JavaScript: QUnit

Persistent Interactive | Persistent University

Key learning points :

- Introduction to QUnit
- Setting up QUnit
- Organizing Tests
- Frequently used assertions
- Running Tests

Introduction to QUnit

- JavaScript unit testing framework
- Used and maintained by JQuery Team
- Can test any JavaScript code including server side !!

How to set up QUnit ?

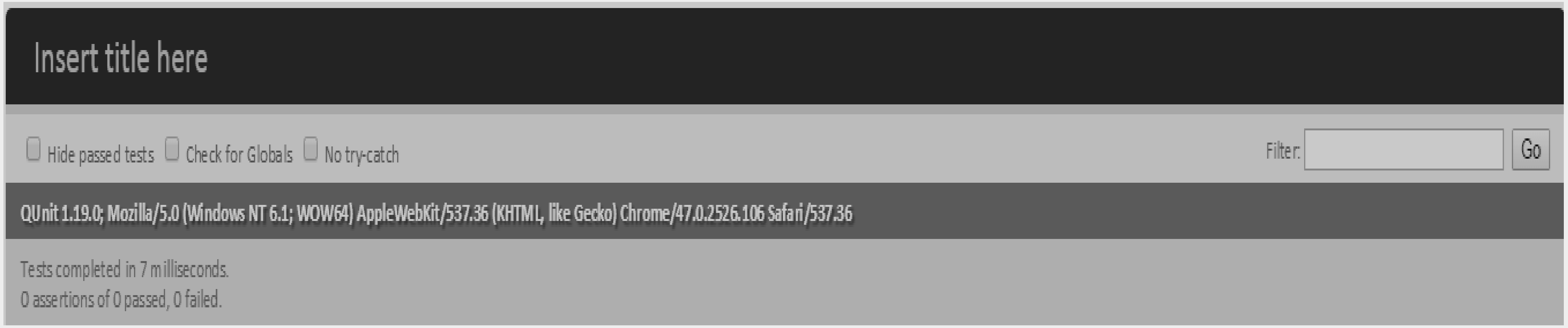
- Include two files in HTML :-
  - qunit.js
  - qunit.css

- Add mark ups in the <body> element as <div> with two ids :-
  - qunit
  - qunit-fixture

```
<link  rel="stylesheet" href="qunit.css"/>

<script type="text/javascript“ src=“qunit.js"></script>

<div id="qunit"></div>

<div id="qunit-fixture"></div>
```

Setting up QUnit continued ..

- QUnit UI once set-up is done :-



Insert title here

```
QUnit 1.19.0; Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/47.0.2526.106 Safari/537.36
Tests completed in 7 milliseconds.
0 assertions of 0 passed, 0 failed.
```

QUnit User Interface

- Header of QUnit UI displays
  - Page title
  - Green bar when all tests are passed
  - Red bar when at least one test gets failed
  - Bar with few checkboxes to filter test results

- Summary
  - Total time to run all tests including total and failed assertions
  - Current test being executed

Persistent

QUnit UI features to filter test results

- Hide passed tests
  - If checked, only failed tests will appear

- Check for global
  - If checked, test will fail if properties are added or removed from the window object

- No try-catch
  - Run test outside of a try-catch block

Organize Tests

- Modules
  - logically organize tests
  - group common code i.e. setup, tear down etc
- Tests

```
QUnit.module("module 1");

QUnit.test("test 1", function(){
                ok(true);
});
```

QUnit Modules explained..

- It has two parameters :-
  - module name
  - callbacks to run before and after test i.e. beforeEach & afterEach
- Can nest sub modules
- Any test that follows a module belong to that module
- Not mandatory in QUnit framework

QUnit Tests explained ..

- It adds a test to run

- Can pass two parameters :-
  - test name
  - callback function, actual code to be tested

- Test names will be preceded by the module name

- Can be filtered based on modules

Commonly used assertions

- ok(expression, message)
  - Boolean check
  - First argument is expression to be tested
  - Second argument is the short description of the assertion

- equal(actual, expected, message)
  - Non strict comparison , uses ‘==‘ operator to compare values
  - First argument is the expression to be tested
  - Second argument is the expected known value
  - Third argument is short description for the assertion

```
ok(true, "always returns true");

equal(true, true, "always returns true");
```

Commonly used assertions continued..

