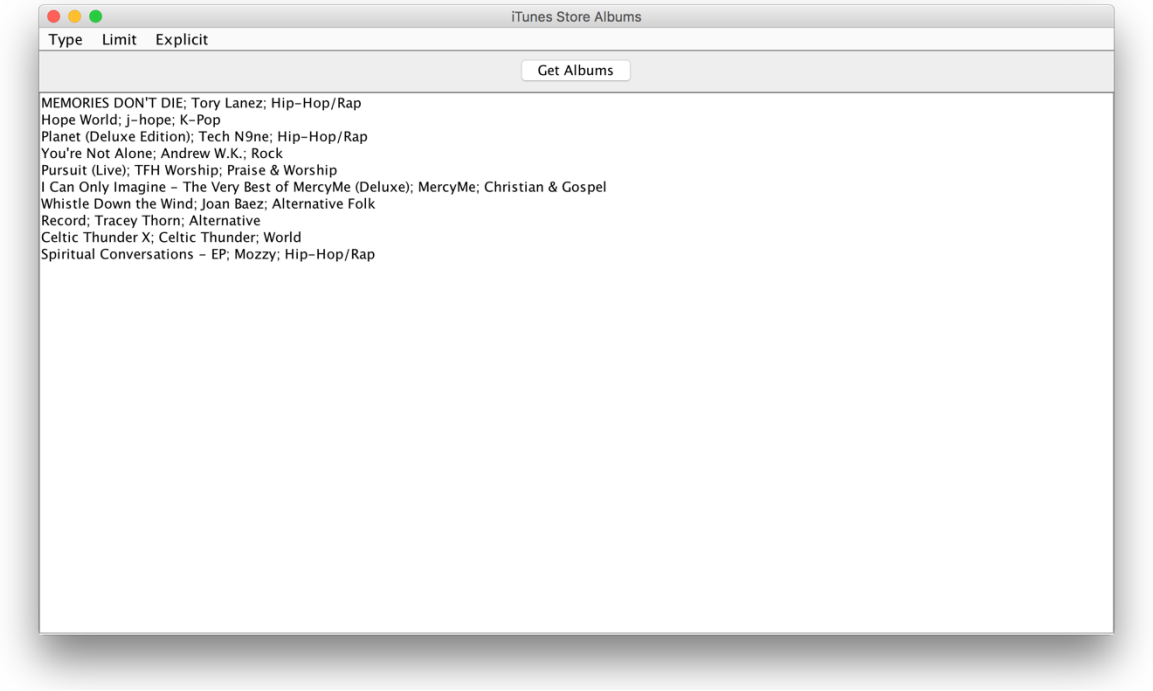
CSCI 470/680 Assignment 3

In this assignment, you will write a Java application to download XML, parse it, and display the results. The XML will contain information about albums available on the Apple iTunes Store.



The code for this assignment will require you to write multiple classes, which are described below. You do not necessarily need to use the same class names for your classes.

The XMLDownloader Class

This will be the main class for the assignment, and should extend `JFrame`. It will have a menu bar and should probably be responsible for handling events from the menu bar's menu items.

Suggested data members

- An `XMLDownloadPanel`.

Suggested methods

- The `main()` method
- A constructor (really just to set the title bar for the frame)
- `createAndShowGUI()` – Sets up the layout for the application as a whole and calls `createMenu()` to create the menu bar. I simply added the `XMLDownloadPanel` to the `CENTER` of the frame's content pane.
- `createMenu()` – Encapsulates code to create the menu bar, menus, and menu items, and to add listeners for the menu items.
- `actionPerformed()` – To handle action events from the menu items.

The Menu

All of the menu items for this assignment will be radio buttons. Clicking on one of them should set the value of a data member in the `XMLDownloadPanel` class. The values of those data members will then be used to construct the URL from which to fetch the album data XML.

The required menus and their menu items are:

- "Type" menu: Specifies the RSS feed type. Menu items should be "New Music", "Recent Releases", and "Top Albums".
- "Limit" menu: Specifies the number of results requested. Menu items should be "10", "25", "50", and "100".
- "Explicit" menu: Specifies whether or not explicit albums are allowed. Menu items should be "Yes" and "No".

One item from each menu should be selected by default (I selected "New Music", "10", and "Yes").

You do not need to set mnemonics or accelerators for the menus or menu items, but a little extra credit is available for doing so.

The XMLDownloadPanel Class

This should be a subclass of `JPanel`, and contain most of the user interface for this assignment. It will handle action events from the "Get Albums" button.

Suggested data members

- A `JButton` to "Get Albums".
- A `JTextArea` in which to display the fetched album data. You'll want to embed this in a scroll pane and make sure that it's wide enough to display all of the data for an album (I used 10 rows and 60 columns).
- An `XMLDownloadTask`.
- The feed type, results limit, and a `boolean` variable to indicate whether or not explicit albums are allowed.

