CIS 422

Developer Documentation for YACC

This document was derived from EyeDraw Programmer's Documentation by Anthony Hornof, Rob Hoselton and Anna Cavender

PDF: https://www.cs.uoregon.edu/Classes/10W/cis422/Handouts/EyeDrawDocumentation.pdf
Online: https://www.cs.uoregon.edu/research/cm-hci/EyeDraw/

Version: 1.2 Date: 02/04/2019

Table of Contents

1. Software Installation

The most current version of our code is available https://github.com/pork3/team499ms.
Prior to installation, the user should ensure the following requirements are met:

1. Java Runtime Environment 11
2. Git (optional) as github distributes archived source, but highly recommended
3. GNU/Linux Operating System

1.1. Binary Install

A binary package is available through github, either visit https://github.com/pork3/team499ms and download the zip file, extract and run the jar file, or use git

```
$ git clone https://github.com/pork3/team499ms.git
$ cd team499ms
$ java -jar YACC.jar
```

1.2. Source Install

Source installation is offered through two means, one is an install script available online. Just download the script, set proper permission, then enjoy.

1.2.1. Installation Script

1. Download installation script

```
$ wget ix.cs.uoregon.edu/~zbower/shell/downloadYACC.sh
$ ./downloadYACC.sh
```

2. There is now a YACC.jar file located in the following directory:

```
$ ../team499ms/Code/src/
```

3. You can move the .jar file to any location you want, and it can be run with the command

```
$ java -jar YACC.jar
```

1.2.2. Manual Installation

1. First download both YACC and JSON dependency

```
$ git clone https://github.com/pork3/team499ms.git
$ git clone https://github.com/stleary/JSON-java
```

2. Compile the JSON library

```
$ mkdir org
```

```
$ mkdir /org/json
$ javac JSON-java/*.java
$ mv JSON-java/*.class JSON-java/org/json
```

3. Copy JSON library to proper location

```
$ cp -r JSON-java/org /team499ms/Code/src
```

4. Compile YACC

```
$ cd team499ms/Code/src
$ javac edu/uoregon/cs/calendar499/*.java
```

5. Create Manifest.mf file

```
$ echo "Manifest-Version: 1.0" >> MANIFEST.MF
$ echo "Class-Path: ." >> MANIFEST.MF
$ echo "Main-Class: edu.uoregon.cs.calendar499.Main" >> MANIFEST.MF
$ echo "Created-By: SICKboy" >> MANIFEST.MF
$ mkdir META-INF
$ mv MANIFEST.MF META-INF
```

6. Create Jar

```
$ jar cvfM YACC.jar META-INF/MANIFEST.MF
edu/uoregon/cs/calendar499/*.class edu/uoregon/cs/calendar499/*.png
org/json/*.class
```

7. There is now a YACC.jar file located in the following directory:

```
 $ ../team499ms/Code/src/
```

8. You can move the .jar file to any location you want, and it can be run with the command

```
$ java -jar YACC.jar
```

1.3. Software Requirements

YACC aims to run on any system capable of handling the Java Runtime Environment. Currently this is the only software requirement. If a user is building from source, the user will have to aquire the org.JSON library. YACC may work on other systems, but currently only GNU/Linux is supported.

1.4. Software Resources

YACC uses multiple resources from the Java library including: JFrames, JPanel, KeyPressListener, KeyReleaseListener, and MouseClickListener. Information regarding the specifics of how these components operate can be found within the Java documentation.

- JFrame: https://docs.oracle.com/javase/7/docs/api/javax/swing/JFrame.html

- JPanel: https://docs.oracle.com/javase/7/docs/api/javax/swing/JPanel.html
- KeyListener: https://docs.oracle.com/javase/tutorial/uiswing/events/keylistener.html
- ClickListener: https://docs.oracle.com/javase/tutorial/uiswing/events/actionlistener.html

YACC uses the JSON parsing library org.json
- Github: https://github.com/stleary/JSON-java
- Documentation: https://stleary.github.io/JSON-java/
- Tutorial/Wiki: https://github.com/stleary/JSON-java/wiki
- License: https://github.com/stleary/JSON-java/blob/master/LICENSE

2. Developer Use

2.1. Introduction

This document covers the source code for the YACC calendar, an open source calendar application developed in Java for University of Oregon class CIS422 Software Methodology.

This document assumes the reader has fundamental knowledge in Java as well as fundamental programming paradigms. This document also assumes the reader is familiar working in a GNU/Linux operating system.

