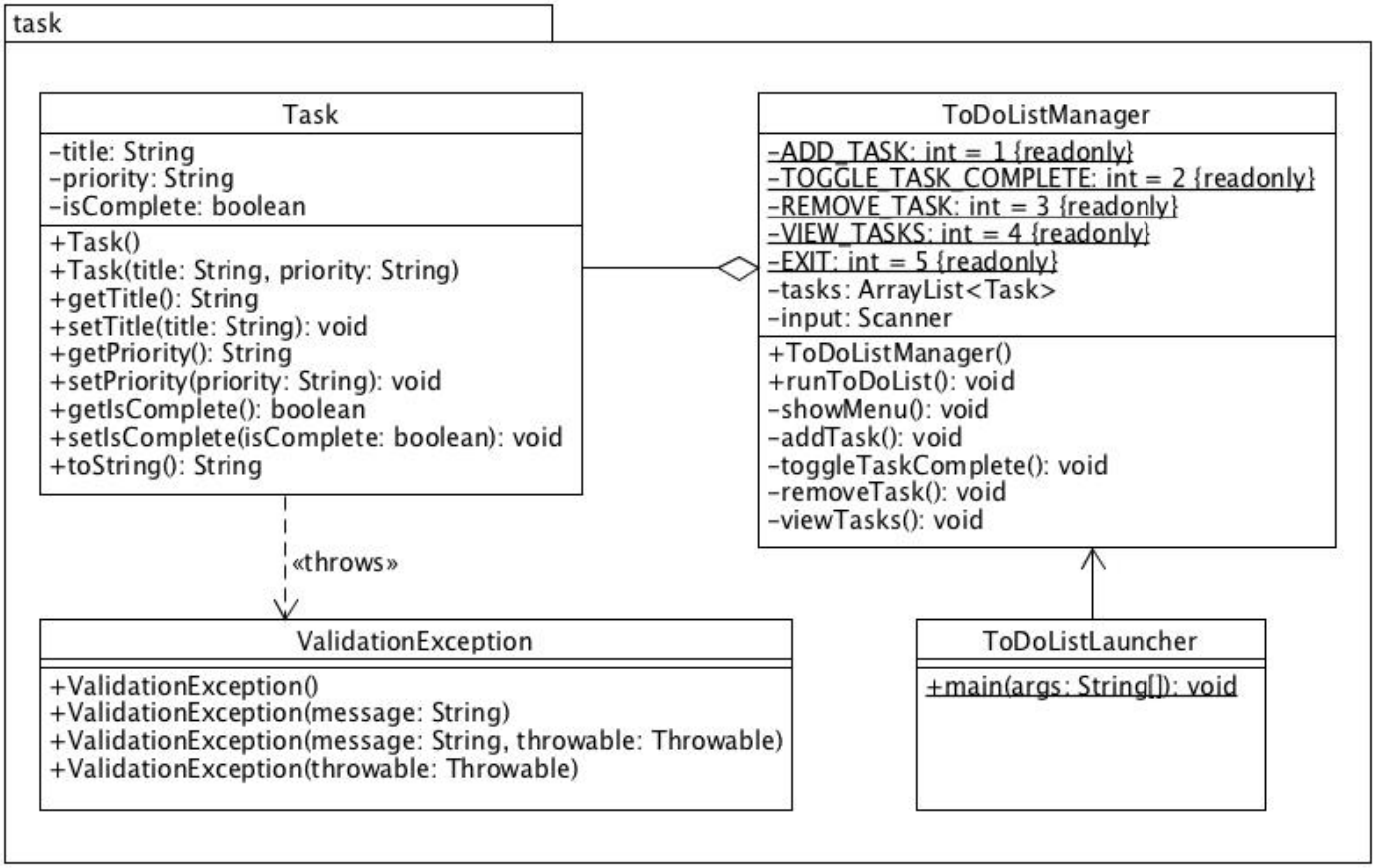
CST8132 Lab Assignment 03 – To Do List

Due: Friday April 1, 2016 by 11:59pm as upload to Blackboard

Problem Description

- In this assignment you will create a program that acts as a simple To-Do-List
- The user should be able to add and remove tasks, view all tasks, toggle as task as complete and vice-versa, and exit the program.
- The core concept for this assignment is Exception Handling, with an introduction to using the ArrayList<E> container from java.util.
- The project design includes 4 classes
    - See the appendix for starter code (Cite professor using programmer comments)
        - classes ValidationException and ToDoListLauncher are provided in full
        - class Task is partly completed
        - A Unit Test class for class Task is also provided in full, it should be used to test your work and will be used by your lab professors to do the same when grading.
            - Review this carefully, it could reappear, in part, on the final exam
    - You need to update class Task, and author class ToDoListManager

Program Design



(Diagram created using UMLet)

Implementation Notes

Class Task

- The constructor should call the set methods for title and priority, leave isComplete alone as it defaults to false. You will need to add an appropriate throws clause to the contructor
- setTitle(String title)
    - verify that the incoming title is:
        - not null
        - not an empty string
        - not longer than 25 characters
    - if any one of the criteria above are met throw a new ValidationException setting an appropriate message.
    - if none of the criteria are met (data is good) then set the title into the title field (trim it as you set it)
    - You will need to add throws ValidationException to this method header
- setPriority(String priority)
    - verify that the incoming priority is:
        - not null
        - not an empty string
        - one of "high", "medium", "low" this should be a case insensitive comparison.
    - if any one of the criteria above are met throw a new ValidationException setting an appropriate message.
    - if none of the criteria are met (data is good) then set the priority into the priority field (trim it as you set it)
    - You will need to add throws ValidationException to this method header

    Tips: don't forget about String methods trim(), length(), toLowerCase(), and equals(String)

- Add a toString() method that overrides the inherited one from class Object
  Use a StringBuilder, or String.format to generate a String similar to:

  ```
  Do homework (low) is not complete
  ```

