CMSC204 **-** Assignment 1

Appointment Book

Concepts tested by this program:

ArrayList

Inheritance

Polymorphism

Tooltips

Mnemonic

Layout Managers

Read Files

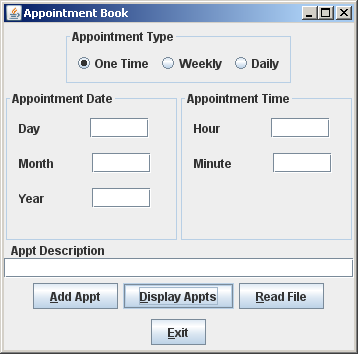
Javadoc

JUnit Tests

New concepts tested by this program

GregorianCalendar

Exceptions



Create an application that allows the user to:

1. Add appointments that occur: daily, one-time, or weekly
   1. One-time appointments occur only on the appointment date
   2. Weekly appointments occur on the appointment date and every 7 days after that.
   3. Daily appointments occur on the appointment date and every day after that.
2. Display appointments that occur on the appointment date (time is not required, and ignored if entered)
3. Read a file. The file will be in the following format:

O|W|D(space)MM(space)DD(space)YYYY(space)HH(space)MM(space)Description

Sample file:

O 01 05 2015 10 00 Dr. Appt

W 01 04 2015 09 30 Zumba Class

D 01 01 2015 08 00 Walk the Dog

O 01 21 2015 20 30 Eric's Bday Party

D 01 15 2015 07 30 Treadmill

D 02 01 2015 18 30 Call Mom

W 02 02 2015 16 00 Band Practice

W 01 27 2015 15 30 Piano Lessons

O 02 14 2015 19 00 Valentine Dinner

1. Exit the application

## Data Element

Appointment Class (abstract)

* Attributes – Type, Gregorian Calendar date, Description
* Methods – occurs\_on

OneTime, Weekly and Daily classes that inherit from Appointment class

* Each class overrides the occurs\_on() method. The occurs\_on() method determines if the appointment occurs on the particular date passed in. Add additional methods as needed.

## Data Structure

Arraylist of Appointment objects

**Data Manager**

Appointments Class.

* Provide a method to add an appointment (addAppt), and a method to display appointments (displayAppt)
* Manage the Data Structure
* Implements AppointmentsInterface

**GUI**

* You may use the GUI Template as the framework for your GUI.
* When the user selects the Add Appointment button, the information is sent to the Appointments class through the **addAppt** method. The fields are then cleared. Only the day, month and year is needed for the Display Appointments action.
* When the user selects the Read File button, the File Chooser is used for user to select file. The file is read and the information is sent to the Appointments class through the **addAppt** method.
* When the user selects the Display Appts button, the appointments returned by the **displayAppt** method of Appointments will be displayed.
* Create a tool tip and a mnemonic for each of the buttons.
* The file will be in the following format:

O|W|D(space)MM(space)DD(space)YYYY(space)HH(space)MM(space)Description

Sample file:

O 01 05 2015 10 00 Dr. Appt

W 01 04 2015 09 30 Zumba Class

D 01 01 2015 08 00 Walk the Dog

O 01 21 2015 20 30 Eric's Bday Party

D 01 15 2015 07 30 Treadmill

D 02 01 2015 18 30 Call Mom

W 02 02 2015 16 00 Band Practice

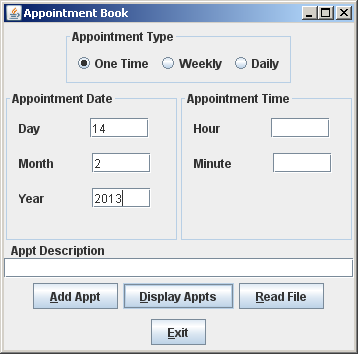
W 01 27 2015 15 30 Piano Lessons

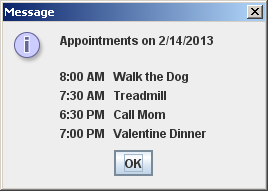
O 02 14 2015 19 00 Valentine Dinner

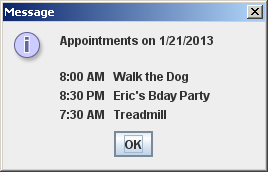
**Exceptions**

* Create exceptions to check for valid numbers, i.e. month should be between 1 and 12. Use a JOptionPane.showMessageDialog to let the user know there was an invalid entry.

