# 单元测试

1. 初识
   1. 测试驱动开发 TDD：测试先行（未写实现，先写测试）
   2. 单元测试：对系统中最小可测单元进行测试
   3. 归类：白盒测试（想象一个盒子，我们能直接看到盒子的内部结构和逻辑结构）
   4. 意义
      1. 方便编码：判断代码是否已符合要求
      2. 方便重构：判断代码对同一输入的输出是否和重构前一致
   5. 辨析
      1. 单元测试：白盒测试（知悉系统内部）
      2. 集成测试：黑盒测试（不知悉系统内部的情况下，对系统整体功能进行测试）
      3. 端到端测试：整体模拟系统与外部的数据交互
      4. 系统测试：
2. 开发流程
   1. Step 1：明确需求，设计测试案例
   2. Step 2：根据测试案例，编写单元测试代码
   3. Step 3：编写业务代码
   4. Step 4：根据单元测试运行结果，判断程序正确性
3. 语法学习&模拟：
   1. 了解 expect 断言语法： Matchers, Assertions
   2. 先写最基础的 it或test函数：it 应该封装的一个最小测试点
   3. 异步测试代码：async/await，promise，callback（测试可用timer模拟）
   4. 多个it，用 describe 函数封装：describe 应该封装最小测试集
   5. 生命周期：setup & teardown
      1. beforeAll()
      2. beforeEach()
      3. afterEach()
      4. afterAll()
   6. Angular testing（由 angular 封装的）

fake 翻译成模拟更妥当

* + 1. TestBed

主要是配置测试环境

* + - 1. configureTestingModule() 配置用于测试的模块
      2. createComponent()创建组件
      3. get()获取模拟注入类
    1. By

通常是查找元素

* + - 1. .css()通过 css 查找
      2. .directive()通过指令查找
      3. .all()
    1. ComponentFixture
       1. detectChanges()
    2. 异步测试：fakeAsync(), tick(), flush()
    3. Spy
       1. spyOn()
    4. dispatchFakeEvent(), dispatchKeyboardEvent()