Using CSS animations on mobile browsers with JavaScript

Many developers tend to ignore the enormous potential of CSS animations when it comes to mobile development. Simply put, CSS animations work better and faster than normal JavaScript animations because they run at browser-level code. In this article we'll see how to use CSS animations on mobile browsers.

Having a series of boxes, we want to dynamically animate their background color with CSS animations. First of all, we create the animation itself:

```
@-moz-keyframes my /* Firefox */
{
from {background-color: #ff9;}
to {background-color: #000;}
}

@-webkit-keyframes my /* Safari and Chrome */
{
from {background-color: #ff9;}
to {background-color: #000;}
}

@-o-keyframes my /* Opera */
{
from {background-color: #ff9;}
to {background-color: #000;}
}
@keyframes my
{
```

```
from {background-color: #ff9;}
to {background-color: #000;}
}

.animate {
        -moz-animation: my 1s; /* Firefox */
        -webkit-animation: my 1s; /* Safari and
Chrome */
        -o-animation: my 1s; /* Opera */
        animation: my 1s;
}
```

The next step is to bind the CSS class `animate` to our elements. We can create a cyclic timer which selects one element at time. Since our animation has a duration of one second, our JavaScript timer must have the same time interval:

```
document.addEventListener('DOMContentLoaded',
function() {
        var boxes =
document.querySelectorAll('div.box'),
                length = boxes.length,
                index = -1;
        var animate = function(element) {
            var $cls = element.className;
                element.className = $cls + '
animate';
                var parent = element.parentNode;
                var children =
parent.getElementsByTagName('div');
                for(var i = 0; i < children.length;
i++) {
```

```
                        var child = children[i];
                        if(child !== element) {
                                var cls =
child.className;
                                child.className =
cls.replace('animate', '');
                        }
                }
        };

        setInterval(function() {
                index++;
                if(index == length) {
                        index = -1;
                }

                var box = boxes[index];
                animate(box);

        }, 1000);

}, false);
```

You can see the above code in action in the following video.