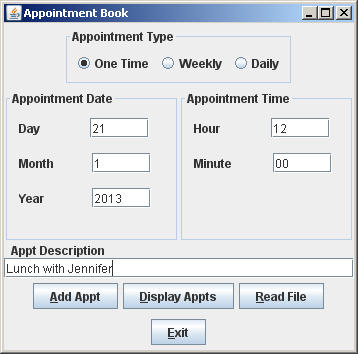
Displaying Appointments based on reading the previous file



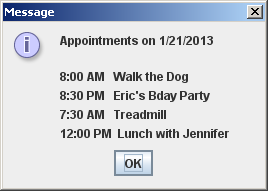




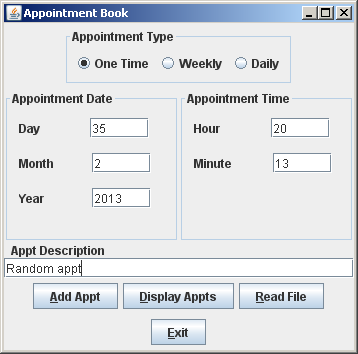
Adding an appointment:

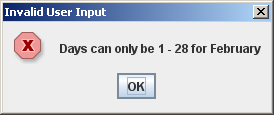


Displaying appointments after adding the previous appointment.



Exceptions: - day is not legal



**Information on the GregorianCalendar class (check the API for additional information)**

* Year must be a four-digit integer.
* Month must be an integer from 0-11. January is 0, April is 3
* Day must be an integer from 1-31.
* Hour must be an integer from 0-23, with 0 being 12am (midnight) and 23 being 11pm.
* Minute and second must be integers from 0-59.
* Any time values that aren’t set will default to 0.

Creates the date/time December 31, 2005, 7:30 am

GregorianCalendar newDate = GregorianCalendar(2005, 11, 31, 7, 30);

Adds 5 days to the newDate

newDate.add(Calendar.DAY\_OF\_MONTH, 5);

Returns the month of the newDate

int month = newDate.get(Calendar.MONTH);

Returns a Date Object from a GregorianCalendar Object

Date newDay = newDate.getTime();

Checks if a GregorianCalendar object is before another GregorianCalendar object

If (thisDate.before(thatDate))

**Information on the DateFormat class (check the API for additional information)**

Field for the DateFormat class

|  |  |  |
| --- | --- | --- |
| Style | Date example | Time example |
| **SHORT** | 12/31/05 | 12:00 AM. |
| **MEDIUM** | Dec 31, 2005 | 7:30:00 PM |
| **LONG** | December 31, 2005 | 7:30:00 AM PST |
| **FULL** | Saturday, December 31, 2005 | 7:30:00 AM PST |

Format a GregorianCalendar object with DateFormat:

GregorianCalendar newDate = new GregorianCalendar(2015, 10, 12, 7, 30);

Date now = newDate.getTime();

DateFormat defaultDate = DateFormat.getTimeInstance(DateFormat.SHORT);

String newString = defaultDate.format(now);

**CMSC 204 Assignment 1 Grade Sheet – Spring 2015**

## DOCUMENTATION 20 points

## Javadoc for all student created classes: 8 pts \_\_\_\_\_

Test Cases 8 pts \_\_\_\_\_

JUnit Test Class

Add additional tests to the AppointmentsTest class

UML Diagram 4 pts \_\_\_\_\_

## PROGRAMMING 80 points

Internal class documentation (within source code) 10 pts \_\_\_\_\_

Class description using Javadoc

Author’s Name, Class, Class Time, @author

Methods commented using Javadoc, @param, @return

Program user interface 4 pts \_\_\_\_\_

Clear to user how data is to be entered

Output is easy to understand

Accuracy – Received correct output

Public tests (JUnit tests provided to you with your additions 10 pts \_\_\_\_\_\_

Private tests 10 pts \_\_\_\_\_

Data Element

Appointment class 6 pts \_\_\_\_\_

* + is abstract
  + contains abstract method occurs\_on()
  + contains GregorianCalendar object

OneTime, Weekly and Daily classes 6 pts \_\_\_\_\_

* + inherit from Appointment class
  + Unique implementation of occurs\_on

Data Structure – ArrayList<Appointment> 2 pts \_\_\_\_\_

Data Manager

Appointments class 10 pts \_\_\_\_\_

* contains an ArrayList of Appointment objects
* implements AppointmentsInterface
* method for adding appointments – **addAppt**
* method for finding appointments for specific day

**displayAppt**

GUI12 pts \_\_\_\_\_

* Uses an Appointments object
* Buttons have tooltips and mnemonics
* Fields are cleared when Add Appointment button

is selected

* Displays error message to user when exception is raised
* Create titled borders
* Reads from a file and adds to the

Appointments class

Exceptions 10 pts \_\_\_\_\_

* + Checks for invalid information

(month, day, hour, min)

Total 100 pts \_\_\_\_\_