Bilkent University

Department of Computer Engineering

Senior Design Project

Project short-name: Augma

Final Report

Ahmet Burak Şahin, Can Bayraktar, Çağdaş Han Yılmaz, Fırat Sivrikaya, Utku Oymak

Supervisor: Ercüment Çiçek
Jury Members: Bülent Özgüç and İbrahim Körpeoğlu

Final Report

May 3, 2018

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS492.

Table of Contents

1.0 Introduction

New tools and ideas emerge in the profession of computer science with advancements in technology. Some of these tools or ideas start to grow and become more popular over time. Augmented Reality [1] [2] [3] is one of these emerging technologies which makes it possible to look at the world from a new perspective. Even though it is a relatively new technology, it holds many great possibilities like helping doctors in a surgery, a visual navigation that doesn't need for the driver to look away from the road, just to name a few. Unfortunately, being new also comes with many drawbacks like most users not willing to change what they are using or lack of tool kits and native platforms. Most AR applications today are designed to work on mobile devices [4] [5] like tablets or phones but true potential of this technology can only be unlocked with devices like Google Glass. We expect to see much more advanced AR applications [6] [7] [8] as native AR platforms [9] like Google Glass starts to become affordable by the public and this can only happen if developers keep creating new and interesting Augmented Reality applications, further increasing the value of the technology and attracting more investors to the area.

Social media applications are the most used day-to-day mobile applications in the market. This is why we saw a need and a potential for a new kind of social media application in an augmented space. Augma creates a world where people can interact with each other in an exciting and creative way. Users will be able to post location-based notes on anywhere in the world. Other people who are near one of these notes will be able see it using Augma like a scope, looking into another dimension from their phone's camera. With Augma, we aim to achieve a deeper level of empathy between the users than any other social media application today by putting the readers into the exact same environment where the post was written.

2.0 System Overview

We are living in an era where conveying information is everything. The way you convey that information is just as important as the information itself. We felt like writing 120 characters or even posting a picture with a lengthy paragraph under it isn't enough. We wanted to add another dimension to our conversations. This is where Augma was born. Augma adds a new layer to our reality and lets our posts come to life.

Augma is a mobile-app which lets user leave location based notes that come to life with the power of Augmented Reality technology. Users can leave pretty much anything they want ranging from a plain text to pictures. After users have left their note to a location, other people who look through their phone cameras will be able to see their creation. These notes will be time-limited and people who see these notes will be able to rate these notes by upvoting or downvoting.

Users will have the choice to make their notes visible to all other people or just certain groups by using Augma's Circle system. Users can create their own small scale circles to leave private messages to their friends and family. Users will have the option to leave notes only they can see for personal use such as a reminder for an upcoming project. With the usage of the Circle system, users will be able to personalize their window to this new layer of reality. Augma will provide the option to see the notes that are only from the certain circles users have specified to filter the notes in the augmented world.

Augma will also feature a heat map in which users can see where the places with lots of notes are. When users get to a "Hot Zone" like this, they will receive a notification on their phones prompting them to have a look around even when the app is running in the background. If they have the map open and got close enough to a note, they will be able to see an indication and a small preview of the note. With this function they will be able to determine if there are notes they want look at without actually having to open their camera. If users have low battery, their phones aren't powerful enough or maybe they don't like having to turn around with their phone to actually see the Augma's world. They can click on these previews to see the notes like a picture taken from the perspective of their creators.

Another one of Augma's big features will be the Augma ads. Companies will be able to leave creative ads in the Augma's world for a much cheaper price than in real life. Think about seeing a giant yellow M on the moon when you look through your camera at night if you are near a McDonald's. This will allow for much more interesting ads and this way the big companies won't have to pollute the scenery with their giant ads.

3.0 Algorithmic Design of Augma

3.1 Note View Filtering

The users can only see the notes that are in the circles that they are subscribed to. Since the user can subscribe to many different circles, the note viewing can be problematic as there might be lots of notes in that area. The users can filter which circles' notes that they want to see.

3.2 Bitmap

All of the photos both taken from camera and selected from gallery are converted into bitmaps for ease of storage and database transfers. These bitmaps are converted to base64 strings later and then sent to the S3 database where they are kept as images.

3.3 OpenGL calculations

A note's size and its motions are calculated according to the distances from the user's location. As the users get closer to the note the size of the note is changing dynamically and the motion of the note gets more intense.

3.4 Light the Beacon calculations

After a note has received "super" likes, than Augma sends the nearby users a notification stating that there is a note nearby to them which is getting attention. According to the number of the "super" likes the radius of the notification grows dynamically.

4.0 Final Architecture and Design

In this section, we are going to explain the subsystem decomposition, subsystem services, client and server architecture.

4.1 Subsystem Decomposition

Augma follows a Client-Server architectural style to effectively respond/process concurrent user requests. On the client side, our phone application handles most of the processing which is mainly AR. Rendering AR elements, data transmission and GPS localization are the primary operations client-side will be dealing with. On the server side, the system will be managing the database which should be efficient and effective as much as possible. Main goal is to achieve the highest performance while sustaining the lowest response time for each user request. Our system falls into 3-Tier system architecture which are: Presentation Tier, Logic Tier and Data Tier. The Presentation Tier, which is the topmost tier, contains the visual components of the system and it can be found mostly on our client-side application. This tier is responsible for managing the interactions with the user such as displaying the environment with AR elements via camera, showing the map of near vicinity, etc. The Logic Tier contains the fundamental operations behind the Augma. It mostly resides on the client-side and includes operations such as AR rendering, GPS localization. The Data Tier is responsible of database management and primarily resides on the server side.

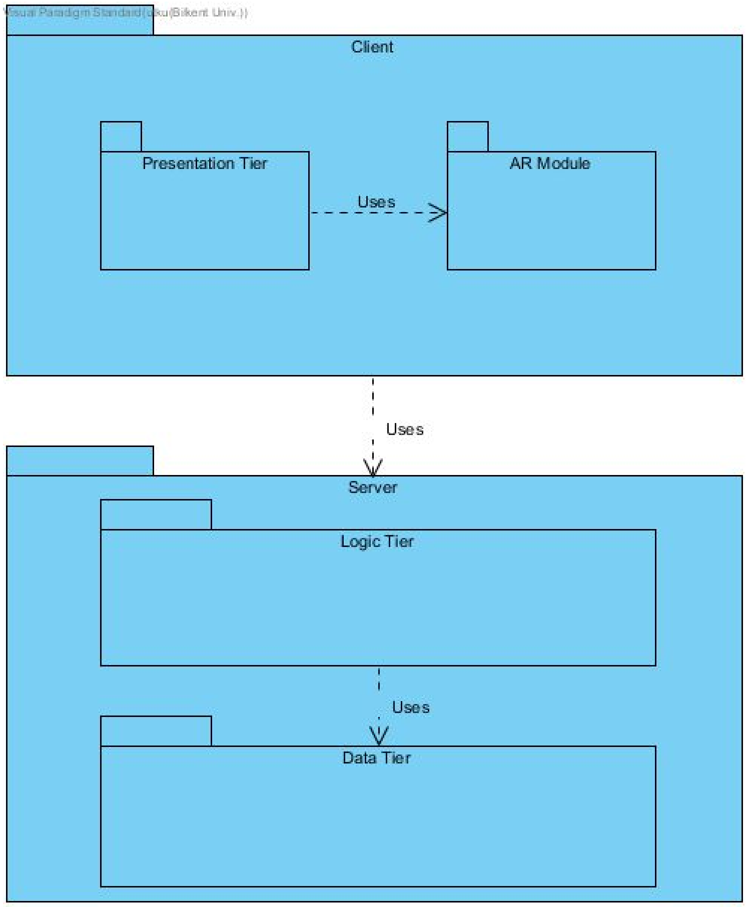


Figure 1: Subsystem Decomposition

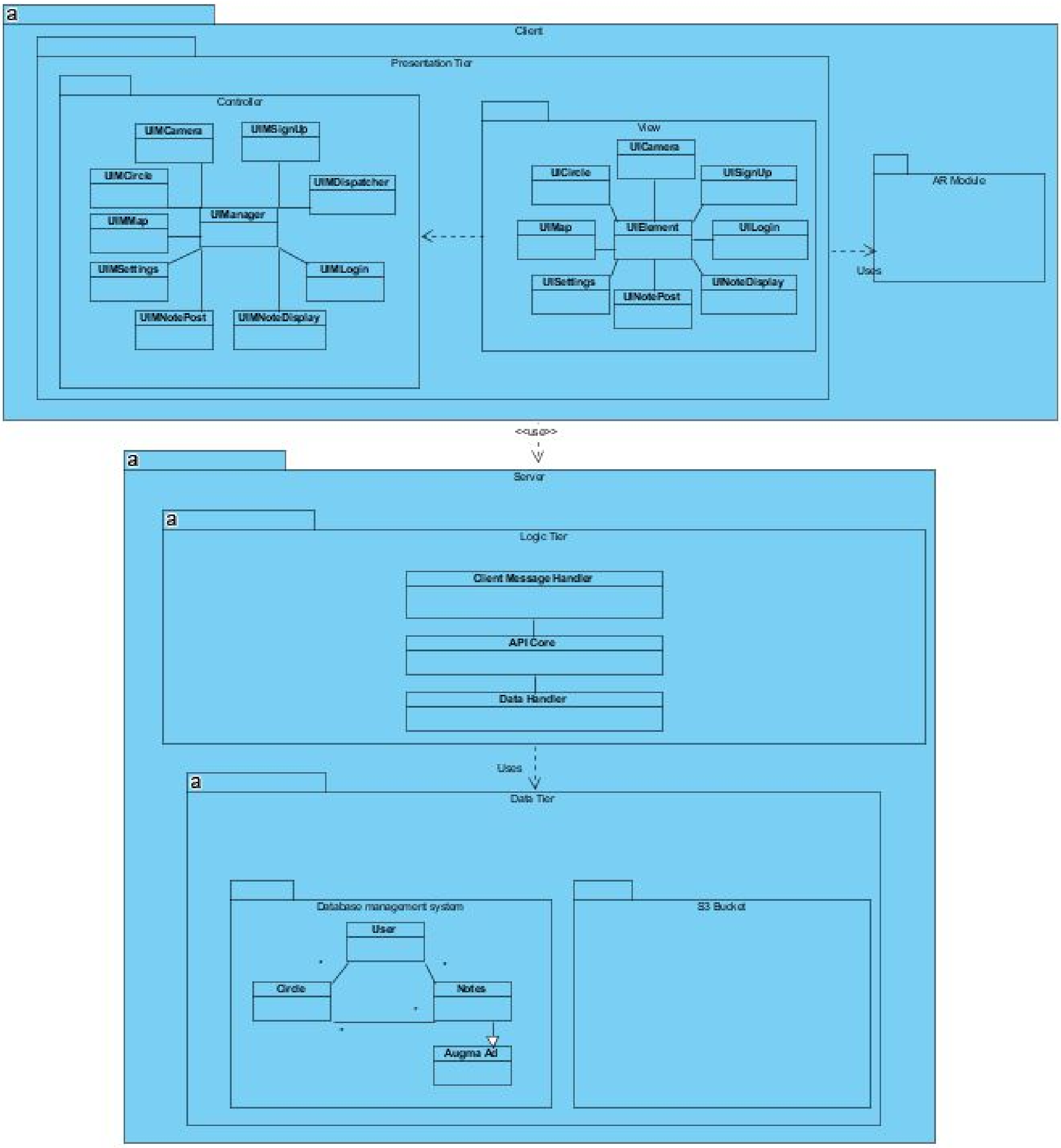


Figure 2: Subsystem Decomposition Detailed View

4.2 Subsystem Services

The subsystem services of our system is too complex to be directly modeled in a single diagram. So, a simpler overview of our subsystem services is provided in the following diagram. For more detailed explanations, class diagrams of our subsystem services are added to Appendix B.

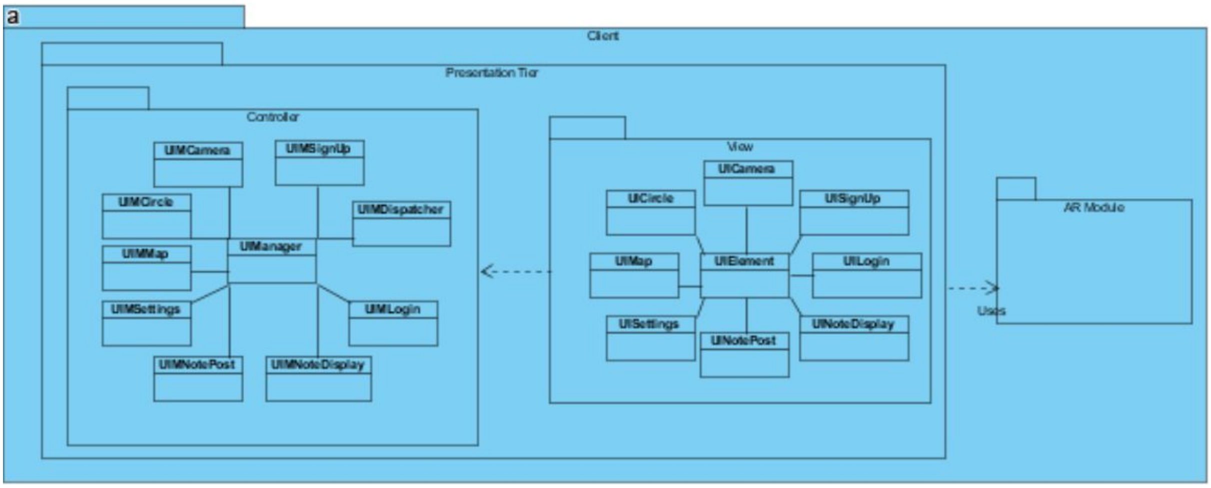


Figure 3: Subsystem Services Overview

4.3 Client

On the client side, Augma utilizes various kinds of technologies ranging from image transformations to inter-activity communications via services. For note-related operations, we have bitmap transformation classes each of which renders certain type of effects such as blur, cropping etc. in order to embellish the user interface. To work with images we have AugmaImager class which functions as a quick and stable interface. It caches the image on disk and ram and converts it to bitmap in order to reduce the time consumed for operations such as reloading same images over and over again. Both original and rendered versions of the image are saved, except for notes which are mostly requested by users only once. To communicate with Amazon Web Service, we developed a class named AWS, which handles JSON composition and parsing. To establish such communication via internet, we have HttpCallHandler that does the job of sending requests and receiving responses; it is basically our façade to the outer world.

For inter-activity communication we established services that provide only the necessary information/functionality to the requesting Activities. The information is secured by implementing a particular interface by the service-providing Activity; only the information that needs to be passed is sent via method parameters, that way we've hidden unnecessary and vulnerable information from getting altered by other Activities.

4.4 Cloud

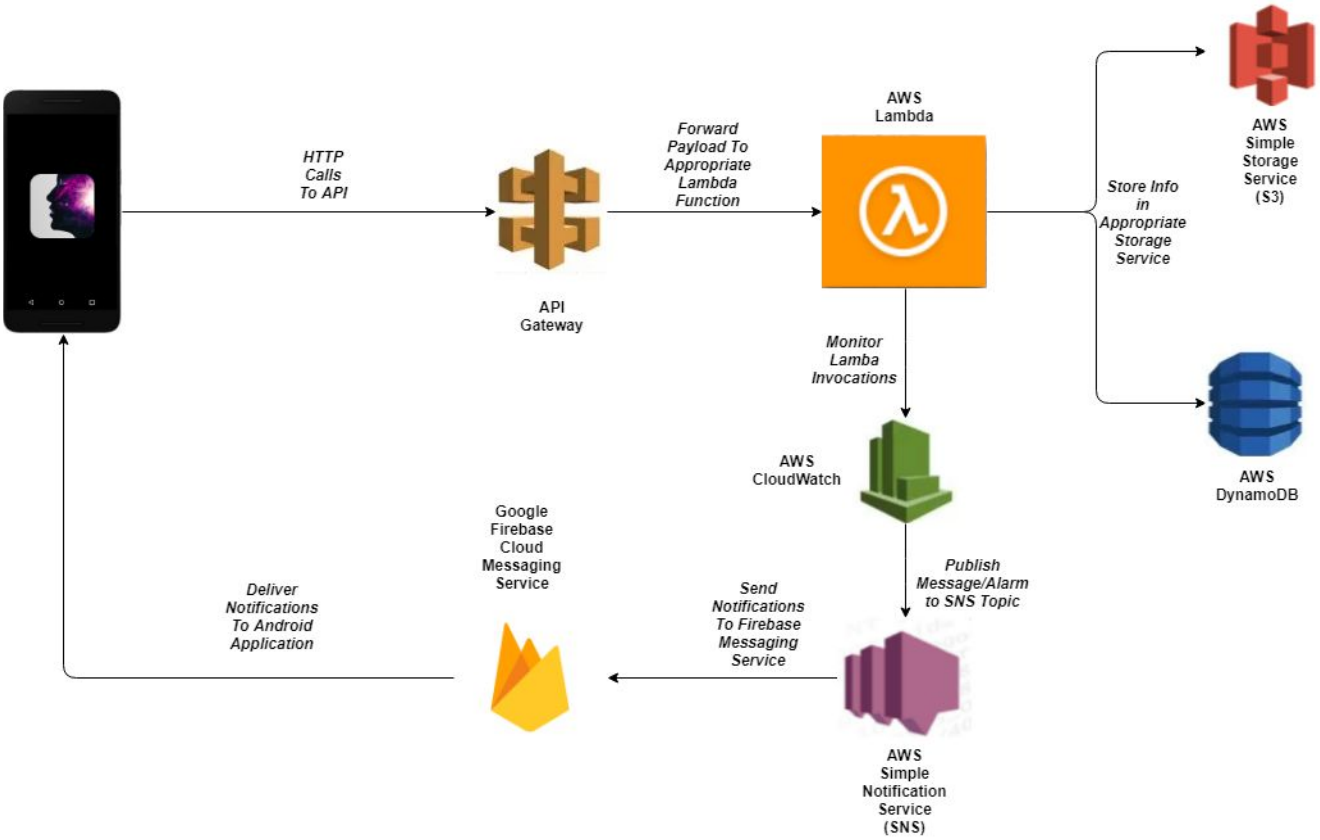


Figure 4: Cloud explanation diagram

Augma has a very sophisticated cloud architecture. Most of the cloud systems comprises of Amazon Web Services. To better explain the workflow of Augma we will be using the use case scenario of posting a new note. Starting from the moment of User hitting the post button, Augma sends an HTTP request with a JSON payload to our API endpoint at AWS API Gateway. This request is controlled and if it passes API Gateways checks, JSON payload gets forwarded to appropriate AWS Lambda function that we implemented in Node.js. This Lambda function gets the payload and parses it into a form that specific AWS storage service expects. These services are S3 for note images and DynamoDB for other text or number based info. After this payload gets send to the storage service Lambda listens to the storage service for feedback on whether the action has succeeded or not. After getting the feedback, Lambda returns it to API Gateway and API Gateway turns it into a proper HTTP call format and answers the client's HTTP request. Meanwhile AWS CloudWatch monitoring system is watching the AWS Lambda functions for invocations. When Post Note function is invoked it publishes an alarm to Augma's AWS SNS topic. SNS then communicates this alarm message to Google’s Firebase Cloud Messaging Service with the firebase tokens of android phones that needs to be notified. Firebase cloud messaging system notifies the Firebase service running behind Augma and gives a push notification on the device.

5.0 Impact of Engineering Solutions

In this section; global, economical and social impacts of Augma is discussed.

5.1 Global Impact

We are living in the era of social media. The number of social media users has reached to 2.34 billion [10]. There are a lot of social media platforms in the market such as Facebook, Twitter, Linkedin etc. According a study conducted by New York Times Consumer, people tend to use social media platforms to support a cause or an issue which they feel strongly about, to share to pass valuable information, to build a personal image and demonstrate who they are and what they stand for, to interact and stay in touch with others, and to participate and feel involved in things happening in the world [11]. Social media has a huge impact on people when it comes to creating business strategies, socializing finance, developing healthcare through public health campaigns, enhancing civil participation and engagement, responding to disasters, tackling difficult challenges from human rights violations to climate change [12].

Because of having the properties of a social media, Augma has a huge global impact in connecting people. But differently than the other social media applications, Augma makes people stand up from their chairs and start moving to explore notes. A note that is posted by a user can only be read by other users providing they are standing at the exact location where the note was initially posted. Instead of sitting all day and browsing through posts that are shared through other social media, Augma users always stay fit and active by travelling distances to discover notes. In this way, Augma changes the way how people interact with each other via social media.

5.2 Economic Impact

One of the benefits of social media for its users that they are no longer bounded by their geographic and demographic boundaries, they can freely group up under the topics that sound interesting to them. These groups are growing organically, as they are established around common interests. These formations have a huge economic impact because it makes the companies be able to reach out to specific audiences who are interested in their focus area. In our case, Augma has circles, which are focused to a huge variety of interests. People can join these circles and post notes under these circles. From the commercial perspective of Augma, partner companies can choose to target specific circles for spreading their Augma Ads. Furthermore, Augma brings the concept of location-based and context-based advertising. Corporate accounts can leave Augma Ads to a specific location of their choice to especially target the people who are living in that area. Augma Ads feature creates a new way for branding and thus, the companies can develop their customer portfolio and increase their earnings. When it is generalized, economic impact of Augma can contribute to the improvement of companies, and thus it will lead to the improvement of industries in which they take place.

5.3 Social Impact

With its context-based and location-based notes property, Augma has a huge social impact on social media users. Differently than the other social media platforms, as we discussed under its global impact, Augma notes are dependent to location and context. Let us give an example of an Instagram story, which was posted in Tomorrowland, an electronic music festival that takes place in Belgium and gets sold-out in under 24 hours every year [13]. The viewers of the story are not likely to get the full understanding of the context by watching the story while sitting on a couch at home. An equivalent case for Augma would be as follows: The note that is posted in Tomorrowland is only accessible in a specific radius and in the circles that the note is attached to. In this case, only the people who are participating in Tomorrowland can read this note. As Augma notes does not only consist of text, but also includes an image, an Augma user who is attending to Tomorrowland 2022 can display a note left in Tomorrowland 2018 and get its full meaning by both being at the same location and observing the image snapshot of that moment which was 4 years ago. In this way, Augma changes how people interact with each other in social media and strengthens the feeling of empathy, by limiting the notes to a location and providing a context with an image.

6.0 Engineering Solutions and Contemporary Issues

We observed that in social media environment, the interactions between people are very shallow. Interactions only consist of simple texts, images that are not able to fully carry the meaning of their intend. In Augma, we solve this contemporary issue with the help of notes. As described in previous sections, Augma notes have location and context information attached. A user is able to read a note only if he is in the radius of the note's location. Also, the user is able to see an image which was taken at the time the note was left. This feature gives the user a better understanding of the context, since he is standing at the location the note was left and seeing an image from the moment the note was initially left.

7.0 Tools and Technologies Used

In this section, we are going to give information about the technologies and the tools we used during the development stage.

Android Studio: IDE for Android Development. Our client side was implemented as an Android Application that we tested both in emulators and our phones.

Gradle: As part of the Android Studio we used Gradle for adding libraries. Building pages using XML and builders helped us too.

GitHub: This was used as a version control tool and as a Git repository. Issues and our plans was also published to the GitHub's issue tracking system.

Discord: Discord was used to communicate via voice between developers. Text channels of the discord server created for this project are also used for sharing important documents and links between developers.

TeamViewer: TeamViewer was used to control and view other developers screens for debugging purposes.

Amazon Web Services: AWS was used for most of Augma’s cloud infrastructure. We user API Gateway, Lambda, DynamoDB, Simple Storage service(S3), CloudWatch and SNS services.

Google Firebase Cloud Messaging: Firebase cloud messaging service was used to send push notifications from Augma’s cloud services to client devices.

Visual Paradigm: A software design tool. This was used to draw diagrams for the reports.

7.1 Library Resources

AppCompatActivity: Base class for activities that use the support library action bar features [14].

ConstraintLayout: A ConstraintLayout is a ViewGroup which allows you to position and size widgets in a flexible way [15].

RecyclerView: A flexible view for providing a limited window into a large data set [16].

Dagger: Dagger 2 is a compile-time evolution approach to dependency injection [17].

Firebase Core: To use Firebase Messaging this library is needed. It connects Augma to the Google’s Firebase. For future use of other functions of Firebase this is required as well.

Firebase Messaging: To send notifications to Augma when a new note is sent or a beacon is lit, this library is used.

Google Services: Google Play services, your app can take advantage of the latest, Google-powered features such as Maps, Google+, and more, with automatic platform updates distributed as an APK through the Google Play store [18].

Google Services Maps: Library that connects Google Places API to Augma.

WaveLoadingView: An Android library that provides a realistic wave-loading effect [19].

Loading Button: Android Button that morphs into a loading progress bar. It is fully customizable in the XML and really simple to use [20].

Circle Menu: Simple, elegant UI menu with a circular layout and material design animations [21].

Gravity View: Gravity View is an Android adaptation of Facebook instant articles. The concept behind the library is to utilize the motion sensors of an Android device and allow the end user to explore the product by rotating his device. It uses gyroscope motion sensor readings to scroll the image [22].

Wave View: A wave view of android, can be used as progress bar [23].

Bubble Picker: An easy-to-use animation which can be used for content picking for Android [24].

Glide: An image loading and caching library for Android focused on smooth scrolling [25].

CardView: Cards provide an easy way to contain a group of views while providing a consistent style for the container [26].

KenBurnsView: Android ImageViews animated by Ken Burns Effect [27].

Sneaker: A lightweight Android library for customizable alerts [28].

7.2 APIs Used

Google Maps Platform (formerly Google Maps API): Used to get location data of the user and to present the notes' whereabouts. Direction info is also calculated and the note filtering is extended by the Platform.

RESTful API: AWS API Gateway was used to create a custom RESTful API that would handle communication between Augma's client side and the cloud infrastructure.

Firebase API: This API was used to receive communication from the cloud infrastructure of Augma while being used in tandem with AWS SNS to send push notification from the cloud side.

References

[1] WoodrowBarfield, Fundamentals of Wearable Computers and Augmented Reality, CRC Press, 2016.

[2] Matt Bower, Cathie Howe, Nerida McCredie, Austin Robinson & David Grover, «Augmented Reality in education – cases, places and potentials,» Educational Media International, 2014.

[3] Mark Billinghurst, Adrian Clark, Gun Lee, «A Survey of Augmented Reality,» Foundations and Trends in Human-Computer Interaction, cilt 8, pp. 73-272, 2014.

