Introduction to JavaScript new.target Metaproperty

Summary: in this tutorial, you learn about the JavaScript `new.target` metaproperty that detects whether a function or constructor was called using the `new` operator.

Introduction to JavaScript new.target

ES6 provides a metaproperty named `new.target` that allows you to detect whether a function or constructor was called using the `new` operator.

The `new.target` consists of the `new` keyword, a dot, and `target` property. The `new.target` is available in all functions.

However, in arrow functions, the `new.targe` t is the one that belongs to the surrounding function.

The `new.target` is very useful to inspect at runtime whether a function is being executed as a function or as a constructor. It is also handy to determine a specific derived class that was called by using the `new` operator from within a parent class.

JavaScript new.target in functions

Let's see the following `Person` constructor function:

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

You can create a new object from the `Person` function by using the `new` operator as follows:

```
let john = new Person('John');
console.log(john.name); // john
```

Or you can call the `Person` as a function:

```
Person('Lily');
```

Because the `this` is set to the global object i.e., the `window` object when you run JavaScript in the web browser, the `name` property is added to the `window` object as follows:

```
console.log(window.name); //Lily
```

To help you detect whether a function was called using the new operator, you use the `new.target` metaproperty.

In a regular function call, the `new.target` returns `undefined`. If the function was called with the `new` operator, the `new.target` returns a reference to the function.

Suppose you don't want the `Person` to be called as a function, you can use the `new.target` as follows:

```
function Person(name) {
    if (!new.target) {
        throw "must use new operator with Person";
    }
    this.name = name;
}
```

Now, the only way to use `Person` is to instantiate an object from it by using the `new` operator. If you try to invoke it like a regular function, you will encounter an error.

JavaScript new.target in constructors

In a class constructor, the `new.target` refers to the constructor that was invoked directly by the `new` operator. It is `true` if the constructor is in the parent class and was delegated from the

constructor of the child class:

```
class Person {
    constructor(name) {
        this.name = name;
        console.log(new.target.name);
    }
}

class Employee extends Person {
    constructor(name, title) {
        super(name);
        this.title = title;
    }
}

let john = new Person('John Doe'); // Person
let lily = new Employee('Lily Bush', 'Programmer'); // Employee
```

In this example, `new.target.name` is the human-friendly name of the constructor reference of new.target

In this tutorial, you have learned how to use the JavaScript `new.target` metaproperty to detect whether a function or constructor was called using the `new` operator.