# Getting Started With QUnit

## Overview

In this lab, you'll refactor some tests that use our custom TestLib.js test driver, so that the tests use the QUnit APIs instead. This is actually quite a straightforward task, because we implemented our custom test driver to mimic the behavior found in standard testing frameworks.

## Source folders

* C:\labs\qunit
* C:\solutions\qunit

## Instructions

Take a look at the existing files in the *Student* folder. This code uses the custom test driver to test the CustomDate object. Your task in this lab is to refactor the tests in CustomDateTest.js so that they use QUnit instead. We're not going to give prescriptive exercise steps in this lab – instead we'll summarize what's needed and you have to figure out how to do it.

The first step is to change the HTML web page so that it links in the QUnit style sheet and JavaScript library. You also need to refactor the web page so that it'll work with QUnit.

The bulk of your changes will be in CustomDateTest.js, obviously. Here are some things you need to take into account:

* The tests currently use the assertEquals(), assert(), and fail() functions from our custom test library. You need to translate these calls into equivalent QUnit functions – think about the most appropriate function to use in each case.
* How are you going to do setup/teardown operations?
* How are you going to deal with exceptions in the tests for the "setter" functions?

Here are some general comments:

* Most of the changes you need to make can be achieved via judicious global search-and-replace operations in the JS file. Just be a bit careful!
* You might want to comment-out all the tests initially and then uncomment them one-by-one, as you work your way through the functions.