[4] Clemens Arth, Lukas Gruber, Raphael Grasset, Tobias Langlotz,Alessandro Mulloni, Dieter Schmalstieg, Daniel Wagner, «The History of Mobile Augmented Reality-Developments in Mobile AR over the last almost 50 years,» Inst. for Computer Graphics and Vision Graz University of Technology, Graz, 2015.

[5] Thomas Olsson, Kaisa Väänänen, «Expected user experience of mobile augmented reality services: A user study in the context of shopping centres,» Personal and Ubiquitous Computing, 2011.

[6] Philip Geiger, Marc Schickler, Rudiger Pryss, Johannes Schobel, Manfred Reichert, «Location-based Mobile Augmented Reality Applications,» 2014.

[7] Panos E. Kourouthanassis, Costas Boletsis, George Lekakos, «Demystifying the design of mobile augmented reality applications,» Multimedia Tools and Applications, 2013.

[8] Rüdiger Pryss,Philip Geiger,Marc Schickler,Johannes Schobel,Manfred Reichert, «Advanced Algorithms for Location-Based Smart Mobile Augmented Reality Applications,» Procedia Computer Science, cilt 94, pp. 97-104, 2016.

[9] Zhihan Lv ,Alaa Halawani , Shengzhong Feng , Shafiq ur Re´hman, Haibo Li, «PreprintTouch-less Interactive Augmented Reality Game on Vision Based Wearable Device,» 2014.

[10] Statisca, “Percentage of US Population with a Social Network Profile”, *statisca.com*. [Online]. Available: https://www.statista.com/statistics/273476/percentage-of-us-population-with-a-social-network-profile/

[11] M. Gaitho, “What is the real impact of social media?”, *simplilearn.com*. [Online]. Available: https://www.simplilearn.com/real-impact-social-media-article [Accessed: May 2, 2018].

[12] A. Guzman “6 ways social media is changing the world”, *weforum.org*. [Online]. Available: https://www.weforum.org/agenda/2016/04/6-ways-social-media-is-changing-the-world/ [Accessed: May 2, 2018].

[13] Tomorrowland, *tomorrowland.com*. [Online]. Avaialble: https://www.tomorrowland.com/global/ [Accessed: May 2, 2018].

[14] Android Developers, “AppCompatActivity”, *developer.android.com*, [Online]. Available: https://developer.android.com/reference/android/support/v7/app/AppCompatActivity [Accessed: May 2, 2018].

[15] Android Developers, “ConstraintLayout”, *developer.android.com*, [Online]. Available: https://developer.android.com/reference/android/support/constraint/ConstraintLayout [Accessed: May 2, 2018].

[16] Android Developers, “RecyclerView”, *developer.android.com,* [Online]. Available: https://developer.android.com/reference/android/support/v7/widget/RecyclerView [Accessed: May 2, 2018].

[17] Dagger, *google.github.io*, [Online]. Available: https://google.github.io/dagger/ [Accessed: May 2, 2018].

[18] Android Developers, “Guides Overview”, *developers.google.com*. [Online]. Available: https://developers.google.com/android/guides/overview [Accessed: May 2, 2018].

[19] Tangqi92, “WaveLoadingView”, *github.com.* [Online]. Available: https://github.com/tangqi92/WaveLoadingView [Accessed: May 2, 2018].

[20] leandroBorgesFerreira, “LoadingButtonAndroid”, *github.com*. [Online]. Available: https://github.com/leandroBorgesFerreira/LoadingButtonAndroid [Accessed: May 2, 2018].

[21] Ramotion, “circle-menu”, *github.com*. [Online]. Available: https://github.com/Ramotion/circle-menu [Accessed: May 2, 2018].

[22] gofynd, “gravity-view”, *github.com*. [Online]. Available: https://github.com/gofynd/gravity-view [Accessed: May 2, 2018].

[23] john990, “WaveView”, *github.com*. [Online]. Available: https://github.com/john990/WaveView [Accessed: May 2, 2018].

[24] igalata, “Bubble-Picker”, *github.com.* [Online] Available: https://github.com/igalata/Bubble-Picker [Accessed: May 2, 2018].

[25] bumptech, “Glide”, *github.com.* [Online]. Available: https://github.com/bumptech/glide [Accessed: May 2, 2018].

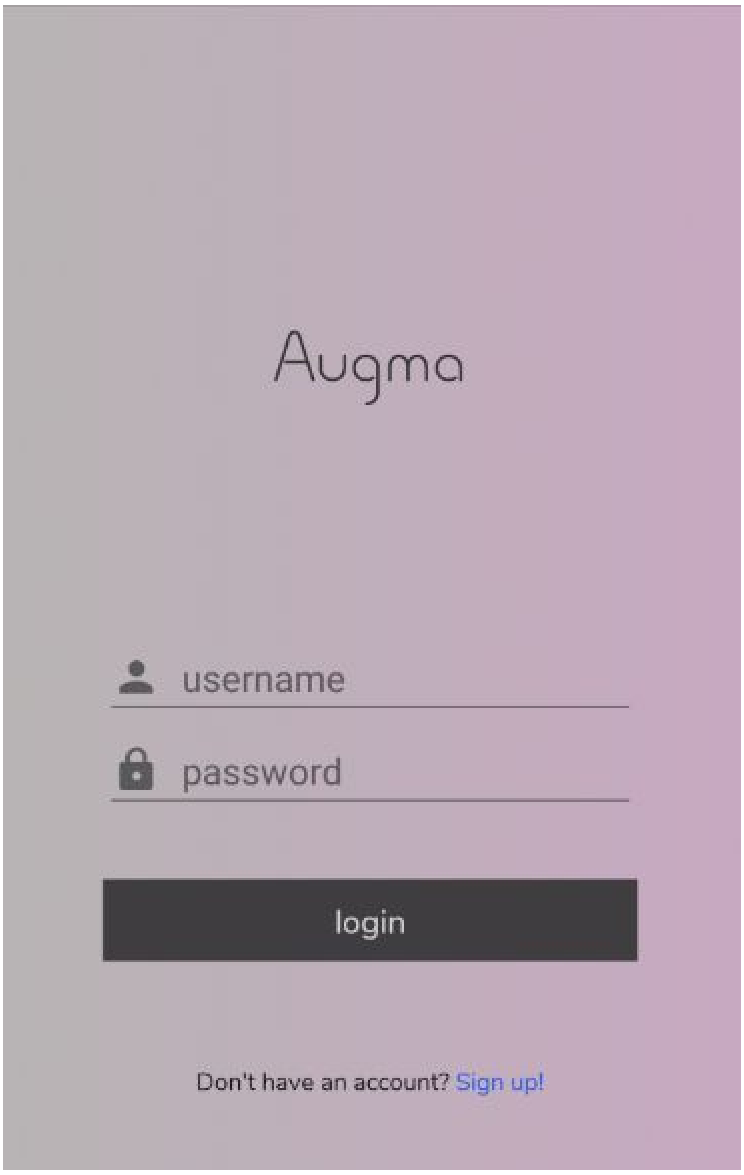
[26] Android Developers, “CardView”, *github.com*. [Online]. Available: https://developer.android.com/guide/topics/ui/layout/cardview [Accessed: May 2, 2018].

[27] flavioarfaria, “KenBurnsView”, *github.com*. [Online]. Available: https://github.com/flavioarfaria/KenBurnsView [Accessed: May 2, 2018].

[28] Google Developers, “Maps Platform”, *github.com.* [Online]. Available: https://cloud.google.com/maps-platform/ [Accessed: May 2, 2018].

Appendix A: User Manual

Login Page



Augma

username

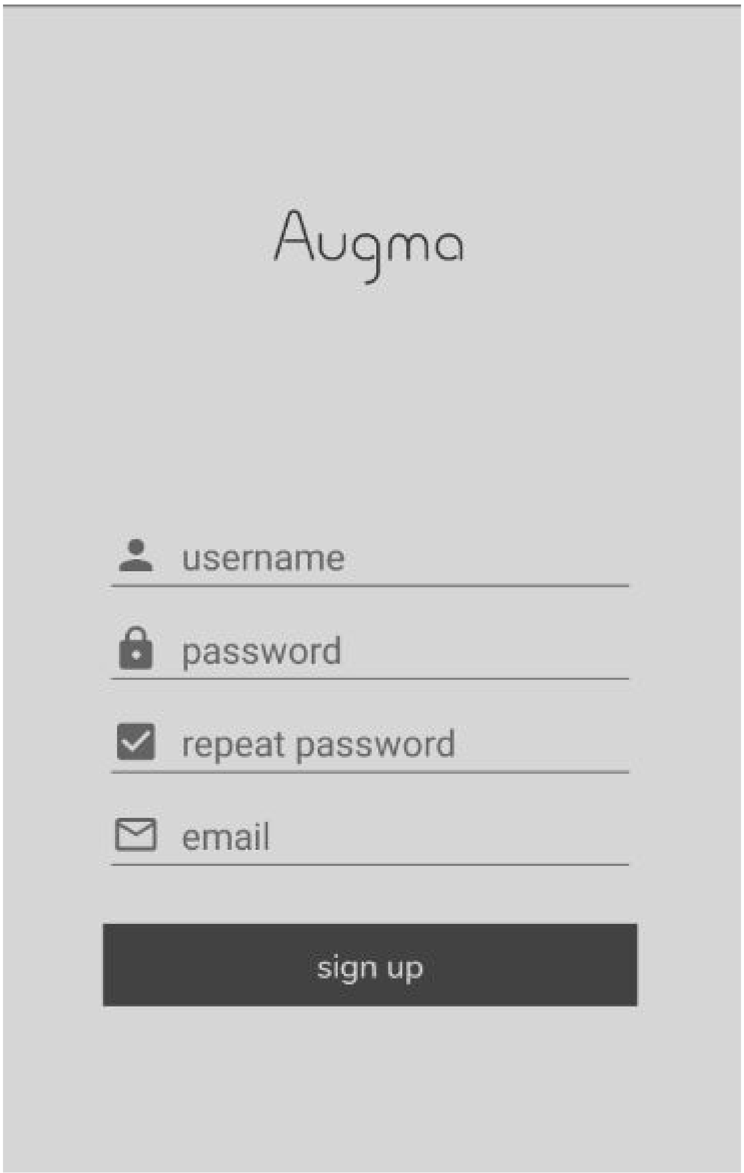
password

login

Don't have an account? Sign up!

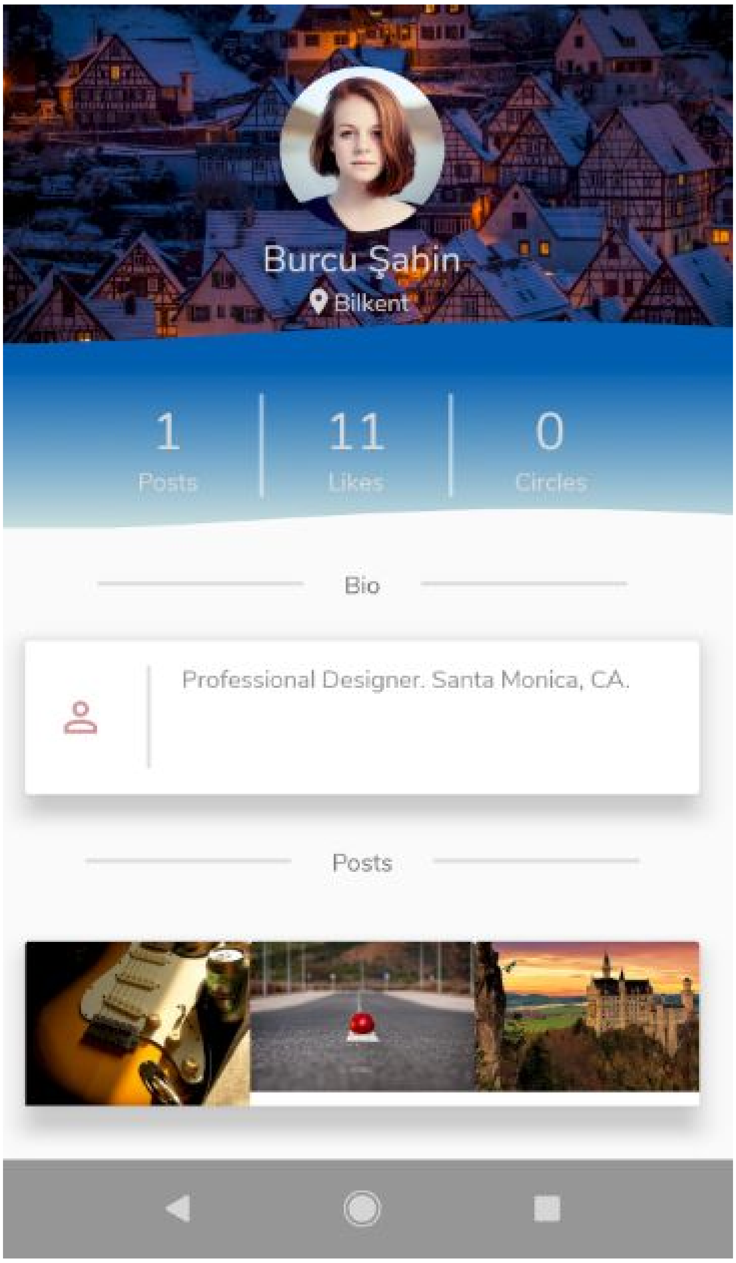
The above screen is the login screen where the user has to enter his/her username and password and click the login button in order to login to Augma. If the user does not have an Augma account, he/she can click the blue “Sign up!” text in order to access the sign up page and create an Augma account.

Sign up Page



The above screen is the sign up screen where the user without an Augma account can create an account by giving information about himself/herself. After filling all of the empty spaces seen above the user can click the sign up button and he/she can create an Augma account if there isn't any problem like: same username, same email.

Profile Page


Burcu Şahin

Bilkent

1 Posts | 11 Likes | 0 Circles

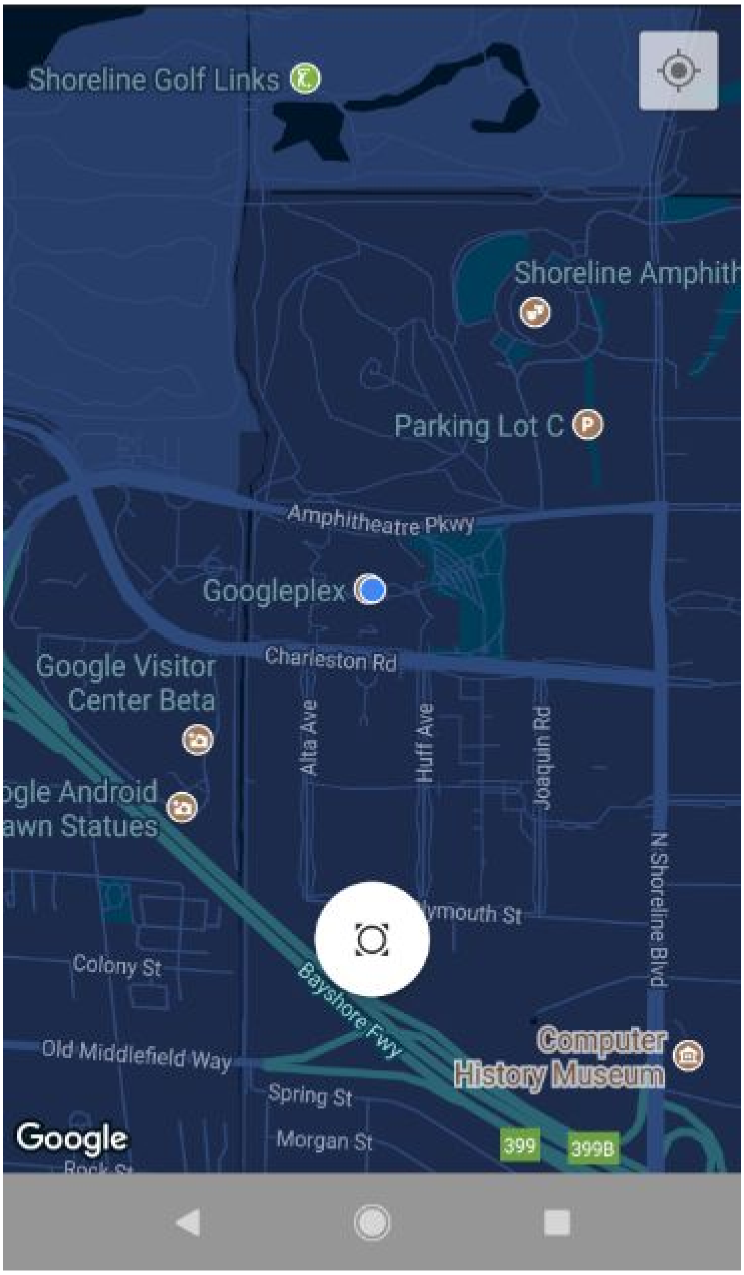
Bio

Professional Designer. Santa Monica, CA.

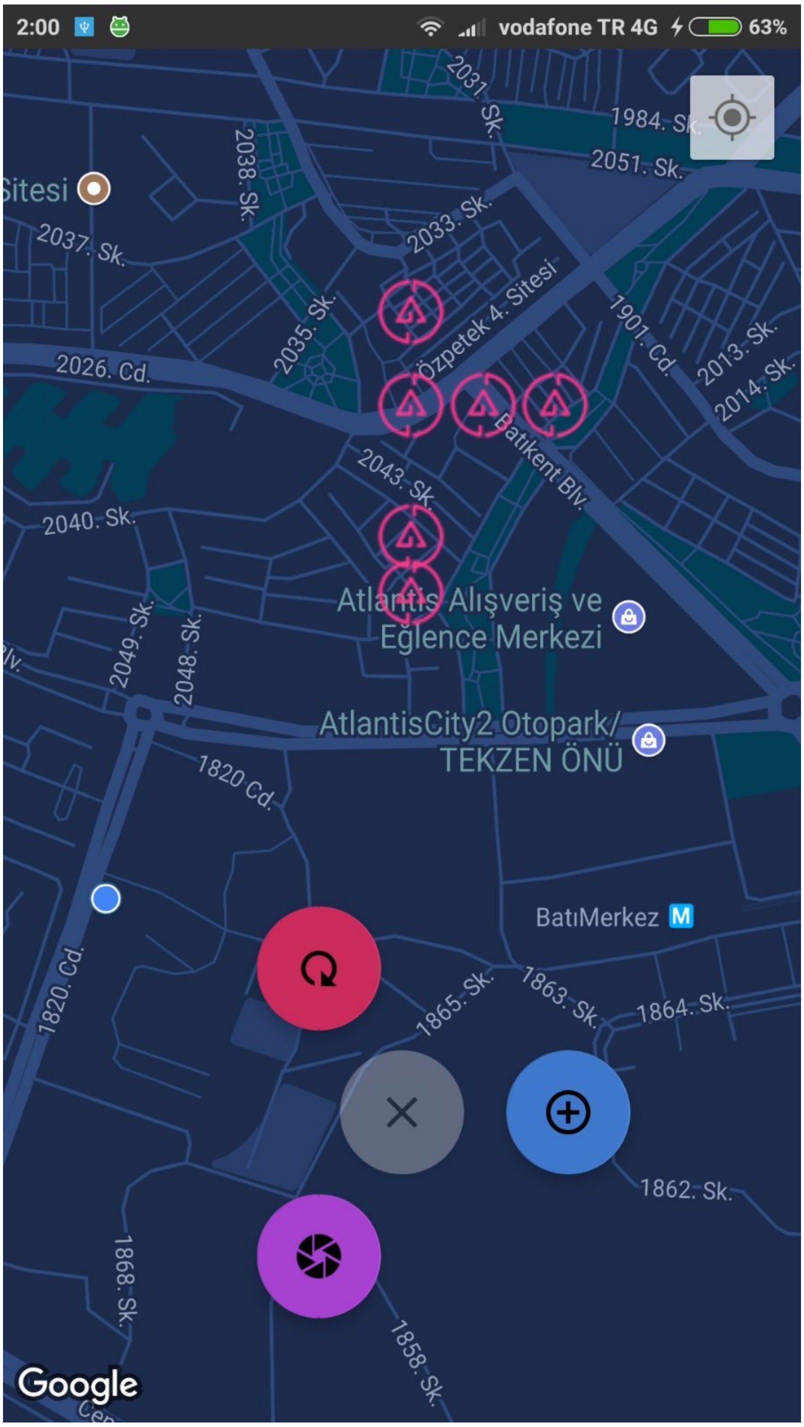
Posts

The above is the personal page of the user. In this page, users can display their name, see how many notes they have posted, see how many likes they have got for their notes, and see how many circles they are a member of. Users can change their public location by tapping on location text. Users can upload a profile picture by tapping on the picture circle that is located at the top of the screen. Furthermore, users can edit their public bio information by tapping on bio section. In posts section, users can display the most recent notes that they have posted.

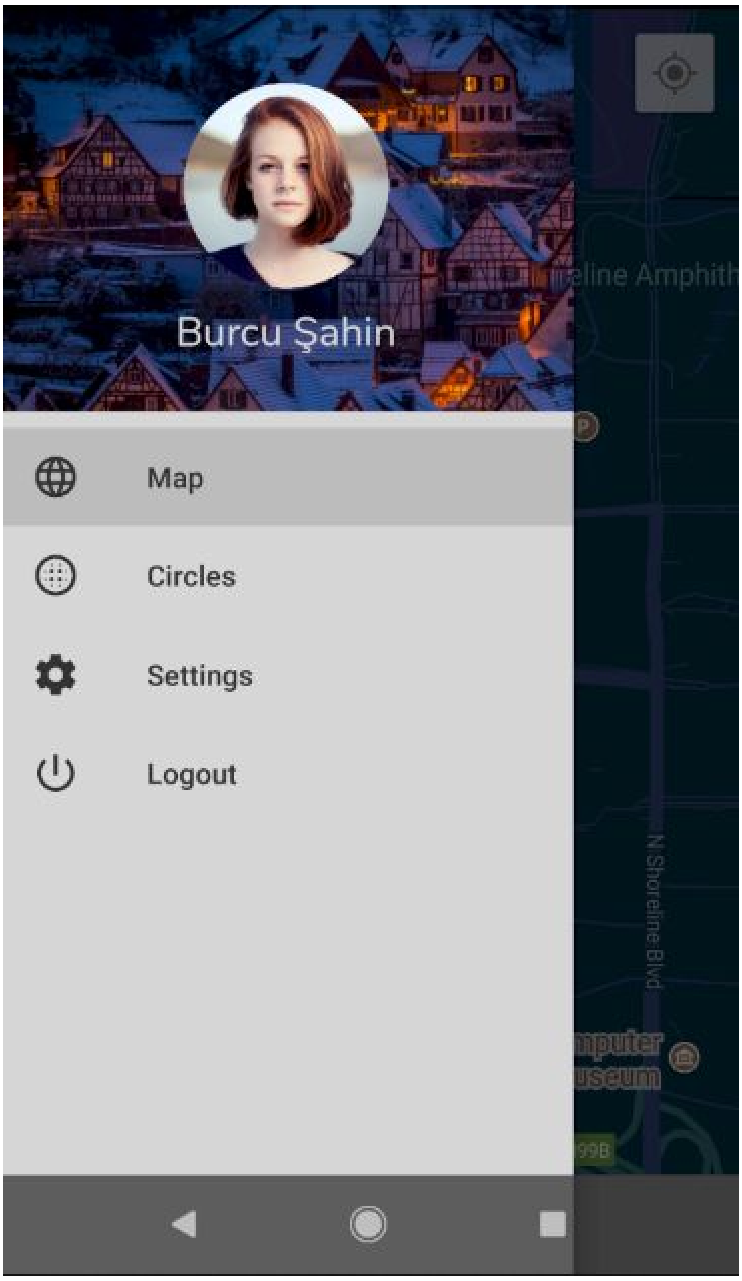
Map Page



The page above is the landing page when the users login the system. In this page, users can display their current location on the map. Also, users can see the markers for the notes that are nearby, provided that the user is member of the circle in which the note was posted.

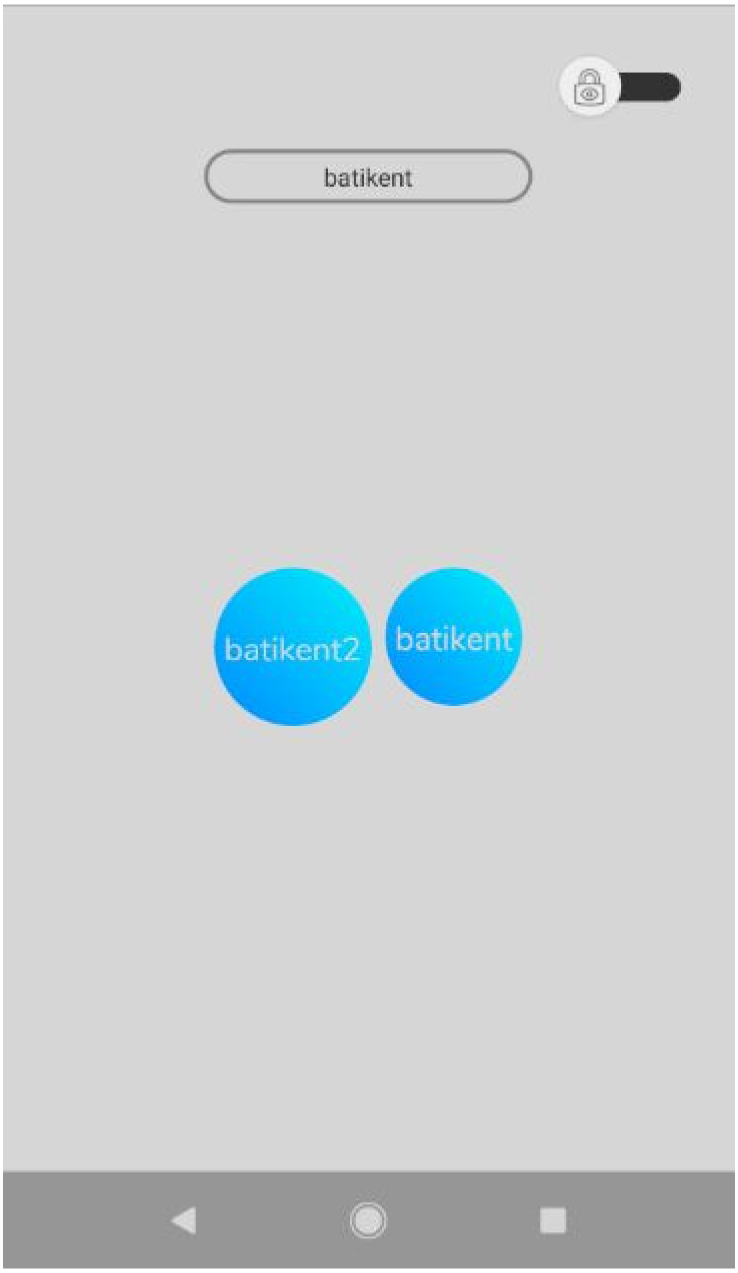


By tapping the circular button located at the middle bottom, users can open the menu for actions such as posting a note, opening the camera to browse notes and filtering the circles.



