TDD & Pair Programming

Konrad Delong Todd Mastumoto

April 21th 2010, Amsterdam

Before we start

```
$ sudo easy_install nose

$ wget http://bit.ly/testdojo -O tdd.py
```

What is eXtreme Programming?

- Good practices pushed to extreme
- Testing $\longrightarrow$ TDD
- Code review $\longrightarrow$ Pair Programming
- Running test suite $\longrightarrow$ Continous Integration
- Client feedback $\longrightarrow$ Client as a part of the team
- Short iterations
- Elasticity

What is Test Driven Development?

- *Short* cycle
  - Write a test
  - Make sure it fails
  - Satisfy test
  - Clean up
- Tested codebase
- Safe refactoring
- One thing at a time
- Specification

What is Pair Programming?

- Two geeks one computer
- Two heads are better than one
- Mutual supervision :)
- Programming becomes social activity

How to run tests?

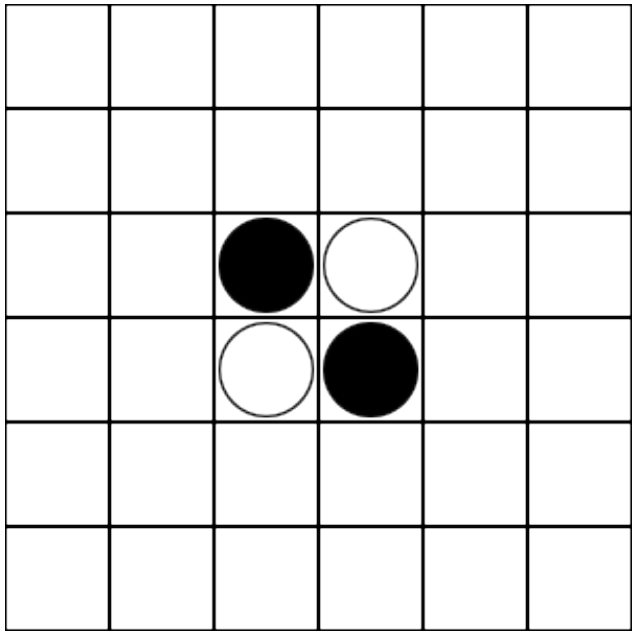
- `$ sudo easy_install nose`
- put your tests in functions with names starting with `test`
- make assertions using `assert_equal`
- `$ nosetests tdd.py` (run from the console)

How to have fun today?

- focus on testing – not a competition
- follow the tdd cycle: write a test, watch it fail, implement, refactor

Problem

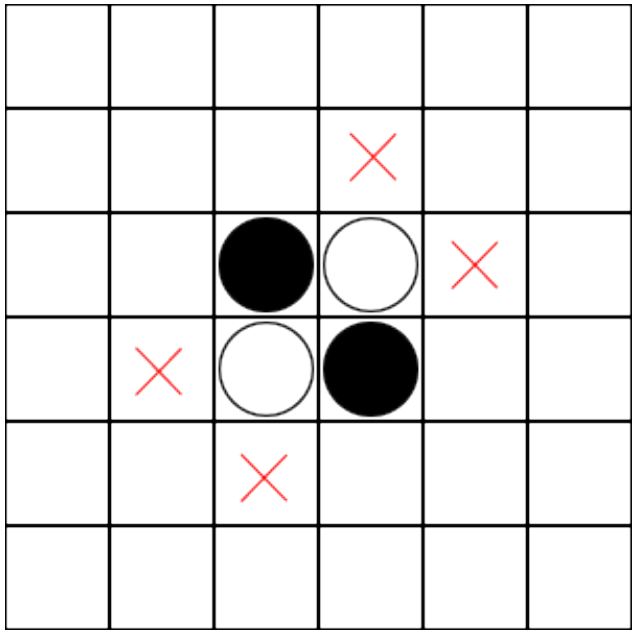
Reversi (Othello)



`http://bit.ly/pun-reversi`

Problem

Given a board and a player, find all legal moves for that player. Return the result as a list of pairs, sorted row-wise.

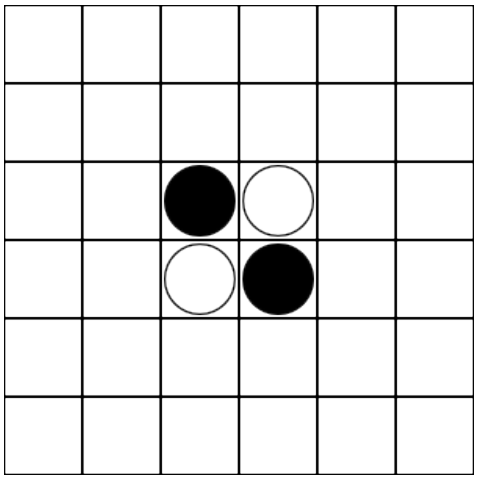


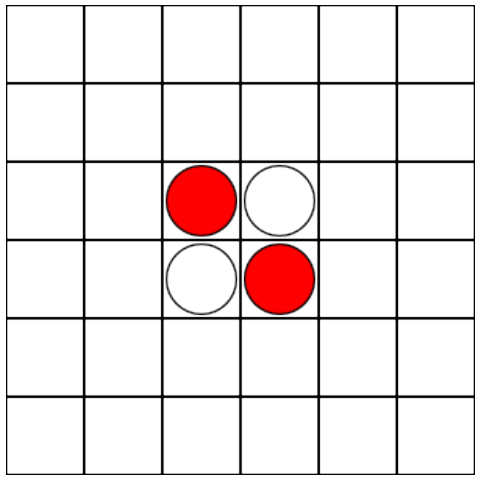
Board representation

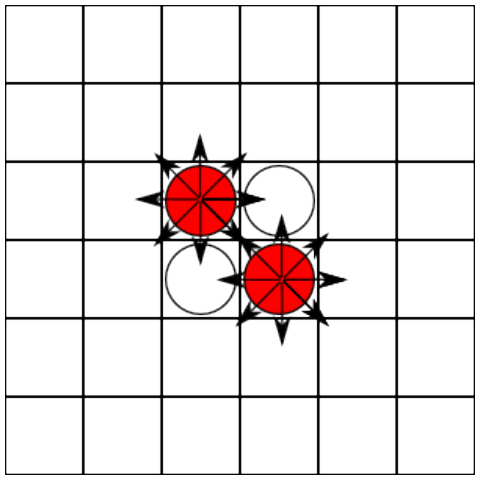
```
>>> board = ["....",
             ".bw.",
             ".wb.",
             "...."]
>>> board[1][2]
'w'
>>> legal_moves(board, 'w')
[(0, 1), (1, 0), (2, 3), (3, 2)]
```

Algorithm suggestion

1. Find your pieces
2. Try walking in every direction to find a valid move







Commands again

```
$ sudo easy_install nose

$ wget http://bit.ly/testdojo -O tdd.py
```