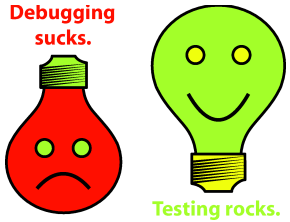

Testing on the Toilet

JavaScript: Simulating Time in jsUnit Tests

March 29, 2007

Sometimes you need to test client-side JavaScript code that uses `setTimeout()` to do some work in the future. **jsUnit** contains the `Clock.tick()` method, which simulates time passing without causing the test to sleep. For example, this function will set up some callbacks to update a status message over the course of four seconds:

```
function showProgress(status) {
  status.message = "Loading";
  for (var time = 1000; time <= 3000; time+= 1000) {
    // Append a '.' to the message every second for 3 secs.
    setTimeout(function() {
      status.message += ".";
    }, time);
  }
  setTimeout(function() {
    // Special case for the 4th second.
    status.message = "Done";
  }, 4000);
}
```

The jsUnit test for this function would look like this:

```
function testUpdatesStatusMessageOverFourSeconds() {
  Clock.reset();  // Clear any existing timeout functions on the event queue.
  var status = {};
  showProgress(status);  // Call our function.

  assertEquals("Loading", status.message);

  Clock.tick(2000);  // Call any functions on the event queue that have been
                     // scheduled for the first two seconds.
  assertEquals("Loading..", status.message);

  Clock.tick(2000);  // Same thing again, for the next two seconds.
  assertEquals("Done", status.message);
}
```

This test will run very quickly - it does not require four seconds to run.

`Clock` supports the functions `setTimeout()`, `setInterval()`, `clearTimeout()`, and `clearInterval()`. The Clock object is defined in `jsUnitMockTimeout.js`, which is in the same directory as `jsUnitCore.js`.

More information, discussion, and archives:

http://googletesting.blogspot.com