CSE 7393

Assignment 5

Due: Sep 29 (in class) / Oct 13 (distance)

Part A.

Jargoneer Version  5

Jargoneer version 4 uses a RecordStore to remember user lookups. Now in Jargoneer 5 we will present the user with words that our midlet already knows about. When the Midlet starts up, the user should see a Form that contains:
  - a text field for the user to enter a word (just like previous Jargonners)
  - a list of words that are available in the RecordStore, displayed in alphabetical order (new)

If the user selects a word from the list, the definition should be displayed with no mention of connecting to the Internet. Do not concern yourself with how long the list gets. Assume the user is willing to scroll.

--

Note on refactoring:

You may want or need to refactor your code to make it more modular and easier to work with in order to add the above modifications. A general rule for reusability and modularity is that a method should only do ONE thing.

When you write a JavaDoc sentence describing your method (you are doing this, aren’t you?) pay attention to how many verbs it has. If it takes more than one verb to describe your method, the method should probably be refactored into two – one associated with each verb.

Also, even though your methods may not need parameters, it’s good practice to pass in the data your method uses as parameters, even if the data is defined globally within the class. For example, if you have a method to load your Hashtable from the a RecordStore, if your method looks like:

```java
public void load() {
    myHash ..... 
}
```

There are two problems with this method:
  - it can only be reused by code that has a Hashtable name myHash and a RecordStore named myStore
it can’t be easily tested using JUNIT

Keep looking over your code and ask yourself, “Isn’t there a cleaner way to do this?”.
Getting the code to work correctly is a first step, but if it’s difficult to understand,
convoluted and has cryptic or non-existent comments, then as Donald Trump says in his
show The Apprentice, “You’re Fired!!”.

Name your new jar and jad: Jargoneer5

Part B.

- Write a Midlet called PickAndShow2. The midlet should display a list of names
  of images AND a reference to the CustomItem you created in the earlier
  exercise. Your list should contain at least 10 items and allow MULTIPLE
  SELECTION.
- Supply the user with an OK command and when the user presses OK, display the
  selected images in a slide show. Each image should appear for 2 seconds followed
  by the display of another image. When all images have been shown, continue the
  slide show by starting again from the first image. Basically, set things up so that
  there is an infinite loop over all the images selected.
- The form you use to display the images should contain an option (button) titled
  Stop. When the user presses Stop, revert back to the original screen and allow the
  user to select again from the list.
- Use a Timer and TimerTask to control a display such that each image displays for
  2 seconds.
- During this slide show display, the user should be able to EXIT the program.
  Make certain that when the user presses EXIT that your Timer thread gets
  terminated. To help with this, it’s suggested that you put a System.out.println
  statement at the beginning of the TimerTask run method. This way, if you
  terminate the Midlet but do not terminate the thread, you will continue to see
  output in the output window.

Submit:
- submit a zip file entitled a5<lastname>.zip to Blackboard using the Assignment 5
  link
- the zip should contain Jargoneer5 and PickAndSave2 jar/jad files plus all your
  source code

Is your Source Code readable?
- getting the program to work correctly makes up only part of your grade – don’t
  forget, your source code will be read!
- make certain that your source code is readable -- have a blank line separating all
  methods and a blank line between each major construct in your methods (for loop,
  while loop)
- each method should have a javadoc comment preceding it.
- avoid using tabs to control indenting – many editors have options for substituting
  tabs with actual spaces so that when the source is opened in another editor it looks
  good and does not wrap.
before you submit your source code, take a look at it in an simple editor such as Notepad or Textpad and print it. If it wraps, fix it before sending it in. Source code that wraps will lose points. One way to do this is to make certain no part of your program exceeds column 80.

- see coder tab vs spaces discussion at: [http://xarg.net/writing/tabs](http://xarg.net/writing/tabs)

NetBeans allows you to insert spaces when you press the Tab key. To check this, go to Tools…Options…Indentation Engines…Java Indentation Engine

Loading Images.

Much thanks to Murat Derici who posted this very fine outline of how to work with png images in your NetBeans project. Thanks much, Murat!

Here are the steps to follow:

1-) Create a res folder under your project folder (outside of NetBeans)
2-) Copy your images into it.
3-) Return back to Netbeans and select your project
4-) Right click on the your project
5-) Select Properties
6-) Under Build, select Libraries & Resources
7-) On the right pane, select Add Folder
8-) Navigate through it and select res folder.
9-) Good news, you are done! You deserve a pat on the back.

Here are some important notes. To reference these files in your program you must add a slash in front of the file name as in: "/xxx.png". When you build your main project, images are automatically moved to build/compile and build/obfuscated dirs. No sweat