Wild Friday Night with Javascript Controllers

Posted At : February 20, 2009 10:53 PM | Posted By : Jon Hartmann
Related Categories: Javascript

All because of Ben Nadel, I'm writing Javascript examples instead of playing Fallout 3 tonight... Ok, so in my post **evolution of a js controller scheme**, Ben commented that he wasn't comfortable with the idea that the this value inside my controller was the window object, rather than the controller itself. So, I've set out to see if I can rework the controller to use "new" and grow up to be a real object. Read more to see how I do.

Here We Go Again

Ok, so my first attempt will be to convert the final controller object we ended up with in **evolution of a js controller scheme** and turn it into an object that can be created with the new keyword. I'm using the same HTML as before:

```
<html>
    <head>
        <title>Test</title>
        <script>
            ...
        </script>
    </head>
    <body>
        <button id="button1">I'm a Button</button>
    </body>
</html>
```

Ok, lets start with the basics and go from there. We create a function called Controller, and use it as a constructor to create a new object. Then we'll alert that new object, just so you can be sure that we are doing it right.

```
<script>
    function Controller() {


    };


    var myController = new Controller();


    alert(myController);
</script>
```

First step down. Now lets add in our config object and the ability to pass in a userID through the options.

```
<script>
    function Controller(options) {
        var config = {
            IDs: {
                button: 'button1'
            }
        };
```

```
        // Private
        var options = options;
    };

    var myController = new Controller({userID: 1234});

    alert(myController);
</script>
```

Now lets add in our private methods getUserID() and handleClick(), and the private variable button.

```
<script>
    function Controller(options) {
        var config = {
            IDs: {
                button: 'button1'
            }
        };

        // Private
        var options = options;
        var button;

        // Private Methods
        function getUserID() {
            return options.userID;
        }

        function handleClick () {
            alert(getUserID());
        }
    };

    var myController = new Controller({userID: 1234});

    alert(myController);
</script>
```

Next up is our public method, setUserID().

```
<script>
    function Controller(options) {
        var config = {
            IDs: {
                button: 'button1'
            }
        };

        // Private
        var options = options;
        var button;

        // Private Methods
        function getUserID() {
            return options.userID;
        }

        function handleClick () {
            alert(getUserID());
```

```
        }

        // Public Methods
        this.setUserID = function (userID) {
            options.userID = userID;
        }
    };

    var myController = new Controller({userID: 1234});

    alert(myController);
</script>
```

If you'll notice, I assigned this function as a public method using the this keyword to refer to to the resulting object itself. Happy yet Ben? :)

The last step we need is our initialization, and since we need the DOM to do it, we'll have to push the code into a window.onload function. Also, we can ditch alerting the myController object, since we will now be able to test the total result by clicking on "I'm a Button".

```
<script>
    function Controller(options) {
        var config = {
            IDs: {
                button: 'button1'
            }
        };

        // Private
        var options = options;
        var button;

        // Private Methods
        function getUserID() {
            return options.userID;
        }

        function handleClick () {
            alert(getUserID());
        }

        // Public Methods
        this.setUserID = function (userID) {
            options.userID = userID;
        }

        // Constructor
        button = document.getElementById(config.IDs.button);
        button.onclick = handleClick;
    };

    window.onload = function () {
        var myController = new Controller({userID: 1234});
        myController.setUserID(6789);
    }
</script>
```

Ok, so there you go, I used the new keyword, and I even used the this keyword correctly as well.

I'm done right? Can I play Fallout 3 now? :)