Front End Testing

## Introduction

This document covers

Introduction

## Jest

### Cheat Sheet

[Code](https://bitbucket.org/kennyrnwilson/typescriptbasics/src/master/jest-testing/)

#### Asynchronous Method Test

test("Testing an asynchronous method", async () => {

    // Assemble

    var myClass = new MyClass();

    // Act

    var s = await myClass.getString();

    // Assert

    expect(s).toBe("Hello World");

})

class MyClass {

    public getString(): Promise<string> {

        return Promise.resolve("Hello World");

    }

}

#### Mocking

test("Testing an asynchronous method", async () => {

    // Assemble - Mock the interface

    const mockEchoFunc = jest.fn((s: string) => Promise.resolve(s + s));

    const sut: IInterface = {

        double: mockEchoFunc

    }

    // Act

    var s = await sut.double("Moi");

    // How many times was the function called

    expect(mockEchoFunc.mock.calls.length).toBe(1);

    // What was it called with on the first invocation

    expect(mockEchoFunc.mock.calls[0][0]).toBe("Moi");

});

interface IInterface {

    double(s: string): Promise<string>;

}

#### Spying

test("Spying Example", async () => {

    // Create an instance of class

    const original = new SourceClass();

    // Use spy to replace method invocation

    const overridenMethod = jest.spyOn(original,"serverCall");

    overridenMethod.mockReturnValue("Overriden Value");

    const result = original.serverCall();

    expect(result).toBe("Overriden Value");

});

class SourceClass

{

    public serverCall() : string

    {

        return "A Value from a server"

    }

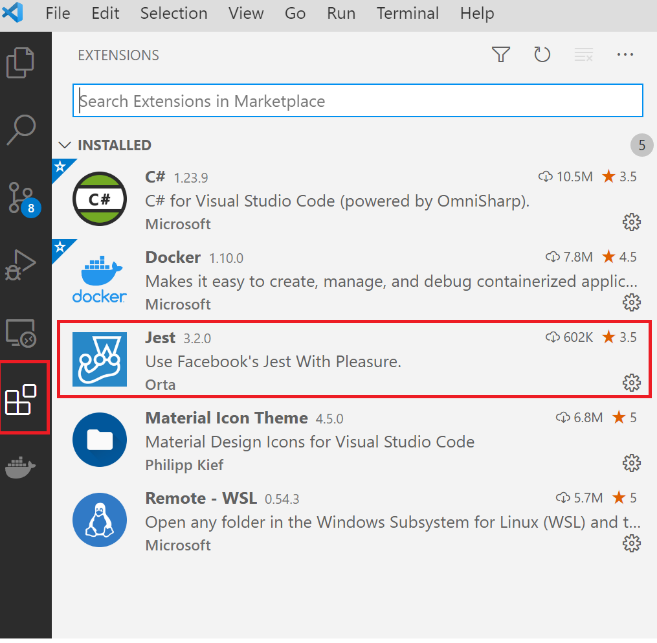
}

### Basic Development Environment Setup

The Jest documentation can be found here.

<https://jestjs.io/>

The [Code](https://bitbucket.org/kennyrnwilson/typescriptbasics/src/master/devenvs/jest-testing/) shows how to setup a simple environment to test TypeScript code using Jest. To run the test first make sure you install the Jest VS Code Plugin.



Once installed make update the workspace settings so we show the Debug menu for tests that succeeded. The file .vscode/settings.json should contain the following section.

{

    "jest.debugCodeLens.showWhenTestStateIn": [

        "pass",

        "skip",

        "fail",

        "unknown",

    ]

}