Suggested methods

- A constructor that sets the layout, adds the button and text area to it, and sets a listener for the button. (I used a `BorderLayout` for the panel. I added the scroll pane its associated text area to the `CENTER`. I added a second panel with a `FlowLayout` to `PAGE_START`, and then added the button to that panel.)
- Methods to set the feed type, results limit, and explicit `boolean`. These will be called from the `XMLDownloader` class in response to the user clicking on menu items.
- `actionPerformed()` – To handle action events from the button. This can just clear the text area and call the `download()` method.
- `download()` – To initiate the download of the XML data. The logic for this method is:

    1. Build a URL string for the requested fetch.
    2. Create a new `XMLDownloadTask` and pass it the URL string and a reference to `this` (so that the task can access the text area in this class to display the data it downloads).

3. To prevent the user from pushing the “Get Albums” button repeatedly, you may want to create a `PropertyChangeListener` for the task. When the task has `STARTED`, you can disable the button. When the task is `DONE`, you can enable the button. (See the Swing Worker example posted on Blackboard for an example of how to set this up.)
4. Call the `execute()` method for the task.

The URL String

A valid URL string for the iTunes Music RSS feed should look like this:

`https://rss.itunes.apple.com/api/v1/us/itunes-music/new-music/all/10/explicit.atom`

The black sections of the string will not change. The three sections in red specify the feed type, the results limit, and whether or explicit albums are allowed. These obviously correspond to the menu items and the three data members described above.

- *Feed type:* Valid values here are `"new-music"`, `"recent-releases"`, and `"top-albums"`.
- *Results limit:* Any numeric value is valid.
- *Explicit allowed:* Valid values here are `"explicit"` and `"non-explicit"`.

The XMLDownloadTask Class

This is a subclass of `SwingWorker` that will be used to download the XML data in a background thread. The Swing Worker example posted on Blackboard is a good model of how to write this class.

This class may add the published Album objects to the text area in the parent class one at a time, and / or add them to an `ArrayList` and return the list when the task is complete so that they can be displayed all at once.

Suggested data members

- A `String` to hold the URL string passed to the constructor.
- A delegate variable to hold the object reference passed to the constructor.
- An empty `ArrayList` of `Album` objects (optional).

Suggested methods

- A constructor.
- `doInBackground()` – Downloads the XML data as a string, creates a SAX parser, creates an instance of the `AlbumHandler` class to handle SAX parse events, and parses the XML. See the **Downloading XML from a Web Page** and **Parsing XML Using SAX** notes on Blackboard for most of the necessary code.
- `process()` – Processes a `List` of `Album` objects that have been published by the `AlbumHandler` class.

The AlbumHandler Class

This should be a subclass of `DefaultHandler`. You may want to make it an inner class of the `XMLDownloadTask`, since that will give its methods direct access to the methods and data members of the outer class. Otherwise, you'll probably need to pass a reference to the XMLDownloadTask into your `AlbumHandler` so that you can access its members.

The content handler example on Blackboard is a good model for how to write the methods this class. To extract the data that we want for this assignment, your SAX event handling code will need to look for the following tags:

- `"entry"`: This tag encloses all of the information for an album.
- `"im:name"`: This tag encloses the name of the album.
- `"im:artist"`: This tag encloses the name of the album's artist.
- `"category"`: The genre of the album can be found as the `"label"` attribute of this tag.

There is one small complication in the XML that is not present in the example on Blackboard. The `"category"` tag appears multiple times in each album entry, but only the first occurrence contains the genre that we want in its `"label"` attribute. Your code will therefore need to process the first occurrence of the `"category"` tag and ignore any others for that entry.

Once all of the data for a particular album entry has been extracted, you can use it to create a new `Album` object. The `Album` object can then be published and / or added to a list in the `XMLDownloadTask` class to be returned when the task is complete.

The Album Class

A class to hold all of the information about an album.

