OU3 - RadioInfo
HT2022
2023-01-09

Theodor Jonsson
ens18trn@cs.umu.se

Applikationsutveckling (Java)
5DV222

1. Introduction

Sverige Radios API publishes data about their channels and radio programs as XML documents. We were tasked to create our own program which parses the XML document and displays the data in a user-friendly manner. This program was created and compiled in Java 17

2. User Guidance

To start the program, either double click the jar file or type “java -jar RadioInfo.jar”. When starting, the list of channels will be empty, the program will get the channels from the API and place them in the list. Figure 1 shows the program running and all channels in the list and the channel P1 is selected.

In this window, you can refresh the list using the “Refresh List” button and access the dropdown menu of “Channel Schedules” which holds loaded channel schedules. The dropdown menu will be empty until a schedule window has been opened and loaded in which can be achieved by double-clicking with mouse 1 on a channel.

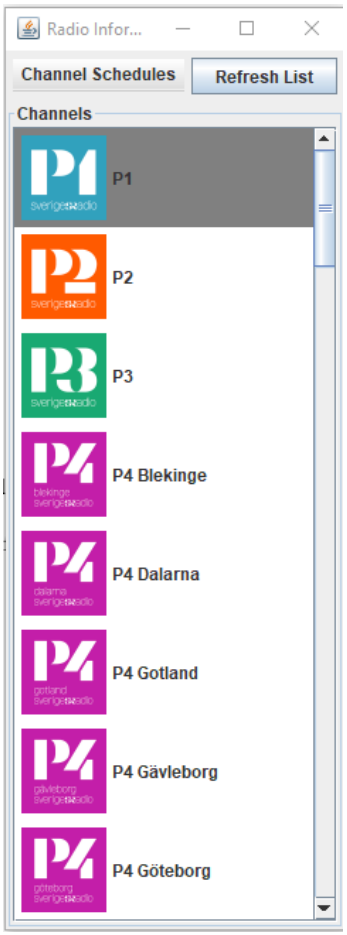
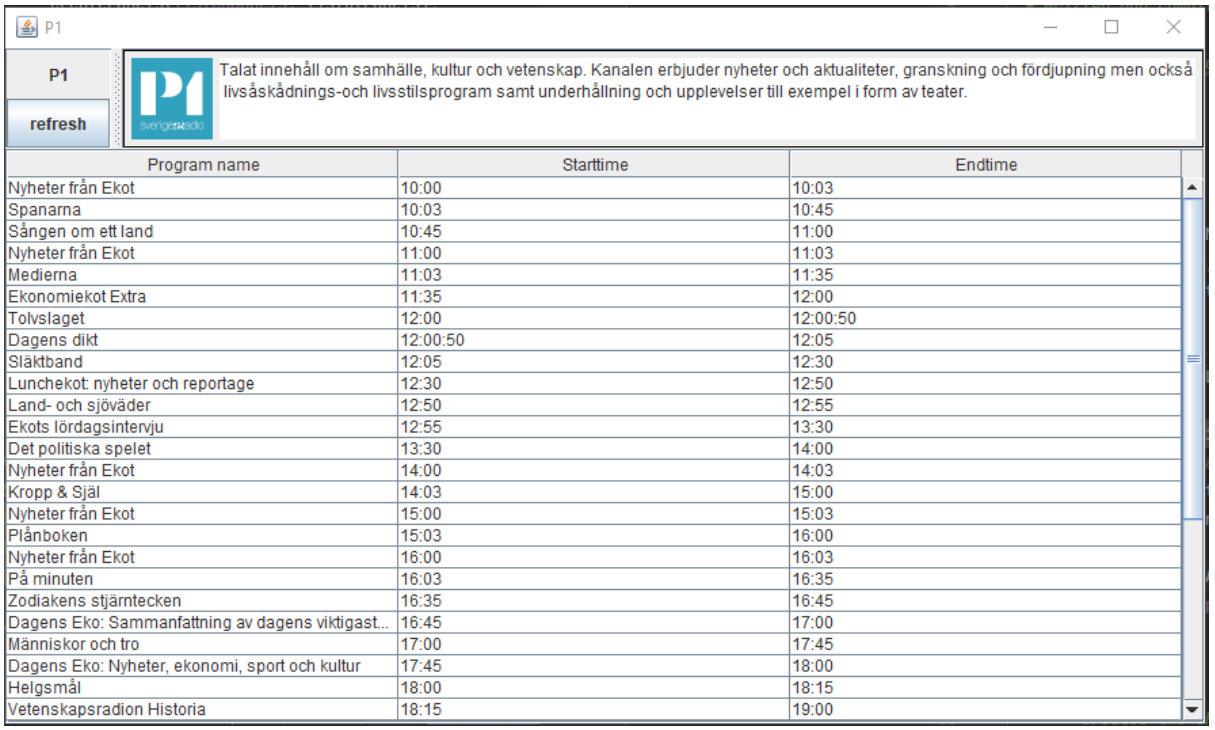


