



When to Write Tests – How to Structure Tests

Lecture Objectives:

- 1) Critique the advantages of writing tests before development has commenced.
- 2) Critique the advantages of having a second person write tests independent from the developer.
- 3) Explain three methods for organizing test cases and the advantages and disadvantages of each one.

Who should write tests?

- You (as the developer)

Familiar w/ code so
you know its weaknesses.

Have a responsibility
to deliver working
code.

Who should write tests?

- Another developer (for you)

Avoids Bias in SW.

Helps to find problems
in spec.

When to write tests

- Before coding

- Might provide a clear understanding of SW purpose
- May help to avoid bias in TC writing.
- Can test as you go.

⇒ TDD

When to write tests

- In parallel with implementation


In Parallel allows more people to work simultaneously maybe reducing delivery time.
⇒ May cause problems w/ synchronization.

- After implementation

When to write tests

- ⇒ May be easier to do DVA (Maybe) ???
- ⇒ If reqs flux, the may save time.
- ⇒ Makes it easier to ensure C. coverage.

Where to put the test cases

- We all agree test cases are good
 - I hope 😊 
- Where should we store them?
- Doesn't matter much for small projects, but can make a huge difference on big ones...
- 3-4 general methods

Same file

- Test cases are written directly in the same file as the production code
 - Works much better for c / c++ than java
- Advantages

easy to find

- Disadvantages

⇒ Makes code much larger

⇒ Merging can be a problem.

⇒ Code and TC's cannot be separated.



- Test classes placed in same directory as production code

Same directory

Adv.

⇒ Less imports

⇒ Package Level Access

⇒ easy to find.

But; Compilation issues

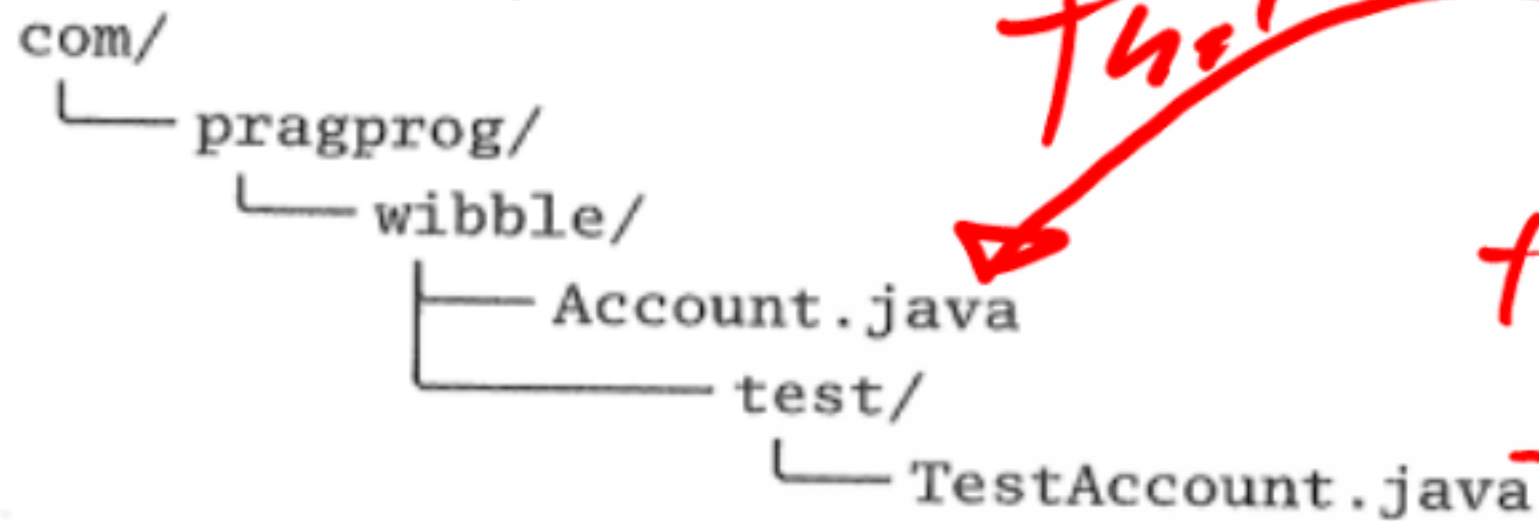
⇒ Bundled w/ SL, which can be bad.

Same directory advantages

- Advantages
 - Easy to find tests
 - Easy to test part of the system
 - Can have visibility of internal variables
 - Use scope other than private to achieve this
- Disadvantages
 - Protected may expose core logic
 - May yield messy directory structure
 - Problems with test cases being released in jars
 - Configuration management issues

Test subdirectory

- Test cases are placed in a “test” subdirectory underneath the production code



Tests are everywhere!

BAD ⇒ Lost package access as well.



Test subdirectory

- Advantages

Test subdirectory

- Disadvantages

Parallel Trees

- Test cases are placed in a separate, parallel tree
 - All of the advantages of the code being in the same directory
 - Avoids the disadvantages
 - Can cause pathing issues with tools
 - Have to watch naming...