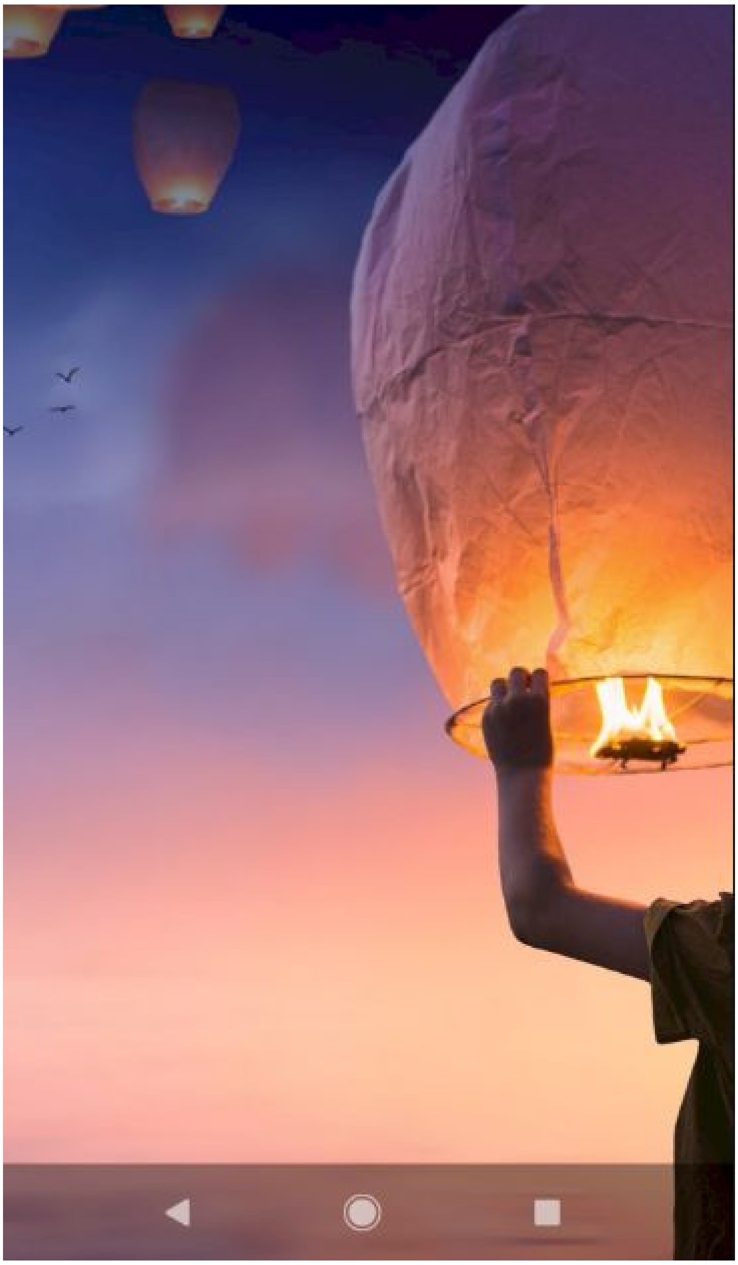
By swiping to the right, users can open the navigation drawer. In this drawer, users can navigate to their personal profile page by simply tapping on the circular profile image of them. Tapping circles will open the circles page, in which the users can search for circles and join. By tapping the settings, users will be navigated to settings screen, where they can change the app settings. By tapping logout, user will simply logout from the system and redirected to the login page.

Circles Page

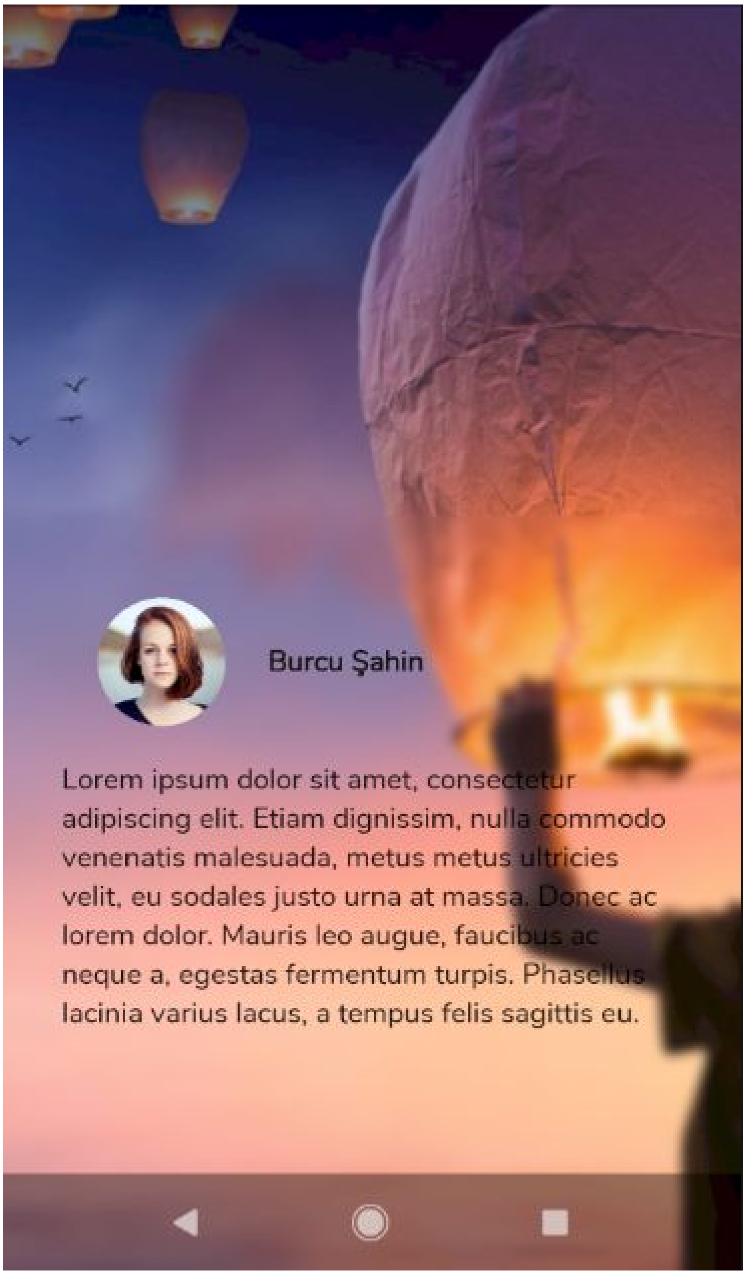


In circles page, users can search for circles by tapping the search bar at the top. After submitting the search request, the results will appear as circles in the middle of the screen. According to the results, users can join the circles of their wish and gain access to the notes that are posted in them.

Note View Page



This is the page where the users display a note. Without tapping anything on the screen, the user only sees the image that is attached to the note.



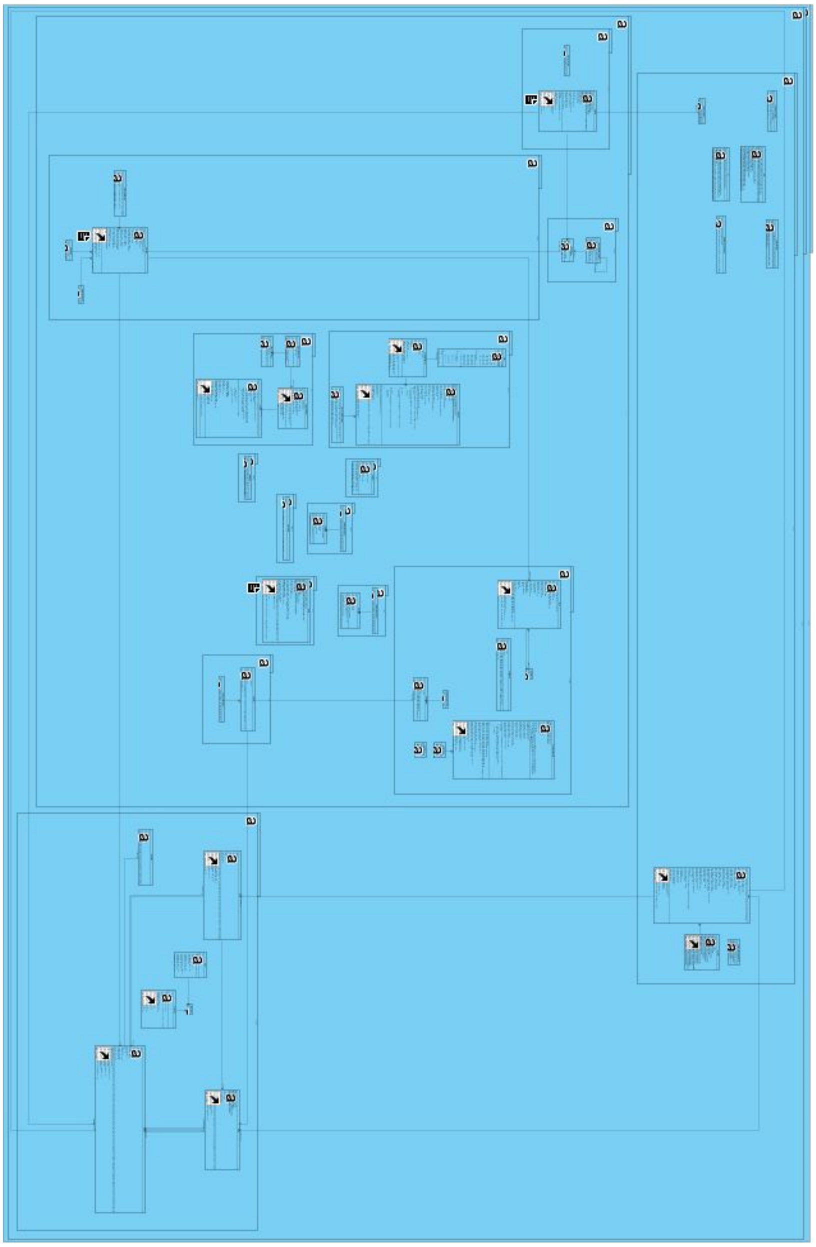
Burcu Şahin

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam dignissim, nulla commodo venenatis malesuada, metus metus ultricies velit, eu sodales justo urna at massa. Donec ac lorem dolor. Mauris leo augue, faucibus ac neque a, egestas fermentum turpis. Phasellus lacinia varius lacus, a tempus felis sagittis eu.

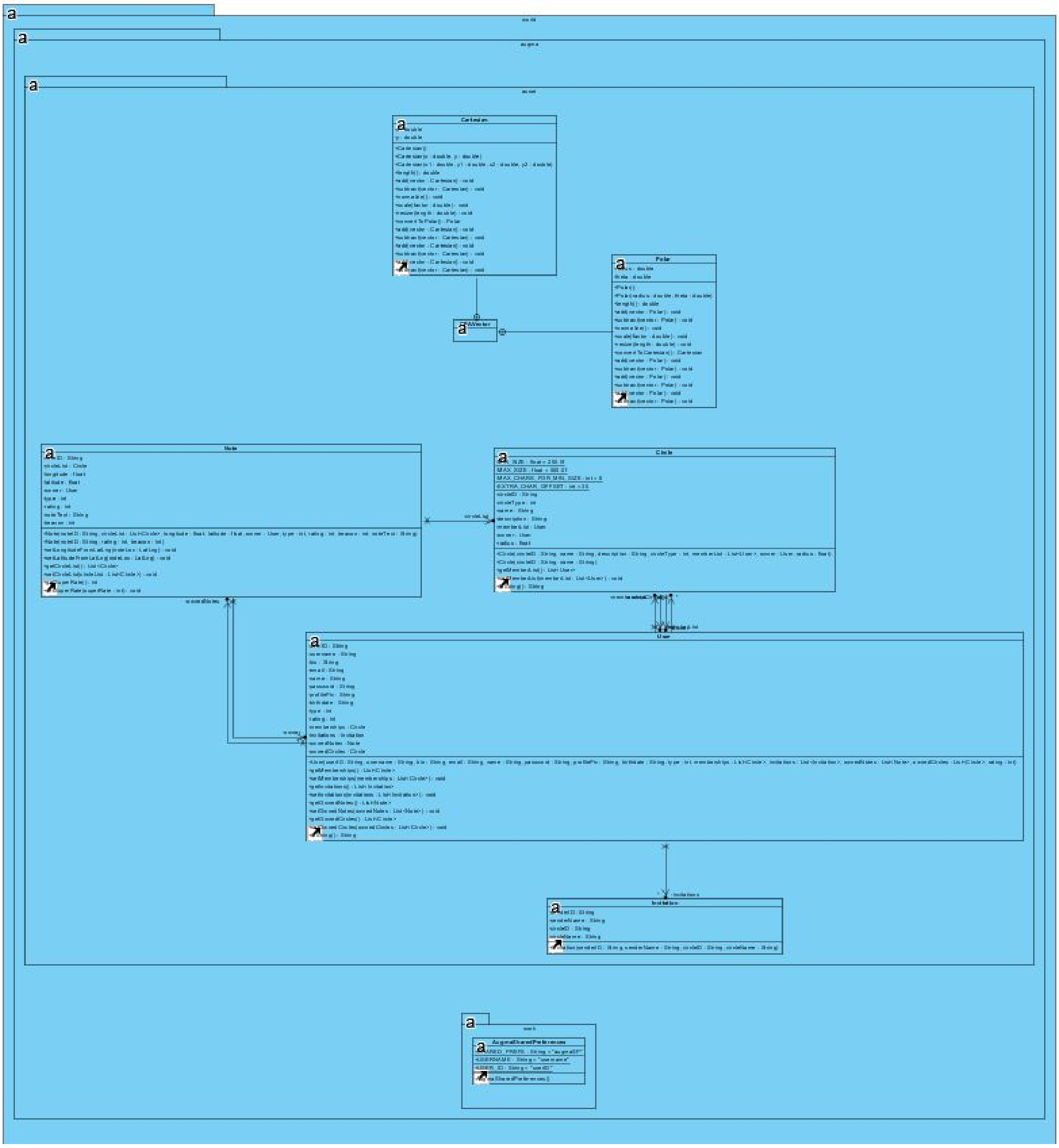
If the users keep their finger pressed on the image, the image will get blurred and the note text will appear. Here, the users can read the note. Also, users can navigate to the owner's profile by tapping on their circular profile picture.

