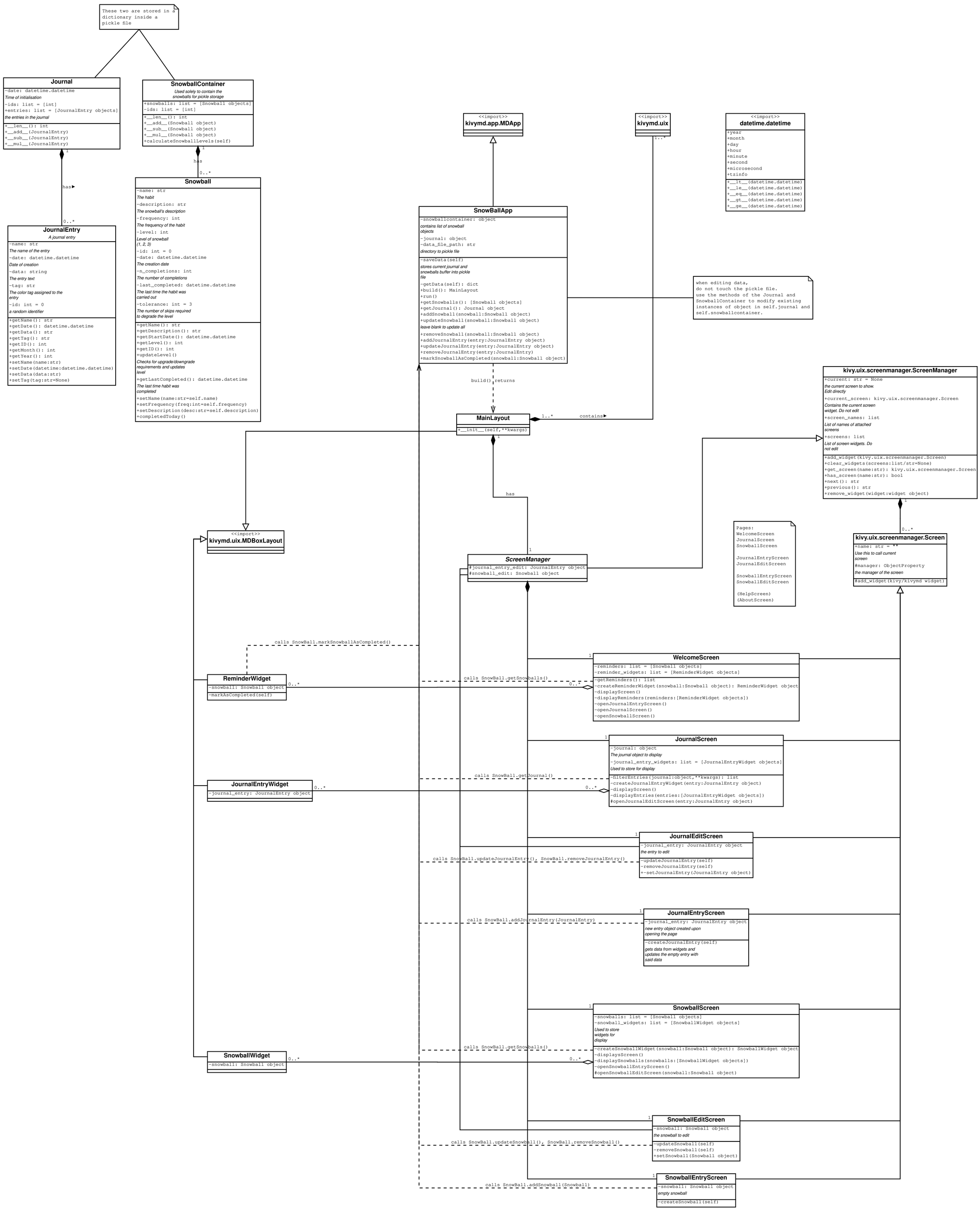
These two are stored in a dictionary inside a pickle file
Journal
-date: datetime.datetime
Time of initialisation
-ids: list = [int]
+entries: list = [JournalEntry objects]
the entries in the journal
+__len__(): int
+__add__(JournalEntry)
+__sub__(JournalEntry)
+__mul__(JournalEntry)
SnowballContainer
Used solely to contain the snowballs for pickle storage
+snowballs: list = [Snowball objects]
-ids: list = [int]
+__len__(): int
+__add__(Snowball object)
+__sub__(Snowball object)
+__mul__(Snowball object)
+calculateSnowballLevels(self)
has
Snowball
-name: str
The habit
-description: str
The snowball's description
-frequency: int
The frequency of the habit
-level: int
Level of snowball (1, 2, 3)
-id: int = 0
-date: datetime.datetime
The creation date
-n_completions: int
The number of completions
-last_completed: datetime.datetime
The last time the habit was carried out
-tolerance: int = 3
The number of skips required to degrade the level
+getName(): str
+getDescription(): str
+getStartDate(): datetime.datetime
+getLevel(): int
+getID(): int
+updateLevel()
Checks for upgrade/downgrade requirements and updates level
+getLastCompleted(): datetime.datetime
The last time habit was completed
+setName(name:str=self.name)
+setFrequency(freq:int=self.frequency)
+setDescription(desc:str=self.description)
+completedToday()
has►
JournalEntry
A journal entry
-name: str
The name of the entry
-date: datetime.datetime
Date of creation
-data: string
The entry text
-tag: str
The color tag assigned to the entry
-id: int = 0
a random identifier
+getName(): str
+getDate(): datetime.datetime
+getData(): str
+getTag(): str
+getID(): int
+getMonth(): int
+getYear(): int
+setName(name:str)
+setDate(datetime:datetime.datetime)
+setData(data:str)
+setTag(tag:str=None)
<<import>>
kivymd.app.MDApp
<<import>>
kivymd.uix
<<import>>
datetime.datetime
+year
+month
+day
+hour
+minute
+second
+microsecond
+tzinfo
+__lt__(datetime.datetime)
+__le__(datetime.datetime)
+__eq__(datetime.datetime)
+__gt__(datetime.datetime)
+__ge__(datetime.datetime)
SnowBallApp
-snowballcontainer: object
contains list of snowball objects
-journal: object
-data_file_path: str
directory to pickle file
-saveData(self)
stores current journal and snowballs buffer into pickle file
-getData(self): dict
+build(): MainLayout
+run()
+getSnowballs(): [Snowball objects]
+getJournal(): Journal object
+addSnowball(snowball:Snowball object)
+updateSnowball(snowball:Snowball object)
leave blank to update all
+removeSnowball(snowball:Snowball object)
+addJournalEntry(entry:JournalEntry object)
+updateJournalEntry(entry:JournalEntry object)
+removeJournalEntry(entry:JournalEntry)
+markSnowballAsCompleted(snowball:Snowball object)
when editing data, do not touch the pickle file. use the methods of the Journal and SnowballContainer to modify existing instances of object in self.journal and self.snowballcontainer.
build() returns
MainLayout
+__init__(self,**kwargs)
contains►
kivy.uix.screenmanager.ScreenManager
+current: str = None