This document provides an overview of each source file, giving the reader information on the inner workings of each file. This document is intended to assist future and current developers in creating new features or editing existing features.

2.2. Source Files

2.2.1. CalendarEvent.java

CalendarEvent.java is a class used to create, store and edit events within YACC. The class contains four data members described in more detail below. These data members hold the information their assigned names indicate. All members follow standard data hiding procedure, with each member being private, but able to be retrieved with a getter function or changed with a setter function.

2.2.1.1. Data Members

- String title
- Calendar timeStart
- Calendar timeEnd
- String note

This class contains these four members to hold the information contained within the calendar event. The title of the event and any notes the user may add are stored as the Java String object. The start and end time are stored as the Java Calendar object.

2.2.1.2. Constructor

- CalendarEvent(String title, Calendar timeStart, Calendar timeEnd)
- CalendarEvent(String title, Calendar timeStart, Calendar timeEnd, String note)

This class contains two constructors, to allow the creation of an event with optional data, or the creation of an event with 'required' data. In YACC's current state, the required information is: title, timeStart and timeEnd.

2.2.1.3. Getters and Setters

- getTitle() / getTimeStart() / getTimeEnd() / getNote()
- setTitle(String title) / setTimeStart(Calendar timeStart) / setTimeEnd(Calendar timeEnd) / setNote(String note)

Each data member described above contains a getter for retrieving the information contained within a class, and a setter for changing the information held within the class.

2.2.2. Cal.java

Cal.java is a class used to store, edit and retrieve all events within YACC. This class contains one public data member that is accessed by multiple classes.

2.2.2.1. Data Members

- HashMap<Calendar, ArrayList<CalendarEvent>> days

This is member is a standard instance of the Java hashmap, with Java Calendar objects used as keys to access a Java ArrayList of CalendarEvents. The calendar object and ArrayList allow for multiple events to be looked up in a single day, while the ArrayList stores all events within a single day.

2.2.2.2. Public Function

- ArrayList<CalendarEvent> grab(Calendar day)

This function takes an instance of the Java Calendar class, and attempts to return the ArrayList associated with the key. If no key exists, the hashmap is updated with a new key and a new instance of the ArrayList class is created and return.

2.2.3. CalMathAbs.java

This file contains many helper functions for getting dates in the calendar view. The file treats the calendar as a matrix performing calculations to change the display to the view.

2.2.3.1. Public Members

- int GetDayN(int i, int j, Calendar copy)

This function takes a Java Calendar object and two integers from the caller. The integers are passed from the view as positions in the calendar matrix. The function calls a helper function GetDayCal to determine which Java calendar object to use for calculations. Once this information is acquired the day is returned to the caller.

- String formatCalendarTime(Calendar c)

This function takes a Java Calendar object and formats the time to comply with the YACC choice of 24 hour time format. The calendar object is passed by the caller, then parsed to comply with a HH:MM time format.

- boolean IsCurrentDay(int i, int j, Calendar copy)

This function determines if the users reported date from their operating systems date and time is the day in integer i and month in integer j of the Java calendar object passed. The function first calls the GetDayCal helper function to determine the correct Java calendar object to work with. The function then creates a Java calendar object based of the users operating system, and determines if the dates: year, month, day are equal then returns true or false.

- String printDate(Calendar c)

This function is used to display the date in a different format. The callers calendar object is parsed for year, month and day, then a string in the form YY/MM/DD is returned.

- Calendar GetDayCal(int i, int j, Calendar copy)

This helper function takes a integer i and j and a Java calendar object then determines the calendar object to display for the day at i and j relative to the GUI matrix display.

- Calendar ClearTime(Calendar C)

This function takes a Java calendar, creates a copy and sets the time fields to 0. This is used by multiple functions to simplify the comparison of calendar objects.

- Calendar setTime(Calendar c, int h, int m, int s)

This function takes a Java calendar object, creates a copy and sets the time fields to those passed. The integers passed are h for hour, m for minute and s for seconds. The copy is returned to the caller.

2.2.4. ErrorMap.java

This file is the single way a component may pass error messages to the view. The files main purpose is to display error strings.

2.2.4.1. Constructor

- ErrorMap()

The constructor is used to create an instance of a Java hashmap object through the use of a helper function. This design was chosen as Java currently (Java 11) does not allow for the static creation of a hashmap.

