

TDD

Test Driven Development

TDD

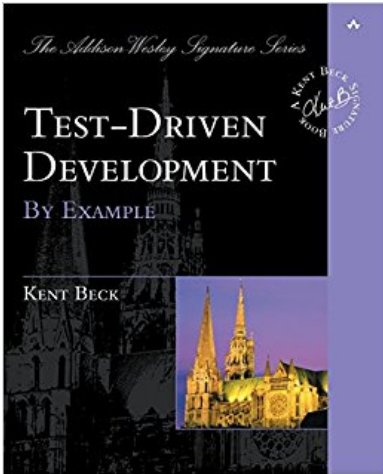
- Le sujet principal n'est pas le test
- Développement guidé par les exemples

Kent Beck



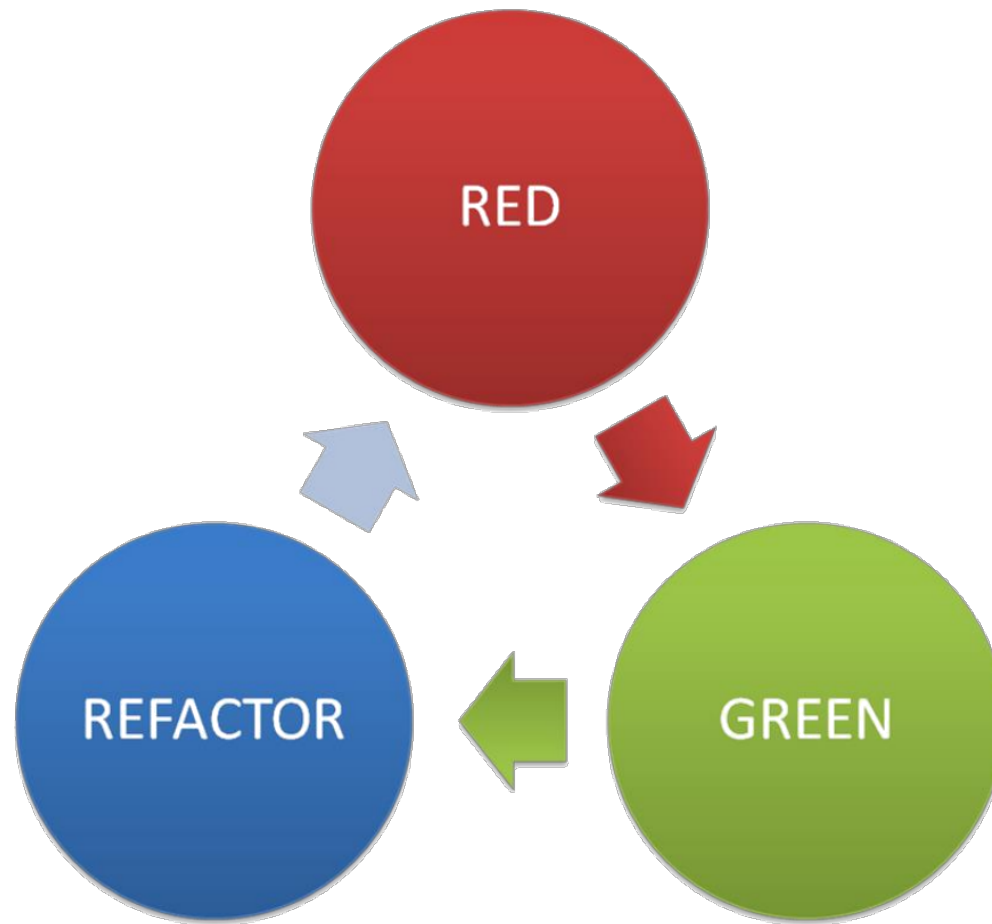
- Extreme programming
- Test-driven development
- Design patterns
- CRC cards
- JUnit
- Manifeste agile

Préface



- N'écrivez pas une ligne de code tant que vous n'avez pas d'abord un test automatique qui échoue
- Eliminez la duplication

