# JavaScriptのthisキーワードをちゃんと理解する

白石俊平（あゆた）
2008/04/09 19:00



**JavaScriptで「自分自身」にアクセスするthisキーワードについてきちんと理解していますか？今回はthisキーワードに関してです。**



## 前回までのおさらいと今回のあらすじ

前回までの記事で、JavaScriptでオブジェクト指向を行うにあたって必要な知識を一通り網羅しました。その中で、オブジェクトに属する関数＝メソッドについても触れました。

今回は、オブジェクトやメソッドと深い関わりのあるキーワード、「this」について解説したいと思います。

最新特集【一覧】


「Adobe Digital Publishing Suite」を試す（3）


画像ファイルをPDFにまとめるSnow Leopardの便利なコマンド

人気記事【一覧】

4 :画像ファイルをPDFにまとめるSnow Leopardの便利なコマンド

## thisキーワードについてきっちり理解する

「thisキーワードについてお話しする」と言いましたが、実はthisキーワードは連載中で既に触れています。連載第９回で、このように述べています。

「（thisキーワードは）実行中のコードが「自分自身」を表すオブジェクトにアクセスするためのキーワードです。」

「実行中のコード」とされているのは、関数（メソッド）などの実行可能なコードということです。つまり、thisキーワードはメソッドをメンバに含むオブジェクトを指す、と言って良いでしょう。

例えば、以下のようなオブジェクトがあったとします。

```
var shiraishi = {
  name: "白石",
  hello: function() {
    alert("こんにちは！" + this.name);
  }
};
```

このhello()メソッドを呼び出すと、「こんにちは！白石」と表示されます。何の不思議もありませんね。thisキーワードを使用しているのはhello()メソッドであり、同メソッドが所属しているのはオブジェクトshiraishiなのですから。

ですが、JavaScriptは非常に動的な言語なので、*あるオブジェクトのメソッドを他のオブジェクトにコピー*することもできます。

```
var tanaka = {
  name: "田中"
};
// shiraishi.helloをtanaka.helloにコピー
tanaka.hello = shiraishi.hello;

tanaka.hello(); // 何が表示される？
```

さあ、最後の「tanaka.hello()」の呼び出しでは何が表示されるでしょうか。

答えはお分かりですね？ そう、「こんにちは！田中」です。

hello()関数の中ではthis.nameを参照しています。そして、thisが指すものは、「メソッドをメンバに含むオブジェクト」です。つまり、「tanaka.hello()」の呼び

出し中、メソッド内のthisオブジェクトは変数tanakaを指しているのです。

このように、JavaScriptにおけるメソッドとは、特定のオブジェクトと密に結びついているものではありません。そのため、プログラムの文脈によっては、thisが指すオブジェクトも全く異なってくることを覚えておいてください。

page: 2

### イベントハンドラ中のthisは混乱のもと

前のページで述べたことは、実際それほど難しいことではありません。ですが、プログラムの書き方によっては（特にオブジェクト指向的なコードを書いているとき）、thisが何を指すのか混乱してしまい、バグの元になることもあります。

以下のコードを見てください。画面にボタンを一つだけ配置し、押されると「こんにちは！白石」と表示するのが目的のプログラムです。

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script type="text/javascript">

var shiraishi = {
  name: "白石",
  // initメソッドは、body.onloadから呼ばれる
  init: function() {
    var button = document.getElementById("button");
    // ボタンのイベントハンドラに、メソッドhelloをセット
    button.onclick = this.hello;
  },
  hello: function() {
    alert("こんにちは！" + this.name);
  }
};
</script>
</head>
<body onload="shiraishi.init()">
  <button id="button" name="ボタン">クリック!</button>
</body>
</html>
```

ですが、このプログラムを実行した結果は「こんにちは！白石」ではなく、以下のようになりました。


なぜこうなったかお分かりですか？（ヒント：buttonタグのname属性）

page: 3

### 前ページの解説

前ページのプログラムが正しく動作しなかった理由は、実は最初にお見せした例と全く変わりません（だから、前ページのバグをすぐに見つけられた方もいらっしゃるでしょう）。

問題は、shiraishi.init()内で行っているこの処理ですね。

```
button.onclick = this.hello;
```

この状態でボタンをクリックされると、onclickイベントハンドラ＝hello()メソッドが実行されます。その際、hello()の中で用いている"this"はshiraishiオブジェクトではなく、*ボタンのDOMオブジェクト*になってしまいます。だから、this.nameが「buttonタグのname属性」を指してしまっていると言うわけです。

### 普通の関数中でthisを使うとどうなる？

最後に、thisに関してもう一つだけ簡単な実験をしておきましょう。

以下のコードを見てください。

```
<script type="text/javascript">
// トップレベルのコード中でthisを使用する
alert(this);
function init() {
  // トップレベルに定義された関数内でthisを使用する
  alert(this);
}
</script>
```

このコードで使われている二つのalert()は、どちらも[object Window]と表示されます。つまり、ブラウザのwindowオブジェクトですね。

この事実を正確に文章で表すなら、「トップレベルのプログラムコード/関数内でthisを利用すると、『*グローバルオブジェクト*』を指す」と言うことになります。

グローバルオブジェクトと言うのは、簡単に言うと「グローバルな変数/関数の置き場所となるオブジェクト」です。ブラウザ上のJavaScriptにおいてはwindowオブジェクトになりますし、その他の実行環境では違うものになることがあります。

### まとめと次回予告

今回は、「this」キーワードについて掘り下げた解説を行いました。オブジェクト指向的なJavaScriptプログラミングを行う際には、今回お見せしたように、thisが指す対象を間違ってしまうことが多いので注意してください。

次回は、今回の知識を踏まえた上で、JavaScriptの関数に関連する事柄（call()/apply()、argumentsオブジェクトなど）を網羅したいと思います。