Exploring Emergent Design With TDD

June 2015

Leon Hewitt leon.hewitt@7digital.com
Neil Kidd neil.kidd@7digital.com

What Is This Session All About?

- Discuss the Merits of TDD
- Explore What We Mean by Well Designed
- Select Some Heuristics
- Practice Applying the Heuristics
- Reflect on Our Findings

The Merits of TDD

The Merits of TDD

- Clarifies acceptance criteria (we know when we're done)
- Assists in creating loosely coupled code (so we can test components in isolation)
- Provides executable specifications of what the software does
- Provides a regression suite, making code safer to change

What Is Well Designed?

What is Well Designed Software

- Easy to change
- Safe to change
- Easy to follow
- Modular
- Idiomatic

OK - So How?

Some Ideas

- Rules of simple design (Beck)
  - Run all the tests
  - Contain no duplicate code
  - Express all the ideas the author wants to express
  - Minimize classes and methods
- Listen to the tests (Goos)
  - Are they easy to write?
  - Do they fail for the reasons you expect and are the failure messages useful and guide you to what to implement?
  - Do they run quickly?
- Seams in the appropriate places
  - Refactoring Legacy Code (Feathers)
- Follow the messages
  - Goos describes this as early as page 14 …
- Listening to and smelling the code
  - Refactoring Legacy Code (Feathers)
- Clean as You Go
  - Refactor Mercilessly (http://c2.com/) and the boy scout rule in Uncle Bobs Clean Code

Select the Rules

Selection Process

1. Post it notes
2. Group the post its
3. Dot vote them
4. Select 5 practical rules \ heuristics
5. Apply them ...

To The Code

The Problem

Our client wishes to sell digital music online.

They have a requirement to price items and report back to suppliers.

We have a sample of the existing catalogue in the repository. (/samples/music_catalogue.csv)

Setup Your Environment

1. Pair up
2. Choose your development environment
3. Download example catalogue from github:

https://github.com/7digital/Spa2015EmergentDesign

Pricing Tracks

Feature

Accept a comma separated list of track codes and return the total price based on the provided catalogue. The software may be a library or command line tool.

Example acceptance criteria:

"" returns 0

"234569" returns 96

"172835, 234572" returns 190

Report Sales of Tracks

Feature

- Report Sales to a log file in the following format, this will be emailed to suppliers on a daily basis. The required format is a csv:
  track upc,track name,artist name,price,sale datestamp
- There will be a report for each distributor (MOP.log and PHO.log)

Example acceptance criteria

1. "" appends nothing to either log
2. "234569" appends to the daily PHO. log:

   "234569,Take The Overground,The Preserves,99,2015-06-26T07:38:56Z"
3. "172835, 234572" appends to the daily MOP.log:

   "172835,Eat It,Bubbles,94,2015-06-26T07:38:56Z"

   "234572,Villiage Called Alice,The Preserves,96,2015-06-26T07:38:56Z"

Selling and Reporting Albums

Feature

- Accept a comma separated list of album upc codes and return the total price.
- Report Sales to a log file in the following format, this will be emailed to them daily.
  album upc,album name,artist name,price,sale datestamp
- There will be a report for each distributor (MOP.log and PHO.log)
- Track and album sales will be reported in the same log

Example acceptance criteria

1. "" appends returns 0 and appends nothing to either report
2. "234569" returns 99 and appends to the daily MOP.log:

   "234569,Take The Overground,The Preserves,99,2015-06-26T07:38:56Z"
3. "172821, 534198" returns 2198 and appends to the daily PHO.log:

   "172821,Crackle And Pop,The Preserves,1199,2015-06-26T07:38:56Z"

   and appends to MOP.log:

   "534198,Chiller,Bubbles,999,2015-06-26T07:38:56Z"

Reflections

Discussion

- Any discoveries?
- Which rules did you follow?
- Which rules were the most commonly used? (show of hands?)
- What worked?
- What was painful?
- Could you apply these techniques in the real world?
- Which of these rules will you try when you're back in the "coding saddle"?