    - Title was "Do homework", priority was "low", isComplete was false

  ```
  Eat pizza (high) is complete
  ```

    - Title was "Eat pizza", priority was "high", isComplete was true

Class ToDoListManager

- Add the required fields
- Make sure all fields are private
- The constructor should instantiate the ArrayList and the Scanner
- runToDoList()
    - Typical menu with switch statement in a loop with the needed menu options
    - Add a try-catch however to handle:
        - Text entries instead of numbers for menu options
    - In the case of a bad input, print out an appropriate message and let the loop continue.
- showMenu()
    - called from runToDoList(), all this does is output the menu on the console

- addTask()
  - get inputs from the user for task title and task priority
  - instantiate a Task using the constructor
  - assign the reference to the Task into the ArrayList named tasks
  - catch any ValidationExceptions and output an appropriate message
  - No need to loop here, if an exception is thrown just let the method end
  - (See Appendix with a sample run of the program)
- toggleTaskComplete()
  - Check that there are tasks before doing anything else
  - Ask user for index of Task in the ArrayList to toggle
  - After obtaining the Task from the ArrayList change the isCompleted into what it is not. (Tip: !) You will need to use getIsComplete() and setIsCompleted(boolean)
  - Two things can go wrong here, InputMismatchException and IndexOutOfBoundsException
  - Trap both and output appropriate message for each
  - No need to loop here, if an exception is thrown just let the method end
  - (See Appendix with a sample run of the program)
- removeTask()
  - Check that there are tasks before doing anything else
  - Ask user for index of Task in the ArrayList to remove
  - Two things can go wrong here, InputMismatchException and IndexOutOfBoundsException
  - Trap both and output appropriate message for each
  - No need to loop here, if an exception is thrown just let the method end
  - (See Appendix with a sample run of the program)
- viewTasks()
  - Check that there are tasks before doing anything else
  - Loop over the ArrayList calling toString() on each Task object as well as appending the index number to the output
  - (See Appendix with a sample run of the program)

Tip:

- This program uses one Scanner(System.in) with a mix of calls to nextInt() and nextLine(), make sure you call nextLine() after every nextInt() to remove leftover line-terminator characters from the input Stream (System.in).
- Also, if an exception is thrown and caught make sure you nextLine() on the Scanner to clean out bad data.