- strictEqual(actual, expected, message)
  - strict comparison using ‘===’ operator

- deepEqual(actual, expected, message)
  - deep recursive comparison
  - useful to compare arrays, objects ,
  - functions, dates etc.

```
strictEqual(true, true, "returns true as both value and data
type are same");

var array = [1,2];

deepEqual(array,[1,2],"arrays are equal");
```

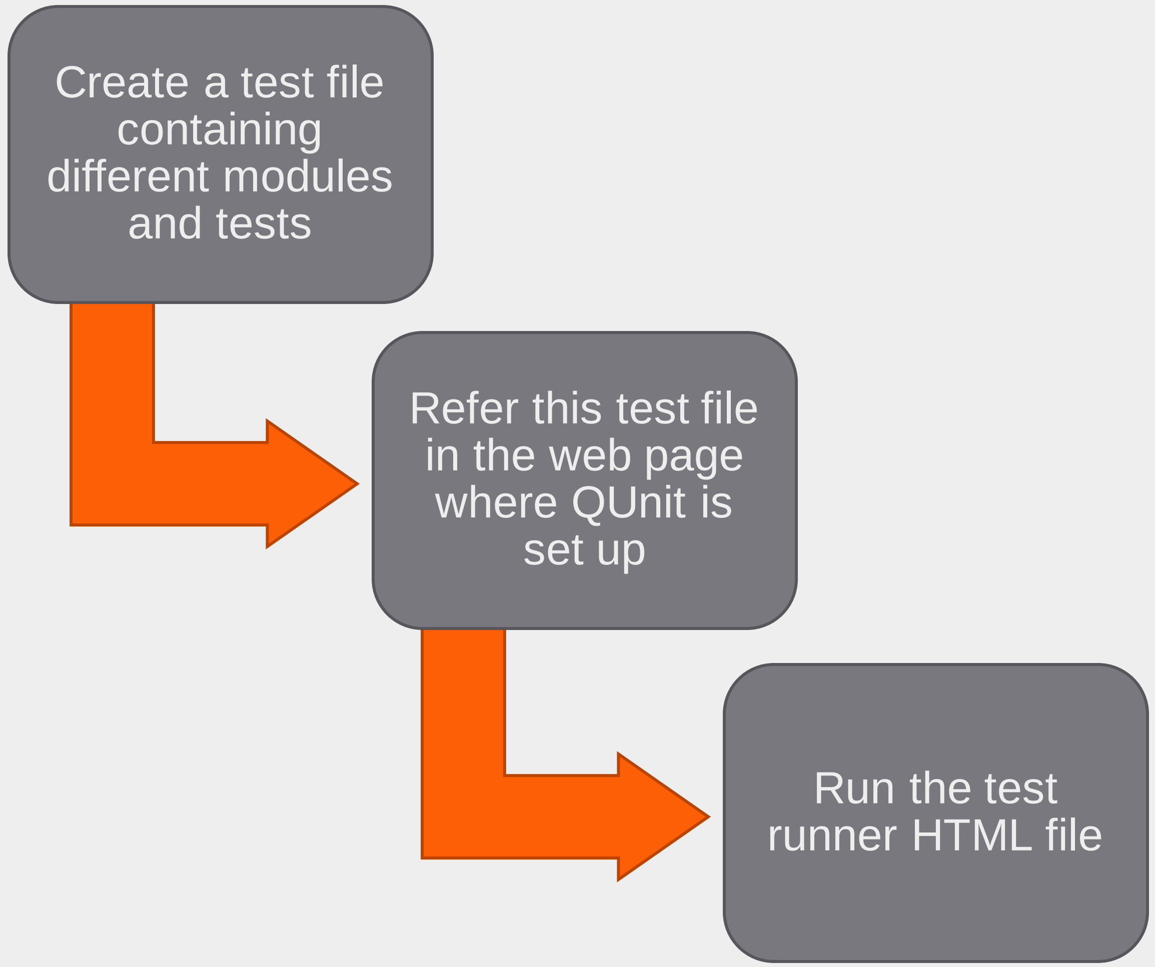
Commonly used assertions continued..