2.2.4.2. Data Members

- HashMap<ErrorNumbers, String> er

This file contains one data member, an instance of the Java hashmap object. The hashmap stores an instance of the Enum ErrorNumbers as a key with a String error message as the value. This hashmap is created on runtime.

2.2.4.3. Public Members

- String GetCustomError(String message)

This function is used to create a custom error and append it to the error hashmap. The function first attempts to see if there is a custom error in the hashmap. If a custom error exists the value is overridden with the callers input string. If a value does not exists, the callers input string is mapped to the key CustomError.

- String GetError(ErrorNumbers en)

This function is used by the caller to return the string error message stored in the hashmap as a value. The caller inputs an instance of an Enum ErrorNumbers which is used as a key. The value is then returned to the caller.

2.2.4.4. Private Member

- void InitMap()

This function is a helper function called by the constructor. The function creates populates the created instance of the Java hashmap object when the ErrorMap object is created. Currently the only method of adding or removing error strings is through this function, excluding CustomError.

2.2.4.5. ErrorNumbers.java

This file is an instance of the Java enum object. This enum was created to allow for the passing of error codes to be more human readable. The possible values are as follows:

- NoError : Default error code, indicates no error.
- OutOfRange: Error message to indicate the date is out of range.
- DateError: Error message to indicate date does not exists, or malformed.
- FileFormat: Error message to indicate file format is unknown.
- FileRead: Error message to indicate a file read error.
- FileNotFound: Error message to indicate unable to locate file.
- LoadError: Error to indicate file was unable to be loaded into memory.
- CustomError: Error message message created and passed by developer.
- SaveError: Error to indicate a save error.

2.2.5. FileIOStatus.java

This class is used by both the LoadEvents module and the SaveEvents module to provide one way message passing to a calling function to denote the status of performing file I/O. This class does not contain any functions, however it does contain a lock that any routine may use to perform thread safe access and modification of values of this class.

2.2.5.1. Data Members

- currentStatus

An instance of a status enum used to indicate the status of a saving and loading event.

- ErrorNumbers errorCode

An instance of a status enum used to pass any errors to the view. Initialized with the value NoError.

- Cal storedValue

A Cal object used to store information from the current state of the users calendar. This value is initialized as null, and populated during the load/save state.

- isFromSaving

A boolean value used to indicate if the instance of the FileIoStatus is loading a files contents (false) or saving to a file (true).

- ReentrantLock lock

An instance of the Java ReentrantLock. This object is used to allow thread safe access to reading and writing from a file.

2.2.6. GUI.java

This file contains all the information for the Graphical User Interface. The class inherits from Javas JPanel interface and implements JavasAction Listener. This class serves as the main interaction between the user and YACC.

This file contains two classes, one being the main GUI and the second being the GUI frame. The main GUI is the window in which YACC will be displayed to the user. The frame is how the program displays the information.

More information regarding the appearance and interaction is located within the SRS and SDS.

2.2.6.1. Constants

- long serialVersionUID
- Dimension windowSize
- View view
- int titleSize
- int bodySize
- int daysSize
- int eventSize
- int editSize
- String fontFamily
- Font titleFont
- Font bodyFont
- Font daysFont
- Font eventFont
- Font editFont
- Font editItalFont
- Font eventItalFont
- int boxThickness
- Stroke boxStroke
- Color backgroundColor
- String calendarName

This class makes use of constants for font sizes, font families, display width, display thickness, colors , and display text. The use of constants allows for human readable code, and the ability to change an instance of a variable easily.

2.2.6.2. Constructors

- GUI(Main main2, UserInput input, View View)

This constructor first creates the GUI according to the Java documentation, calling instances of the super class JPanel to create the GUI. The constructor then enables ActionListeners to react to events such as a mouse click on the JPanel or a keyboard press.

- GUIFrame(GUI gui)

This constructor handles biding the frame and action events to the window. This constructor also handles ‘drawing’ the GUI on the screen.

2.2.6.3. Derived Functions

- void paintComponent(Graphics g)

This function handles the ‘painting’ of the GUI on the screen. It makes uses of Java internals to draw on the screen. For more information regarding Java internals, specifically with AWT and Swing, visit: https://www.oracle.com/technetwork/java/painting-140037.html

- void actionPerformed(ActionEvent arg0)

This function handles the frame count within the GUI. The frame count is increased during runtime and the GUI repainted. This allows the user to GUI to change and interact with the user.

