TA Guide for SED Ex7 - System Integration (Weather)

To prepare, watch the lecture video linked below and download the exercise spec from Scientia:

https://imperial.cloud.panopto.eu/Panopto/Pages/Viewer.aspx?id=b38faf70-4dcd-49fd-b686-af5300b48c11

During the lab (Tuesday 4-6pm in Level 2 computer labs) you should:

This week's exercise (hopefully) brings together a number of themes from previous weeks.

From the lecture they should recognise that the Proxy pattern is a good fit for the caching problem.

If they design a Proxy, they may want to test it mocking out the downstream source (that's a good idea). They may then complain that they can't mock Forecaster as it doesn't implement an interface (correct). So the problem is they have a class that they wish implemented a particular interface, but it doesn't. Which pattern from the lecture helps with that problem? Adapter to the rescue.

What people naturally do is start with the code from the third party library, and write a cache around that. This tends to lead to them using the third party types throughout their code, which causes coupling.

A good question to ask is "if you had to change from this third party weather provider to a different one, and delete this jar from the codebase, how much of your code would stop compiling - could it be less?"

Region may not be the most appropriate type for their application - it could be they want to use a PostCode or LatLong or something else for a position. For the exercise, using a String to represent a place name is fine. A useful method to know about is Region.valueOf("…") which converts a String to an enum value.

There is no need for them to write a Client class - although they can if they want to.

If they are getting mysterious errors, check for a hash map key that does not override equals()/hashcode()

The extension involves the passing of time. The idea is not "start a timer and clear the cache every 60 minutes", rather "whenever you retrieve data from the cache, check if that data is more than 60 mins old".

Testing this extension is tricky - but possible. They should aim for tests where no real time elapses.

Example solution for the exercise:

https://imperial.cloud.panopto.eu/Panopto/Pages/Viewer.aspx?id=26ffe7f2-7598-4e3c-a6a4-ad5f00caa57a

Target architecture: https://www.doc.ic.ac.uk/~rbc/50002/weather-integration.jpeg

Marking scheme:

+ 10 base if everything compiles, any tests pass, style is ok and they seem to have a working cache.

+ 1 for an adapter that wraps the third party library
+ 1 for the adapter implementing an interface
+ 1 the interface does not feature any types from the third party library (Region/Day/Forecast)

+ 1 There is a Proxy class to implement the cache

+ 1 Proxy implement the same interface as adapter, and composes a field of the same type
+ 1 cache behaviour is tested by mocking the downstream source

+ 1 for generally good code hygiene
+ 1 for good naming (types, tests, etc)

+ 1 for implementing expiry after 60 minutes
+ 1 for testing the expiry using virtual time, not real time (e.g. with a mock clock)