**CalendarsApp**

Attributes:

Added: User currentUser, Scanner scan

Removed: settings: ArrayList<Setting>

No function existed to access a user so I added currentUser to denote who was being accessed and to keep track of it. I added scan as a public attribute so that I could use a single Scanner across the entire application rather than multiple as a means of input from the user. No input settings were in the original design besides the GUI but since that was out of scope for this implementation, this was the next best thing. Removed settings because no more settings were required. Additionally, “//Note: All enums implicitly extend java.lang.Enum. Because a class can only extend one parent (see Declaring Classes), the Java language does not support multiple inheritance of state (see Multiple Inheritance of State, Implementation, and Type), and therefore an enum cannot extend anything else.”. As such, ThemeType could not extend abstract class Setting so, instead, I scrapped abstract class setting. ThemeType is kept as an enum but included in the User class because the requirement is that the user has the choice. As it is text based, the only change is to a parameter. More in User class.

Methods:

Added: chooseUser(), displayUsers(), showOptions(), getCurrentUserId(), getCurrentUser()

Removed: addSetting(), updateSetting()

The added functions were to help aid the console-controlled interface that I implemented. The removed are because the settings attribute was removed.

**User**

Attributes:

Added: Theme theme, Calendar currentCalendar

Removed: None

In this class, the enum Theme was made following from the reasoning above. Private member theme holds this value. The other Added attribute, currentCalendar is to help with the interface I implemented.

Methods:

Rather than list out all the changes since there are many, I will just explain this section briefly, hitting the key points. Most of the added functions are helper functions, getters and setters, and, in general, to aid my interface. The key removed methods are those dealing with filtering and searching which I did not implement in this edition of Calendar. The last big change is that addEvent is not included in my User class. Due to my implementation of current objects, it makes more sense to move this method down the line into Calendar. However, the addEvent there mirrors the addEvent that the original UML has in their Calendar class. Essentially, addEvent was not necessary here so I removed it.

**Calendar**

Attributes:

Added: Event currentEvent

Removed: int daysPerWeek, Boolean private, ArrayList<CalendarItem> items, Year year

Added attribute is following my implementation of interface style. daysPerWeek was removed because it was not necessary under the requirements picked. Private likewise removed. Items and years are in the similar boat but their classes were removed altogether as well. Although the original design uses CalendarItems for Event to inherit from, there was no need for it because the requirement to add other media and attach items to a Calendar was not implemented. Hence, I shortened the implementation by removing the abstract class CalendarItem and gave its attributes to the Event class.

Methods:

As with the previous section, this one will be explained rather than fully listed. The majority of additions are, once again, interface helper functions. The biggest removal is again the filter and search methods because it was not implemented and hence unnecessary. Some minor changes also occurred- for example, updateTimes() became adjustTime(newTimezone:ZoneId). These changes were necessary to pass information down the line of classes. Although shareWith does not have real functionality in my interface, I did implement it within my program. This is because I initially planned to add a requirement dealing with it but then decided against that requirement. However, since it is still functional, I decided to keep it within the program in case I have time to continue building upon it.

**Event**

Attributes:

Added: String name, ArrayList<String> sharedWith

Removed: None

Since I decided not to include abstract class CalendarItem, its attribute name was transferred over to Event. The sharedWith attribute was added to fulfill one of the 8 requirements, one stating events are shareable.

Methods:

Added: getEventId(), setEventId(newId: String), adjustTIme(newTimezone: ZoneId).

Removed: None

Getters and Setters were added to help with my interface. I had to had adjustTime() to this class because I found interacting with the class to flow better at this level. The original display() method was altered to better fit my implementation with displayEvent().

**OverArching Changes**

With the thorough documentation given by the original UML design, I was able to build a Calendar App of sorts. However, because the assignment only requires 8/16 or ½ of the total requirements, that left a lot of extra classes and functions in the original design. This includes the entire day setup with classes Year, Month, Week, and Day being excluded along with many other classes and design pieces. That design went so far as to cover the futures changes. Since I was not building such an extensive calendar, many classes were trimmed for me to implement this project within a reasonable time. If I have more time in the future, several changes will need to be made to my design in order for it to reach the level of the design set forth by the original creator. However, for the time being, those changes will be left undone.