Tasks

- Use the UML, Starter Code, provided unit test class, and implementation notes to make a working program.
- Find one web tutorials on Exception handling in Java, and one web tutorial on using ArrayList and include these in your discussion document as APA references.
- Write one-page essay, using MS Word, that addresses the discussion questions (below) as a guide to discuss what was accomplished in the assignment, cite and reference any sources used online to help you answer questions. (You may use more than one page, but professors will stop reading after 3 pages).

Discussion Questions

- The ValidationException used in the lab extended Exception. Is ValidationException a runtime exception or not? What does this mean with regards to throws clauses as well as catch blocks?
- In general terms would using an Array for this type of program be easier or harder than using ArrayList? List a few problems with using an array over an ArrayList. Tip: Search the web with "Java Array vs ArrayList" (without the quotes) for ideas, make sure you cite your source(s).
- When you called input.nextLine() as part of your exception catching did you place it into every catch block or just inside a finally block at the end of the try-catch. What is the advantage or disadvantage to either approach?

Submission Requirements

- Place source code files and essay document into a zip archive and upload it to Blackboard by the due date.
- The essay should be an electronic document, Microsoft Word format, include your name inside the document.
- Your lab professor will indicate any additional submission requirements to you in the lab

Notes on citations and references

- Please do not cite or reference other students, ask for the original sources from them and cite and reference those instead
- You will not get credit for an assignment or project if large portions are copied directly from other sources, even if you cite and reference the source(s) correctly. Assignments are to be your own original work; other works can be used for help in solving problems or as small pieces of your larger program. Determinations on this are up to the discretion of the professor, if in doubt check with your professor.
- Note: One exception to this is in the case of lecture handouts, and code samples posted to Blackboard. You are free to use these as a starting point just cite + reference them as a personal communication from your professor e.g. Stanley Pieda (2015) personal communication.

UML Class Diagram refresher / review:

Donald Bell. (2004). UML basics: The class diagram. Retrieved from http://www.ibm.com/developerworks/rational/library/content/RationalEdge/sep04/bell/#N100A5

JUnit Testing – Learning Resources (See Blackboard for larger list)

Lars Vogel. (2015). Unit Testing with JUnit – Tutorial. Retrieved from http://www.vogella.com/tutorials/JUnit/article.html

Java Coding Conventions (Reference only)

Oracle. (April 20, 1999). Code Conventions for the Java Programming Language. [webpage] Retrieved from http://www.oracle.com/technetwork/java/index-135089.html
You will need to follow links and use the table of contents.
(PDF version available here: http://www.oracle.com/technetwork/java/codeconventions-150003.pdf)

Grading Guide

Criteria* (Equal Weight)	Needs Work (0)	Poor (1)	Intermediate (2)	Excellent (3)	Value Scored
Code Conventions	Java coding conventions are not followed	Java coding conventions are not well followed.	Java coding conventions are followed with several inconsistencies	Java coding conventions are closely followed.	
Comments (Javadoc not required Assign 3)	Comments missing or incorrect.	Many classes and / or class members missing meaningful comments	Very few classes and / or members missing comments, comments are meaningful.	Nearly everything is commented, comments are meaningful, brief, well written.	
Discussion	Missing or only repeats the questions	Provides brief answers to questions with no explanations or examples	Questions are answered, explained, and examples are used from the code to illustrate	Questions are answered and explained with examples from code, student discusses how concepts in this program will be beneficial in future programing.	
Citations Minimum 3 references (Exceptions, ArrayList, Array vs ArrayList)	No referenced work cited when it needed to be. See Algonquin College Policy AA20 on Plagiarism.	References and citations present but not APA style, e.g. only the URL provided.	References and citations loosely follow APA style	References and citations closely follow APA style	
Compiles	Program does not compile, too many syntax mistakes for the professor to track or debug without major re-write.	Program does not compile, has several syntax mistakes	Program does not compile, has a few small syntax mistakes	Program compiles	
Concepts (Exceptions, ArrayList)	Concepts are not demonstrated, program does not follow UML design.	One or two concepts demonstrated, program loosely follows UML design	Program demonstrates most of the concepts, closely follows UML design	Program demonstrates understanding and application of concepts notably inheritance, polymorphism, abstract and follows UML design exactly.	
Execution	Program does not run, or crashes on startup	Program runs but crashes on unexpected input	Program runs and does not crash, outputs are a close match to handout example	Program runs, does not crash, outputs are an exact match to handout example.	
Unit Tests	Unit tests omitted from student submission (or modified to pass tests that would normally fail)	Provided unit test has many failed tests	Provided unit test has few failed tests	Provided unit test has no failed tests.	
				Total:	