2.2.6.4. Public Functions

- Long getFrameCount()

This function returns the current frame count.

- void setFrameCount(long frameCount)

This function sets the current frame count.

2.2.7. LoadEvents.java

This file is responsible for loading the calendar information from a file, parsing it, then passing the information to the view. This action is performed asynchronously; if the file is read and parsed correctly a FileIOStatus object is returned with the storedValue field pointing to a Cal object containing the files information. If any error occurs a FileIOStatus is returned with an errorCode set and the storedValue set to null.

2.2.7.1. Public Functions

- FileIOStatus loadCalendar(String fileName)

This function creates a new FileIOStatus and a new thread for the loading action to be performed on. The caller passes a file name which the created thread attempts to open and parse. After the thread is started the FileIOStatus is returned to the caller.

2.2.7.2. Private Functions

- void threadedLoad(String fileName, FileIOStatus status)

The function the thread initially calls upon creation. The caller passes in a file name which is parsed as a list of strings. The string list is then united into one string if the file was read successfully, or an exception is thrown and the caller is notified.
A Cal object is created if the files contents are read without error, and passed to a helper function convertFromString. If the convert from string returns successfully the thread takes control of the lock within the callers FileIOStatus lock. The callers FileIOStatus.storedValue member is then set to the Cal object returned from the helper function, and the Status is set to success. The thread then releases the lock.
If an error occurs during the process the thread takes control of the lock, sets the status field to failed, and sets the proper error code depending on the error encountered.

Error Codes Returned

- ➔ NoError: Indicates no error.
- ➔ LoadError: Indicates the thread was unable to load the files contents.
- ➔ FileNotFound: Indicates the filename given was not found.

- Cal convertFromString(String contents)

A helper function used to parse the contents of a string. This function makes use of the org.json library to parse the JSON format. The function first creates a JSONObject using the string passed from the caller. The function first checks the version field to parse the file with respect to the version of YACC. Using the version field allows for backwards compatibility between different version of YACC. The version string is then run through a switch statement and a parsing function is called depending on which version the file is in, or an error is thrown if the version is not found.

- void handle1_0(JSONObject obj, Cal c)

This function handles version 1.0 of the file format. The function begins by first checking the file contains the required fields. If the required fields are not found an exception is thrown the function exits. The function then parses the JSON object getting the fields held within the JSONArray.

Data Fields Possible

- ➔ Year: A required field holding the year of an event.
- ➔ Month: A required field holding the month of an event.
- ➔ Day: A required field holding the day of an event.
- ➔ Start_hour : A required field holding the start hour of an event.
- ➔ Start_minute: A required field holding the start minute of an event.
- ➔ End_hour: A required field holding the end hour of an event.
- ➔ End_minute: A required field hold the end minute of an event.

- ➔ Title: A required field holding a title of an event.
- ➔ Note: An optional field holding notes of an event.
- ➔ Is_all_day : A required field indicating if the event is all day

2.2.8. SaveEvents.java

This file is responsible for saving the calendar information from a file, saving the information in the format specified by the current version of YACC. This action is performed asynchronously; if the information is read, parsed correctly, and saved a FileIOStatus object is returned with the status field set to success indicating the file was updated.

2.2.8.1. Public Functions

- FileIOStatus saveCalendar(Cal calendar, String fileName)

This function creates a new FileIOStatus and a new thread for saving. The caller passes a file name and a Cal object. The Cal object contains the information to be saved to the file and the file name is the name which the thread will attempt to save the information to. After the thread is started the FileIOStatus is returned to the caller.

2.2.8.2. Private Functions

- void threadedSave(Cal calendar, String fileName, FileIOStatus status)

The function the thread initially calls upon creation. The caller passes in a Cal object which contains the information to be saved, the name of the file to be saved, and a FileIOStatus to be returned by the thread upon success or failure with the proper status indicated.
A Cal object is passed to the function containing all the information to be saved to the file. The function first checks to see if the file passed by the caller exists. If the file exists it is deleted, otherwise the file is created.
If an error occurs during the process the thread takes control of the lock, sets the status field to failed, and sets the proper error code depending on the error encountered.

Error Codes Returned

- ➔ NoError: Indicates no error during save.
- ➔ SaveError: Indicates an error occurred during the parsing and saving of the Cal object.

- JSONObject convertToJSON(Cal calendar)

A helper function called by threadedSave to parse the passed Cal object. The function makes use of the org.json library and returns a JSONObject to the caller containing the information from the Cal object parsed in JSON format.

