QUnit 2.xで変わること

QUnit 2.xで変わること

現バージョン: QUnit 1.16.0

基本方針

- 2.0 では互換レイヤーを入れて互換性を維持
  - 既存のメソッド名の変更もDeprecatedだが動く
- 2.1で互換レイヤーを破棄して完全に移行
- 詳しくは QUnit 2.x Upgrade Guide | QUnit を読む

Try Own

- azu/qunit-examples
- 今回でてきたサンプルコード

Avoid Global

➔ グローバルに合ったものがQunit.*に移動

```
// OLD Style
module("module name");
test("old test", function (assert) {
    expect(1);// 1つのassertがあるという宣言

    assert.ok(true);
});
```

Avoid Global

➔ グローバルに合ったものがQunit.*に移動

```
// NEW Style
QUnit.module("module name");
QUnit.test("new test", function (assert) {
    assert.expect(1);// 1つのassertがあるという宣言

    assert.ok(1);
});
```

Avoid Global

deepEqual(), equal(), notDeepEqual(), notEqual(), notPropEqual(), notStrictEqual(), ok(), propEqual(), strictEqual(), throws()

などのassertionもassert.*に移動

doneでの非同期テストの導入

➔ asyncTestやstartなどはDeprecatedに

```
asyncTest("old async test", function (assert) {
    setTimeout(function () {
        assert.ok(true);
        start();
    }, 16);
});
```

doneでの非同期テストの導入

- assert.asyncの返り値がdone関数
- asyncTestという宣言は不要

```
QUnit.test("new async test", function (assert) {
    var done = assert.async();
    setTimeout(function () {
        assert.ok(1);
        done();
    }, 16);
});
```

setup/teardown のリネーム

→ setup/teardown が beforeEach/afterEach にリネーム

```
QUnit.module( "router", {
    setup: function( assert ) {
        this.router = new Router();
    },
    teardown: function( assert ) {
        this.router.destroy();
    }
});
```

setup/teardown のリネーム

➔ setup/teardown が beforeEach/afterEach にリネーム

```
QUnit.module( "router", {
    beforeEach: function( assert ) {
        this.router = new Router();
    },
    afterEach: function( assert ) {
        this.router.destroy();
    }
});
```

Promiseサポート

- MochaやBuster.jsなどで採用されてるスタイル
- reutrn promise;というかんじでThenableを返すと認識

```
QUnit.test("fulfilled Promise", function (assert) {
    return Promise.resolve("value").then(function (value) {
        assert.equal(value, "value")
    });
});
```

Promiseテスト

- assert.expectが動く！(利用assert数を宣言する機能)
- 意図しないテスト結果 となるのを宣言で防止出来る

```
QUnit.test("fulfilled Promise", function (assert) {
    assert.expect(1);// it's work!
    return Promise.resolve("value").then(function (value) {
        assert.equal(value, "value")
    });
});
```

Promiseテスト

- assertがひとつも呼ばれないテストは自動で失敗する!
- Promiseテストのミスがかなり軽減される GREATE!

```
QUnit.test("Year! Fail test", function (assert) {
    // `resolve`なので`catch`は呼ばれない

    return Promise.resolve().catch(function (value) {
        assert.equal(value, "value")
    });
});
```

レポーターの標準化

- **js-reporters/js-reporters**
- テストフレームワークのレポーターの標準化活動
- テスト結果の**Status**の標準化 **- Pass/Fail/Pending...**
- テストの**Hook Event**の標準化 **- SuiteStart/testEnd...**
- レポーター**API**の標準化 **-** イベントベース**?**

まとめ

- グローバル空間には`QUnit`のみ
- `done`を使った非同期テストのサポート
- `return promise`での`Promise`テストのサポート
- `assert.expect`と`Promise`テストの相性が良い
- 2.0では互換レイヤーあり、2.1で完全移行
- 続きは QUnit 2.x Upgrade Guide | QUnit で

azu/qunit-examples