ATTD with:

RAILS + Cucumber behaviour driven development with elegance and joy

WHO THE HELL AM I?

Al Grimes

SENIOR QA DOGSBODY,

ThoughtWorks®

@al_grimes

HTTP://WWW.TESTUPSTREAM.COM/

WHAT WE UP TO, THEN?

DRINKING!

PAIRING!

TESTING!

DEVELOPING!

THE THINKING I WANTED TO BRING TODAY:

INQUISITIVE
DEMONSTRATE SIMPLE, COLLABORATIVE, AND LIGHTNING-QUICK ATDD IN ACTION

SEASONED
PRACTISE IN TALKING ABOUT AND WRITING PRODUCTION CODE

CYNICAL
PRACTISE OF PAIRING SKILLS, SHARING SOME EXPERIENCE, HECKLING

GLOSSARY OF TERMS:

RUBY

A DYNAMIC, OBJECT-ORIENTED LANGUAGE, WITH FUNCTIONAL INFLUENCES, INVENTED BY YUKIHIRO MATSUMOTO IN 1993

RAILS

A COLLECTION OF RUBY LIBRARIES TO SUPPORT WEB DEVELOPMENT, BORN FROM DAVID HANSSON'S WORK ON BASECAMP

CUCUMBER

AN OPEN SOURCE, CROSS-PLATFORM, BDD TEST RUNNER MADE BY ASLAK HELLESOY FROM WORK OUT OF DAN NORTH'S STORY RUNNER

TODAY, WE WILL, FOR A SMALL STORY:

TURN REQUIREMENTS INTO FEATURES

WRITE FAILING ACCEPTANCE TESTS

IMPLEMENT THE CODE

WATCH IT PASS

BUT FIRST.....

FORM GROUPS !

STORY 1

ADD A NEW MEMBER

- SEE CHECKED-IN DESIGN DOC
- FORM TO CAPTURE FIRST NAME, SURNAME AND HOBBIES (HOBBIES IS A FREE TEXT AREA)
- SURNAME IS MANDATORY
- VALIDATION ERRORS KEEP USER ON FORM WITH ERROR MESSAGES AT TOP
- SUCCESS TAKES USERS TO ALL MEMBERS PAGE WITH A SUCCESS MESSAGE

THE FEATURE

```
Feature: A one or two word phrase describing the feature

  Scenario: Short sentences describing the activity and any business rules

    Given a setup action which allows me to perform the desired action
    When I perform this action
    Then I expect some outcome

        And you can also use "parameters"

        And even inline tables:
          | Like | And  |
          | this | this |
```

FEATURES:

CREATE A FEATURE THAT CONTAINS A SCENARIO THAT INVOLVES:

- GETTING TO THE RIGHT INITIAL STATE (START PAGE ETC)
- PERFORMING THE DOMAIN ACTION (TABLE MIGHT BE USEFUL FOR NEATNESS)
- EXPECTING A RESULT

YOU HAVE 5 MINS

RAILS CONTROLLERS

RAILS CONTROLLERS:

1. SERVICE REQUESTS FOR RESOURCES (PRODUCTS, PEOPLE, ORDERS)
2. PUT TOGETHER MODELS (DOMAIN OBJECTS, VIEWMODELS) WITH VIEWS (HTML, JSON, XML)
3. CAN SERVICE MANY TYPES OF ACTIONS ON A RESOURCE (RETRIVE, CREATE, DELETE, UPDATE etc)
4. ARE CREATED USING THE COMMAND-LINE LIKE SO:

```
bundle exec rails g controller [name] [action1, action2, action3 etc]
```

RAILS CONTROLLER ACTIONS:

NEW : A REPRESENTATION OF HOW TO CREATE A RESOURCE

CREATE: HANDLING OF REQUEST TO CREATE NEW RESOURCE

UPDATE: HANDLING OF REQUEST TO UPDATE EXISTING RESOURCE

INDEX: A LIST OF RESOURCES

SHOW: RETRIEVE A PARTICULAR RESOURCE

DESTROY: DELETE A PARTICULAR RESOURCE

```
bundle exec rails g controller [name] [action1, action2, action3 etc]
```

RAILS VIEWS

RAILS VIEWS:

1. RESPONSIBLE FOR REPRESENTATIONS OF A RESOURCE (HTML, JSON, XML ETC)

2. FOLLOW THE CONVENTION OF THE CONTROLLER ACTION NAME (INDEX.HTML.ERB)

3. CAN HAVE RUBY INJECTED INTO THEM (SIMILAR TO ASP + JSP)

RAILS MODELS

RAILS MODELS:

1. EMBODY BUSINESS OBJECTS (CUSTOMER, TELEPHONE, ORDER, PRODUCT) AND CONTAIN ATTRIBUTES ABOUT THEM INSIDE

2. CONTAIN BUSINESS RULES AND LOGIC ABOUT THOSE DOMAIN OBJECTS (VALID EMAIL ADDRESS, ETC)

3. ARE LINKED TO DATABASE TABLES BEARING THEIR NAME

4. CAN BE GENERATED LIKE SO:

```
bundle exec rails g model [name] [name:datatype name:datatype etc]
```