The JavaScript this Keyword

- description : The JavaScript this Keyword
- author : 오션
- email : shlim@repia.com
- lastupdate : 2021-04-07

Ref

The JavaScript this Keyword

예제

```
<!DOCTYPE html>
<html>
<body>
    <h2>The JavaScript <i>this</i> Keyword</h2>
    <p>In this example, <b>this</b> represents the <b>person</b> object.</p>
    <p>Because the person object "owns" the fullName method.</p>
    <p id="demo"></p>
    <script>
        // Create an object:
        var person = {
            firstName: "John",
            lastName: "Doe",
            id: 5566,
            fullName: function () {
                return this.firstName + " " + this.lastName;
            }
        };
        // Display data from the object:
        document.getElementById("demo").innerHTML = person.fullName();
    </script>
</body>
</html>
```

예제 설명

오브젝트 메서드에서, this는 메서드의 “ 소유자 ” 를 나타냅니다.

본 페이지 상단의 첫 번째 예제에서, this는 person 오브젝트를 나타냅니다.

person 오브젝트는 **fullName** 메서드의 **소유자**입니다.

What is this?

JavaScript `this` 키워드는 그것이 속한 오브젝트를 참조합니다.

`this` 키워드는 사용 위치에 따라 다른 값을 가집니다.

- 메서드에서 `this`는 소유자 오브젝트**(owner object)**를 나타냅니다.
- 단독으로는, `this`는 전역 오브젝트**(global obejct)**를 나타냅니다.
- 함수에서, `this`는 전역 오브젝트**(global obejct)**를 나타냅니다.
- 스트릭트 모드(strict mode)에서 `this`는 `undefined`입니다.
- 이벤트에서, `this`는 이벤트를 받은 해당 요소를 나타냅니다.
- `call()` 및 `apply()`와 같은 메서드는 `this`가 모든 객체를 참조하게 할 수 있습니다.

this Alone

`this`가 단독으로 사용되는 경우, **소유자**는 전역 개체이므로, `this`는 전역 개체를 참조합니다.

브라우저 창에서, 전역 개체는 `[object window]`입니다.

예제

```
<!DOCTYPE html>
<html>
<body>
    <h2>The JavaScript <i>this</i> Keyword</h2>
    <p>In this example, <b>this</b> refers to the window Object.</p>
    <p id="demo"></p>
    <script>
        var x = this;
        document.getElementById("demo").innerHTML = x;
    </script>
</body>
</html>
```

예제

스트릭트 모드(strict mode)에서, `this`가 단독으로 사용될 경우,
`this`는 전역 오브젝트 `[object Window]`를 참조합니다.

```
<!DOCTYPE html>
<html>
```

```
<body>
    <h2>The JavaScript <i>this</i> Keyword</h2>
    <p>In this example, <b>this</b> refers to the window Object:</p>
    <p id="demo"></p>
    <script>
        "use strict";
        var x = this;
        document.getElementById("demo").innerHTML = x;
    </script>
</body>
</html>
```

this in a Function (Default)

JavaScript 함수에서 함수의 소유자는 this에 대한 기본 바인딩입니다.

그래서 함수에서, this는 전역 오브젝트 [object Window]를 의미합니다.

예제

```
<!DOCTYPE html>
<html>
<body>
    <h2>JavaScript <i>this</i> Keyword</h2>
    <p>In this example, <b>this</b> represents the object that "owns"
myFunction:</p>
    <p id="demo"></p>
    <script>
        document.getElementById("demo").innerHTML = myFunction();
        function myFunction() {
            return this;
        }
    </script>
</body>
</html>
```

this in a Function (Strict)

JavaScript **스트릭트 모드(strict mode)**는 기본 바인딩을 허용하지 않습니다.

그래서 스트릭트 모드의 함수에서 사용되는 경우, this는 undefined입니다.

예제

```
<!DOCTYPE html>
```

```
<html>
<body>
    <h2>The JavaScript <i>this</i> Keyword</h2>
    <p>In a function, by default, <b>this</b> refers to the Global
object.</p>
    <p>In strict mode, <b>this</b> is <b>undefined</b>, because strict mode
does not allow default binding:</p>
    <p id="demo"></p>
    <script>
        "use strict";
        document.getElementById("demo").innerHTML = myFunction();
        function myFunction() {
            return this;
        }
    </script>
</body>
</html>
```

this in Event Handlers

HTML 이벤트 핸들러에서, this는 해당 이벤트를 받은 HTML 요소를 참조합니다.

예제

```
<!DOCTYPE html>
<html>
<body>
    <h2>The JavaScript <i>this</i> Keyword</h2>
    <button onclick="this.style.display='none'">Click to Remove Me!</button>
</body>
</html>
```

Object Method Binding

하기 예제들에서, this는 **person** 오브젝트입니다.(person 오브젝트는 함수의 소유자입니다.)

예제

```
<!DOCTYPE html>
<html>
<body>
    <h2>The JavaScript <i>this</i> Keyword</h2>
    <p>In this example, <b>this</b> represents the person object that "owns"
myFunction method.</p>
```

```
    <p id="demo"></p>
    <script>
        //Create an object:
        var person = {
            firstName: "John",
            lastName: "Doe",
            id: 5566,
            myFunction: function () {
                return this;
            }
        };
        // Display data from the object:
        document.getElementById("demo").innerHTML = person.myFunction();
    </script>
</body>
</html>
```

예제

```
<!DOCTYPE html>
<html>
<body>
    <h2>The JavaScript <i>this</i> Keyword</h2>
    <p>In this example, <b>this</b> represents the person object</p>
    <p>Because the person object "owns" the fullname method.</p>
    <p id="demo"></p>
    <script>
        //Create an object:
        var person = {
            firstName: "John",
            lastName: "Doe",
            id: 5566,
            fullName: function () {
                return this.firstName + " " + this.lastName;
            }
        };
        // Display data from the object:
        document.getElementById("demo").innerHTML = person.fullName();
    </script>
</body>
</html>
```

즉, **this.firstName**은 **this**(person) 오브젝트의 **firstName** 속성을 의미합니다.

Explicit function Binding

`call()` 및 `apply()` 메서드는 미리 정의된 JavaScript 메서드입니다.

둘 다 다른 오브젝트를 인수(argument)로 사용하여 오브젝트 메서드를 호출하는 데 사용할 수 있습니다.

아래 예에서 person2를 인수로 사용하여 person1.fullName을 호출하면, this는 person1의 메소드인 경우에도 person2를 참조합니다.

예제

```
<!DOCTYPE html>
<html>
<body>
    <h2>The JavaScript <i>this</i> Keyword</h2>
    <p>In this example, <b>this</b> refers to person2, even if it is a
method of person1:</p>
    <p id="demo"></p>
    <script>
        var person1 = {
            fullName: function () {
                return this.firstName + " " + this.lastName;
            }
        }
        var person2 = {
            firstName: "John",
            lastName: "Doe",
        }
        var x = person1.fullName.call(person2);
        document.getElementById("demo").innerHTML = x;
    </script>
</body>
</html>
```

오션,, The, JavaScript, this, Keyword

From:
http://125.132.25.164/dokuwiki/ - **문제를 잘 정의하는 것은 문제를 절반 해결한 것이다. - 2023.12**

Permanent link:
http://125.132.25.164/dokuwiki/doku.php?id=wiki:javascript:javascript_note:js_this_keyword&rev=1617780475

Last update: **2022/03/10 19:52**