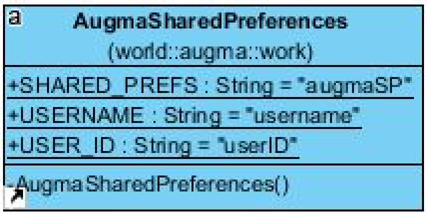
Appendix B: Class Diagrams

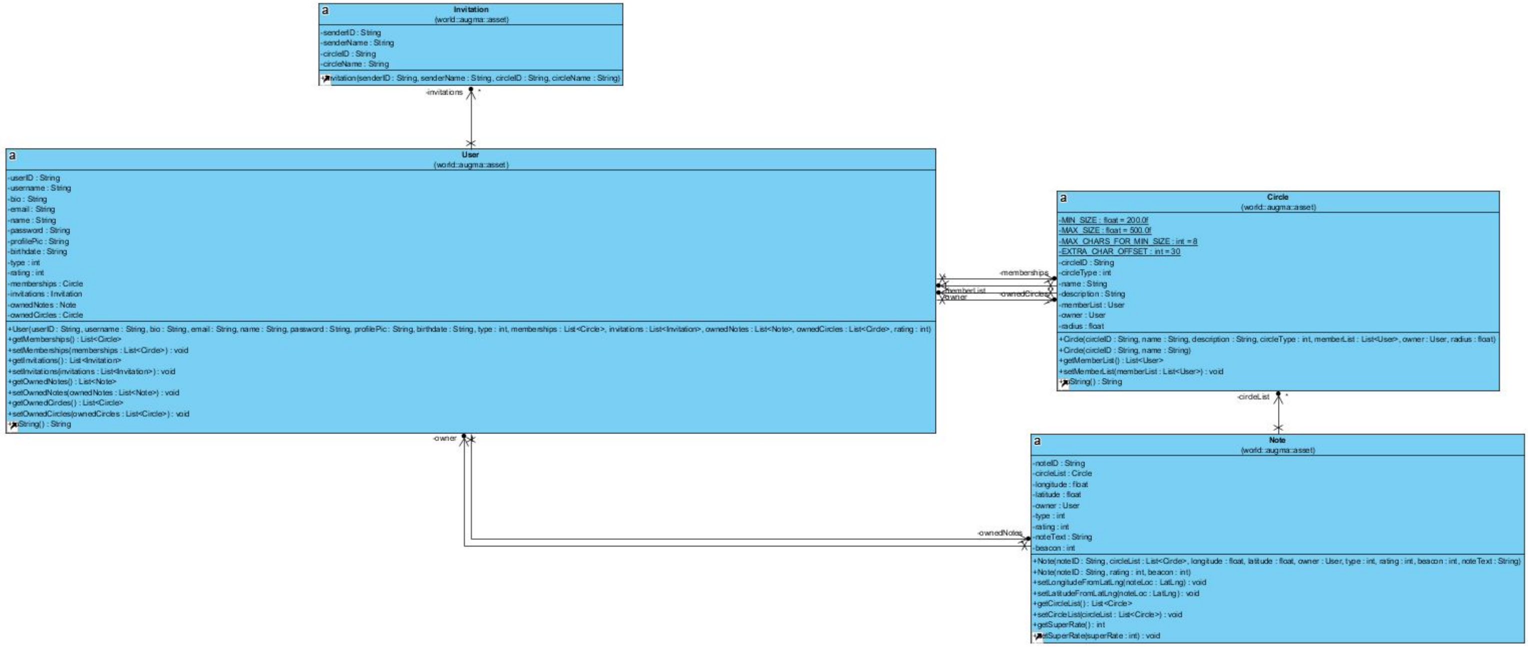
Overview of Class Diagram of Client

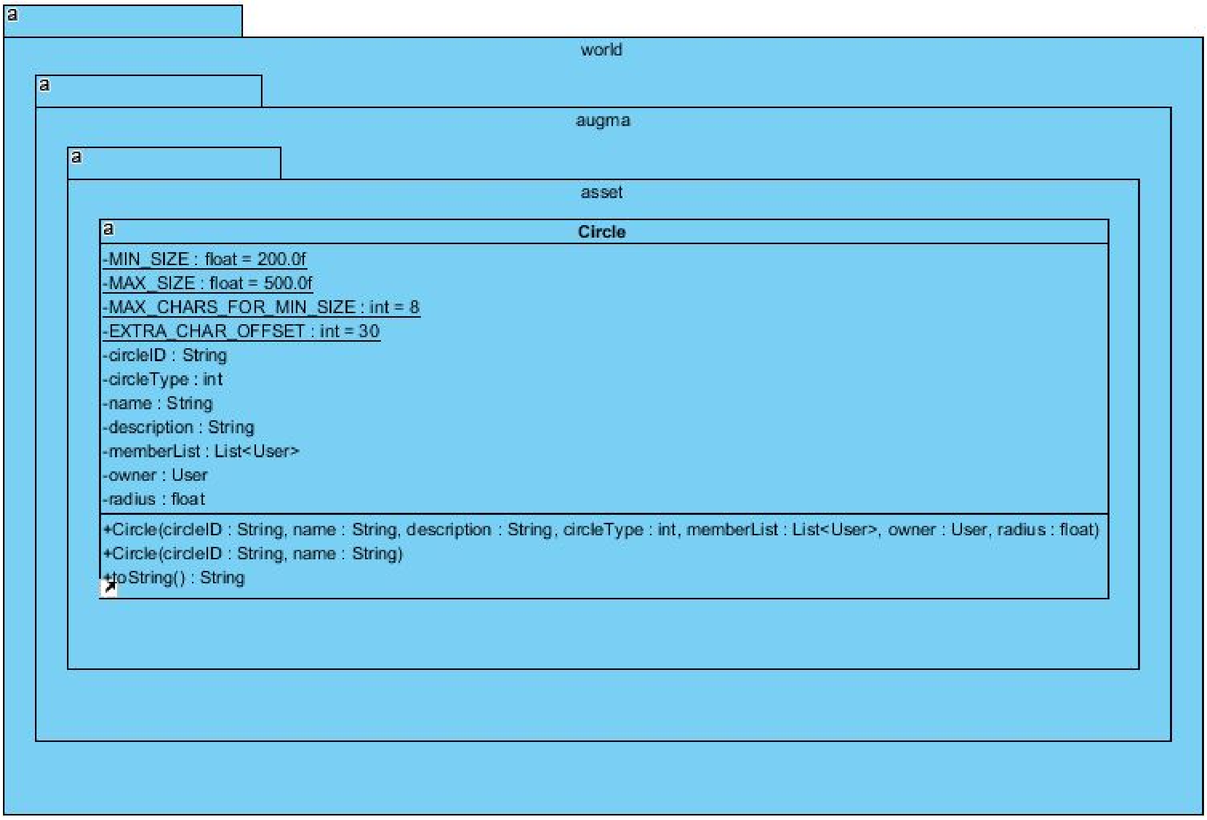


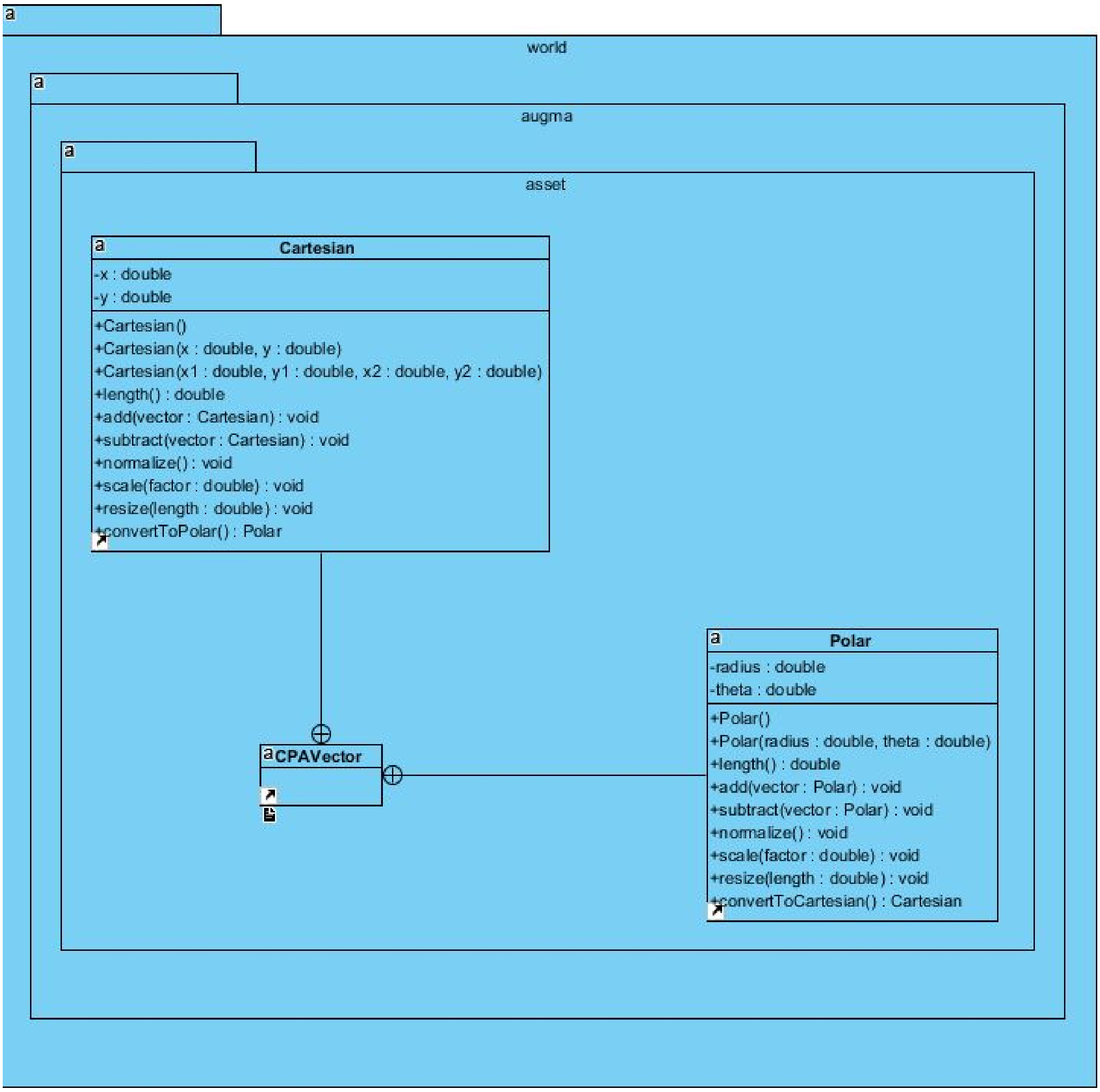
Overview of Asset Package of Client

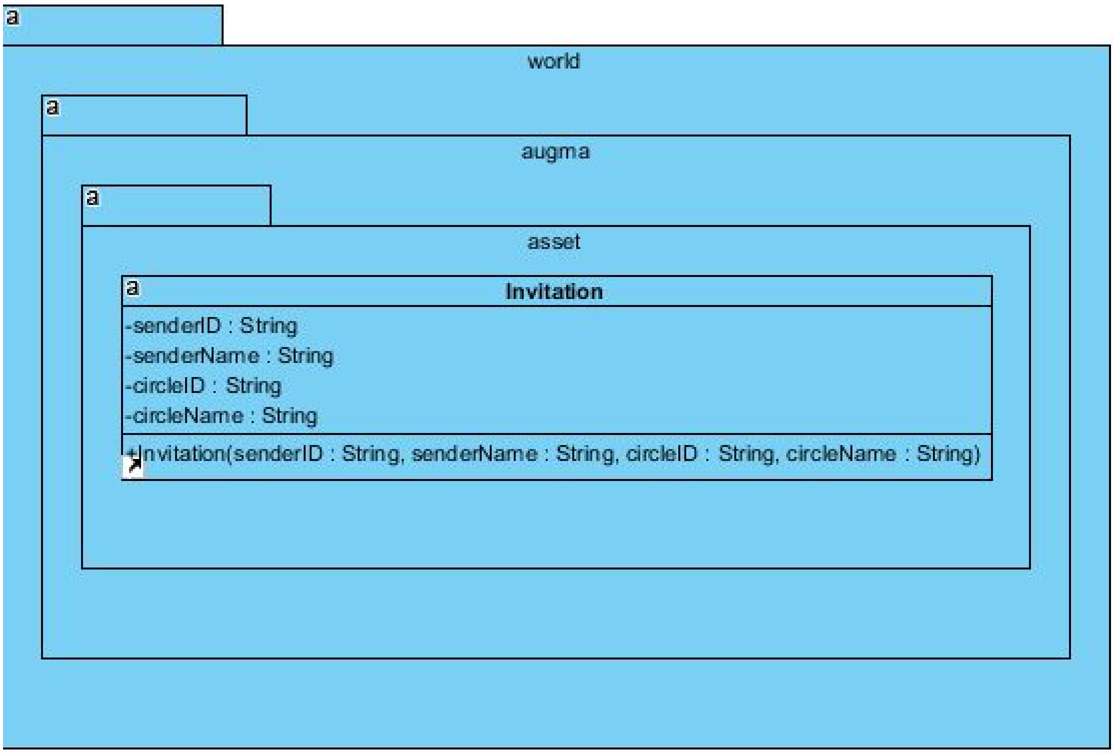


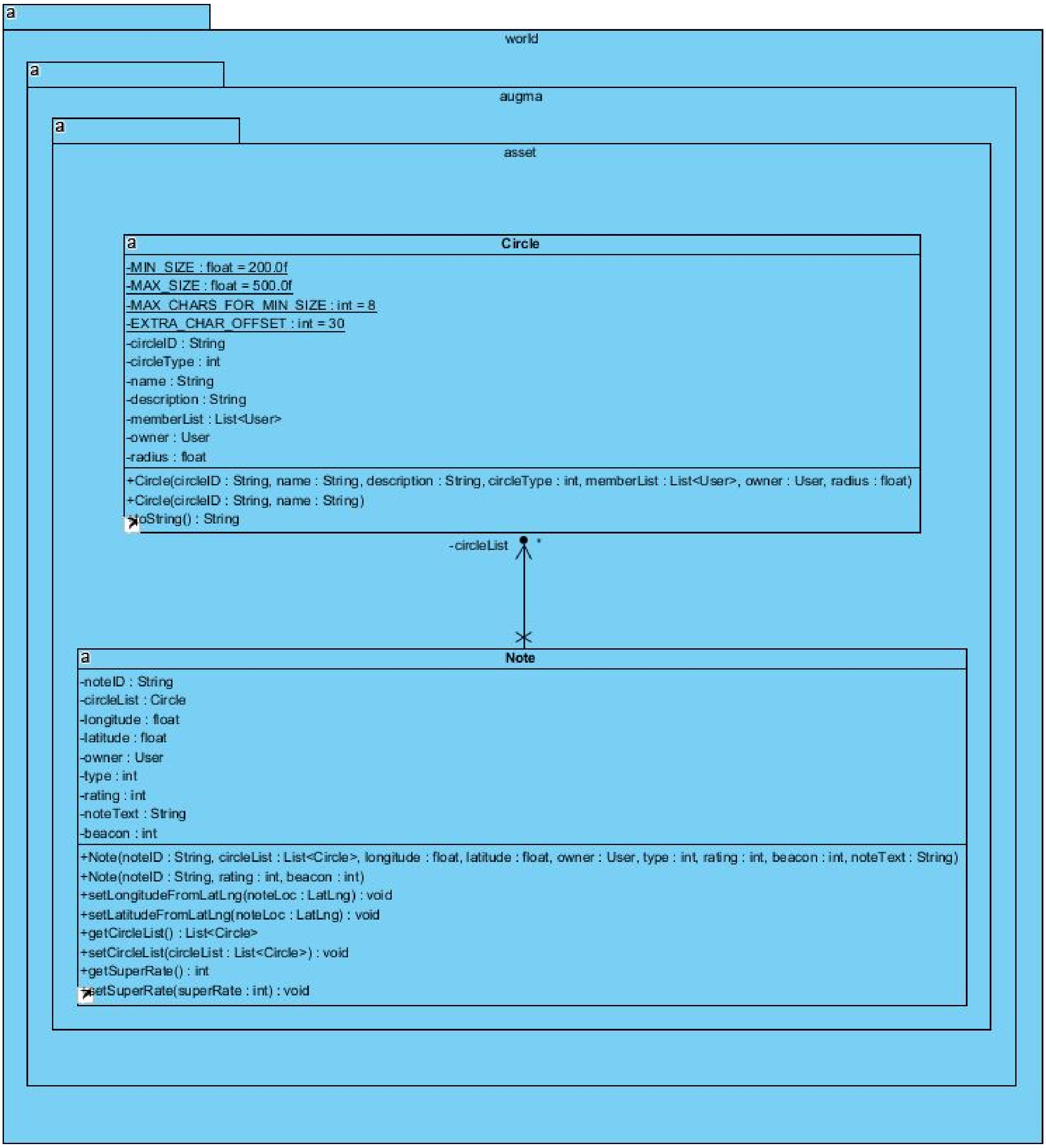

AugmaSharedPreferences
(world::augma::work)
+SHARED_PREFS : String = "augmaSP"
+USERNAME : String = "username"
+USER_ID : String = "userID"
-AugmaSharedPreferences()

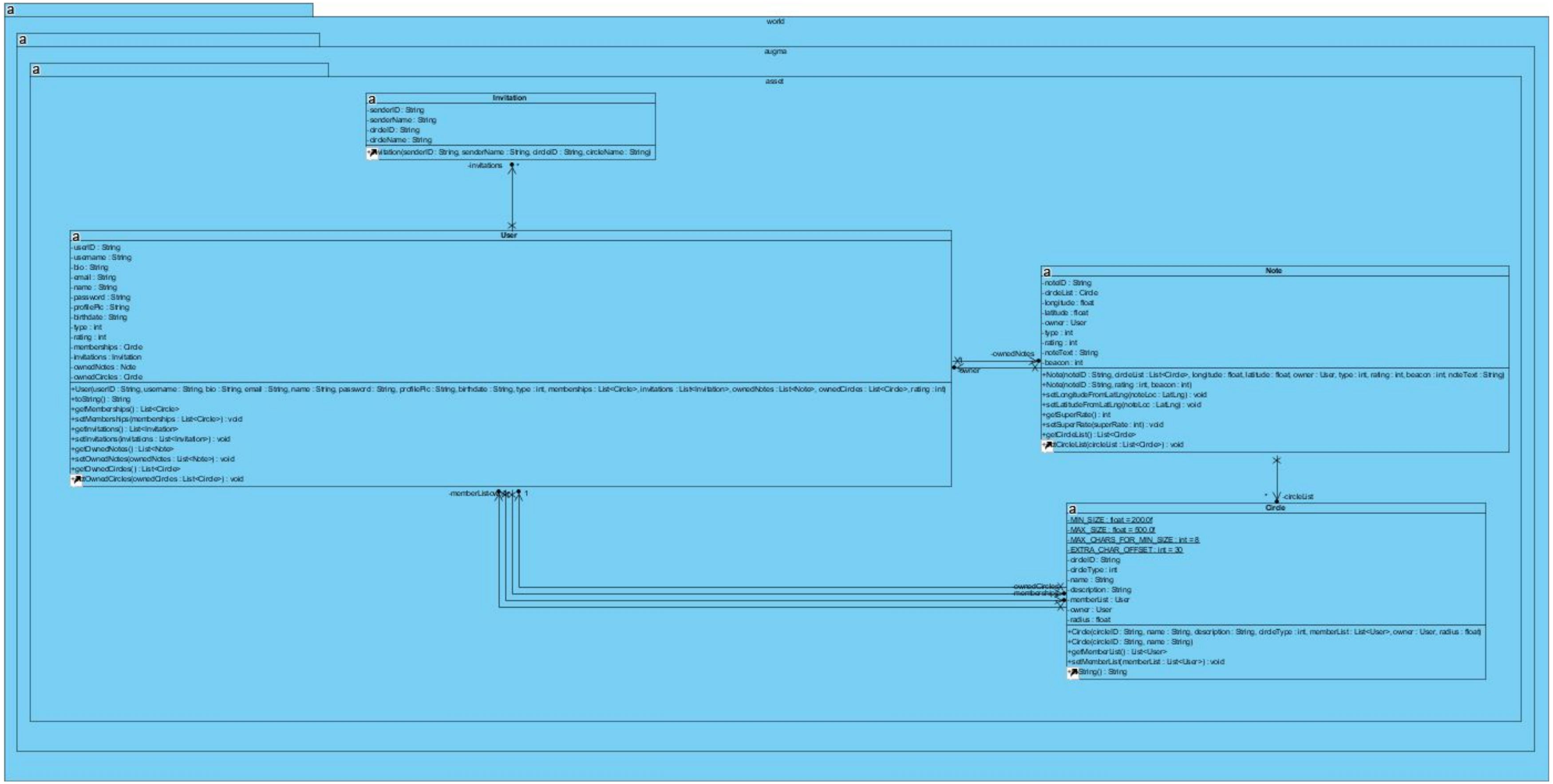

a
Invitation
(world::augma::asset)
-senderID : String
-senderName : String
-circleID : String
-circleName : String
+vitation(senderID : String, senderName : String, circleID : String, circleName : String)
-invitations
*
a
User
(world::augma::asset)
-userID : String
-username : String
-bio : String
-email : String
-name : String
-password : String
-profilePic : String
-birthdate : String
-type : int
-rating : int
-memberships : Circle
-invitations : Invitation
-ownedNotes : Note
-ownedCircles : Circle
+User(userID : String, username : String, bio : String, email : String, name : String, password : String, profilePic : String, birthdate : String, type : int, memberships : List<Circle>, invitations : List<Invitation>, ownedNotes : List<Note>, ownedCircles : List<Circle>, rating : int)
+getMemberships() : List<Circle>
+setMemberships(memberships : List<Circle>) : void
+getInvitations() : List<Invitation>
+setInvitations(invitations : List<Invitation>) : void
+getOwnedNotes() : List<Note>
+setOwnedNotes(ownedNotes : List<Note>) : void
+getOwnedCircles() : List<Circle>
+setOwnedCircles(ownedCircles : List<Circle>) : void
+oString() : String
-owner
-memberships
-memberList
-owner
-ownedCircles
a
Circle
(world::augma::asset)
-MIN_SIZE : float = 200.0f
-MAX_SIZE : float = 500.0f
-MAX_CHARS_FOR_MIN_SIZE : int = 8
-EXTRA_CHAR_OFFSET : int = 30
-circleID : String
-circleType : int
-name : String
-description : String
-memberList : User
-owner : User
-radius : float
+Circle(circleID : String, name : String, description : String, circleType : int, memberList : List<User>, owner : User, radius : float)
+Circle(circleID : String, name : String)
+getMemberList() : List<User>
+setMemberList(memberList : List<User>) : void
+oString() : String
-circleList
*
a
Note
(world::augma::asset)
-noteID : String
-circleList : Circle
-longitude : float
-latitude : float
-owner : User
-type : int
-rating : int
-noteText : String
-beacon : int
+Note(noteID : String, circleList : List<Circle>, longitude : float, latitude : float, owner : User, type : int, rating : int, beacon : int, noteText : String)
+Note(noteID : String, rating : int, beacon : int)
+setLongitudeFromLatLng(noteLoc : LatLng) : void
+setLatitudeFromLatLng(noteLoc : LatLng) : void
+getCircleList() : List<Circle>
+setCircleList(circleList : List<Circle>) : void
+getSuperRate() : int
+etSuperRate(superRate : int) : void
-ownedNotes


world
augma
asset
Circle
-MIN_SIZE : float = 200.0f
-MAX_SIZE : float = 500.0f
-MAX_CHARS_FOR_MIN_SIZE : int = 8
-EXTRA_CHAR_OFFSET : int = 30
-circleID : String
-circleType : int
-name : String
-description : String
-memberList : List<User>
-owner : User
-radius : float
+Circle(circleID : String, name : String, description : String, circleType : int, memberList : List<User>, owner : User, radius : float)
+Circle(circleID : String, name : String)
+toString() : String

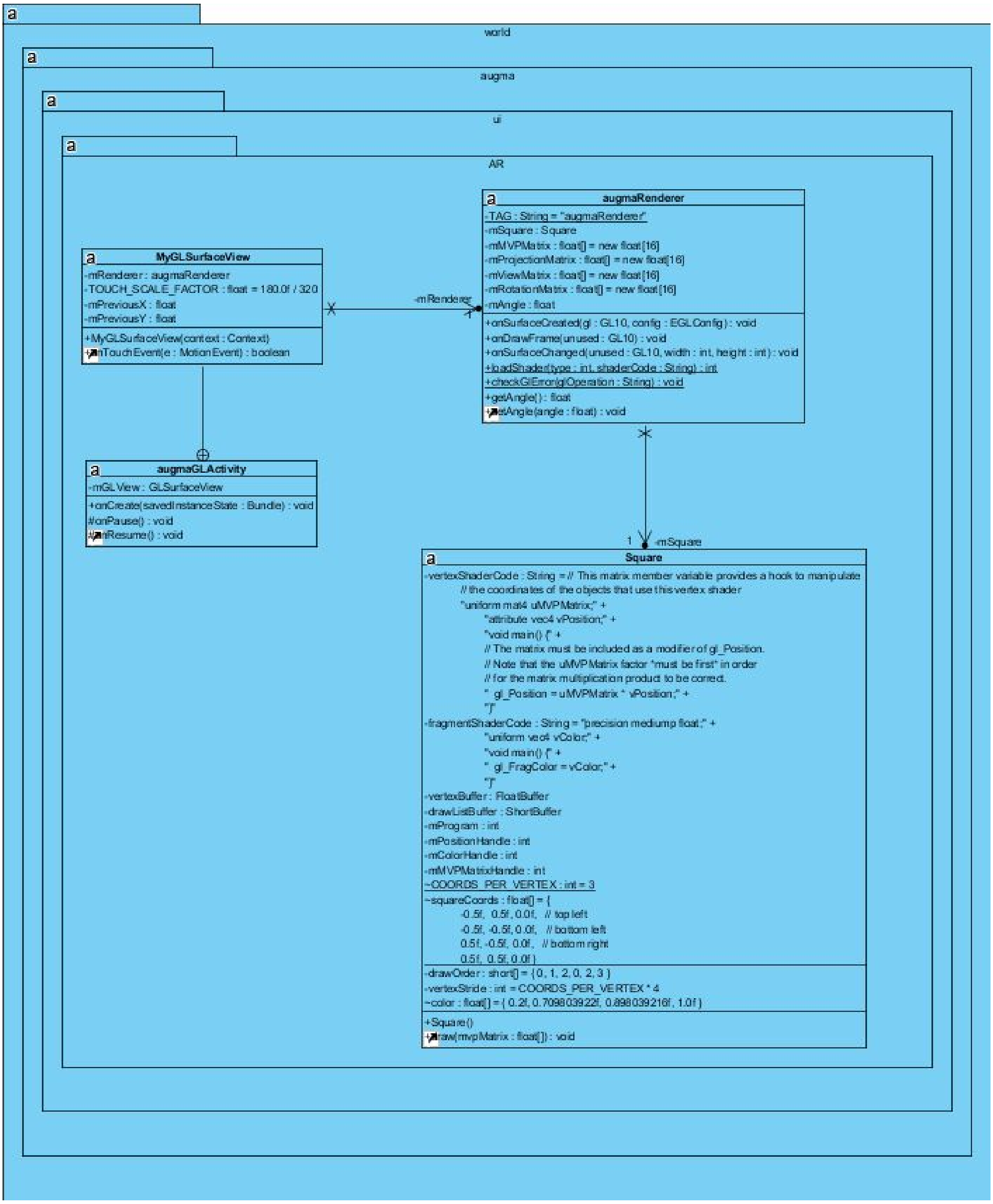

world
augma
asset
Cartesian
-x : double
-y : double
+Cartesian()
+Cartesian(x : double, y : double)
+Cartesian(x1 : double, y1 : double, x2 : double, y2 : double)
+length() : double
+add(vector : Cartesian) : void
+subtract(vector : Cartesian) : void
+normalize() : void
+scale(factor : double) : void
+resize(length : double) : void
+convertToPolar() : Polar
CPAVector
Polar
-radius : double
-theta : double
+Polar()
+Polar(radius : double, theta : double)
+length() : double
+add(vector : Polar) : void
+subtract(vector : Polar) : void
+normalize() : void
+scale(factor : double) : void
+resize(length : double) : void
+convertToCartesian() : Cartesian

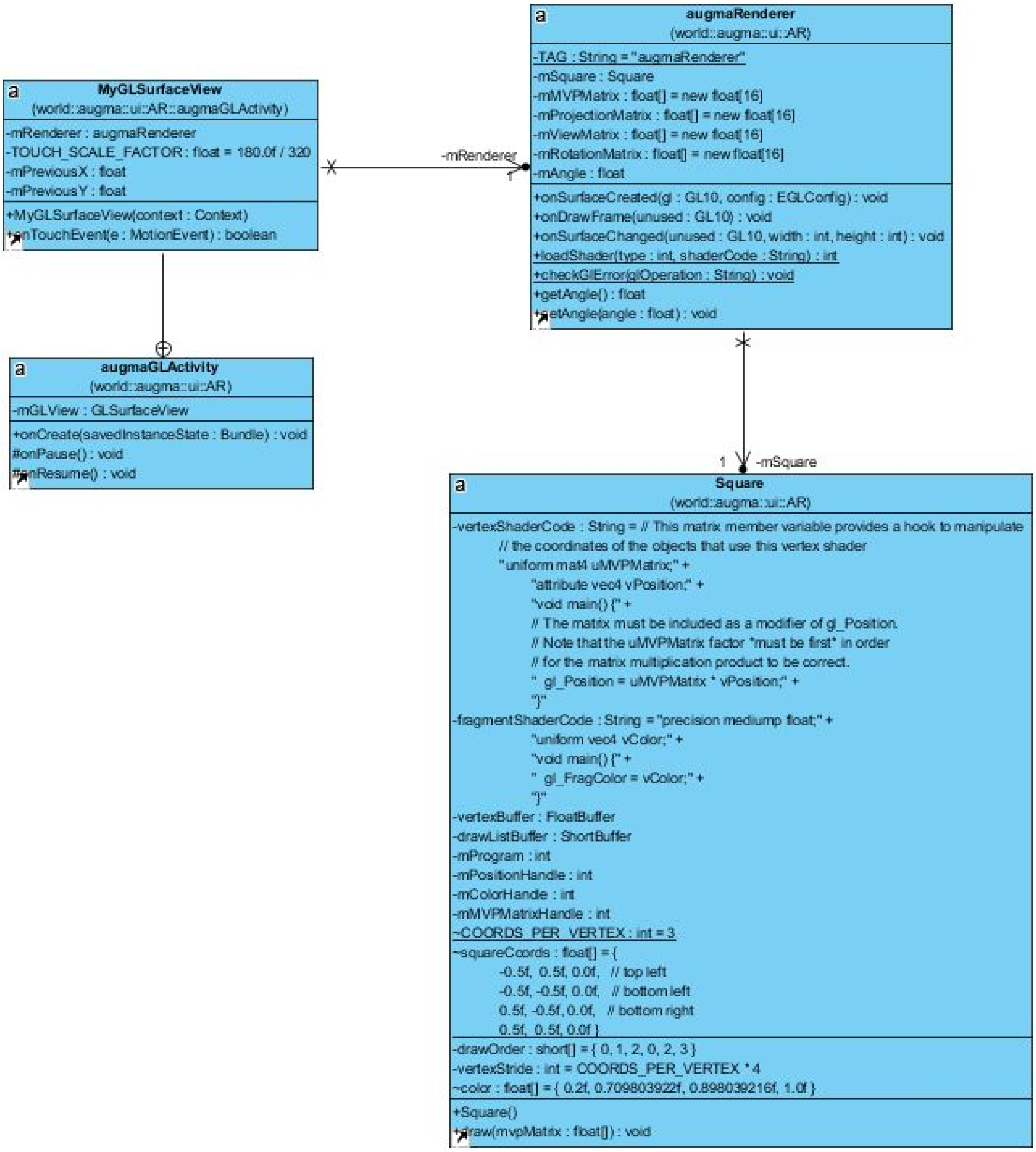

world
augma
asset
Invitation
-senderID : String
-senderName : String
-circleID : String
-circleName : String
+Invitation(senderID : String, senderName : String, circleID : String, circleName : String)

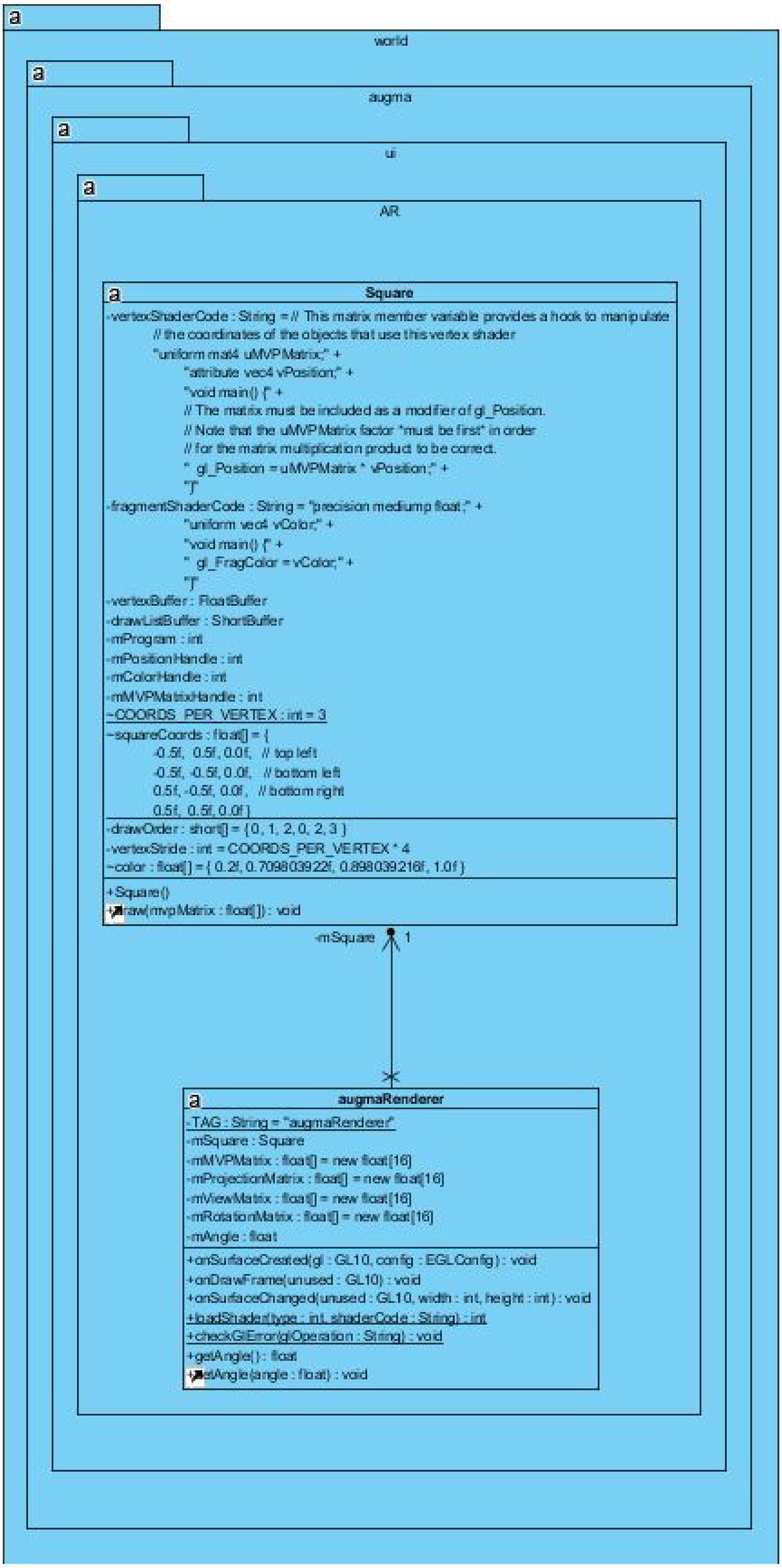

a
world
a
augma
a
asset
a
Circle
-MIN_SIZE : float = 200.0f
-MAX_SIZE : float = 500.0f
-MAX_CHARS_FOR_MIN_SIZE : int = 8
-EXTRA_CHAR_OFFSET : int = 30
-circleID : String
-circleType : int
-name : String
-description : String
-memberList : List<User>
-owner : User
-radius : float
+Circle(circleID : String, name : String, description : String, circleType : int, memberList : List<User>, owner : User, radius : float)
+Circle(circleID : String, name : String)
+toString() : String
-circleList
*
a
Note
-noteID : String
-circleList : Circle
-longitude : float
-latitude : float
-owner : User
-type : int
-rating : int
-noteText : String
-beacon : int
+Note(noteID : String, circleList : List<Circle>, longitude : float, latitude : float, owner : User, type : int, rating : int, beacon : int, noteText : String)
+Note(noteID : String, rating : int, beacon : int)
+setLongitudeFromLatLng(noteLoc : LatLng) : void
+setLatitudeFromLatLng(noteLoc : LatLng) : void
+getCircleList() : List<Circle>
+setCircleList(circleList : List<Circle>) : void
+getSuperRate() : int
+setSuperRate(superRate : int) : void


world
augma
asset
Invitation
-senderID : String
-senderName : String
-circleID : String
-circleName : String
+Invitation(senderID : String, senderName : String, circleID : String, circleName : String)
-invitations
*
User
-userID : String
-username : String
-bio : String
-email : String
-name : String
-password : String
-profilePic : String
-birthdate : String
-type : int
-rating : int
-memberships : Circle
-invitations : Invitation
-ownedNotes : Note
-ownedCircles : Circle
+User(userID : String, username : String, bio : String, email : String, name : String, password : String, profilePic : String, birthdate : String, type : int, memberships : List<Circle>, invitations : List<Invitation>, ownedNotes : List<Note>, ownedCircles : List<Circle>, rating : int)
+toString() : String
+getMemberships() : List<Circle>
+setMemberships(memberships : List<Circle>) : void
+getInvitations() : List<Invitation>
+setInvitations(invitations : List<Invitation>) : void
+getOwnedNotes() : List<Note>
+setOwnedNotes(ownedNotes : List<Note>) : void
+getOwnedCircles() : List<Circle>
+setOwnedCircles(ownedCircles : List<Circle>) : void
-memberList
-owner
1
-ownedNotes
-owner
Note
-noteID : String
-circleList : Circle
-longitude : float
-latitude : float
-owner : User
-type : int
-rating : int
-noteText : String
-beacon : int
+Note(noteID : String, circleList : List<Circle>, longitude : float, latitude : float, owner : User, type : int, rating : int, beacon : int, noteText : String)
+Note(noteID : String, rating : int, beacon : int)
+setLongitudeFromLatLng(noteLoc : LatLng) : void
+setLatitudeFromLatLng(noteLoc : LatLng) : void
+getSuperRate() : int
+setSuperRate(superRate : int) : void
+getCircleList() : List<Circle>
+setCircleList(circleList : List<Circle>) : void
*
-circleList
Circle
-MIN_SIZE : float = 200.0f
-MAX_SIZE : float = 800.0f
-MAX_CHARS_FOR_MIN_SIZE : int = 8
-EXTRA_CHAR_OFFSET : int = 30
-circleID : String
-circleType : int
-name : String
-description : String
-memberList : User
-owner : User
-radius : float
+Circle(circleID : String, name : String, description : String, circleType : int, memberList : List<User>, owner : User, radius : float)
+Circle(circleID : String, name : String)
+getMemberList() : List<User>
+setMemberList(memberList : List<User>) : void
+toString() : String
-ownedCircles
-memberships

