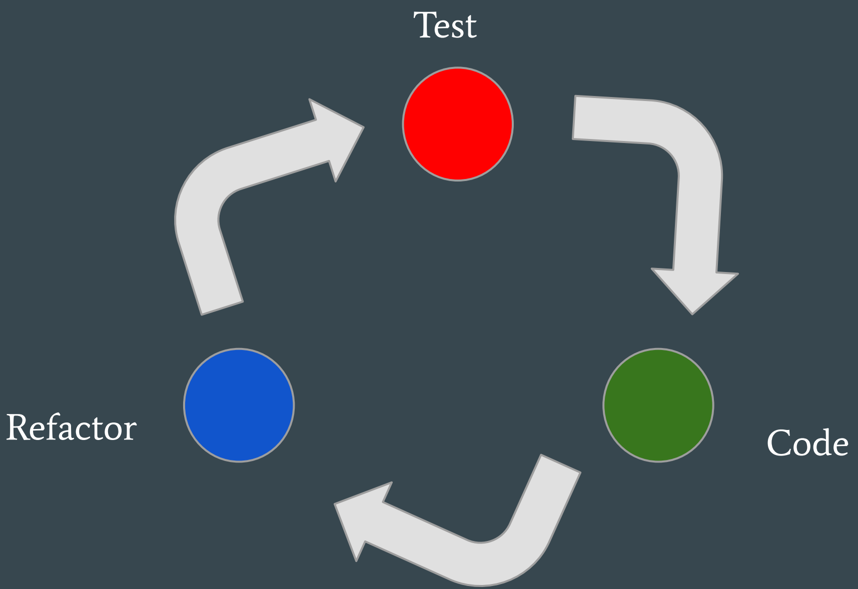
Test Driven Development

What is Test Driven Development

- Iterative software development methodology
- Test, code, refactor

Basics of TDD



Rules [from Uncle Bob/Robert Martin]

- Not allowed to write any production code unless it is to make a failing test pass
- Not allowed to write any more of a unit test than is sufficient to fail. Compilation errors are failures
- Not allowed to write any more production code than is sufficient to pass the one failing unit test

Motivations

- Improve code quality
- Better test coverage
- Documentation from test cases

Coding example

- Requirements: Make a shopping cart
- The shopping cart should tally the total amount when performing final checkout

Iteration 1

test_code.py

```
from shopping import ShoppingCart

def test_add_products():
    cart = ShoppingCart()
    assert (cart.get_total()==0)
```

shopping.py

```
class ShoppingCart:
    def __init__(self):
        pass
    def get_total(self):
        return 0
```

Iteration 2

`test_code.py`

```python
from shopping import ShoppingCart

def test_add_products():
    cart = ShoppingCart()
    assert (cart.get_total()==0)
    cart.add_product("apple", 2)
    assert (cart.get_total()==2)
```

`shopping.py`

```python
class ShoppingCart:
    def __init__(self):
        self.total_price = 0
    def get_total(self):
        return self.total_price
    def add_product(self, product, price):
        self.total_price += price
```

Iteration 3

test_code.py

```
from shopping import ShoppingCart

def test_add_products():
    cart = ShoppingCart()
    assert (cart.get_total()==0)
    cart.add_product("apple", 2)
    assert (cart.get_total()==2)
    cart.add_product("orange", 3)
    assert (cart.get_total()==5)
```

shopping.py

```
class Product:
    def __init__(self, name, price):
        self.name = name
        self.price = price

class ShoppingCart:
    def __init__(self):
        self.products = []
    def get_total(self):
        return sum([product.price for product in
self.products])
    def add_product(self, product, price):
        self.products.append(Product(product,
price))
```

When not to use TDD

- Rapid prototyping/POC/demo
- Legacy applications without any unit/automated tests

Source

1. Ferdinando Santacroce [https://semaphoreci.com/blog/test-driven-development]
2. Andrea Koutifaris[https://www.freecodecamp.org/news/test-driven-development-what-it-is-and-what-it-is-not-41fa6bca02a2/]
3. Matthew Renze [https://app.pluralsight.com/library/courses/clean-architecture-patterns-practices-principles/]
4. Chiradeep BasuMallick [https://www.spiceworks.com/tech/devops/articles/what-is-tdd/]
5. Amy Dredge [https://app.pluralsight.com/library/courses/tdd-is-not-unit-testing-executive-briefing/table-of-contents]
6. Robert Martin [http://butunclebob.com/ArticleS.UncleBob.TheThreeRulesOfTdd]
7. Ron Jeffries, Grigori Melnik [https://www.computer.org/csdl/magazine/so/2007/03/s3024/13rRUygT7kK]
8. David Fucci, A Dissection of the Test-Driven Development Process: Does It Really Matter to Test-First or to Test-Last?