TDD mantra



Un exemple

- Une calculatrice gérant l'addition de plusieurs entiers

Concepts clés

- Ecrire un test et s'assurer qu'il échoue
 - Les valeurs en « dur » orientent le test suivant
- Ecrire « du » code pour satisfaire le test
 - Passer rapidement au vert
 - Implémentation rapide
- Ecrire « le » code que l'on va garder
 - Renommage, restructuration
 - Changement d'implémentation

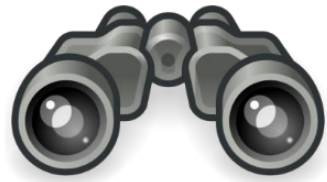
Questions



Premier exercice - FizzBuzz

- Le programme FizzBuzz retourne le nombre qu'on lui donne et qui est compris entre 1 et 100
- Si le nombre est un multiple de 3, retourner Fizz
- Si le nombre est un multiple de 5, retourner Buzz
- Pour les nombres multiple de 3 et de 5, retourner FizzBuzz
- Dans les autres cas, on retourne le nombre

Rétrospective



Points d'attention

- Se focaliser sur le comportement/besoin et non la manière
- Penser utilisation avant implémentation
- Ne pas modifier le test et le code en même temps
- Prendre autant soin des tests que du code

Bonnes pratiques de test

- Indépendance
- Rapidité d'exécution
- Reproductibilité
- Lisibilité
- Tester une seule chose à la fois
- Le nom du test indique l'objectif

Outillage

- Junit, TestNG
- Maven, Ant
- Eclipse, IntelliJ, NetBean
- Jenkins, Travis CI, GitLab CI
- Cobertura, Emma
- Sonar, Checkstyle, PMD, ...
- MoreUnit, Inifinitest

Spécification / Documentation

- Spécification exécutable
 - Code lisible
 - Indépendant de l'implémentation
- Documentation à jour
 - Exemple d'utilisation du code
 - Spécification du comportement

Concepts agiles

- KISS (Keep It simple, stupid)
 - On commence par une implémentation triviale
 - On restructure pour simplifier
- YAGNI (You Ain't Gonna Need It)
 - On ne développe que ce qui est nécessaire pour faire passer un test
 - On écrit un test que pour décrire un cas utilisateur

Feedback

- Baby steps
 - Approche itérative très courte
- Feedback
 - Retour immédiat

Mesure de la couverture

- La couverture est assurée par construction
- On ne s'en préoccupe pas spécialement

Qualité de code

- Refactoring
 - Modification de l'implémentation sans changer le comportement
 - Elimination de la duplication
 - Amélioration de l'implémentation

BDD: Behavior Driven Development

- Continuité du TDD
- Encore plus orienté vers le métier
- Rédaction en collaboration avec le métier
- Syntaxe Gerkhin: Given / When / Then