Overview of AR Package

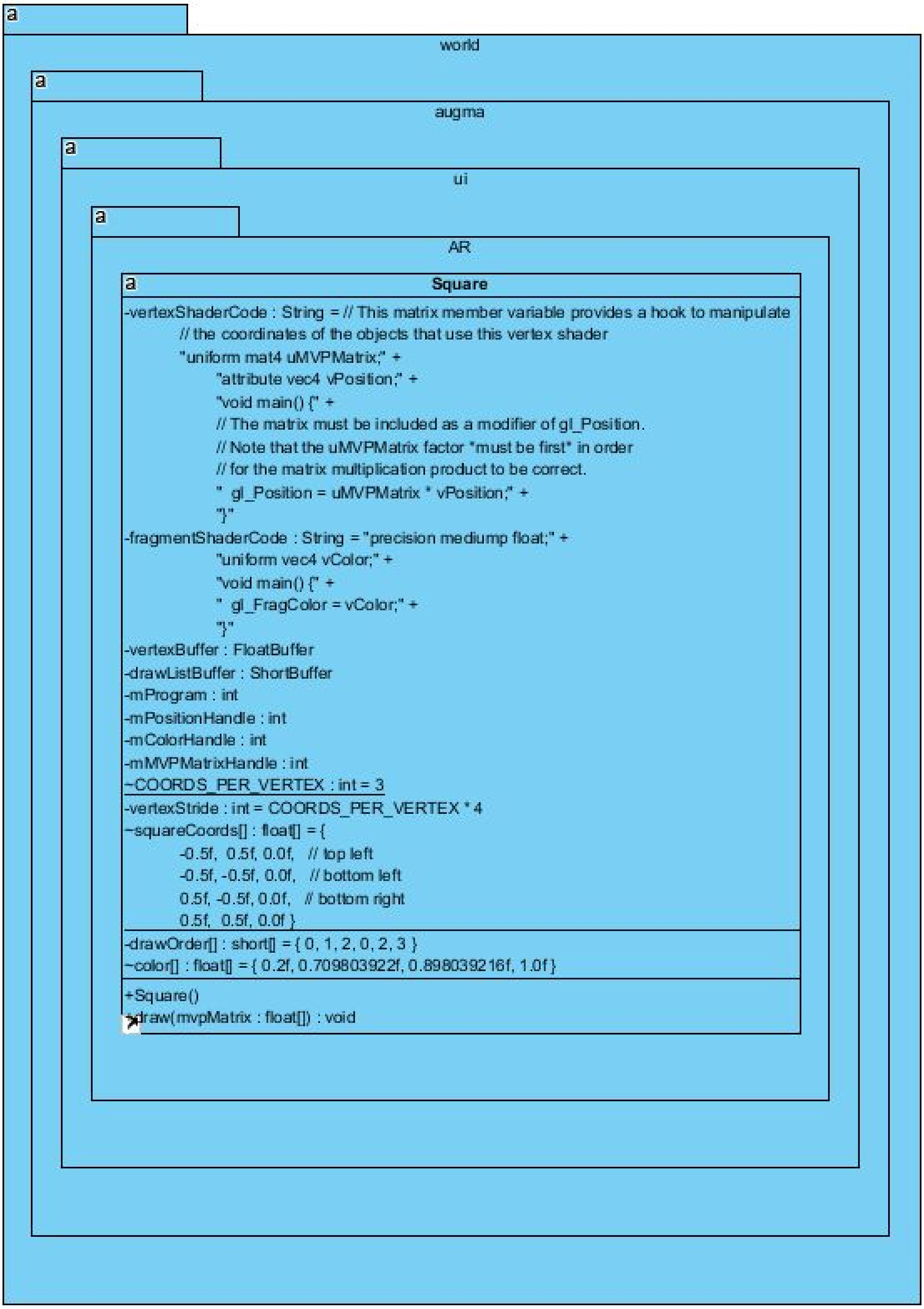

world
augma
ui
AR
MyGLSurfaceView
-mRenderer : augmaRenderer
-TOUCH_SCALE_FACTOR : float = 180.0f / 320
-mPreviousX : float
-mPreviousY : float
+MyGLSurfaceView(context : Context)
+nTouchEvent(e : MotionEvent) : boolean
-mRenderer
augmaRenderer
-TAG : String = "augmaRenderer"
-mSquare : Square
-mMVPMatrix : float[] = new float[16]
-mProjectionMatrix : float[] = new float[16]
-mViewMatrix : float[] = new float[16]
-mRotationMatrix : float[] = new float[16]
-mAngle : float
+onSurfaceCreated(gl : GL10, config : EGLConfig) : void
+onDrawFrame(unused : GL10) : void
+onSurfaceChanged(unused : GL10, width : int, height : int) : void
+loadShader(type : int, shaderCode : String) : int
+checkGlError(glOperation : String) : void
+getAngle() : float
+etAngle(angle : float) : void
augmaGLActivity
-mGLView : GLSurfaceView
+onCreate(savedInstanceState : Bundle) : void
#onPause() : void
#nResume() : void
1
-mSquare
Square
-vertexShaderCode : String = // This matrix member variable provides a hook to manipulate
// the coordinates of the objects that use this vertex shader
"uniform mat4 uMVPMatrix;" +
"attribute vec4 vPosition;" +
"void main() {" +
// The matrix must be included as a modifier of gl_Position.
// Note that the uMVPMatrix factor *must be first* in order
// for the matrix multiplication product to be correct.
" gl_Position = uMVPMatrix * vPosition;" +
"}"
-fragmentShaderCode : String = "precision mediump float;" +
"uniform vec4 vColor;" +
"void main() {" +
" gl_FragColor = vColor;" +
"}"
-vertexBuffer : FloatBuffer
-drawListBuffer : ShortBuffer
-mProgram : int
-mPositionHandle : int
-mColorHandle : int
-mMVPMatrixHandle : int
~COORDS_PER_VERTEX : int = 3
~squareCoords : float[] = {
-0.5f, 0.5f, 0.0f,   // top left
-0.5f, -0.5f, 0.0f,   // bottom left
0.5f, -0.5f, 0.0f,   // bottom right
0.5f, 0.5f, 0.0f }
-drawOrder : short[] = { 0, 1, 2, 0, 2, 3 }
-vertexStride : int = COORDS_PER_VERTEX * 4
~color : float[] = { 0.2f, 0.709803922f, 0.898039216f, 1.0f }
+Square()
+raw(mvpMatrix : float[]) : void


a
MyGLSurfaceView
(world::augma::ui::AR::augmaGLActivity)
-mRenderer : augmaRenderer
-TOUCH_SCALE_FACTOR : float = 180.0f / 320
-mPreviousX : float
-mPreviousY : float
+MyGLSurfaceView(context : Context)
+onTouchEvent(e : MotionEvent) : boolean
-mRenderer
1
a
augmaGLActivity
(world::augma::ui::AR)
-mGLView : GLSurfaceView
+onCreate(savedInstanceState : Bundle) : void
#onPause() : void
#onResume() : void
a
augmaRenderer
(world::augma::ui::AR)
-TAG : String = "augmaRenderer"
-mSquare : Square
-mMVPMatrix : float[] = new float[16]
-mProjectionMatrix : float[] = new float[16]
-mViewMatrix : float[] = new float[16]
-mRotationMatrix : float[] = new float[16]
-mAngle : float
+onSurfaceCreated(gl : GL10, config : EGLConfig) : void
+onDrawFrame(unused : GL10) : void
+onSurfaceChanged(unused : GL10, width : int, height : int) : void
+loadShader(type : int, shaderCode : String) : int
+checkGlError(glOperation : String) : void
+getAngle() : float
+setAngle(angle : float) : void
1
-mSquare
a
Square
(world::augma::ui::AR)
-vertexShaderCode : String = // This matrix member variable provides a hook to manipulate
// the coordinates of the objects that use this vertex shader
"uniform mat4 uMVPMatrix;" +
"attribute vec4 vPosition;" +
"void main() {" +
// The matrix must be included as a modifier of gl_Position.
// Note that the uMVPMatrix factor *must be first* in order
// for the matrix multiplication product to be correct.
" gl_Position = uMVPMatrix * vPosition;" +
"}"
-fragmentShaderCode : String = "precision mediump float;" +
"uniform vec4 vColor;" +
"void main() {" +
" gl_FragColor = vColor;" +
"}"
-vertexBuffer : FloatBuffer
-drawListBuffer : ShortBuffer
-mProgram : int
-mPositionHandle : int
-mColorHandle : int
-mMVPMatrixHandle : int
~COORDS_PER_VERTEX : int = 3
~squareCoords : float[] = {
-0.5f, 0.5f, 0.0f,   // top left
-0.5f, -0.5f, 0.0f,   // bottom left
0.5f, -0.5f, 0.0f,   // bottom right
0.5f, 0.5f, 0.0f }
-drawOrder : short[] = { 0, 1, 2, 0, 2, 3 }
-vertexStride : int = COORDS_PER_VERTEX * 4
~color : float[] = { 0.2f, 0.709803922f, 0.898039216f, 1.0f }
+Square()
+draw(mvpMatrix : float[]) : void



```
Square
-vertexShaderCode : String = // This matrix member variable provides a hook to manipulate
          // the coordinates of the objects that use this vertex shader
          "uniform mat4 uMVPMatrix;" +
                "attribute vec4 vPosition;" +
                "void main() {" +
                // The matrix must be included as a modifier of gl_Position.
                // Note that the uMVPMatrix factor *must be first* in order
                // for the matrix multiplication product to be correct.
                "  gl_Position = uMVPMatrix * vPosition;" +
                "}"
-fragmentShaderCode : String = "precision mediump float;" +
                "uniform vec4 vColor;" +
                "void main() {" +
                "  gl_FragColor = vColor;" +
                "}"
-vertexBuffer : FloatBuffer
-drawListBuffer : ShortBuffer
-mProgram : int
-mPositionHandle : int
-mColorHandle : int
-mMVPMatrixHandle : int
~COORDS_PER_VERTEX : int = 3
~squareCoords : float[] = {
          -0.5f,  0.5f, 0.0f,   // top left
          -0.5f, -0.5f, 0.0f,   // bottom left
          0.5f, -0.5f, 0.0f,   // bottom right
          0.5f,  0.5f, 0.0f }
-drawOrder : short[] = { 0, 1, 2, 0, 2, 3 }
-vertexStride : int = COORDS_PER_VERTEX * 4
~color : float[] = { 0.2f, 0.709803922f, 0.898039216f, 1.0f }
+Square()
+raw(mvpMatrix : float[]) : void
```

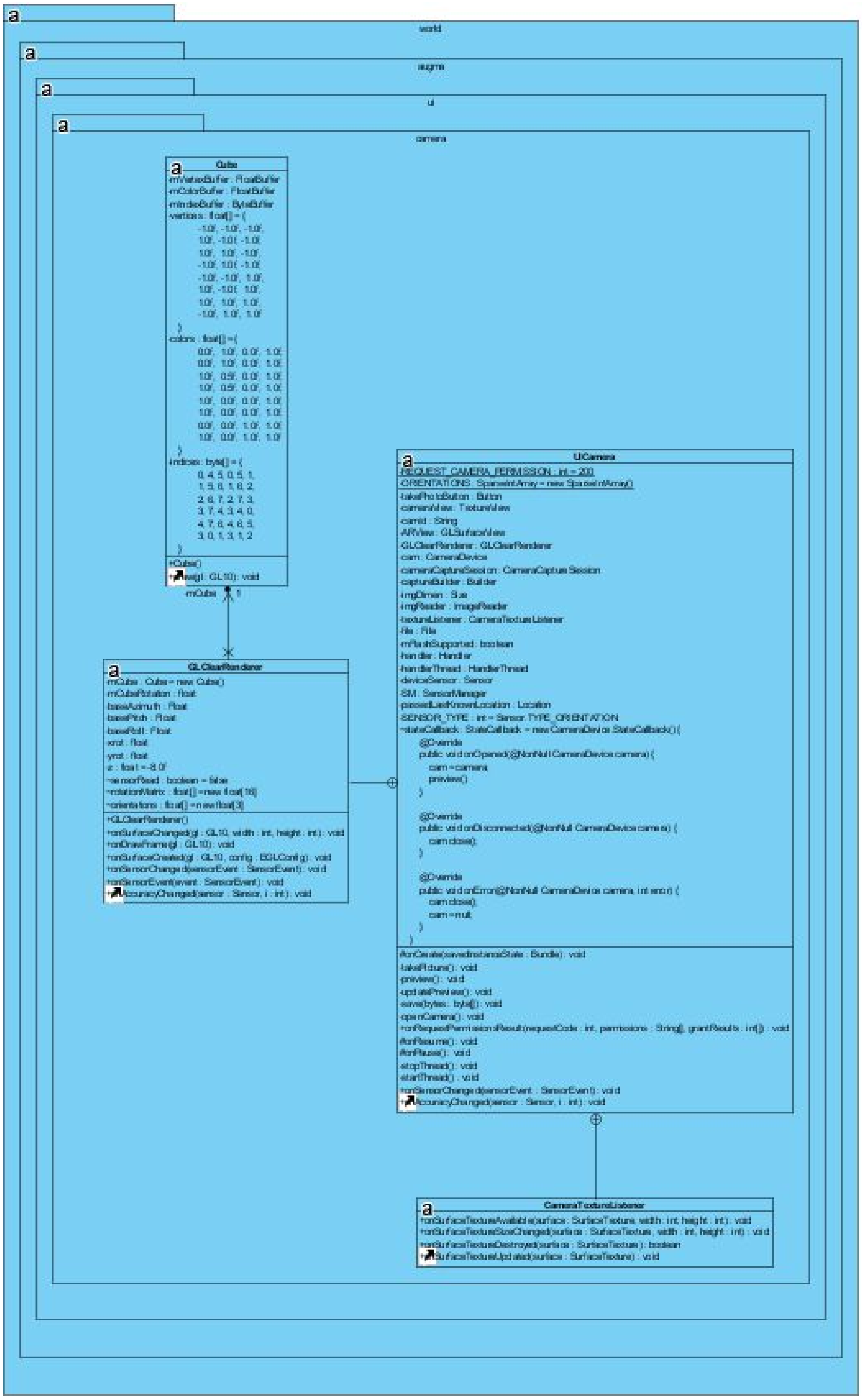
```
augmaRenderer
-TAG : String = "augmaRenderer"
-mSquare : Square
-mMVPMatrix : float[] = new float[16]
-mProjectionMatrix : float[] = new float[16]
-mViewMatrix : float[] = new float[16]
-mRotationMatrix : float[] = new float[16]
-mAngle : float
+onSurfaceCreated(gl : GL10, config : EGLConfig) : void
+onDrawFrame(unused : GL10) : void
+onSurfaceChanged(unused : GL10, width : int, height : int) : void
+loadShader(type : int, shaderCode : String) : int
+checkGlError(glOperation : String) : void
+getAngle() : float
+etAngle(angle : float) : void
```

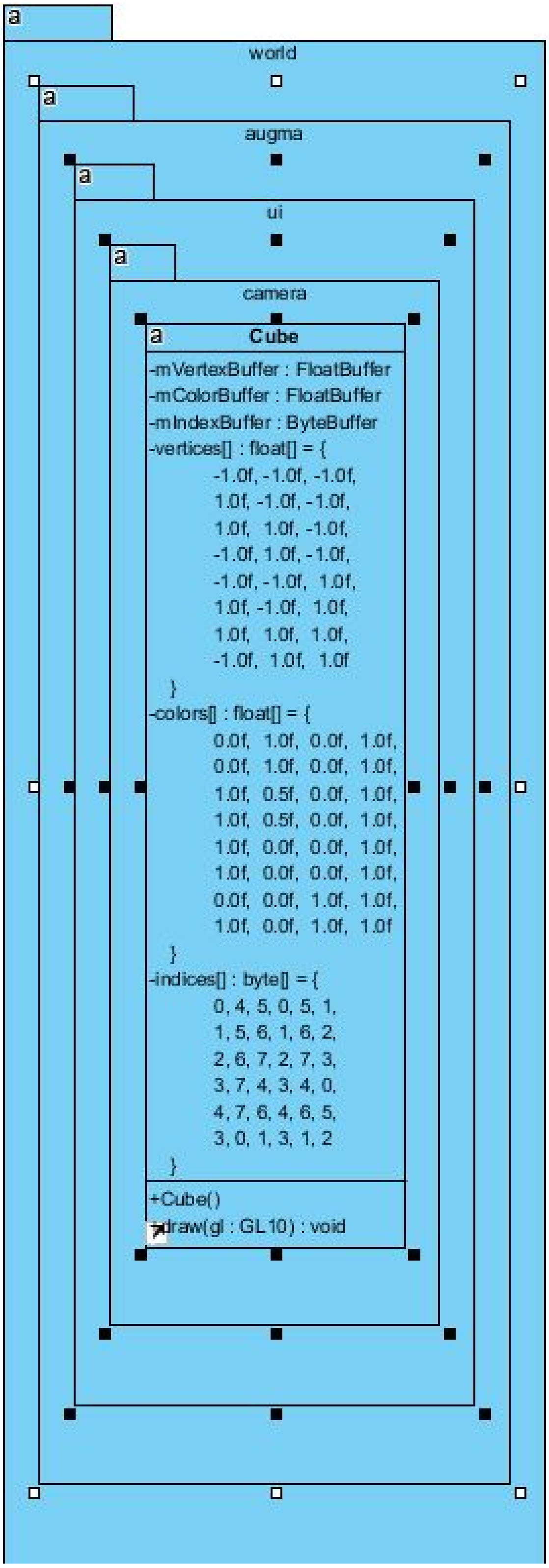


Square

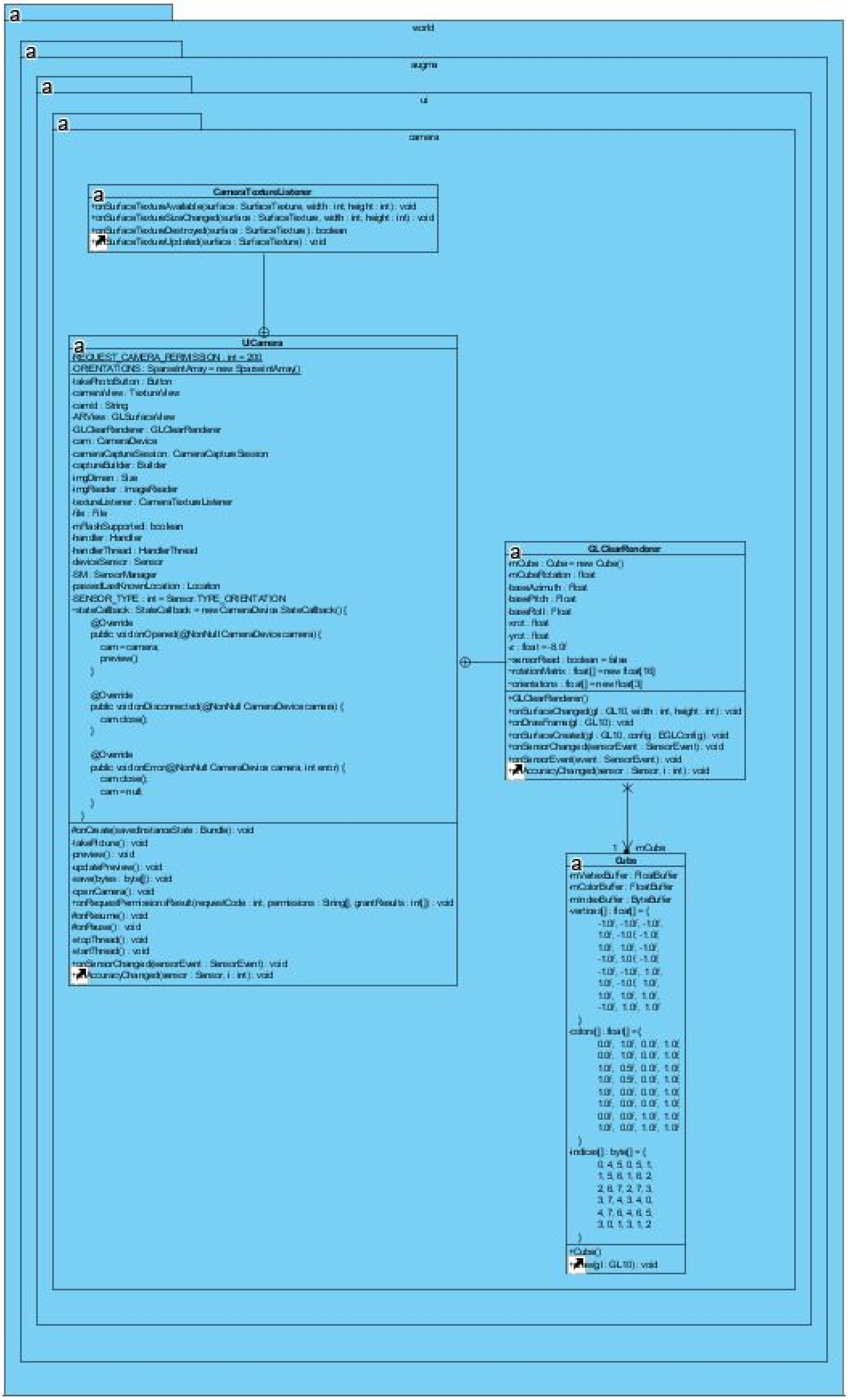
```
-vertexShaderCode : String = // This matrix member variable provides a hook to manipulate
        // the coordinates of the objects that use this vertex shader
        "uniform mat4 uMVPMatrix;" +
              "attribute vec4 vPosition;" +
              "void main() {" +
              // The matrix must be included as a modifier of gl_Position.
              // Note that the uMVPMatrix factor *must be first* in order
              // for the matrix multiplication product to be correct.
              "  gl_Position = uMVPMatrix * vPosition;" +
              "}"
-fragmentShaderCode : String = "precision mediump float;" +
              "uniform vec4 vColor;" +
              "void main() {" +
              "  gl_FragColor = vColor;" +
              "}"
-vertexBuffer : FloatBuffer
-drawListBuffer : ShortBuffer
-mProgram : int
-mPositionHandle : int
-mColorHandle : int
-mMVPMatrixHandle : int
~COORDS_PER_VERTEX : int = 3
-vertexStride : int = COORDS_PER_VERTEX * 4
~squareCoords[] : float[] = {
        -0.5f,  0.5f, 0.0f,   // top left
        -0.5f, -0.5f, 0.0f,   // bottom left
        0.5f, -0.5f, 0.0f,   // bottom right
        0.5f,  0.5f, 0.0f }
-drawOrder[] : short[] = { 0, 1, 2, 0, 2, 3 }
~color[] : float[] = { 0.2f, 0.709803922f, 0.898039216f, 1.0f }
+Square()
+draw(mvpMatrix : float[]) : void
```