- throws(block, expected, message)
  - tests if ‘block’ throws exception
  - can compare thrown error
  - ‘block’ should be a function

- practice below assertions ☺
  - notOk
  - notEqual
  - notStrickEqual
  - notDeepEqual

```
throws(function(){
                    throw "fatal error";
}, "fatal error" ,"error message matches");
```

Run QUnit Tests


Create a test file containing different modules and tests
Refer this test file in the web page where QUnit is set up
Run the test runner HTML file

Sample HTML file and test file

HTML File

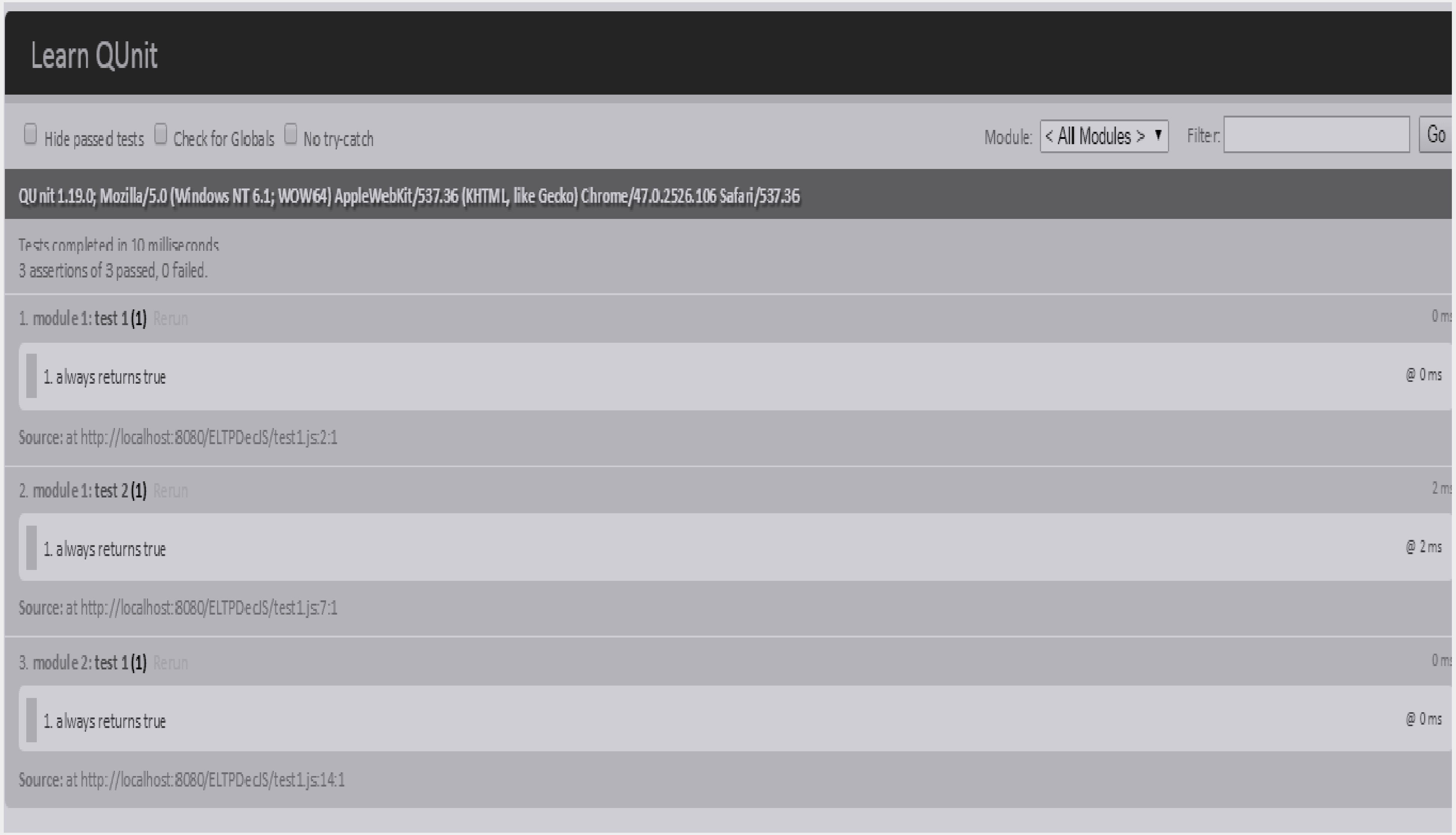
```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Q</title>
<link rel="stylesheet" href="qunit.css" />
<script type="text/javascript" src="qunit.js"></script>
</head>

<body>
<div id="qunit"></div>
<div id="qunit-fixture"></div>
<script type="text/javascript" src="test1.js"></script>
</body>
</html>
```