Scénario: Compl
éter toute ma
todo liste

Etant donné
que j'ai 2 t

âches dans ma
todo liste

Lorsque je
complète toutes
mes tâches

Alors ma todo
liste est vide

Second exercise

- Bowling Game
- Game of life
- How much water ?
- Tennis score

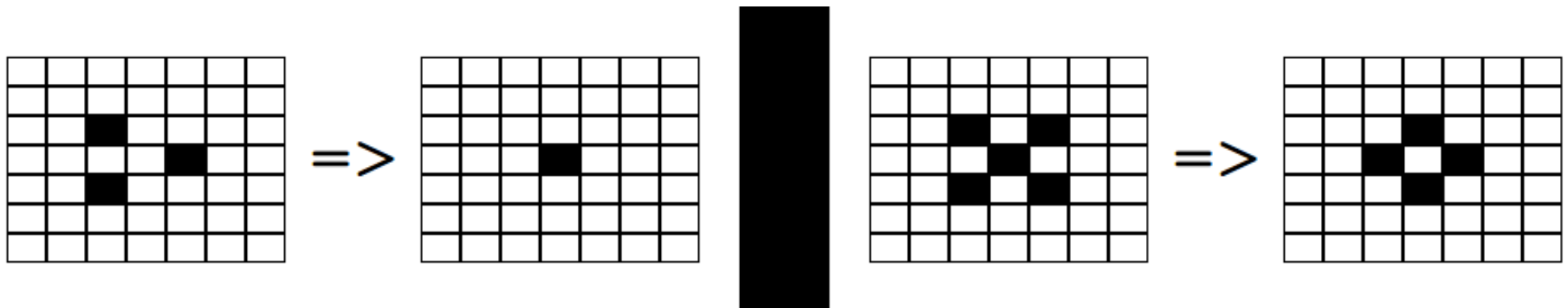
Score de bowling

- La grille de bowling est constituée de 10 cadres.
- Pour chaque cadre, le joueur à deux lancers pour faire tomber les 10 quilles.
- Le score du cadre est le nombre de quilles tombées plus un bonus en cas de spare ou de strike.
- Il y a spare lorsque qu'un joueur fait tomber toutes les quilles en deux coups. Le bonus est le nombre de quilles tombées au coup suivant
- Il y a strike lorsque toutes les quilles tombent au premier essai. Le bonus est le score des deux coups suivants.

Joueur A	1	5	3	1	6	/	2	1	5	4	X	-	3	4						
	6	10	18	21	30	47	54													

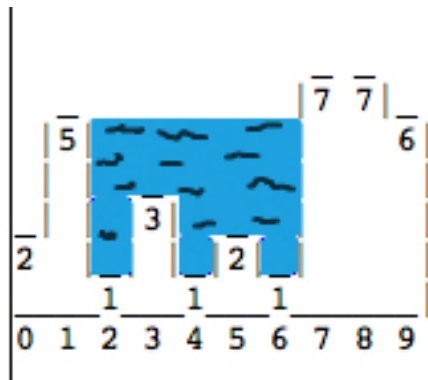
Le jeu de la vie: John conway

- **Pour un emplacement 'peuplé':**
 - Une cellule avec un ou aucun voisin meurt de solitude.
 - Une cellule avec quatre voisins ou plus meurt de surpopulation.
 - Une cellule avec deux ou trois voisins survit.
- **Pour un emplacement 'vide' ou 'non peuplé'**
 - Une cellule avec trois voisins devient peuplée.



How much water ?

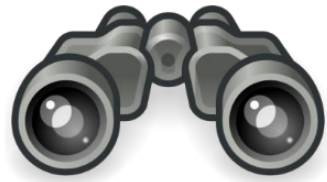
- Etant donnée une liste d'entiers représentant les hauteurs de colonnes
- On cherche la quantité d'eau qui resterait prisonnière des cuvettes formées par les colonnes



Tennis score

- Afficher le score d'un match de tennis
- On donne une suite indiquant qui a marqué chaque point et on retourne le score
- Exemples
 - AAAB \Rightarrow 40 - 15
 - AABB \Rightarrow 30A
 - BBBB \Rightarrow Jeu B

Rétrospective



Les points difficiles

- Les méthodes privées
- Les contributeurs
- Rester indépendant de l'implémentation
- Tester sur du code existant
- Conception émergente

Bénéfices

- Composants prévus pour être testés
- Composants prévus pour être réutilisés
- Capacité à faire évoluer/modifier le code
- On sait ce qui marche ou pas
- Projet auto validé
- Rapidité d'analyse des défauts

Bénéfices

- Les tests ne sont plus une option "lorsqu'il reste du temps"
- On ne perd pas du temps à écrire les tests, on gagne du temps pour écrire le code

Références

Extreme programming explained: embrace change.

— Kent Beck. Addison-Wesley, 1999

Test-Driven Development: By Example

— Kent Beck. Addison-Wesley, 2002

Test-Driven Development: A Practical Guide

— David Astels. Prentice Hall, 2003

Growing Object-Oriented Software, Guided by Tests

— Steve Freeman, Nat Pryce, 2009

Sites

- Cyber dojo: <http://cyber-dojو.org/>
- CodingDojo: <http://codingdojo.org/kata/>
- Yosethegame: <http://yosethegame.com/>
- Coding Game: <https://www.codinggame.com/start>