Assignment 1 – Student Guide

ICTPRG501 Apply advanced object-oriented language skills

ICTPRG501 JEE Essentials

The Assignment will assess competencies for **ICTPRG501 Apply advanced object-oriented language skills**.

Instructions

For this assignment you will create a 'Filtextulator'.

Filtextulator is a new made-up word. It refers to a program that transforms text in weird and wonderful ways, using filters.

You will create a server application that accepts some text and some instructions, alters the text according to the instructions, and returns the result.

You will also create an Android client. This client will accept some text and instructions regarding the filters from the user, ask the server to process the information, and display the result to the user.

A Server Application

Using JEE, build a server application that:

- contains at least four text filters (some suggestions are listed below)
- can accept a request for, and respond with, a list of available filters
- can accept some text, plus a list of filters in a specific order, and respond with the text after those filters have been applied

The server application should include

- unit tests, using mocking where appropriate
  - each filter should have at least one unit test
- some integration tests
  - some of the tests should test the result of applying multiple filters
- JavaDoc

Filter Suggestions

This is a list of possible text filters you could implement. You are not required to implement these specific ones, however this list should indicate the type and range of filters expected.

- change each word to start with uppercase, with remaining letters lowercase
- remove all spaces
- remove all vowels
- apply simple encryption using ROT13 ('a' becomes 'n', 'b' becomes 'o' and so on. See http://en.wikipedia.org/wiki/ROT13)
- encrypt using the latin alphabet version of Atbash ('a' becomes 'z', 'b' becomes 'y' and so on. See http://en.wikipedia.org/wiki/Atbash)
- use 'pig latin' (move the first consonant to the end, and add 'ay'. So 'pig' becomes 'igpay' and 'hello' becomes 'ellohay'. See http://en.wikipedia.org/wiki/Pig_latin)
- use 'opish' ('op' is added after each consonant. So 'pig' becomes 'popigop' and 'hello' becomes 'hopeloplopo')
- reverse each word (So 'pig' becomes 'gip' and 'hello' becomes 'olleh')

A Client Application

Build an Android application that:

1. requests from the server a list of available filters
2. displays that list to the user
3. allows the user to reorder the list of filters using drag and drop
4. allows the user to turn off one or more of the filters
5. allows the user to type in some text
6. when the user clicks a specific button, the client sends the text plus the list of chosen filters to the server for processing, then displays to the user the altered text returned from the server

The application should include:

- documentation for the user
- JavaDoc
- error handling: if the server returns an error, or cannot be contacted, the application should detect this and inform the user

Design Guidelines and Suggestions

- Create an interface or an abstract class for use by all filters. Your server code should use these filters by their interface/abstract class type. That way you can later add more filters with minimal change to your server.
- Think about design patterns you could use. For example, depending on how you decide to design your filters, the *Factory Pattern* might be useful.
- The body of your requests should be formatted in your choice of either XML or JSON. The format should also be documented.
- A servlet or controller should contain as little code as possible. Your filters, and the code to use them, should mostly not be in a servlet, but instead called upon by the servlet.
- The list of filters sent to the client should include some sort of description of the filter, for display to the user, as well as some sort of 'short' name, that the server uses to recognise a specific filter.
- Think about handling the following sorts of errors in the client:
  - server not reachable
  - server's response missing expected information
  - server responds with error message
- Think about handling the following sorts of errors in the server:
  - client requests unknown filter
  - client sends empty text to be filtered
  - client requests no filters

If you haven't created a program of this size before, it can seem a bit daunting. Pick one small part that seems easier than the rest, that will let you see something. Create a program that does this thing, only. Then pick another small feature, and add that. Keep going, and you end up with a full program!

For example, you might approach the Android app by first creating an app that shows a screen wth a list. The list might be String resources, and might be a list of fruit, or your favourite movie titles. Try it out, make sure it works as you want. Then you might add a text box where the user can type something. Then add a button. Then add code so that when the user taps the button, a new screen displays the text. Keep going in this way, and eventually you will have an entire app.

Grading of Assignments

This is how your assignment will be marked:

Has the student demonstrated the following?

ICTPRG501 Apply advanced object-oriented language skills		
	C	**NYC**
Performance Evidence (Note: If a specific volume or frequency is not stated, then evidence must be provided at least once) **Evidence of the ability to:**		
develop client-server applications using an object oriented language		
produce a graphical user interface (GUI)		
build, debug and test the applications		
produce documentation for the applications.		
Knowledge Evidence **To complete the unit requirements safely and effectively, the individual must:**		
describe the architecture of a framework for web-enabled application development		
summarise the techniques for implementing inter-process communication		
describe the process for the development of large-size application		
identify and outline testing techniques as applied to distributed application development		
identify and outline techniques for implementing third-party supplied code.		

For the % grade, you will be also be marked according to the following:

Code Design and Quality	**Mark**
• Has a coding standard been followed consistently? • Are classes divided into logically sensible packages? • Are controller methods short? *Non-trivial work should be in a separate, potentially reusable class.* • Is invalid input checked for and handled? • Does the design support easy addition of new filters, preferably via polymorphism? • Is there evidence of considered use of Design Patterns? • Are all Exceptions handled?	**/40**
Unit Testing	
• Are Unit Tests included? *There should be, at minimum, one for each text filter.* • Is each unit test small, and easy to understand? • Has mocking been used appropriately? • Is there at least one integration test that tests applying more than one filter? • Does each test have either a comment, or suitable name, so it is easy to tell what it is designed to test and what the expected outcome is? • Do all tests pass?	**/40**
Documentation	
• Is every public method fully documented with JavaDoc?	

• Are there normal comments in the code to assist understanding of complicated sections of logic? • Is the JSON or XML data exchange format fully documented?	/20
TOTAL	**/100**
WEIGHTED TOTAL	**/50**

Android Client

Worth 50% of the total assignment.

Code Design and Quality	**Mark**
• Has a coding standard been followed consistently? • Are classes divided into logically sensible packages? • Are Exceptions handled? • Is an error returned by the server handled sensibly? • Is the absence of a network handled? • Is a network connection timeout handled? • Has logging been used? • Is invalid user input handled safely?	/40
Documentation	
Programmer:	
• Is every public method fully documented with JavaDoc? • Are there normal comments in the code to assist understanding of complicated sections of logic?	
User:	
• Does the app contain documentation for the user? • Is the documentation written with non-technical language? • Does the documentation sufficiently cover how to use the program?	/25
GUI	
• Has attention been paid to support of different screen sizes? • Has drag and drop been used? • Is some form of thread management in use so the network request does not run on the UI thread? Perhaps an AsyncTask? • Is there any user feedback mechanism while waiting for a response from the server?	/35
TOTAL	**/100**
WEIGHTED TOTAL	**/50**