Figure 1. Main window of RadioInfo after channels have loaded in.

Figure 2 shows the channel P1 and its schedule opened in a separate window. The table shows all episodes of that channel ranging from 6 hours before and 12 hours after the time of loading in. The table shows program name, start time and endtime of all the episodes within the range.

The table will update automatically after an hour of being opened and then every hour after that, it can also be manually updated using the “refresh” button.
There is no limit to how many of these different windows you can have opened and when they are closed they can still be opened again from the dropdown menu in the main window.



Program name	Starttime	Endtime
Nyheter från Ekot	10:00	10:03
Spanarna	10:03	10:45
Sången om ett land	10:45	11:00
Nyheter från Ekot	11:00	11:03
Medierna	11:03	11:35
Ekonomiekot Extra	11:35	12:00
Tolvslaget	12:00	12:00:50
Dagens dikt	12:00:50	12:05
Släktband	12:05	12:30
Lunchekot: nyheter och reportage	12:30	12:50
Land- och sjöväder	12:50	12:55
Ekots lördagsintervju	12:55	13:30
Det politiska spelet	13:30	14:00
Nyheter från Ekot	14:00	14:03
Kropp & Själ	14:03	15:00
Nyheter från Ekot	15:00	15:03
Plånboken	15:03	16:00
Nyheter från Ekot	16:00	16:03
På minuten	16:03	16:35
Zodiakens stjärntecken	16:35	16:45
Dagens Eko: Sammanfattning av dagens viktigast...	16:45	17:00
Människor och tro	17:00	17:45
Dagens Eko: Nyheter, ekonomi, sport och kultur	17:45	18:00
Helgsmål	18:00	18:15
Vetenskapsradion Historia	18:15	19:00

Figure 2. Schedule window of the channel P1

3. System Description

The class RadioInfo is where main is located which will create an EDT (Event Dispatch Thread) where the GUI will be created. MainMenu is responsible for the creation of the GUI for the main window which can be seen in Figure 1. MainMenu will also create a class called ChannelList which is responsible for creating the class parseXMLAllChannels.

This class is a SwingWorker that parses the XML document that holds all channels. The XML documents will be parsed using a DOMParser. To make the XML document into a Document object it will use a factory to get a builder that then will parse the XML document into a Document object. Then it will go through the Document to get the name, id, image and tagline of all the channel elements and add them to a Channel object and from these also create a JPanel with the name and image. The SwingWorker will add the channels and JPanels to two different lists which it sends back to ChannelList. The list of JPanels will be added as the data for the JList in MainMenu.

When selecting an item in the JList the visual part is handled by ChannelListCellRenderer and the creation of a new schedule window as seen in Figure 2 is handled by ListSelectionHandler.

