(`${greeting}, my name is ${this.name}`);
};

greet.apply(obj, ['Hello']); // 'Hello, my name is John'
```

Aquí hay algunos ejemplos de cómo se pueden usar <a href="bind(")">bind(")</a>, <a href="call(")">call(")</a> y <a href="apply(")" en una aplicación real:

bind() se puede usar para crear una nueva función que tenga un valor de this fijo para su uso posterior. Por ejemplo:

```
let obj = {
  name: 'John',
  age: 30
};

// Creamos una nueva función que muestra el nombre y la edad de una persona
let showPerson = function() {
  console.log(`Name: ${this.name}, Age: ${this.age}`);
}.bind(obj);

// Podemos usar la nueva función en cualquier momento
showPerson(); // 'Name: John, Age: 30'
```

call() se puede usar para llamar a una función con un valor de this específico y pasar argumentos adicionales a la función. Por ejemplo:

```
let obj1 = {
  name: 'John',
  age: 30
};

let obj2 = {
  name: 'Jane',
  age: 35
};

let showPerson = function(greeting) {
  console.log(`${greeting}, my name is ${this.name} and I am ${this.age} years old`);
};

// Podemos usar call() para llamar a la función con diferentes valores de this y argumento s
  showPerson.call(obj1, 'Hello'); // 'Hello, my name is John and I am 30 years old'
  showPerson.call(obj2, 'Hi'); // 'Hi, my name is Jane and I am 35 years old'
```

apply() se puede usar de la misma manera que call(), pero se usa para pasar los argumentos a la función como un array en lugar de como una lista separada por comas. Por ejemplo:

```
let obj1 = {
  name: 'John',
  age: 30
};

let obj2 = {
  name: 'Jane',
  age: 35
};

let showPerson = function(greeting, exclamation) {
    console.log(`${greeting}, my name is ${this.name}${exclamation} and I am ${this.age} yea
  rs old`);
};

// Podemos usar apply() para llamar a la función con diferentes valores de this y argument
  os
  showPerson.apply(obj1, ['Hello', '!']); // 'Hello, my name is John! and I am 30 years old'
  showPerson.apply(obj2, ['Hi', '!']); // 'Hi, my name is Jane! and I am 35 years old'
```

Espero que estos ejemplos te hayan ayudado a entender mejor cómo se pueden usar bind(), call() y apply()