Data Fields Possible

- ➔ Year: A required field holding the year of an event.
- ➔ Month: A required field holding the month of an event.
- ➔ Day: A required field holding the day of an event.

- ➔ Start_hour : A required field holding the start hour of an event.
- ➔ Start_minute: A required field holding the start minute of an event.
- ➔ End_hour: A required field holding the end hour of an event.
- ➔ End_minute: A required field hold the end minute of an event.
- ➔ Title: A required field holding a title of an event.
- ➔ Note: An optional field holding notes of an event.
- ➔ Is_all_day : A required field indicating if the event is all day
- ➔ Version: A required field taken from Main.VERSION_NUMBER.

2.2.9. Status.java

An enum of possible outcomes of a FileIOStatus object. This allows the FileIOStatus object to communicate with other components indicating an error if one occurs or success.

Possible Values

- ➔ Waiting: Value indicating the FileIOStatus is waiting to acquire a lock.
- ➔ Success: Value indicating the FileIOStatus has completed its job.
- ➔ Failed: Value indicating an error occurred.

2.2.10. UserInput.java

This file contains all logic related to interaction between the user and the view. The class is derived from Java classes MouseListener, WindowListener and KeyListener. The uses event based programming practices to interact with the user.

2.2.10.1. Public Functions

- GUI getGui()

A function to return the GUI object currently the event listeners are bound to.

- void setGui(GUI gui)

A function to set the GUI object the event listener will be bound to.

- void mousePressed(MouseEvent arg0)

An implementation of Java's mousePressed event listener. The function first gets the coordinates of a mouse press event, then determines the current view and where the user clicked using functions from CalMathAbs.

If the users click is determined to have been from a 'previous' or 'next' month, the view is updated to the user, otherwise the day or event view is displayed to the user depending on the location of the users mouse and objects the user interacted with.

- void windowClosing(WindowEvent arg0)

An implementation of Java's WindowListener. The function is called when the users selects the 'X' button in the window. The function first stops the frame timer to pause all 'painting' of the GUI to the user. The function then displays a message box to indicate to indicate a saving action is being performed.

A new thread is then created. This thread calls the saveCalendar function to save the current state of the calendar. The thread then waits for the saveCalendar to save the current state of the calendar. If any errors occur the user is informed through a message box, otherwise the information is saved and the GUI is terminated.

- void keyReleased(KeyEvent arg0)

An implementation of Java's KeyListener. The function is called during a key release event.

Key Release Values:

➔ Esc: Closes the event/day view.
➔ Left Arrow: Moves the event one day to the left.
➔ Right Arrow: Moves the event one day to the right.
➔ Home: Switches the view to the current month.
➔ PageUp: Moves the current screen one month backward.
➔ PageDown: Moves the current screen one month forward.

- void keyTyped(KeyEvent arg0)

An implementation of Java's KeyListener. The function is called during a key press event. The function is used to filter invalid characters from user input.
The function runs a switch statement to find the current view and field the user has selected, then depending on the view filters keystrokes accordingly.

Filtering Example

If the user has selected a number entry field and attempts to enter non numeric characters, the users keystroke will be ignored.

2.2.11. View.java

This class is the main module to display information to the user.

2.2.11.1. Constants

- String[] months
- String[] days
- FontMetrics titleMetric
- FontMetrics bodyMetric
- FontMetrics eventMetric

- FontMetrics editMetric
- FontMetrics daysMetric
- Color currentDayColor
- Color currentDayColor2
- Color titleColor
- Color weeksColor
- Color otherMonthColor
- Color currentMonthColor
- Color eventBackgroundColor
- Color eventPrevMonthBackgroundColor
- Color eventCurrentDayBackgroundColor
- Color eventEditHintColor
- Color eventEditCheckBoxColor
- Color eventEditDisabledColor
- Color eventEditHighlightColor
- Image leftImage
- Image leftImage1
- Image upImage
- Image xImage
- Point LeftArrowMonthPosition
- Point RightArrowMonthPosition

2.2.11.2. Public Members

- void Display(Graphics2D g, Calendar c)

This function is the main function used to render drawings on the screen. The function first sets up fonts, strokes, and colors from the Graphics2D object passed by the caller. The function then uses the string array constant to populate the names of the months. The function then renders the days of week based off the information within the days string array, then draws the left and right navigation arrows. The function then loads and displays all saved events the user has stored within the saved file.

