5つのステップ

1, テストコードを書く
2. テストを失敗させる（RED）
3. プロダクションコードを書く
4. テストを成功させる（GREEN）
5. リファクタリングする

TDDのこころ

少しずつ、ひとつずつ

小さなステップをひとつずつ登る

ひとりずつ仕留める

複数の問題を一度に解決してはならない

すばやくまわす

自明な実装とリファクタリングを繰り返す

自分が最初のユーザー

自分がマズイと思うものを提供してはならない

不安をテストに

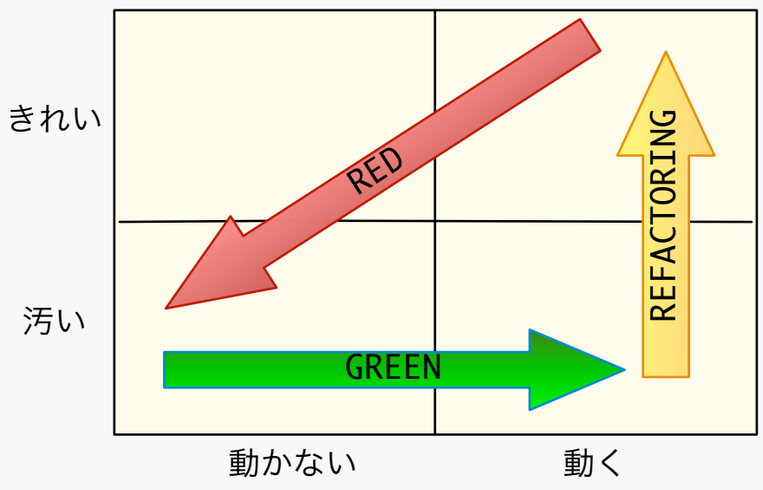
祈るのではダメ

テストとグリーンバー以外は信用してはならない

テストが命綱

セーフティネットとしてのテストを先に書く

黄金の回転



テスト駆動開発チートシート version 0.1

テスト駆動開発

テスト

自動化されたテストを作成し、テストする

独立したテスト

テストの実行は、テスト間で決して影響すべきではない

テストリスト

開始する前に想定されるテストのリストを書き出す

テストファースト

テスト対象のコードを書く前に、テストコードを書く

アサートファースト

テストコードでは、アサーションをはじめに書く

テストデータ

テストが読みやすくなるようにテストデータを記述する

明示的データ

期待するデータと実際の結果について、関係を明示的にする

テスト

下位のテスト

大きすぎるテストは、小さなテストに抽出してから実行する

モックオブジェクト

複雑なリソースは、定数を返す仮バージョンを使ってテストする

自己接続

テスト対象と他のオブジェクトが通信している事をテストするには、テストケースと通信させてテストする

ログ文字列

メッセージの呼び出し順序をテストするには、呼び出し毎にログ文字列を追加する

クラッシュテストダミー

呼び出されないコードをテストするには、例外をおこす特別なオブジェクトを利用する

失敗するテスト

ひとりでプログラミングしているならば、レッドで終わらせる

綺麗なチェックイン

チームでプログラミングしているならば、グリーンで終わらせる

レッドバー

１歩を表現するテスト

何かを教え、実装できる自信を持てるテストからはじめる

最初のテスト

何もしないバリエーションのテストからはじめる

説明用のテスト

テストを用いた説明を求め、提供する

学習用のテスト

外部ライブラリに関しては、パッケージ内の新機能をはじめて使用する前にテストする

その他のテスト

脱線的な考えが浮かんだら、テストリストに追加し本題に戻る

回帰テスト

欠陥が報告されたならば、再現する最小限のテストを作る

休憩

疲れたら、休憩を取る

やり直し

迷ったならば、コードを捨ててやり直す

安価な机、良質な椅子

他の家具は全てけちっても、良質の椅子を用意する

グリーンバー

仮実装

テストを成功させるには、最初に定数を返す実装を行う

三角測量

２つ目のテストを作成して、正しい実装を導く

明白な実装

答えが明確ならば、ギアチェンジして一気に実装する

１から多

コレクションを使用しない実装からはじめる

Uncle BobのTDD三原則

失敗するユニットテストのコードを書く前に、プロダクションコードを書いてはならない

コンパイルが通り、適切に失敗するユニットテストが出来るまでは、次のユニットテストのコードを書いてはならない

現在失敗しているユニットテストが通るまで、次のプロダクションコードを書いてはならない