Sample HTML file and test file(Cont.…)

- Test File

```
module("module 1");
test("test 1", function(){
   ok(true,"always returns true");
});
test("test 2", function(){
   ok(1,"always returns true");
});

module("module 2");
test("test 1", function(){
ok("true","always returns true");
});
```

QUnit User Interface After Running Tests


Learn QUnit

Hide passed tests Check for Globals No try-catch Module: < All Modules > Filter: Go

QUnit 1.19.0; Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/47.0.2526.106 Safari/537.36

Tests completed in 10 milliseconds.
3 assertions of 3 passed, 0 failed.

1. module 1: test 1 (1) Rerun 0 ms
   1. always returns true @ 0 ms

Source: at http://localhost:8080/ELTPDecJS/test1.js:2:1

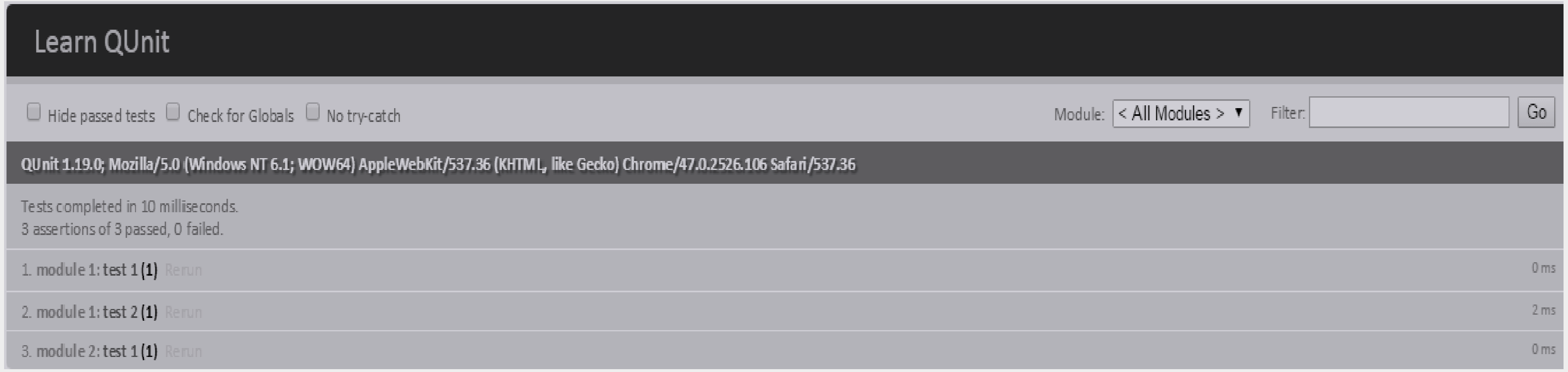
2. module 1: test 2 (1) Rerun 2 ms
   1. always returns true @ 2 ms