Overview of Camera Package

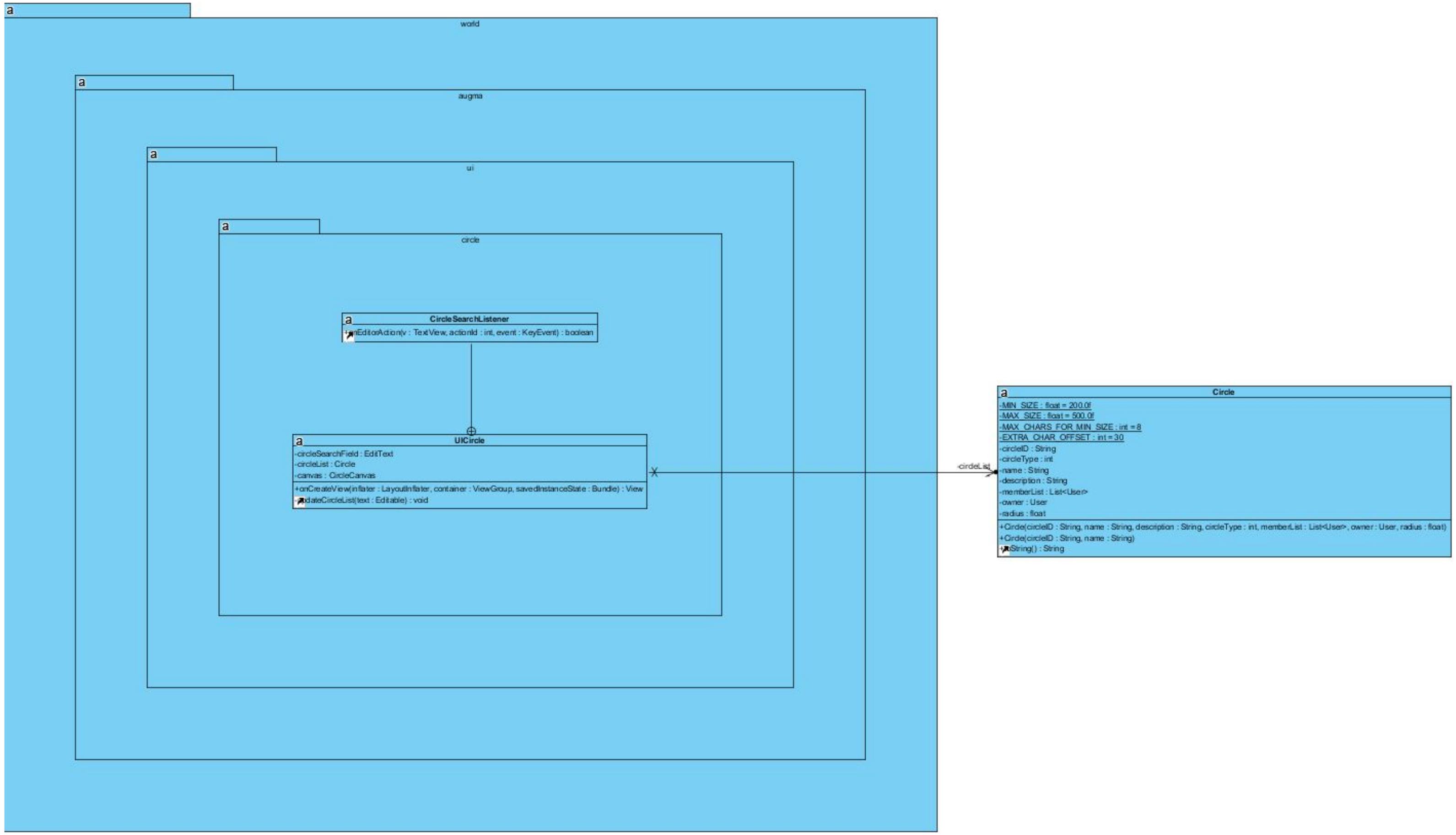

world
augmr
ui
camera
Cube
-mVertexBuffer : FloatBuffer
-mColorBuffer : FloatBuffer
-mIndexBuffer : ByteBuffer
-vertices : float[] = {
-1.0f, -1.0f, -1.0f,
1.0f, -1.0f, -1.0f,
1.0f, 1.0f, -1.0f,
-1.0f, 1.0f, -1.0f,
-1.0f, -1.0f, 1.0f,
1.0f, -1.0f, 1.0f,
1.0f, 1.0f, 1.0f,
-1.0f, 1.0f, 1.0f
}
-colors : float[] = {
0.0f, 1.0f, 0.0f, 1.0f,
0.0f, 1.0f, 0.0f, 1.0f,
1.0f, 0.5f, 0.0f, 1.0f,
1.0f, 0.5f, 0.0f, 1.0f,
1.0f, 0.0f, 0.0f, 1.0f,
1.0f, 0.0f, 0.0f, 1.0f,
0.0f, 0.0f, 1.0f, 1.0f,
1.0f, 0.0f, 1.0f, 1.0f
}
-indices : byte[] = {
0, 4, 5, 0, 5, 1,
1, 5, 6, 1, 6, 2,
2, 6, 7, 2, 7, 3,
3, 7, 4, 3, 4, 0,
4, 7, 6, 4, 6, 5,
3, 0, 1, 3, 1, 2
}
+Cube()
+draw(gl : GL10): void
mCube
1
GLClearRenderer
-mCube : Cube = new Cube()
-mCubeRotation : float
-baseAzimuth : Float
-basePitch : Float
-baseRoll : Float
-xrot : float
-yrot : float
-z : float = -8.0f
~sensorRead : boolean = false
~rotationMatrix : float[] = new float[16]
~orientations : float[] = new float[3]
+GLClearRenderer()
+onSurfaceChanged(gl : GL10, width : int, height : int) : void
+onDrawFrame(gl : GL10) : void
+onSurfaceCreated(gl : GL10, config : EGLConfig) : void
+onSensorChanged(sensorEvent : SensorEvent) : void
+onSensorEvent(event : SensorEvent) : void
+onAccuracyChanged(sensor : Sensor, i : int) : void
UCamera
-REQUEST_CAMERA_PERMISSION : int = 200
-ORIENTATIONS : SparseIntArray = new SparseIntArray()
-takePhotoButton : Button
-cameraView : TextureView
-camId : String
-ARView : GLSurfaceView
-GLClearRenderer : GLClearRenderer
-cam : CameraDevice
-cameraCaptureSession : CameraCaptureSession
-captureBuilder : Builder
-imgDimen : Size
-imgReader : ImageReader
-textureListener : CameraTextureListener
-file : File
-mFlashSupported : boolean
-handler : Handler
-handlerThread : HandlerThread
-deviceSensor : Sensor
-SM : SensorManager
-passedLastKnownLocation : Location
-SENSOR_TYPE : int = Sensor.TYPE_ORIENTATION
~stateCallback : StateCallback = new CameraDevice.StateCallback() {
@Override
public void onOpened(@NonNull CameraDevice camera) {
cam = camera;
preview();
}
@Override
public void onDisconnected(@NonNull CameraDevice camera) {
cam.close();
}
@Override
public void onError(@NonNull CameraDevice camera, int error) {
cam.close();
cam = null;
}
}
#onCreate(savedInstanceState : Bundle) : void
-takePicture() : void
-preview() : void
-updatePreview() : void
-save(bytes : byte[]) : void
-openCamera() : void
+onRequestPermissionsResult(requestCode : int, permissions : String[], grantResults : int[]) : void
#onResume() : void
#onPause() : void
-stopThread() : void
-startThread() : void
+onSensorChanged(sensorEvent : SensorEvent) : void
+onAccuracyChanged(sensor : Sensor, i : int) : void
CameraTextureListener
+onSurfaceTextureAvailable(surface : SurfaceTexture, width : int, height : int) : void
+onSurfaceTextureSizeChanged(surface : SurfaceTexture, width : int, height : int) : void
+onSurfaceTextureDestroyed(surface : SurfaceTexture) : boolean
+onSurfaceTextureUpdated(surface : SurfaceTexture) : void



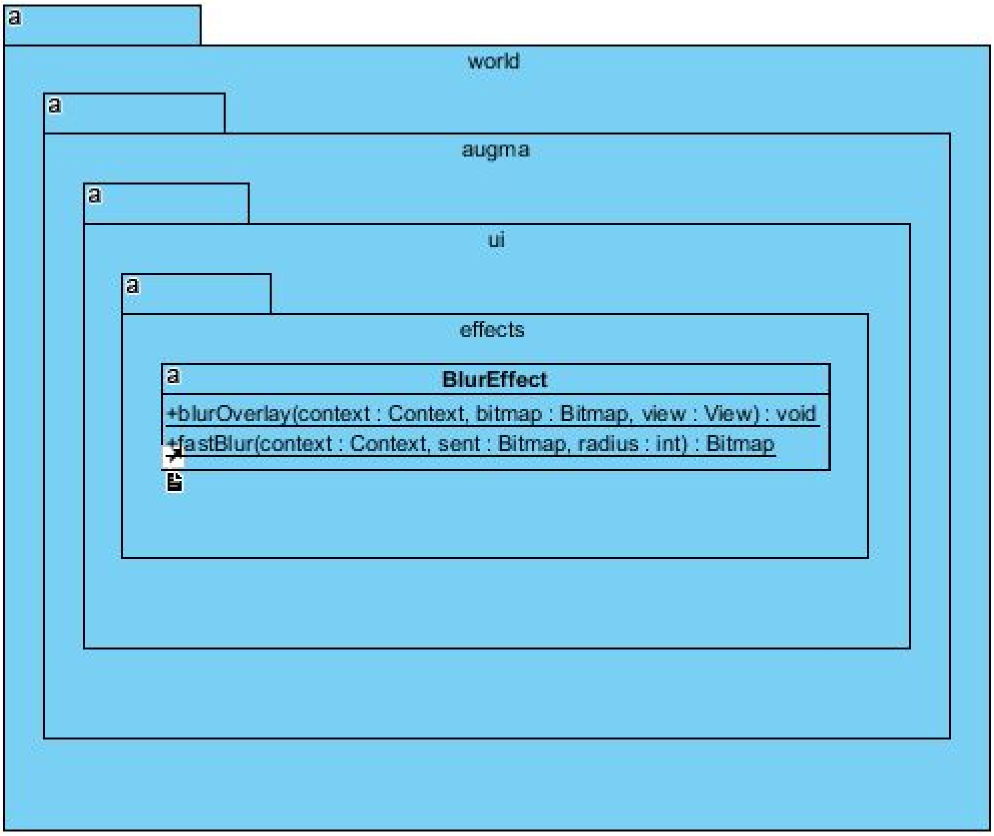
```
-mVertexBuffer : FloatBuffer
-mColorBuffer : FloatBuffer
-mIndexBuffer : ByteBuffer
-vertices[] : float[] = {
            -1.0f, -1.0f, -1.0f,
            1.0f, -1.0f, -1.0f,
            1.0f,  1.0f, -1.0f,
            -1.0f, 1.0f, -1.0f,
            -1.0f, -1.0f,  1.0f,
            1.0f, -1.0f,  1.0f,
            1.0f,  1.0f,  1.0f,
            -1.0f,  1.0f,  1.0f
    }
-colors[] : float[] = {
            0.0f,  1.0f,  0.0f,  1.0f,
            0.0f,  1.0f,  0.0f,  1.0f,
            1.0f,  0.5f,  0.0f,  1.0f,
            1.0f,  0.5f,  0.0f,  1.0f,
            1.0f,  0.0f,  0.0f,  1.0f,
            1.0f,  0.0f,  0.0f,  1.0f,
            0.0f,  0.0f,  1.0f,  1.0f,
            1.0f,  0.0f,  1.0f,  1.0f
    }
-indices[] : byte[] = {
            0, 4, 5, 0, 5, 1,
            1, 5, 6, 1, 6, 2,
            2, 6, 7, 2, 7, 3,
            3, 7, 4, 3, 4, 0,
            4, 7, 6, 4, 6, 5,
            3, 0, 1, 3, 1, 2
    }
+Cube()
+draw(gl : GL10) : void
```


world
augme
ui
camera
CameraTextureListener
+onSurfaceTextureAvailable(surface : SurfaceTexture, width : int, height : int) : void
+onSurfaceTextureSizeChanged(surface : SurfaceTexture, width : int, height : int) : void
+onSurfaceTextureDestroyed(surface : SurfaceTexture) : boolean
+onSurfaceTextureUpdated(surface : SurfaceTexture) : void
UCamera
-REQUEST_CAMERA_PERMISSION : int = 200
-ORIENTATIONS : SparseIntArray = new SparseIntArray()
-takePhotoButton : Button
-cameraView : TextureView
-camId : String
-ARView : GLSurfaceView
-GLClearRenderer : GLClearRenderer
-cam : CameraDevice
-cameraCaptureSession : CameraCaptureSession
-captureBuilder : Builder
-imgDimen : Size
-imgReader : ImageReader
-textureListener : CameraTextureListener
-file : File
-mFlashSupported : boolean
-handler : Handler
-handlerThread : HandlerThread
-deviceSensor : Sensor
-SM : SensorManager
-passedLastKnownLocation : Location
-SENSOR_TYPE : int = Sensor.TYPE_ORIENTATION
~stateCallback : StateCallback = new CameraDevice.StateCallback() {
@Override
public void onOpened(@NonNull CameraDevice camera) {
cam = camera;
preview();
}
@Override
public void onDisconnected(@NonNull CameraDevice camera) {
cam.close();
}
@Override
public void onError(@NonNull CameraDevice camera, int error) {
cam.close();
cam = null;
}
}
#onCreate(savedInstanceState : Bundle) : void
-takePicture() : void
-preview() : void
-updatePreview() : void
-save(bytes : byte[]) : void
-openCamera() : void
+onRequestPermissionsResult(requestCode : int, permissions : String[], grantResults : int[]) : void
#onResume() : void
#onPause() : void
-stopThread() : void
-startThread() : void
+onSensorChanged(sensorEvent : SensorEvent) : void
+onAccuracyChanged(sensor : Sensor, i : int) : void
GLClearRenderer
-mCube : Cube = new Cube()
-mCubeRotation : float
-baseAzimuth : Float
-basePitch : Float
-baseRoll : Float
-xrot : float
-yrot : float
-z : float = -8.0f
~sensorRead : boolean = false
~rotationMatrix : float[] = new float[16]
~orientations : float[] = new float[3]
+GLClearRenderer()
+onSurfaceChanged(gl : GL10, width : int, height : int) : void
+onDrawFrame(gl : GL10) : void
+onSurfaceCreated(gl : GL10, config : EGLConfig) : void
+onSensorChanged(sensorEvent : SensorEvent) : void
+onSensorEvent(event : SensorEvent) : void
+onAccuracyChanged(sensor : Sensor, i : int) : void
1
mCube
Cube
-mVertexBuffer : FloatBuffer
-mColorBuffer : FloatBuffer
-mIndexBuffer : ByteBuffer
-vertices[] : float[] = {
-1.0f, -1.0f, -1.0f,
1.0f, -1.0f, -1.0f,
1.0f, 1.0f, -1.0f,
-1.0f, 1.0f, -1.0f,
-1.0f, -1.0f, 1.0f,
1.0f, -1.0f, 1.0f,
1.0f, 1.0f, 1.0f,
-1.0f, 1.0f, 1.0f
}
-colors[] : float[] = {
0.0f, 1.0f, 0.0f, 1.0f,
0.0f, 1.0f, 0.0f, 1.0f,
1.0f, 0.5f, 0.0f, 1.0f,
1.0f, 0.5f, 0.0f, 1.0f,
1.0f, 0.0f, 0.0f, 1.0f,
1.0f, 0.0f, 0.0f, 1.0f,
0.0f, 0.0f, 1.0f, 1.0f,
1.0f, 0.0f, 1.0f, 1.0f
}
-indices[] : byte[] = {
0, 4, 5, 0, 5, 1,
1, 5, 6, 1, 6, 2,
2, 6, 7, 2, 7, 3,
3, 7, 4, 3, 4, 0,
4, 7, 6, 4, 6, 5,
3, 0, 1, 3, 1, 2
}
+Cube()
+draw(gl : GL10) : void

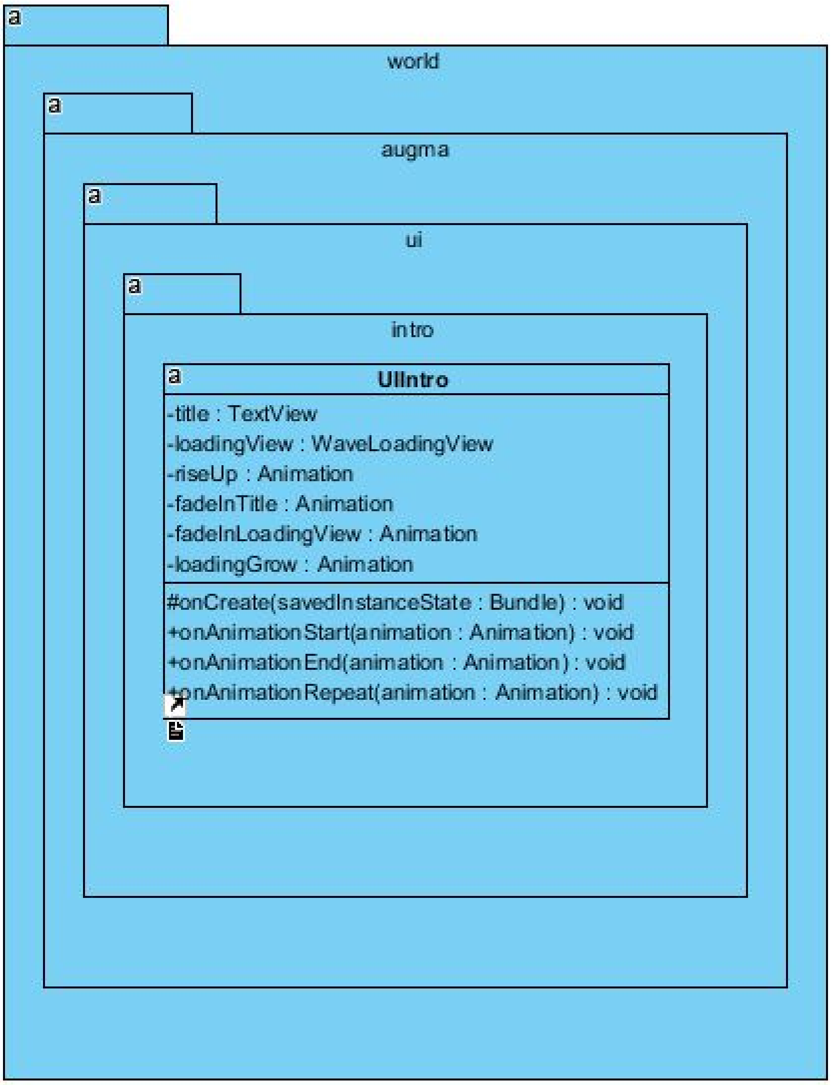
Overview of Circle Package


world
augma
ui
circle
CircleSearchListener
+onEditorAction(v : TextView, actionId : int, event : KeyEvent) : boolean
UICircle
-circleSearchField : EditText
-circleList : Circle
-canvas : CircleCanvas
+onCreateView(inflater : LayoutInflater, container : ViewGroup, savedInstanceState : Bundle) : View
-updateCircleList(text : Editable) : void
-circleList
Circle
-MIN_SIZE : float = 200.0f
-MAX_SIZE : float = 500.0f
-MAX_CHARS_FOR_MIN_SIZE : int = 8
-EXTRA_CHAR_OFFSET : int = 30
-circleID : String
-circleType : int
-name : String
-description : String
-memberList : List<User>
-owner : User
-radius : float
+Circle(circleID : String, name : String, description : String, circleType : int, memberList : List<User>, owner : User, radius : float)
+Circle(circleID : String, name : String)
+toString() : String

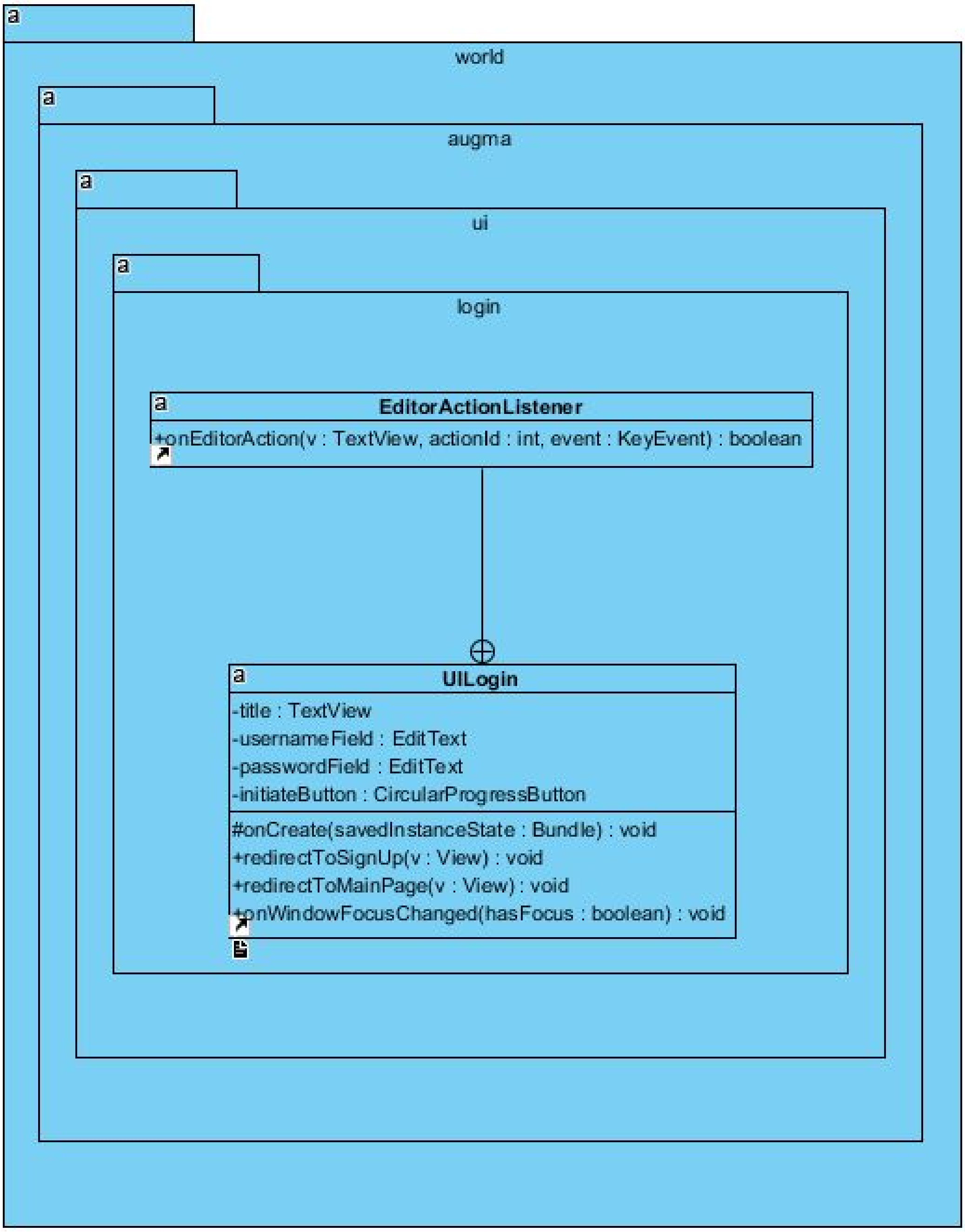
Overview of Effects Package


world
augma
ui
effects
BlurEffect
+blurOverlay(context : Context, bitmap : Bitmap, view : View) : void
+fastBlur(context : Context, sent : Bitmap, radius : int) : Bitmap

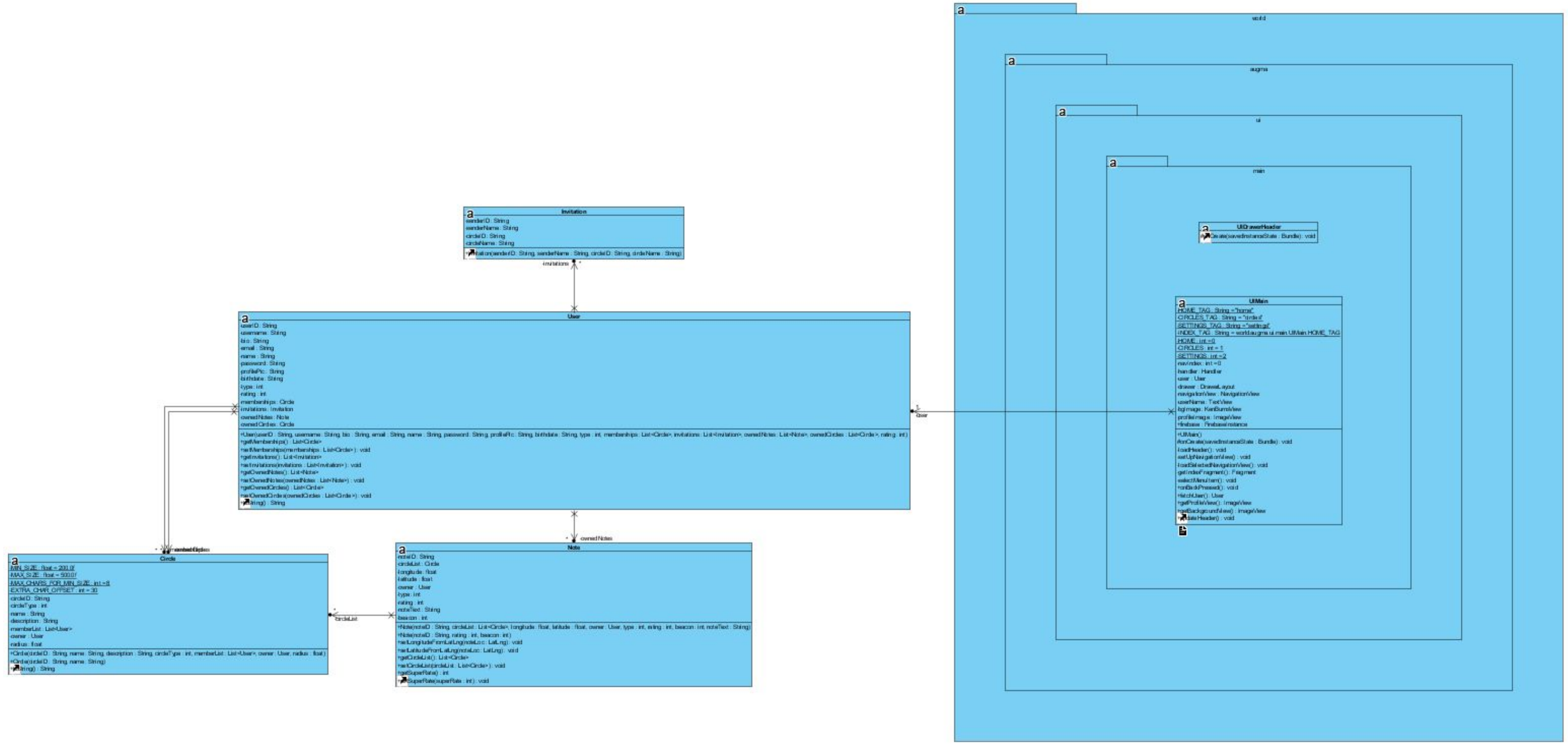
Overview of Intro Package


world
augma
ui
intro
UIIntro
-title : TextView
-loadingView : WaveLoadingView
-riseUp : Animation
-fadeInTitle : Animation
-fadeInLoadingView : Animation
-loadingGrow : Animation
#onCreate(savedInstanceState : Bundle) : void
+onAnimationStart(animation : Animation) : void
+onAnimationEnd(animation : Animation) : void
+onAnimationRepeat(animation : Animation) : void

