QUnit

Victor Herrero Cazurro

Contenidos

1. Introduccion

Framework de test para jQuery, jQuery UI y jQuery Mobile y para javascript en general.

2. Instalacion

Dos formas de uso:

- Con **html**, con lo que se habria de añadir el link al **js**

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width">
  <title>QUnit Example</title>
  <link rel="stylesheet" href="https://code.jquery.com/qunit/qunit-
2.7.1.css">
</head>
<body>
  <div id="qunit"></div>
  <div id="qunit-fixture"></div>
  <script src="https://code.jquery.com/qunit/qunit-2.7.1.js"></script>
  <script>
    QUnit.test( "a basic test example", function( assert ) {
      var value = "hello";
      assert.equal( value, "hello", "We expect value to be hello" );
    });
  </script>
</body>
</html>
```

- Con Node.js, para lo que se recomienda inicializar un proyecto **Node.js** lanzando el comando

```
> npm init
```

Obteniendo así el fichero **package.json**

Se ha de lanzar el siguiente comando para crear un proyecto **qunit**

```
> npm install qunit
```

Se han de crear por defecto los test en la carpeta **test**

Y se ha de lanzar el siguiente comando para ejecutar las pruebas con **Qunit**

```
> node node_modules/qunit/bin/qunit
```

Se puede incorporar a los scripts de **npm**, añadiendo al fichero **package.json** lo siguiente

```
"scripts": {
    "test": "qunit"
  }
```

Y luego ejecutarlos con el comando

```
> npm test
```

3. Modulos

Permiten agrupar **test**, definiendo una preparacion del entorno para los test del modulo

Se pueden anidar.

```
QUnit.module( "parent module", function( hooks ) {
    hooks.beforeEach( function( assert ) {
       assert.ok( true, "beforeEach called" );
    });

    hooks.afterEach( function( assert ) {
       assert.ok( true, "afterEach called" );
    });

    QUnit.test( "hook test 1", function( assert ) {
       assert.expect( 2 );
    });

    QUnit.module( "nested hook module", function( hooks ) {
       // This will run after the parent module's beforeEach hook
       hooks.beforeEach( function( assert ) {
          assert.ok( true, "nested beforeEach called" );
       });

       // This will run before the parent module's afterEach
       hooks.afterEach( function( assert ) {
          assert.ok( true, "nested afterEach called" );
       });

       QUnit.test( "hook test 2", function( assert ) {
          assert.expect( 4 );
       });
    });
});
```

4. Test

Sirven para definir una prueba, definen un texto que completa el texto del **modulo** donde se incluyen y son los encargados de ejecutar la pueba sobre el **SUT**.

Dentro contienen las validaciones **assert** necesarias.

```
QUnit.test( "test case 1", function( assert ) {
  assert.ok( true, "Module A: in test case 1" );
});
```

Se pueden saltar test con **skip**

```
QUnit.skip( "test case 2", function( assert ) {
  assert.ok( true, "Module A: in test case 2" );
});
```

5. Assert

Permiten validar condiciones que han de cumplir los test, ofrece funciones como

- equal
- notEqual
- ok
- notOk
- deepEqual
- notDeepEqual
- throws

```
QUnit.test( "TestSuite", function( assert ) {
    //test data
    var str1 = "abc";
    var str2 = "abc";
    var str3 = null;
    var val1 = 5;
    var val2 = 6;
    var expectedArray = ["one", "two", "three"];
    var resultArray =  ["one", "two", "three"];

    //Check that two objects are equal
    assert.equal(str1, str2, "Strings passed are equal.");

    //Check that two objects are not equal
    assert.notEqual(str1,str3, "Strings passed are not equal.");

    //Check that a condition is true
    assert.ok(val1 < val2, val1 + " is less than " + val2);

    //Check that a condition is false
    assert.notOk(val1 > val2, val2 + " is not less than " + val1);

    //Check whether two arrays are equal to each other.
    assert.deepEqual(expectedArray, resultArray ,"Arrays passed are
equal.");

    //Check whether two arrays are equal to each other.
    assert.notDeepEqual(expectedArray, ["one", "two"],
       "Arrays passed are not equal.");
});
```

6. Preparacion

Se proporcionan funciones que permiten preparar el entorno de ejecucion de la prueba, a nivel del **modulo** con las funciones:

- **beforeEach()**, la función dentro de beforeEach se va a ejecutar antes de cada test dentro del describe o context.

```
QUnit.module( "parent module", function( hooks ) {
  hooks.beforeEach( function( assert ) {
     assert.ok( true, "beforeEach called" );
  });
});
```

- **afterEach()**, la función dentro de afterEach se va a ejecutar después de cada test dentro del describe o context.

```
QUnit.module( "parent module", function( hooks ) {
  hooks.afterEach( function( assert ) {
     assert.ok( true, "afterEach called" );
  });
});
```