- Point getPopupBox()

This function will, based upon the currently selected day on the calendar, return the top left corner position for the popup box. The function calculates how close an event display is to any edge, and adjusts the display accordingly.

- int getPopupBoxHeight()

This function calculates the height of the popup box. If the popup box is for displaying events occuring in a day and it is less than 4, decrease the size of the box.

- void scrollDown()

This function allows the user to scroll through available events for the day.

- void scrollUp()

This function allows the user to scroll through available events for the day.

- int getObjectClicked(Point mousePos)

Attempts to determine where on the screen did the user click, and returns an int representing what was clicked.

Codes Returned

- ➔ -1: Not clicked in popup window
- ➔ -2: Clicked the 'x' icon
- ➔ 0: User clicked in the popup window, but on nothing in particular
- ➔ 2: User clicked the previous day arrow
- ➔ 3: User clicked the next day arrow
- ➔ 1: User clicked the new event button
- ➔ 6 + i: (i+1)th event clicked in day view
- ➔ 4: User clicked on the 'scroll up' arrow
- ➔ 5: User clicked on the 'scroll down' arrow
- ➔ -3: User clicked on the title field
- ➔ -14: User clicked on the hours part of the start time field
- ➔ -4: User clicked on the minutes part of the start time field
- ➔ -15: User clicked on the hours part of the end time field
- ➔ -5: User clicked on the minutes part of the end time field
- ➔ -6: User clicked on the checkbox for all day event
- ➔ -7: User clicked on the notes field
- ➔ -8: User clicked the save event button
- ➔ -9: User clicked the cancel/delete event

- void moveToLeft()

This function will attempt to move the popup box to the previous day, changing the month if required.

- void moveToRight()

This function will attempt to move the popup box to the next day, changing the month if required.

- String displayTime(Calendar sc, Calendar ec)

A local function that will determine how the times should be rendered to the user. If the event is all day, determined by the seconds field of the start calendar, “All day - “ is returned, otherwise a string in the format HH:MM is returned.

- void renderEvent(Graphics2D g, int topLeftX, int topLeftY)

This function handles drawing the GUI elements for the event editor, including: start time, end time, notes, title and an all day checkmark.

- boolean isDisplayingEvents()

A function to determine if the popup window is visible (because if isDisplayingEvent is true, then it must be the case that this function returns true.

- boolean isDisplayingEvent()

A function to determine if the popup window is displaying the event editor.

- boolean isShowingThisEvent(int gridX, int gridY)

A helper function to determine if the popup box is displayed on this cell on the screen.

- void createEvent()

This functions displays the user interface for the users to enter information regarding events.

- boolean attemptSaving()

This function parses the information on the event and attempts to save the information to a file. This function returns true if the save was successful, false if not.

- void hideEvent()

A simple function that will set necessary variables to default values to allow for rendering logic to not display the event editor.

<u>Default Values</u>

- ➔ isEventShown = false;
- ➔ selectedField = -1;
- ➔ eventTitle = "";
- ➔ eventEvent = null;
- ➔ this.eventIndex = -1;
- ➔ eventAllDay = false;
- ➔ eventTime1 = "";
- ➔ eventTime2 = "";

- ➔ this.delta = 0;
- ➔ this.overField = -1;
- ➔ this.prevContents = "";
- ➔ this.redBox = false;
- ➔ scrollBar = 0;
- ➔ eventNotes = "";
- ➔ selectedField = -1;

- void showEvents(int gridX, int gridY)

A simple function that takes a col and a row and will display the event popup on that day. Setting necessary values as needed.

- void hideEvents()

A simple function that will set necessary variables to default values to allow for rendering logic to not display the day view

- int getTitleBox(Point mousePos)

A function for determining if the left or right arrow at the top of the screen was clicked. Returns -1 for left month clicked, +1 for right month clicked, and 0 otherwise.

- String chopTitle(String title, int length)

A helper function that will take a string and chop it ,adding an ellipse if it is too long, returns the new string with ellipse added.

- centerString(String s, Rectangle rect, FontMetrics fm)

A helper function that will take a string and will center it vertically and horizontally in the rectangle provided, returns the center point to display the text.

- Point leftCenterString(String s, Rectangle rect, FontMetrics fm)

A helper function that will take a string and will center it vertically in the rectangle provided, still left justified, returns the point to display the text.