the current screen to show. Edit directly
+current_screen: kivy.uix.screenmanager.Screen
Contains the current screen widget. Do not edit
+screen_names: list
List of names of attached screens
+screens: list
List of screen widgets. Do not edit
+add_widget(kivy.uix.screenmanager.Screen)
+clear_widgets(screens:list/str=None)
+get_screen(name:str): kivy.uix.screenmanager.Screen
+has_screen(name:str): bool
+next(): str
+previous(): str
+remove_widget(widget:widget object)
<<import>>
kivymd.uix.MDBoxLayout
has
ScreenManager
#journal_entry_edit: JournalEntry object
#snowball_edit: Snowball object
Pages:
WelcomeScreen
JournalScreen
SnowballScreen
JournalEntryScreen
JournalEditScreen
SnowballEntryScreen
SnowballEditScreen
(HelpScreen)
(AboutScreen)
kivy.uix.screenmanager.Screen
+name: str = ""
Use this to call current screen
#manager: ObjectProperty
the manager of the screen
#add_widget(kivy/kivymd widget)
calls SnowBall.markSnowballAsCompleted()
ReminderWidget
-snowball: Snowball object
-markAsCompleted(self)
calls SnowBall.getSnowballs()
WelcomeScreen
-reminders: list = [Snowball objects]
-reminder_widgets: list = [ReminderWidget objects]
-getReminders(): list
-createReminderWidget(snowball:Snowball object): ReminderWidget object
-displayScreen()
-displayReminders(reminders:[ReminderWidget objects])
-openJournalEntryScreen()
-openJournalScreen()
-openSnowballScreen()
JournalScreen
-journal: object
The journal object to display
-journal_entry_widgets: list = [JournalEntryWidget objects]
Used to store for display
-filterEntries(journal:object,**kwargs): list
-createJournalEntryWidget(entry:JournalEntry object)
-displayScreen()
-displayEntries(entries:[JournalEntryWidget objects])
#openJournalEditScreen(entry:JournalEntry object)
calls SnowBall.getJournal()
JournalEntryWidget
-journal_entry: JournalEntry object
JournalEditScreen
-journal_entry: JournalEntry object
the entry to edit
-updateJournalEntry(self)
-removeJournalEntry(self)
+-setJournalEntry(JournalEntry object)
calls SnowBall.updateJournalEntry(), SnowBall.removeJournalEntry()
calls SnowBall.addJournalEntry(JournalEntry)
JournalEntryScreen
-journal_entry: JournalEntry object
new entry object created upon opening the page
-createJournalEntry(self)
gets data from widgets and updates the empty entry with said data
SnowballScreen
-snowballs: list = [Snowball objects]
-snowball_widgets: list = [SnowballWidget objects]
Used to store widgets for display
-createSnowballWidget(snowball:Snowball object): SnowballWidget object
-displaysScreen()
-displaySnowballs(snowballs:[SnowballWidget objects])
-openSnowballEntryScreen()
#openSnowballEditScreen(snowball:Snowball object)
calls SnowBall.getSnowballs()
SnowballWidget
-snowball: Snowball object
SnowballEditScreen
-snowball: Snowball object
the snowball to edit
-updateSnowball(self)
-removeSnowball(self)
+setSnowball(Snowball object)
calls SnowBall.updateSnowball(), SnowBall.removeSnowball()
SnowballEntryScreen
-snowball: Snowball object
empty snowball
-createSnowball(self)
calls SnowBall.addSnowball(Snowball)
1
0..*
1..*