





1. Distill your Specification by Example tables into acceptance tests

a.Add them to a feature file / fitnesse wiki *(Cucumber/Specflow: tag them with @acceptance)*

b. Wire them up into a step file / fixture that calls your public methods in your GameController class

2. Add system operations, as needed, to your GameController class

a. Empty implementations only ({}, return null)

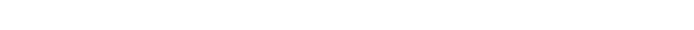
b. Just enough to get it to compile - you should not be creating any other classes

3. Make your tests living documentation

a. Gather images, pictures, links, description, screenshots, etc that describe the feature

b. Add them to your feature file / fitnesse wiki

(make test-acceptance)



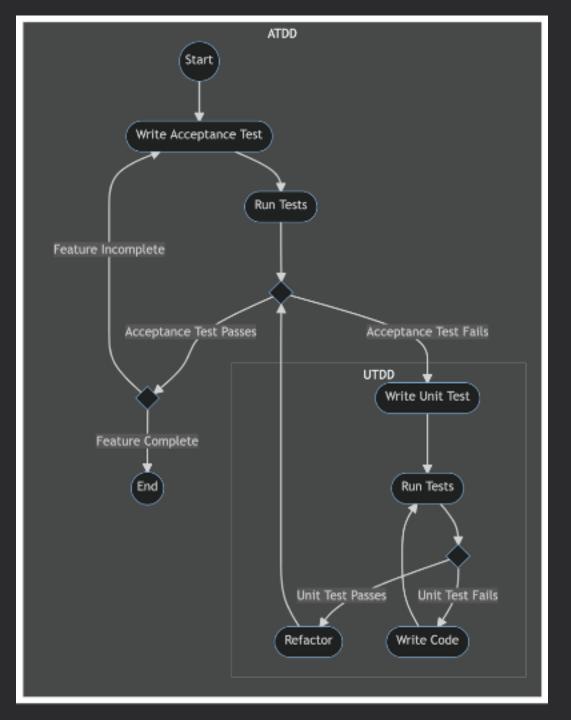
Get ready!

- 1. <u>Distill your Specification by Example tables into acceptance tests</u>
 - a. Add them to a feature file / fitnesse wiki (Cucumber/Specflow: tag them with @acceptance)
 - b. Wire them up into a step file / fixture that calls your public methods in your GameController class
- 2. Add system operations, as needed, to your GameController class
 - a. Empty implementations only ({}, return null)
 - b. Just enough to get it to compile you should not be creating any other classes
- 3. Make your tests living documentation
 - a. Gather images, pictures, links, description, screenshots, etc that describe the feature
 - b. Add them to your feature file / fitnesse wiki

Run your builds (make test-acceptance), push your changes, review your output. You should now have FAILING acceptance tests.

If you are done, or going to take a break, let a facilitator know.





ATDD & UTDD Algorithm

- 1. Write a failing acceptance test
- 2. UTDD it passes
- 3. Write a failing acceptance test...

OR

- 1. Write all failing acceptance tests for the feature
- 2. UTDD until they pass