Source: at http://localhost:8080/ELTPDecJS/test1.js:7:1

3. module 2: test 1 (1) Rerun 0 ms
   1. always returns true @ 0 ms

Source: at http://localhost:8080/ELTPDecJS/test1.js:14:1

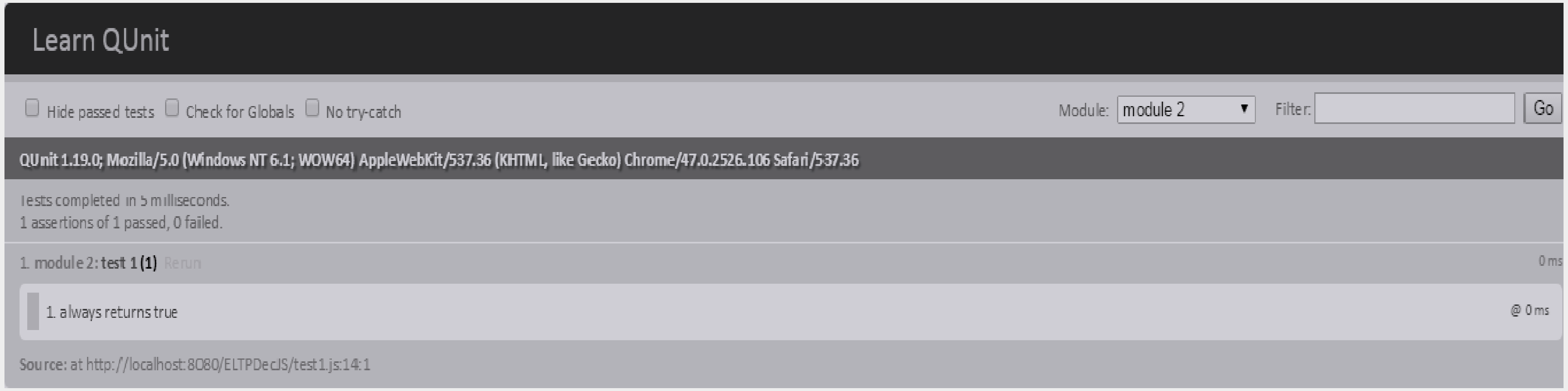
QUnit UI continued..



Learn QUnit

Hide passed tests Check for Globals No try-catch Module: < All Modules > Filter: Go

```
QUnit 1.19.0; Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/47.0.2526.106 Safari/537.36

Tests completed in 10 milliseconds.
3 assertions of 3 passed, 0 failed.

1. module 1: test 1 (1) Rerun                                    0 ms
2. module 1: test 2 (1) Rerun                                    2 ms
3. module 2: test 1 (1) Rerun                                    0 ms
```



Learn QUnit

Hide passed tests Check for Globals No try-catch Module: module 2 Filter: Go

```
QUnit 1.19.0; Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/47.0.2526.106 Safari/537.36

Tests completed in 5 milliseconds.
1 assertions of 1 passed, 0 failed.

1. module 2: test 1 (1) Rerun                                    0 ms
   1. always returns true                                        @ 0 ms
Source: at http://localhost:8080/ELTPDecJS/test1.js:14:1
```

Advantages of QUnit

- Automate unit testing

- Test DOM interactions

- Can test asynchronous code
  - Timeouts
  - Ajax
  - Events

Summary : Session

With this we have come to an end of our session, where we discussed :

- Overview of QUnit , its pre-requisites and set up.
- Different QUnit assertions.
- Organizing and running QUnit tests.

At the end of this session, we expect you to :

- Understand introductory concepts of QUnit.
- Use QUnit for efficient development.

Appendix

- References
- Key Contacts

Reference Material : Websites

- https://qunitjs.com/

Reference Material : Books

Instant Testing with QUnit

- By: Dmitry Sheiko
- Publisher: Packt Publishing

Testable JavaScript

- By: Mark Ethan Trostler
- Publisher: O'Reilly Media, Inc.

Test-Driven JavaScript Development

- By: Christian Johansen
- Publisher: Addison-Wesley Professional

Key Contacts

Persistent University

Tarun Kr. Joshi

tarun_joshi@persistent.co.in

Priya Singh

priya_singh@persistent.co.in

Shubhangi Kelkar

shubhangi_kelkar@persistent.co.in

Persistent

Thank you!

Persistent Interactive | Persistent University