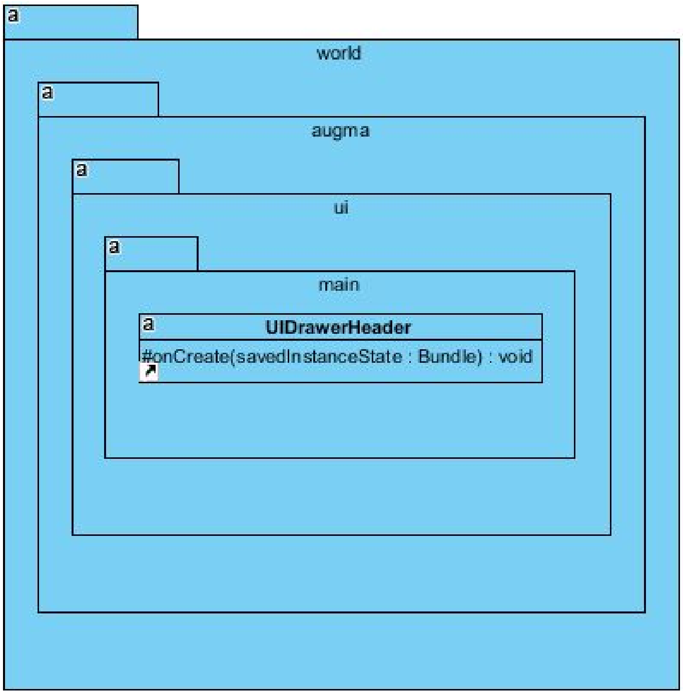
Overview of Login Package

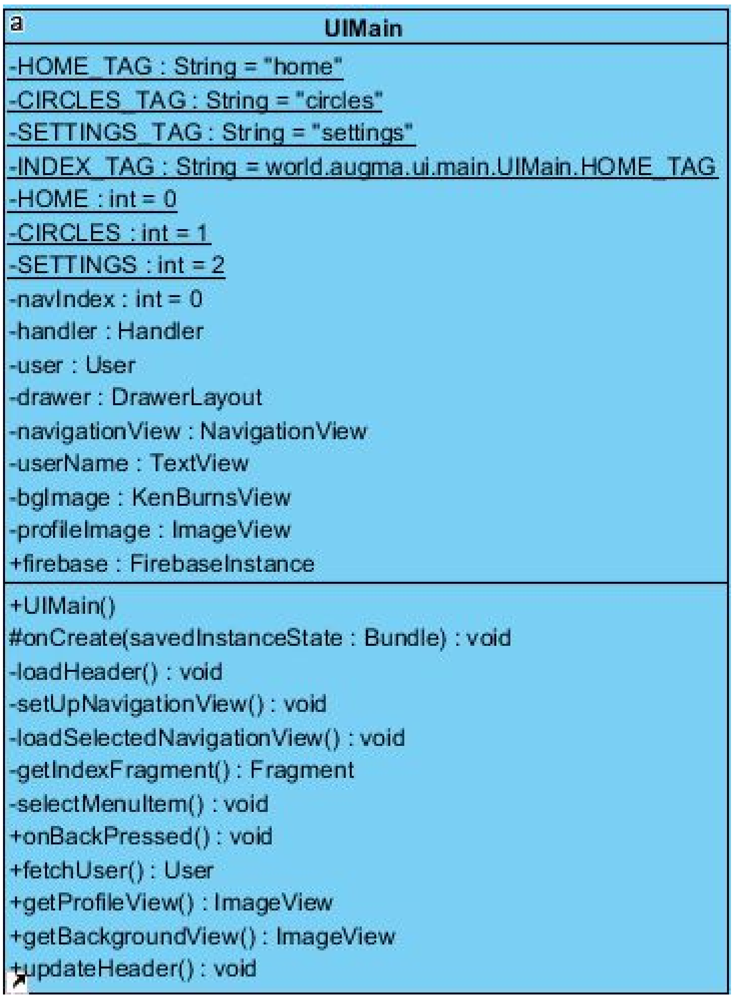

world
augma
ui
login
EditorActionListener
+onEditorAction(v : TextView, actionId : int, event : KeyEvent) : boolean
UILogin
-title : TextView
-usernameField : EditText
-passwordField : EditText
-initiateButton : CircularProgressButton
#onCreate(savedInstanceState : Bundle) : void
+redirectToSignUp(v : View) : void
+redirectToMainPage(v : View) : void
+onWindowFocusChanged(hasFocus : boolean) : void

Overview of Main Package



world
augma
ui
main
Invitation
-senderID : String
-senderName : String
-circleID : String
-circleName : String
+Invitation(senderID : String, senderName : String, circleID : String, circleName : String)
-invitations
*
User
-userID : String
-username : String
-bio : String
-email : String
-name : String
-password : String
-profilePic : String
-birthdate : String
-type : int
-rating : int
-memberships : Circle
-invitations : Invitation
-ownedNotes : Note
-ownedCircles : Circle
+User(userID : String, username : String, bio : String, email : String, name : String, password : String, profilePic : String, birthdate : String, type : int, memberships : List<Circle>, invitations : List<Invitation>, ownedNotes : List<Note>, ownedCircles : List<Circle>, rating : int)
+getMemberships() : List<Circle>
+setMemberships(memberships : List<Circle>) : void
+getInvitations() : List<Invitation>
+setInvitations(invitations : List<Invitation>) : void
+getOwnedNotes() : List<Note>
+setOwnedNotes(ownedNotes : List<Note>) : void
+getOwnedCircles() : List<Circle>
+setOwnedCircles(ownedCircles : List<Circle>) : void
+toString() : String
1
-user
-ownedNotes
*
-memberships
-ownedCircles
Circle
-MIN_SIZE : float = 200.0f
-MAX_SIZE : float = 500.0f
-MAX_CHARS_FOR_MIN_SIZE : int = 8
-EXTRA_CHAR_OFFSET : int = 30
-circleID : String
-circleType : int
-name : String
-description : String
-memberList : List<User>
-owner : User
-radius : float
+Circle(circleID : String, name : String, description : String, circleType : int, memberList : List<User>, owner : User, radius : float)
+Circle(circleID : String, name : String)
+toString() : String
*
-circleList
Note
-noteID : String
-circleList : Circle
-longitude : float
-latitude : float
-owner : User
-type : int
-rating : int
-noteText : String
-beacon : int
+Note(noteID : String, circleList : List<Circle>, longitude : float, latitude : float, owner : User, type : int, rating : int, beacon : int, noteText : String)
+Note(noteID : String, rating : int, beacon : int)
+setLongitudeFromLatLng(noteLoc : LatLng) : void
+setLatitudeFromLatLng(noteLoc : LatLng) : void
+getCircleList() : List<Circle>
+setCircleList(circleList : List<Circle>) : void
+getSuperRate() : int
+setSuperRate(superRate : int) : void
UIDrawerHeader
#onCreate(savedInstanceState : Bundle) : void
UIMain
-HOME_TAG : String = "home"
-CIRCLES_TAG : String = "circles"
-SETTINGS_TAG : String = "settings"
-INDEX_TAG : String = world.augma.ui.main.UIMain.HOME_TAG
-HOME : int = 0
-CIRCLES : int = 1
-SETTINGS : int = 2
-navIndex : int = 0
-handler : Handler
-user : User
-drawer : DrawerLayout
-navigationView : NavigationView
-userName : TextView
-bgImage : KenBurnsView
-profileImage : ImageView
+firebase : FirebaseInstance
+UIMain()
#onCreate(savedInstanceState : Bundle) : void
-loadHeader() : void
-setUpNavigationView() : void
-loadSelectedNavigationView() : void
-getIndexFragment() : Fragment
-selectMenuItem() : void
+onBackPressed() : void
+fetchUser() : User
+getProfileView() : ImageView
+getBackgroundView() : ImageView
+updateHeader() : void



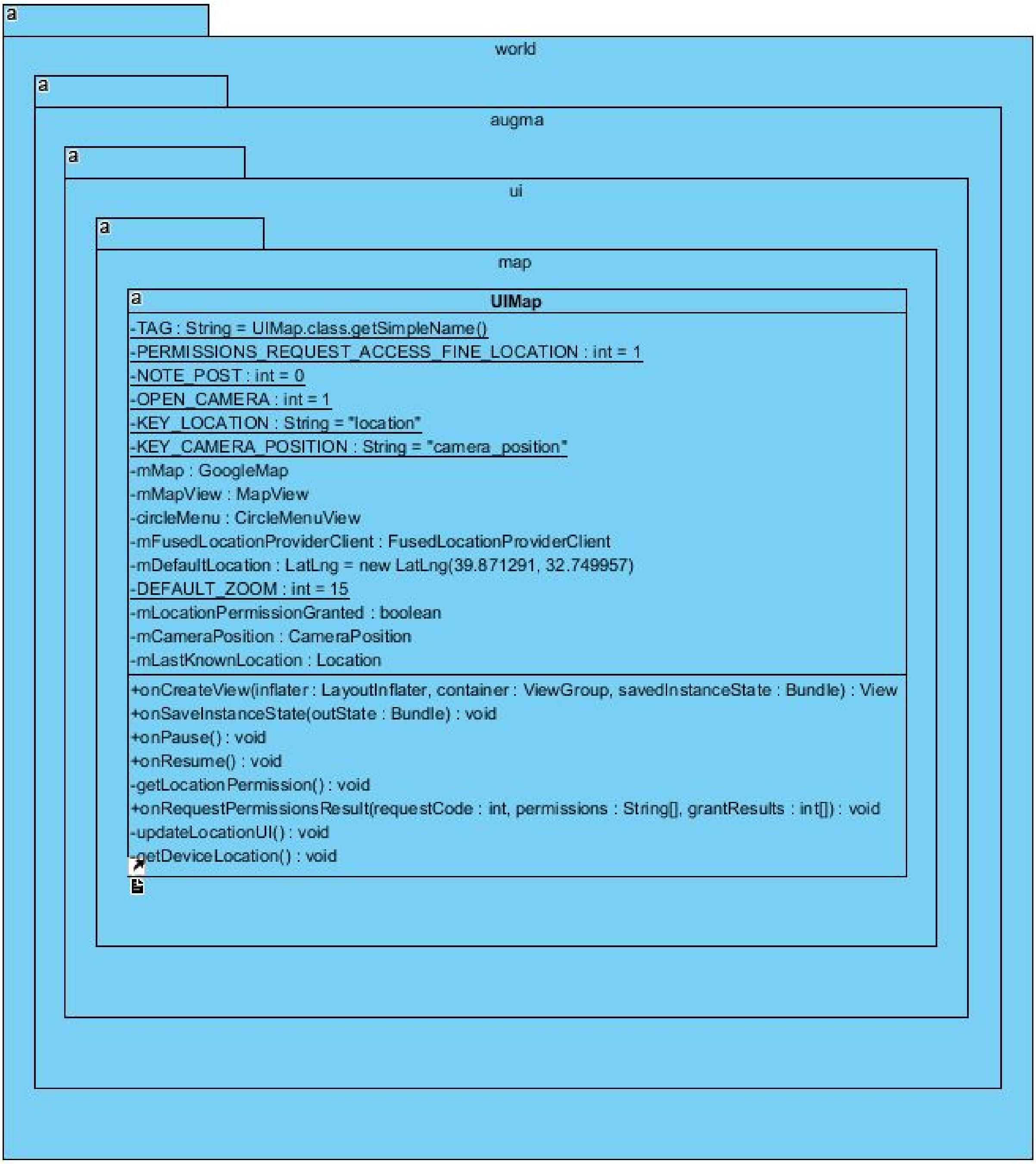


UIMain

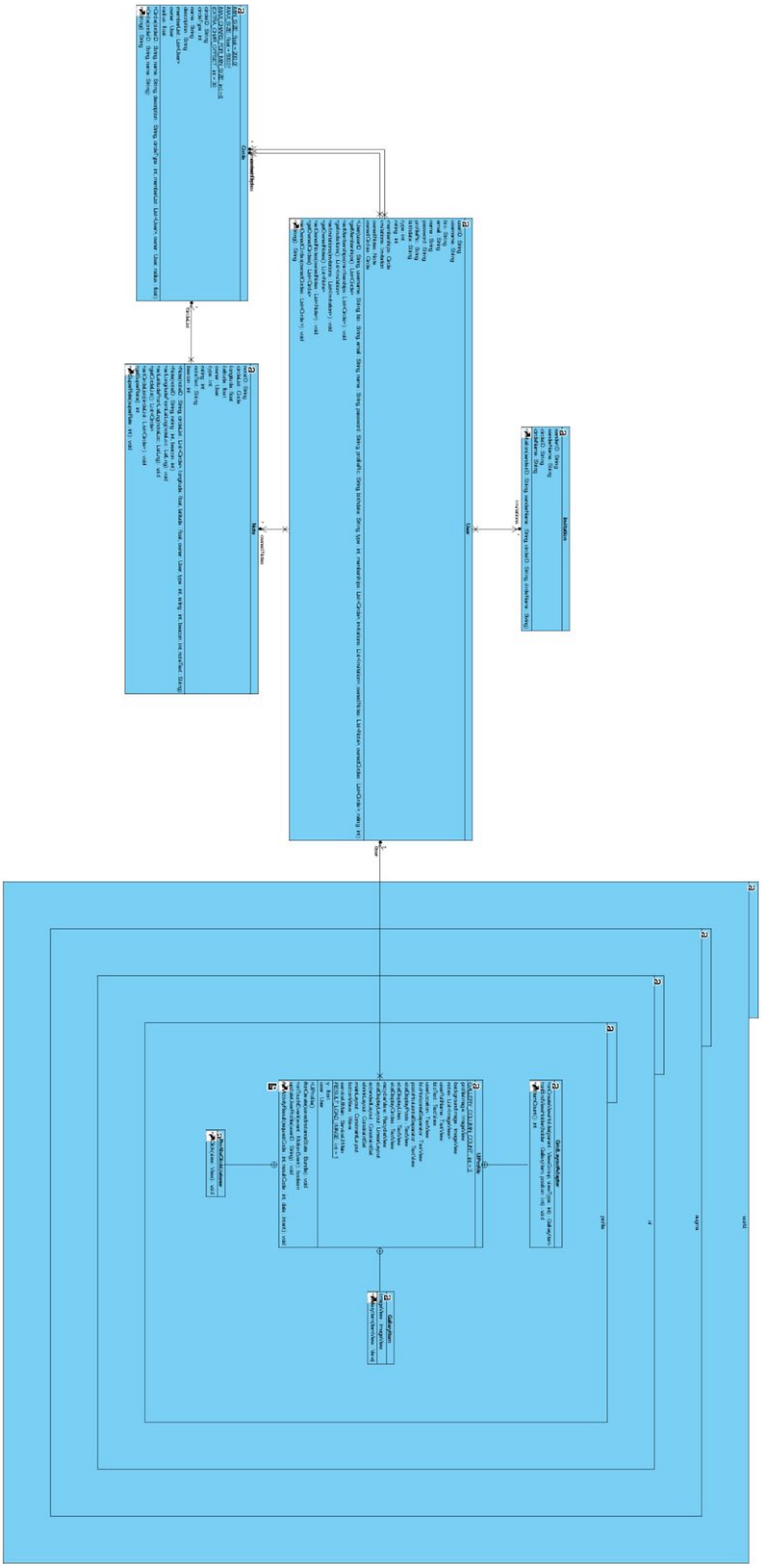
-HOME_TAG : String = "home"
-CIRCLES_TAG : String = "circles"
-SETTINGS_TAG : String = "settings"
-INDEX_TAG : String = world.augma.ui.main.UIMain.HOME_TAG
-HOME : int = 0
-CIRCLES : int = 1
-SETTINGS : int = 2
-navIndex : int = 0
-handler : Handler
-user : User
-drawer : DrawerLayout
-navigationView : NavigationView
-userName : TextView
-bgImage : KenBurnsView
-profileImage : ImageView
+firebase : FirebaseInstance

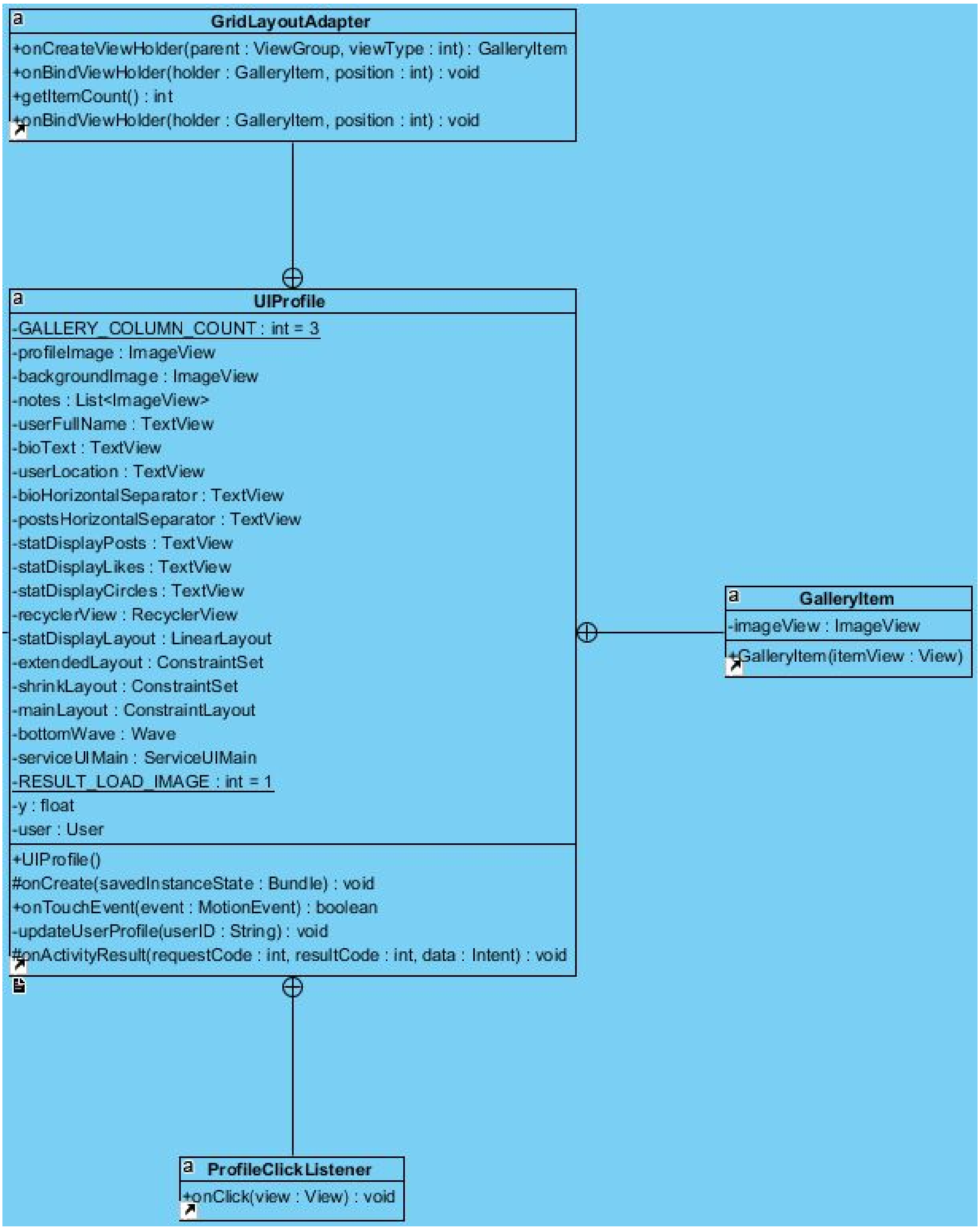
+UIMain()
#onCreate(savedInstanceState : Bundle) : void
-loadHeader() : void
-setUpNavigationView() : void
-loadSelectedNavigationView() : void
-getIndexFragment() : Fragment
-selectMenuItem() : void
+onBackPressed() : void
+fetchUser() : User
+getProfileView() : ImageView
+getBackgroundView() : ImageView
+updateHeader() : void

Overview of Map Package

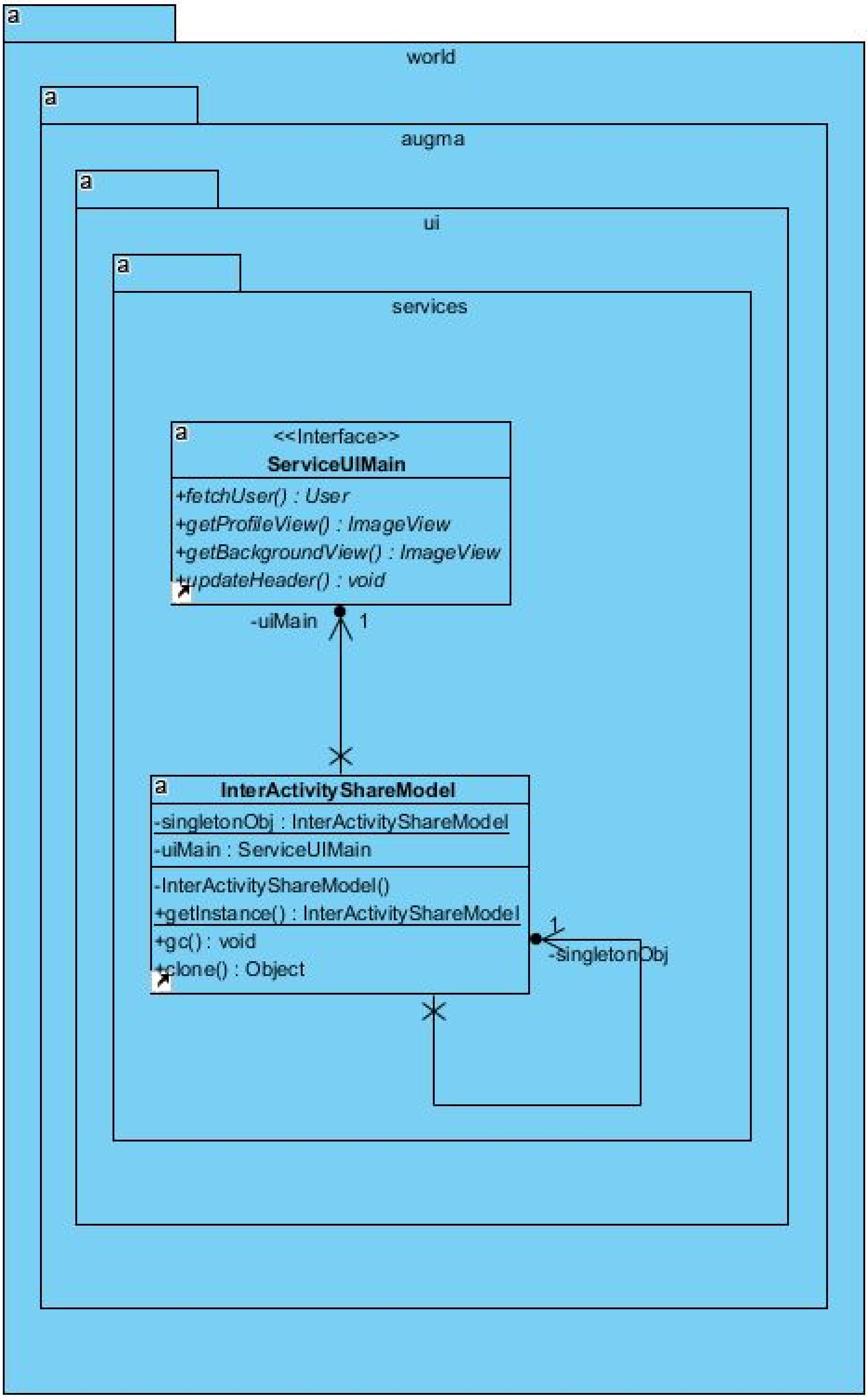

a
world
a
augma
a
ui
a
map
a
UIMap
-TAG : String = UIMap.class.getSimpleName()
-PERMISSIONS_REQUEST_ACCESS_FINE_LOCATION : int = 1
-NOTE_POST : int = 0
-OPEN_CAMERA : int = 1
-KEY_LOCATION : String = "location"
-KEY_CAMERA_POSITION : String = "camera_position"
-mMap : GoogleMap
-mMapView : MapView
-circleMenu : CircleMenuView
-mFusedLocationProviderClient : FusedLocationProviderClient
-mDefaultLocation : LatLng = new LatLng(39.871291, 32.749957)
-DEFAULT_ZOOM : int = 15
-mLocationPermissionGranted : boolean
-mCameraPosition : CameraPosition
-mLastKnownLocation : Location
+onCreateView(inflater : LayoutInflater, container : ViewGroup, savedInstanceState : Bundle) : View
+onSaveInstanceState(outState : Bundle) : void
+onPause() : void
+onResume() : void
-getLocationPermission() : void
+onRequestPermissionsResult(requestCode : int, permissions : String[], grantResults : int[]) : void
-updateLocationUI() : void
-getDeviceLocation() : void

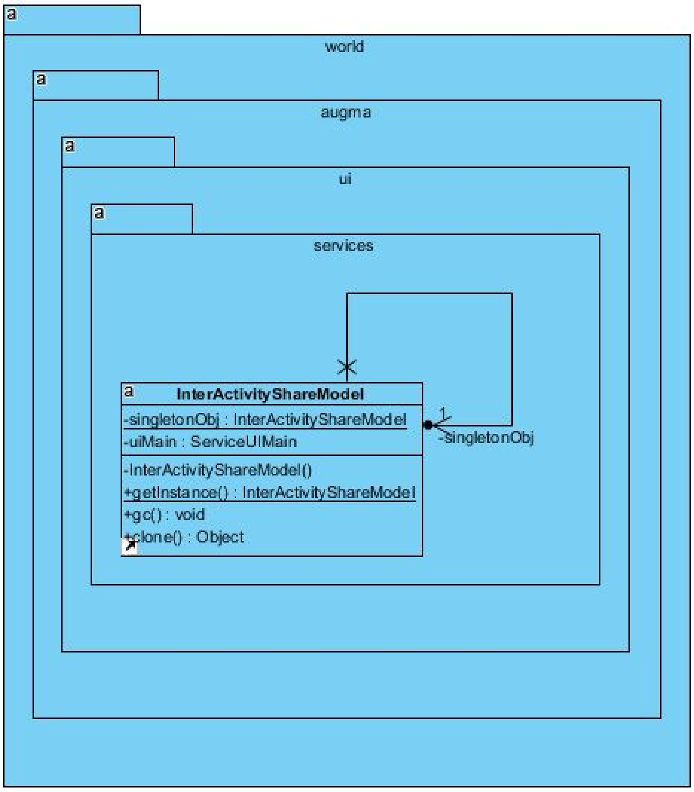
Overview of Profile Package

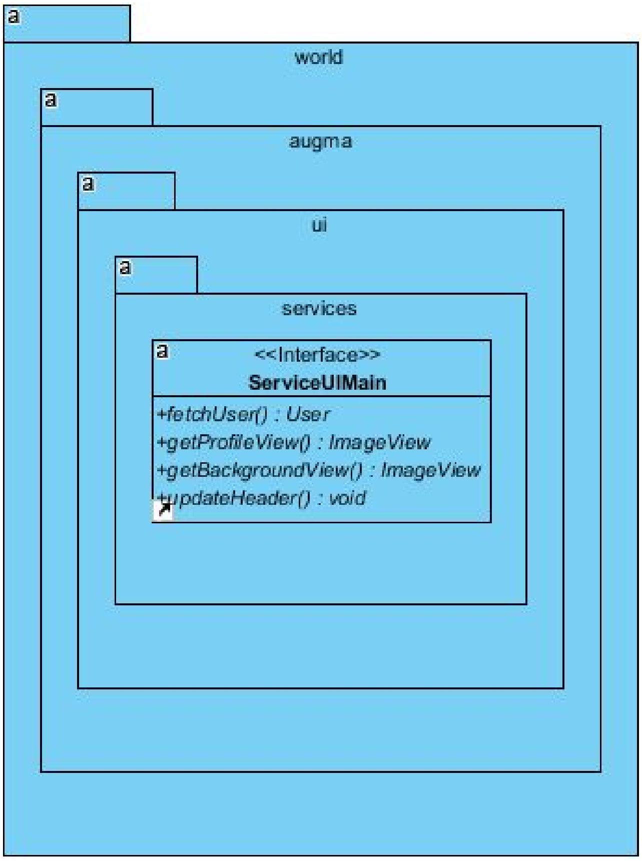



GridLayoutAdapter
+onCreateViewHolder(parent : ViewGroup, viewType : int) : GalleryItem
+onBindViewHolder(holder : GalleryItem, position : int) : void
+getItemCount() : int
+onBindViewHolder(holder : GalleryItem, position : int) : void
UIProfile
-GALLERY_COLUMN_COUNT : int = 3
-profileImage : ImageView
-backgroundImage : ImageView
-notes : List<ImageView>
-userFullName : TextView
-bioText : TextView
-userLocation : TextView
-bioHorizontalSeparator : TextView
-postsHorizontalSeparator : TextView
-statDisplayPosts : TextView
-statDisplayLikes : TextView
-statDisplayCircles : TextView
-recyclerView : RecyclerView
-statDisplayLayout : LinearLayout
-extendedLayout : ConstraintSet
-shrinkLayout : ConstraintSet
-mainLayout : ConstraintLayout
-bottomWave : Wave
-serviceUIMain : ServiceUIMain
-RESULT_LOAD_IMAGE : int = 1
-y : float
-user : User
+UIProfile()
#onCreate(savedInstanceState : Bundle) : void
+onTouchEvent(event : MotionEvent) : boolean
-updateUserProfile(userID : String) : void
#onActivityResult(requestCode : int, resultCode : int, data : Intent) : void
GalleryItem
-imageView : ImageView
+GalleryItem(itemView : View)
ProfileClickListener
+onClick(view : View) : void