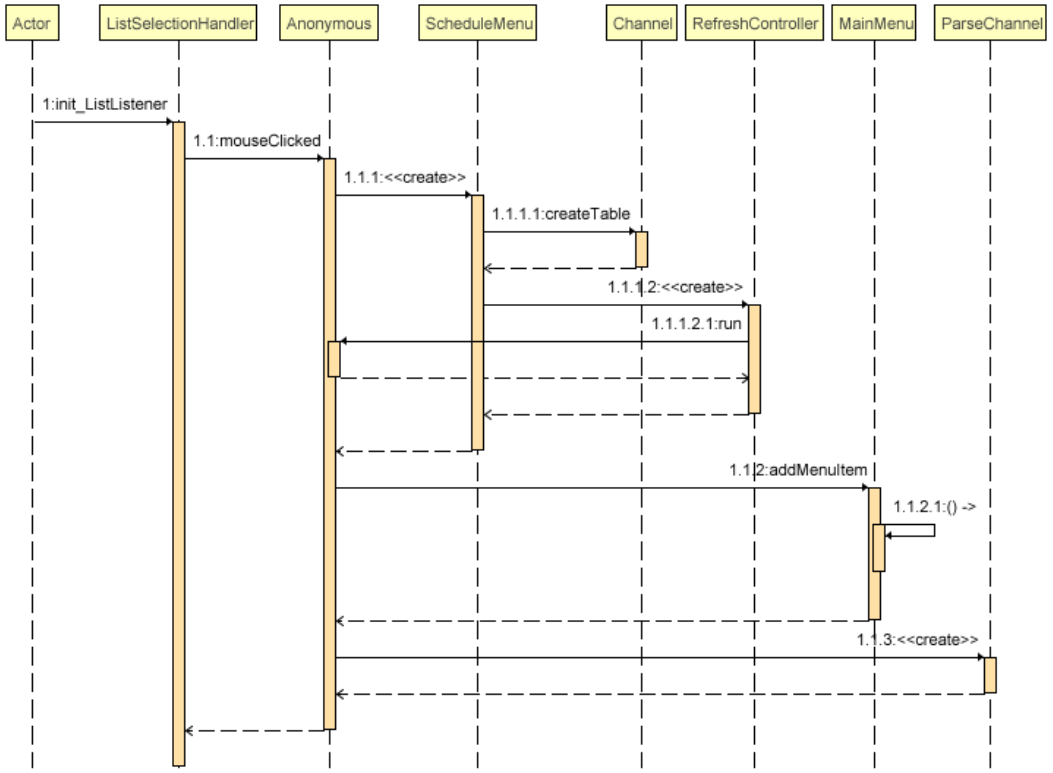


Figure 3. Sequence Diagram over the creation of individual channel menu

The ListSelectionHandler adds a MouseListener to the JList which when “mouse 1” is double clicked it will get the selected index and create a ScheduleMenu for that index then create a ParseChannel which is a SwingWorker this class will parse the XML documents for the individual channels. The class will go through the same process when parsing the XML document as the other Swingworker. It will go through the document to find the title, start time and end time for all episodes and store these in a 2D String array but it will only store the

episodes that fit the timeframe discussed in User Guidance. When the SwingWorker is done it will send the data to the Channel to be added to a table. The ScheduleMenu creates a GUI for the individual Channels which can be seen in Figure 2 the table of episodes is obtained from Channel. ScheduleMenu also creates a RefreshController which creates a TimerTask that will after an hour then every hour create a new ParseChannel and repeats its process, it also adds an ActionListener which will do the same thing. Figure 3 shows and UML diagram over the entire system