Max Points: 24

Appendix – Additional Notes

Program Comments Note:

• At the top of each source code file include the following comment header, adding the needed information:	`/* File Name:`
	` * Course Name:`
	` * Lab Section:`
	` * Student Name:`
	` * Date:`
	` * Author: Stanley Pieda (2015) Personal Communication`
	` * Updated By: (Student's name)`
	` */`

• Classes and class members (class level fields, constructors, methods) should have a brief description as a comment immediately above in the code listing. Example:	`/*`
	` * Provides interaction with the user in the`
	` * form of a console menu system. Options`
	` * include Practice test, Print Report,`
	` * and Exit Program.`
	` */`
	`public void runMenu(){`

Javadoc comments are not required for Assignment 3 but if you want to explore using them go ahead. (We will learn about Javadoc comments shortly in a future lecture)

Appendix: Starter Code (If you copy and paste fix your indentation)

```
package task;
public class Task {
    private String title;
    private String priority;
    private boolean isComplete;

    public Task(){
        title = "No title";
        priority = "No priority";
    }

    // things to do here:
    public Task(String title, String priority){
    }

    public String getTitle(){
        return title;
    }

    // things to do here
    public void setTitle(String title){

        this.title = title.trim();
    }

    public String getPriority(){
        return priority;
    }

    // things to do here
    public void setPriority(String priority){
        this.priority = priority.trim();
    }

    public boolean getIsComplete(){
        return isComplete;
    }

    public void setIsComplete(boolean isComplete){
```

```
        this.isComplete = isComplete;
    }

}
===== ===== ===== ===== ===== =====
package task;
public class ValidationException extends Exception{

    public ValidationException(){
        super("There was a problem when validating data");
    }

    public ValidationException(String message){
        super(message);
    }

    public ValidationException(String message, Throwable throwable){
        super(message, throwable);
    }

    public ValidationException(Throwable throwable){
        super(throwable);
    }
}
===== ===== ===== ===== ===== =====
package task;
public class ToDoListLauncher {

    public static void main(String[] args) {
        (new ToDoListManager()).runToDoList();
    }
}
```

Appendix: Sample run of program

Note:
In the Eclipse console:

- user input is in green text
- System.err.println() outputs are in red.

Here

- user inputs were made black font, bolded, and highlighted in yellow
- System.err.println() outputs were made black font, underlined, and bolded

```
1 to add a task
2 to toggle a task's is completed
3 to remove a task
4 to view all tasks
5 to exit program
delicious tuna
Please enter integer numbers for menu selection
1 to add a task
2 to toggle a task's is completed
3 to remove a task
4 to view all tasks
5 to exit program
42
unrecognized command
1 to add a task
2 to toggle a task's is completed
3 to remove a task
4 to view all tasks
```

```
5 to exit program
2
Nothing to toggle, no tasks
1 to add a task
2 to toggle a task's is completed
3 to remove a task
4 to view all tasks
5 to exit program
3
Nothing to remove, no tasks
1 to add a task
2 to toggle a task's is completed
3 to remove a task
4 to view all tasks
5 to exit program
4
Nothing to show, no tasks
1 to add a task
2 to toggle a task's is completed
3 to remove a task
4 to view all tasks
5 to exit program
1
Please enter task title
(Title cannot be empty, max 25 characters)
Homework
Please enter task priority (high, medium, low)
low
1 to add a task
2 to toggle a task's is completed
3 to remove a task
4 to view all tasks
5 to exit program
1
Please enter task title
(Title cannot be empty, max 25 characters)
Eat pizza
Please enter task priority (high, medium, low)
high
1 to add a task
2 to toggle a task's is completed
3 to remove a task
4 to view all tasks
5 to exit program
4
index: 0 Homework (low) is not complete
index: 1 Eat pizza (high) is not complete
1 to add a task
2 to toggle a task's is completed
3 to remove a task
4 to view all tasks
5 to exit program
1
Please enter task title
(Title cannot be empty, max 25 characters)
This title is too long to use in this program
Please enter task priority (high, medium, low)
low
1 to add a task
2 to toggle a task's is completed
3 to remove a task
4 to view all tasks
5 to exit program
```