Suggested data members

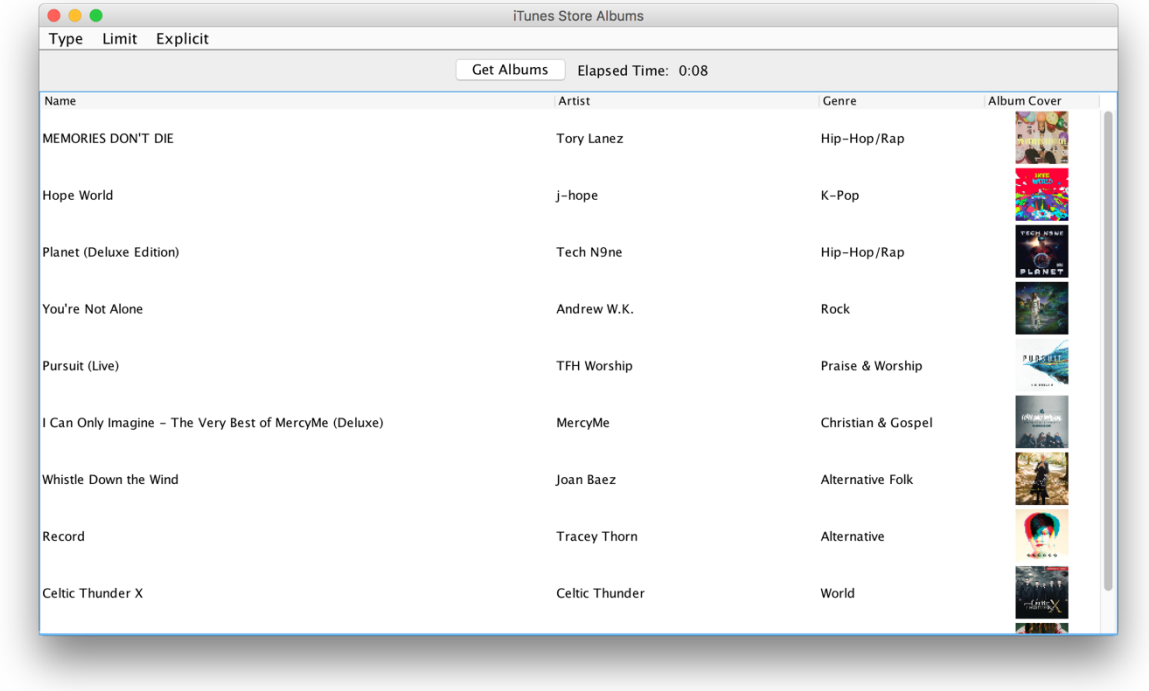
- Album name, artist name, and genre.

Suggested methods

- A constructor
- Methods to get the value of each of the data members
- Overriding the `toString()` method may be useful, especially if you choose not to do the extra credit and are simply displaying albums in a text area.

Extra Credit

A number of options for extra credit are available on this assignment. Because these things are extra credit, if you choose to attempt them, I will expect you to do the bulk of the work. I'll be happy to answer questions but if you can't get something to work, I'm not going to show you how to write the code.



- (2 points) Write accelerators for the items of the "Type" menu items. Your accelerators should use the correct shortcut key mask based on the platform on which the program is running (see the menu bar example on Blackboard for the code to do this). If you want to create mnemonics for the menus as well, you are welcome to do so.
- (2 points) Create tool tip text for the three menus.
- (10 points) Use a clock timer to show the time elapsed while the download is running. You can track the elapsed time in seconds using an integer counter, with the value displayed in a label (converted to minutes and seconds). Create a `Timer` object to call a method to increment the counter on a fixed schedule of once every 1000 milliseconds. Create the timer when the download task begins and cancel it when the download task is done.
- (15 points) Display the downloaded data in a `JTable` instead of simply appending it to a text area as text strings. By default, the columns of a JTable are all the same width. The following link has some code that will let you vary the widths of the columns as a percentage of the total width of the table:

  https://stackoverflow.com/questions/19012691/set-column-width-of-jtable-by-percentage

- (20 points) If you get the `JTable` working and want an additional challenge, also download and display the thumbnail images of the album covers in your table. `JTable` has a cell renderer for the `ImageIcon` class, so that's a good data type to use to store the downloaded image in the `Album` object. The image URL is enclosed in the XML tag `"im:image"`. You will probably need to write your own subclass of `DefaultTableModel` and override the `getColumnClass()` method to return the correct data type for each of the table's columns. You will also need to set the row height of the table rows to the height of your ImageIcons (plus

a couple of extra pixels if you want a little space separating the rows). The downloaded images will be 200 x 200, which is rather large; you can scale them down to 50 x 50 using the following method (you'll need to research the relationship between the `Image` and `ImageIcon` classes):

```
private Image getScaledImage(Image sourceImage) {
    BufferedImage resizedImage = new BufferedImage(50, 50,
        BufferedImage.TYPE_INT_ARGB);
    Graphics2D g2 = resizedImage.createGraphics();

    g2.setRenderingHint(RenderingHints.KEY_INTERPOLATION,
        RenderingHints.VALUE_INTERPOLATION_BILINEAR);
    g2.drawImage(sourceImage, 0, 0, 50, 50, null);
    g2.dispose();

    return resizedImage;
}
```

Note that your download will take significantly longer if you are also downloading images, especially with the larger result limits.

Other Notes

- Question: "How do you eat an elephant?" Answer: "One bite at a time." This is a fairly large assignment, and the key to getting it working is not trying to do everything all at once.
- You may not use a GUI form designer for this assignment. The UI is very simple, and it's easier for the TAs to grade the assignments if they have access to all of the code.
- Caught exceptions may simply print error messages to the console.
- The iTunes RSS Feed Generator can be found here:

  https://rss.itunes.apple.com/en-us

  If you paste the generated URLs into a web browser like Chrome, you can see the XML that will be downloading.
- Since the output from this assignment is being produced from an RSS feed, the albums displayed will likely change from day to day.