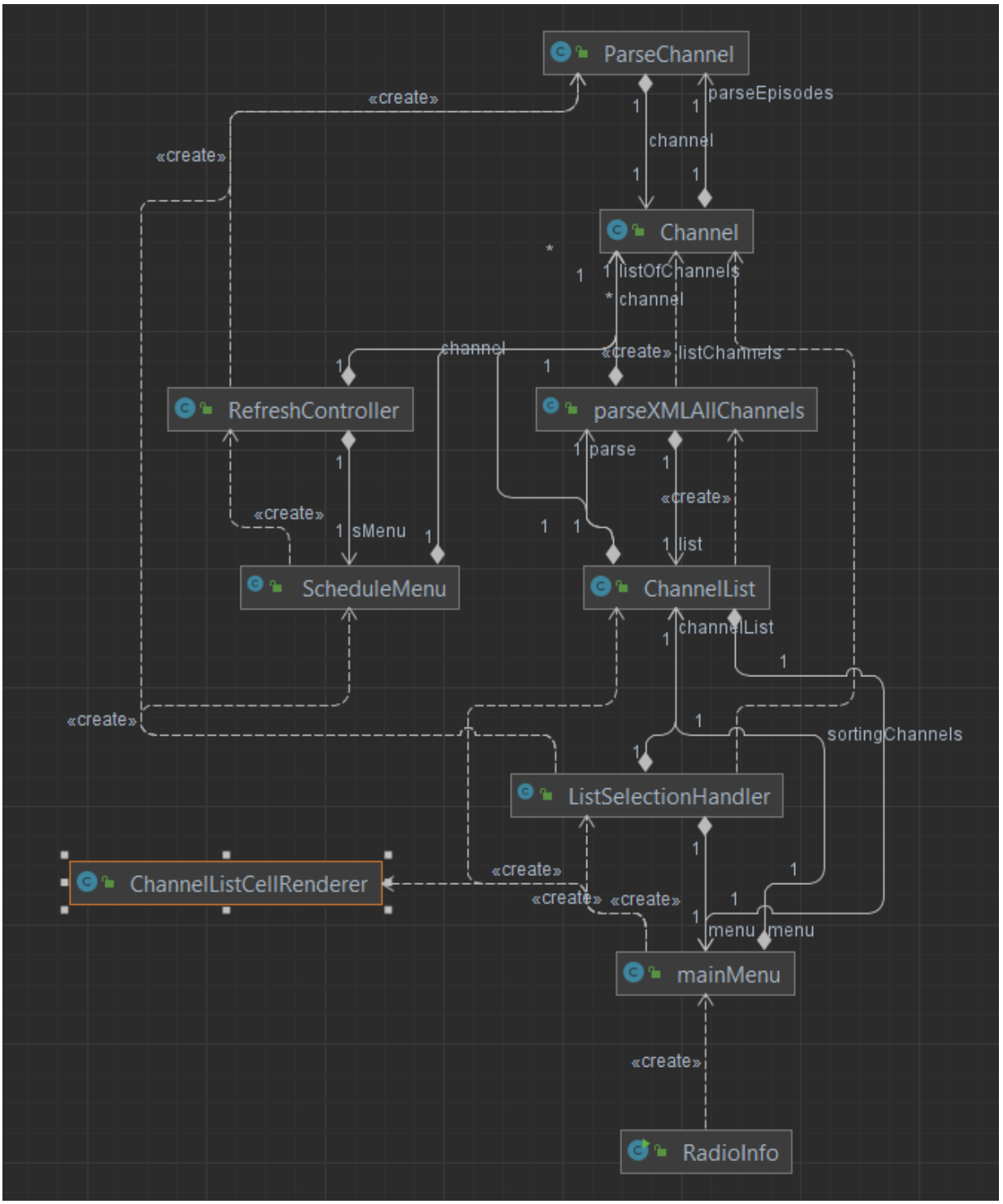


Figure 3. UML diagram over the project

4. Thread-Safety

In this system there are three different places a new thread can be created firstly at the start when the EDT is started for handling the GUI, secondly when parsing the and processing the data from the XML document of all channels a SwingWorker is started and lastly when parsing and processing the data from the XML documents of individual channels. The SwingWorker for parsing individual channels is the easiest to show thread-safety because the method doInBackGround only handles with variables it created itself and no Swing methods are used in it only when done is called and the thread is closed can it use an JOptionpane to notify if something when wrong.

The parser for all Channels is similar to the other parser class in which it only deals with variables it created itself but it does create JPanel but this is in the done method which means that the thread is closed.

Lastly the EDT thread no Swing methods are used in the main thread because SwingUtilities.invokeLater is used and all Swing methods in the SwingWorkers are in the done method which means they are processed on the EDT thread

5. Design Patterns

The design pattern Observer is used for the ActionListener of the refresh buttons, Mouselistener of the JList and for the table model of the JTable. Factory is used for the DOMParser in creating the document object used for converting the XML document into something useful.

6. Limitations

The system has some limitations that I know that I didn't have the time to fix.
The schedule windows should have been made into singletons to stop the user from opening more than one of the same schedule windows.

The schedule window doesn’t show episodes that wouldn’t have ended 6 hours before the start time has to be within the 6 hour time limit similarly with the episodes that start within the 12 hour time limit but end outside of the limit won’t be shown.

Would have wanted to use the process and publish methods in the SwingWorker to get some data out to the GUI when it’s processing
But probably the biggest thing is that big parts of the classes are very codependent of each other which isn’t desirable.