There was a problem adding a task:
Title cannot exceed 25 characters
Please try again.
1
Please enter task title
(Title cannot be empty, max 25 characters)

User only used enter key

Please enter task priority (high, medium, low)
high
There was a problem adding a task:
Title cannot be empty
Please try again.
1 to add a task
2 to toggle a task's is completed
3 to remove a task
4 to view all tasks
5 to exit program
1
Please enter task title
(Title cannot be empty, max 25 characters)

User entered only whitespace

Please enter task priority (high, medium, low)
low
There was a problem adding a task:
Title cannot be empty
Please try again.
1 to add a task
2 to toggle a task's is completed
3 to remove a task
4 to view all tasks
5 to exit program
1
Please enter task title
(Title cannot be empty, max 25 characters)
task title
Please enter task priority (high, medium, low)
maximum
There was a problem adding a task:
Priority must be high, medium, or low
Please try again.
1 to add a task
2 to toggle a task's is completed
3 to remove a task
4 to view all tasks
5 to exit program
1
Please enter task title
(Title cannot be empty, max 25 characters)
task title
Please enter task priority (high, medium, low)

User only used enter key

There was a problem adding a task:
Priority cannot be empty
Please try again.
1 to add a task
2 to toggle a task's is completed
3 to remove a task
4 to view all tasks
5 to exit program
1
Please enter task title
(Title cannot be empty, max 25 characters)
task title

User entered only whitespace

```
Please enter task priority (high, medium, low)

There was a problem adding a task:
Priority cannot be empty
Please try again.
1 to add a task
2 to toggle a task's is completed
3 to remove a task
4 to view all tasks
5 to exit program
2
Please enter index number of task to toggle complete / incomplete
Tuna
Could not toggle task complete
Please enter only integers for index
Please try again
1 to add a task
2 to toggle a task's is completed
3 to remove a task
4 to view all tasks
5 to exit program
2
Please enter index number of task to toggle complete / incomplete
42
Could not toggle task complete
Please enter a valid index
0 to 1
1 to add a task
2 to toggle a task's is completed
3 to remove a task
4 to view all tasks
5 to exit program
2
Please enter index number of task to toggle complete / incomplete
1
1 to add a task
2 to toggle a task's is completed
3 to remove a task
4 to view all tasks
5 to exit program
4
index: 0 Homework (low) is not complete
index: 1 Eat pizza (high) is complete
1 to add a task
2 to toggle a task's is completed
3 to remove a task
4 to view all tasks
5 to exit program
3
Please enter index number of task to remove
moar tunas
Could not remove task
Please enter only integers for index
Please try again
1 to add a task
2 to toggle a task's is completed
3 to remove a task
4 to view all tasks
5 to exit program
3
Please enter index number of task to remove
452
Could not remove task
```

Please enter a valid index
0 to 1
1 to add a task
2 to toggle a task's is completed
3 to remove a task
4 to view all tasks
5 to exit program
3
Please enter index number of task to remove
1
1 to add a task
2 to toggle a task's is completed
3 to remove a task
4 to view all tasks
5 to exit program
4
index: 0 Homework (low) is not complete
1 to add a task
2 to toggle a task's is completed
3 to remove a task
4 to view all tasks
5 to exit program
55555555555555555555555555555555555555555555555555555555555555555555555555555555555555555555555555
Please enter integer numbers for menu selection
1 to add a task
2 to toggle a task's is completed
3 to remove a task
4 to view all tasks
5 to exit program
5
program will exit