cucumber school

Lesson 8 Answers

Problems & Solutions

1 - Reflection questions

1.1 - Investment in a support API

*We talked about making an investment to do the refactoring work to keep your step definitions clean and create a good support API. What would be the consequences of **not** making this investment?*

Lots of teams make the mistake of treating their test code as less valuable than their production application code, leaving it to become untidy. Just like regular application code, messy test code becomes increasingly awkward and expensive to maintain over time. Eventually the cost of modifying your tests can become so great you want to throw them away and start over.

1.2 - Modelling by example

How does the language of your scenarios help you design your domain model?

Eric Evan's concept of a Ubiquitious Language means that we try to use consistent terminology right the way down the stack, from conversations with business people, to documentation, to the artefacts in our code. Taking care to choose the nouns and verbs that appear in your scenarios means you'll already have names for the classes and methods you need to create in your solution domain.

1.3 - Example mapping

In the example mapping, we used four colours of cards to represent four different kinds of information we were hunting for in our conversation. What were they?

Yellow: User story

Blue: business rule or acceptance criterion

Red: question

Green: example

1.4 - Example mapping - timebox

How long should an example mapping session take for a single user story? What should you do if you run out of time?

Half an hour should be enough for a team who are already familiar with the domain problem, and have some practice with example mapping. If the story needs longer, it's a signal that either:

- the group don't understand this part of the domain problem well enough
- the story is too big

If you run out of time, have one person take the action to resolve the questions you've identified, and book a time to run the session again with that new knowledge.

1.5 - Rules vs examples

In the three amigos session, we used both rules and examples to understand the user story. Which is best? Complete the following sentences:

- *Rules are better than examples because…*
- *Examples are better than rules because…*

Hopefully this exercise taught you that both are valuable. Rules are great because they can sum up a whole large category of examples in a concise way. Examples are great because they're unambiguous, and illustrate a rule clearly. You need both.

1.5 - Domain language

Take a large piece of paper and draw two overlapping circles. Label the one on the left problem and the one on the right solution. Now think about the words your team use when talking about your own project. Place each word either in the problem domain, the solution domain or, if it's a word that everyone on the team would understand, in the middle.

Pay special attention to words that appear separately in problem and solution domain circles, but mean different things in those domains.

Also look for concepts where both domains use a different term for the same thing.