JS JavaScript TUTORIAL

JavaScript Static Methods

Summary: in this tutorial, you'll learn about the JavaScript static methods and how to use them effectively.

Introduction to the JavaScript static methods

By definition, static methods are bound to a class, not the instances of that class. Therefore, static methods are useful for defining helper or utility methods.

To define a static method before ES6, you add it directly to the constructor of the class. For example, suppose you have a `Person` type as follows:

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

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

The following adds a static method called `createAnonymous()` to the `Person` type:

```
Person.createAnonymous = function (gender) {
    let name = gender == "male" ? "John Doe" : "Jane Doe";
    return new Person(name);
};
```

The `createAnonymous()` method is considered a static method because it doesn't depend on any instance of the `Person` type for its property values.

To call the `createAnonymous()` method, you use the `Person` type instead of its instances:

```
var anonymous = Person.createAnonymous();
```

JavaScript static methods in ES6

In ES6, you define static methods using the `static` keyword. The following example defines a static method called `createAnonymous()` for the `Person` class:

```
class Person {
        constructor(name) {
                this.name = name;
        }
        getName() {
                return this.name;
        }
        static createAnonymous(gender) {
                let name = gender == "male" ? "John Doe" : "Jane Doe";
                return new Person(name);
        }
}
```

To invoke the static method, you use the following syntax:

```
let anonymous = Person.createAnonymous("male");
```

If you attempt to call the static method from an instance of the class, you'll get an error. For example:

```
let person = new Person('James Doe');
let anonymous = person.createAnonymous("male");
```

Error:

```
TypeError: person.createAnonymous is not a function
```

Calling a static method from the class constructor or an instance method

To call a static method from a class constructor or an instance method, you use the class name, followed by the `.` and the static method:

```
className.staticMethodName();
```

Alternatively, you can use the following syntax:

```
this.constructor.staticMethodName();
```

Summary

- JavaScript static methods are shared among instances of a class. Therefore, they are bound to the class.
- Call the static methods via the class name, not the instances of that class.
- Use the `className.staticMethodName()` or `this.constructor.staticMethodName()` to call a static method in a class constructor or an instance method.