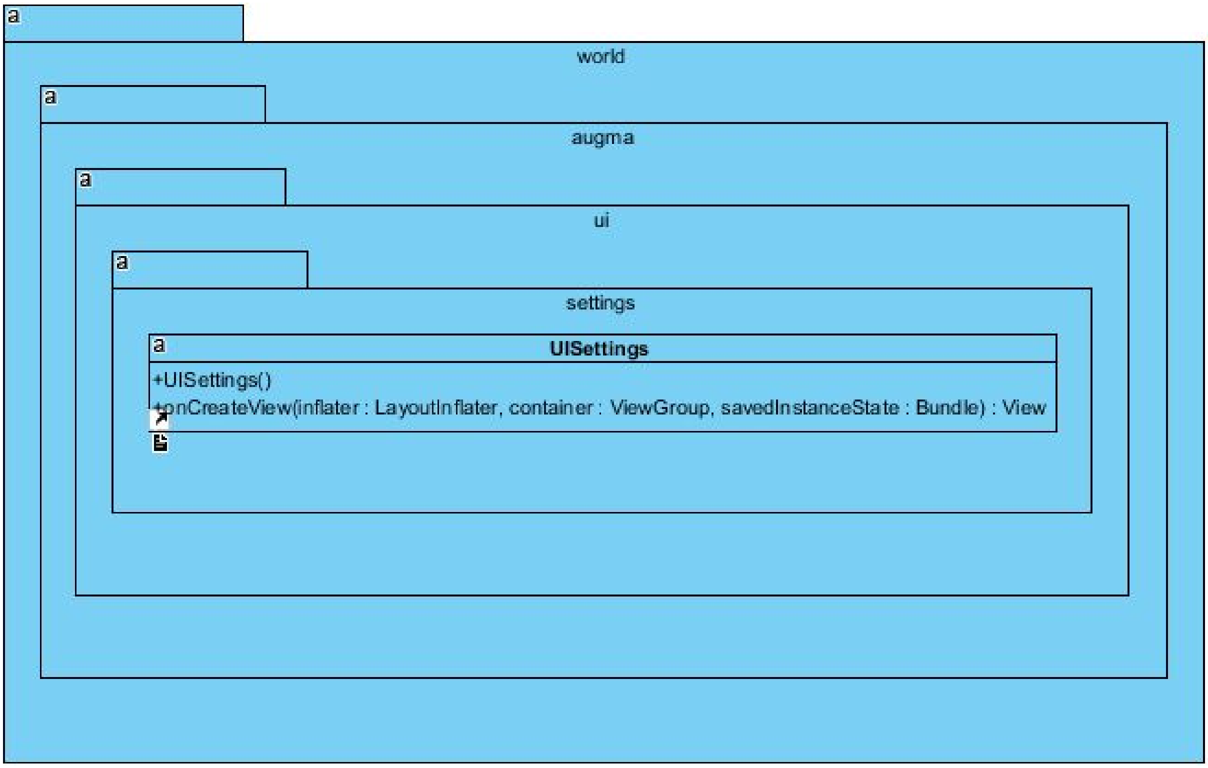
Overview of Services Package



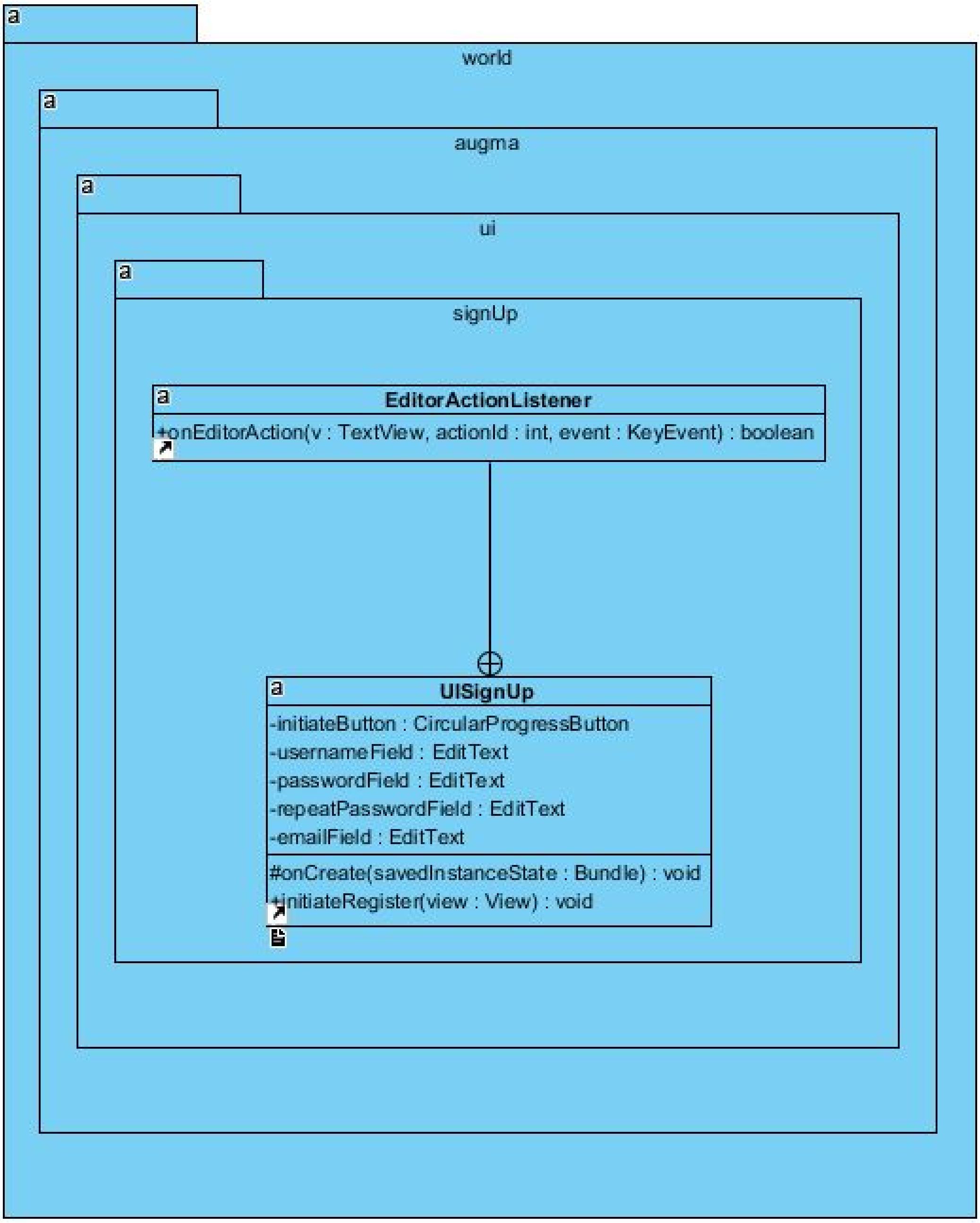

world
augma
ui
services
InterActivityShareModel
-singletonObj : InterActivityShareModel
-uiMain : ServiceUIMain
-InterActivityShareModel()
+getInstance() : InterActivityShareModel
+gc() : void
+clone() : Object
1
-singletonObj


world
augma
ui
services
<<Interface>>
ServiceUIMain
+fetchUser() : User
+getProfileView() : ImageView
+getBackgroundView() : ImageView
+updateHeader() : void

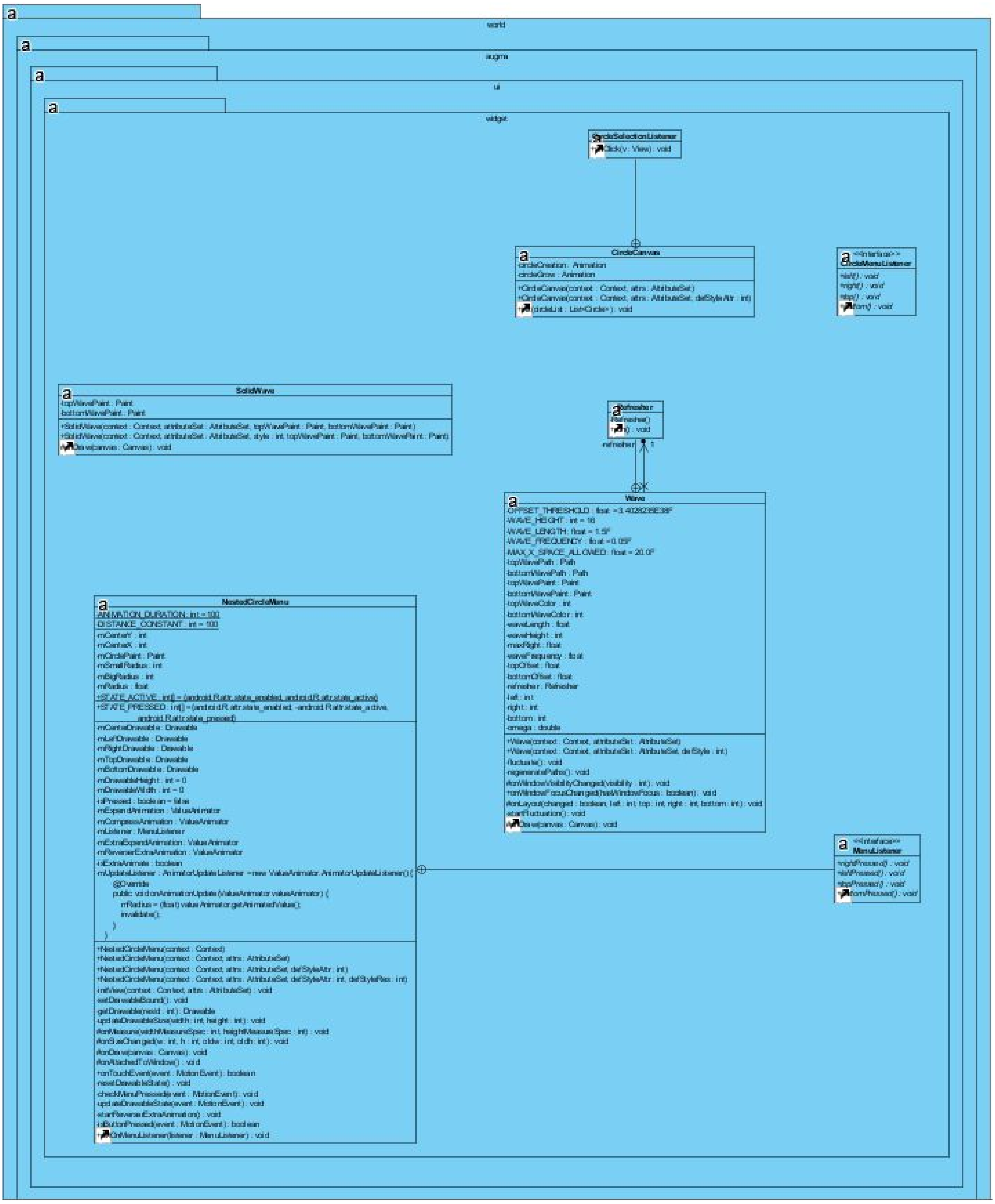
Overview of Settings Package


world
augma
ui
settings
UISettings
+UISettings()
+onCreateView(inflater : LayoutInflater, container : ViewGroup, savedInstanceState : Bundle) : View

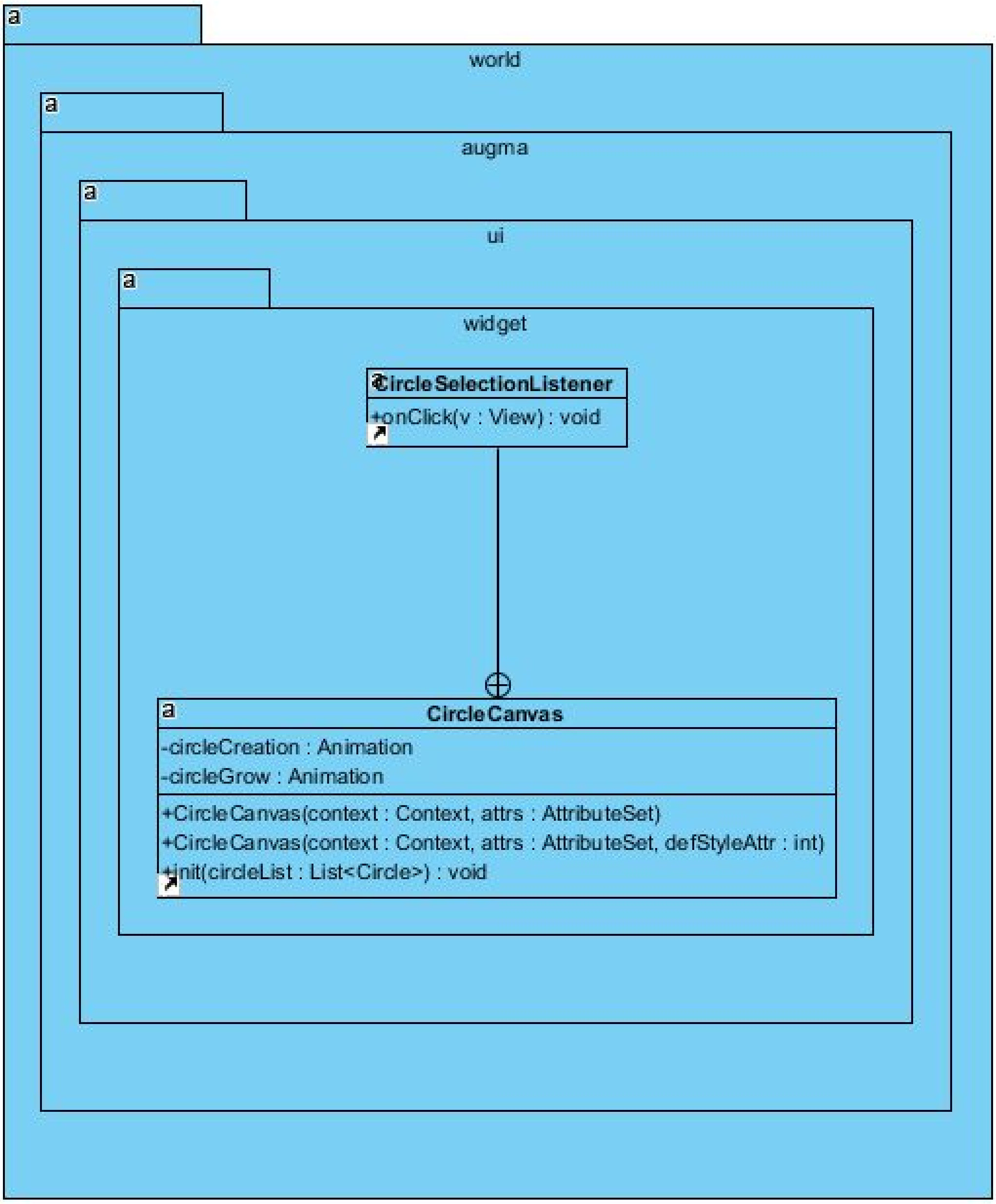
Overview of Signup Package

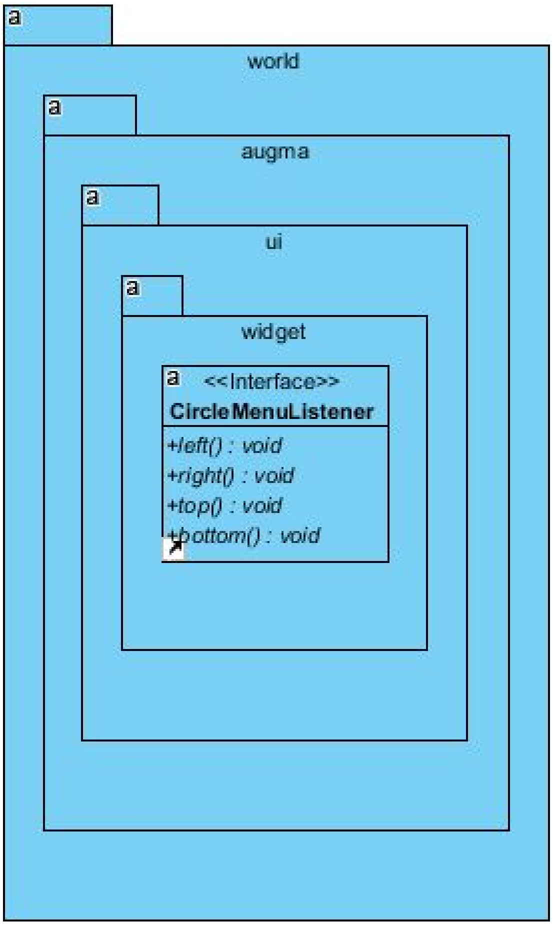

world
augma
ui
signUp
EditorActionListener
+onEditorAction(v : TextView, actionId : int, event : KeyEvent) : boolean
UISignUp
-initiateButton : CircularProgressButton
-usernameField : EditText
-passwordField : EditText
-repeatPasswordField : EditText
-emailField : EditText
#onCreate(savedInstanceState : Bundle) : void
+initiateRegister(view : View) : void

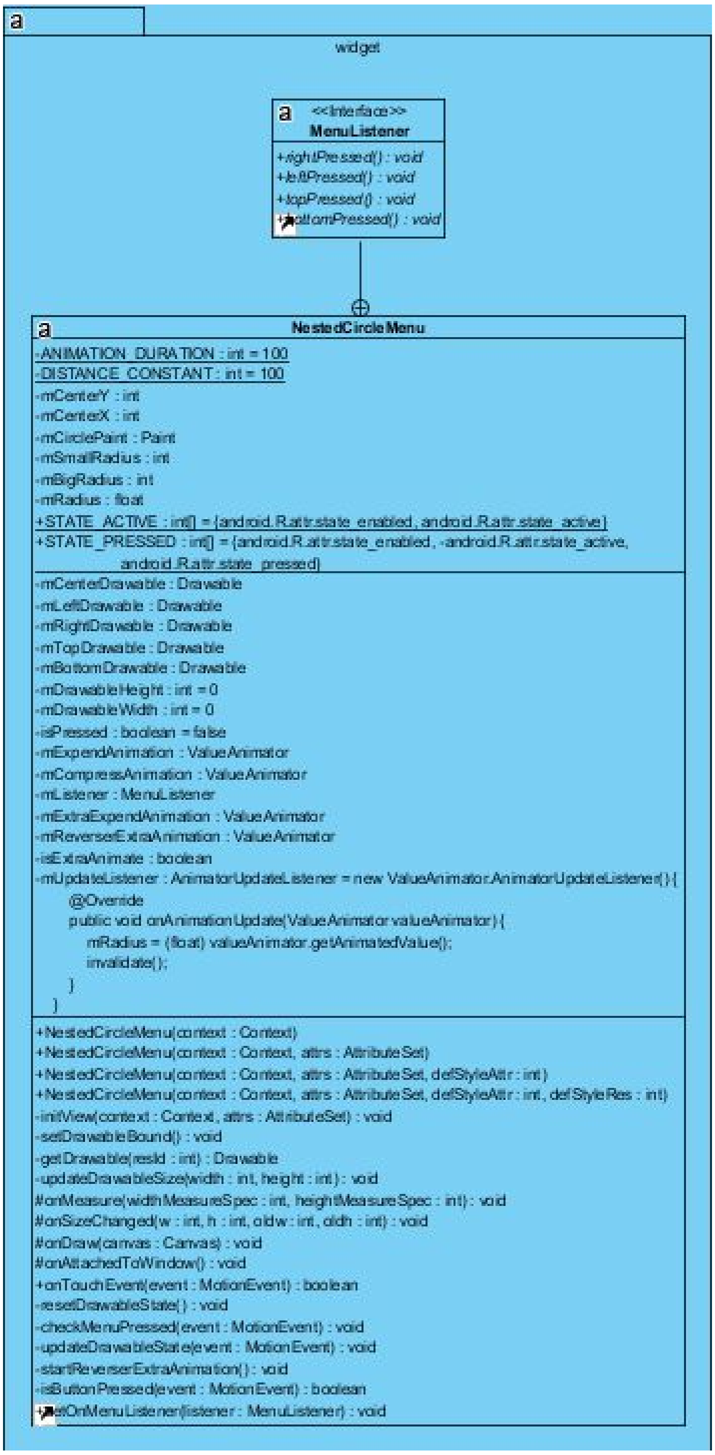
Overview of Widget Package



world
augma
ui
widget
CircleSelectionListener
+onClick(v : View) : void
CircleCanvas
-circleCreation : Animation
-circleGrow : Animation
+CircleCanvas(context : Context, attrs : AttributeSet)
+CircleCanvas(context : Context, attrs : AttributeSet, defStyleAttr : int)
+[illegible](circleList : List<Circle>) : void
<<Interface>>
CircleMenuListener
+left() : void
+right() : void
+top() : void
+bottom() : void
SolidWave
-topWavePaint : Paint
-bottomWavePaint : Paint
+SolidWave(context : Context, attributeSet : AttributeSet, topWavePaint : Paint, bottomWavePaint : Paint)
+SolidWave(context : Context, attributeSet : AttributeSet, style : int, topWavePaint : Paint, bottomWavePaint : Paint)
#onDraw(canvas : Canvas) : void
Refresher
-Refresher()
+run() : void
-refresher
1
Wave
-OFFSET_THRESHOLD : float = 3.4028235E38F
-WAVE_HEIGHT : int = 16
-WAVE_LENGTH : float = 1.5F
-WAVE_FREQUENCY : float = 0.05F
-MAX_X_SPACE_ALLOWED : float = 20.0F
-topWavePath : Path
-bottomWavePath : Path
-topWavePaint : Paint
-bottomWavePaint : Paint
-topWaveColor : int
-bottomWaveColor : int
-waveLength : float
-waveHeight : int
-maxRight : float
-waveFrequency : float
-topOffset : float
-bottomOffset : float
-refresher : Refresher
-left : int
-right : int
-bottom : int
-omega : double
+Wave(context : Context, attributeSet : AttributeSet)
+Wave(context : Context, attributeSet : AttributeSet, defStyle : int)
-fluctuate() : void
-regeneratePaths() : void
#onWindowVisibilityChanged(visibility : int) : void
+onWindowFocusChanged(hasWindowFocus : boolean) : void
#onLayout(changed : boolean, left : int, top : int, right : int, bottom : int) : void
-startFluctuation() : void
#onDraw(canvas : Canvas) : void
NestedCircleMenu
-ANIMATION_DURATION : int = 100
-DISTANCE_CONSTANT : int = 100
-mCenterY : int
-mCenterX : int
-mCirclePaint : Paint
-mSmallRadius : int
-mBigRadius : int
-mRadius : float
+STATE_ACTIVE : int[] = {android.R.attr.state_enabled, android.R.attr.state_active}
+STATE_PRESSED : int[] = {android.R.attr.state_enabled, -android.R.attr.state_active, android.R.attr.state_pressed}
-mCenterDrawable : Drawable
-mLeftDrawable : Drawable
-mRightDrawable : Drawable
-mTopDrawable : Drawable
-mBottomDrawable : Drawable
-mDrawableHeight : int = 0
-mDrawableWidth : int = 0
-isPressed : boolean = false
-mExpandAnimation : ValueAnimator
-mCompressAnimation : ValueAnimator
-mListener : MenuListener
-mExtraExpandAnimation : ValueAnimator
-mReverseExtraAnimation : ValueAnimator
-isExtraAnimate : boolean
-mUpdateListener : AnimatorUpdateListener = new ValueAnimator.AnimatorUpdateListener() {
@Override
public void onAnimationUpdate(ValueAnimator valueAnimator) {
mRadius = (float) valueAnimator.getAnimatedValue();
invalidate();
}
}
+NestedCircleMenu(context : Context)
+NestedCircleMenu(context : Context, attrs : AttributeSet)
+NestedCircleMenu(context : Context, attrs : AttributeSet, defStyleAttr : int)
+NestedCircleMenu(context : Context, attrs : AttributeSet, defStyleAttr : int, defStyleRes : int)
-initView(context : Context, attrs : AttributeSet) : void
-setDrawableBound() : void
-getDrawable(resId : int) : Drawable
-updateDrawableSize(width : int, height : int) : void
#onMeasure(widthMeasureSpec : int, heightMeasureSpec : int) : void
#onSizeChanged(w : int, h : int, oldw : int, oldh : int) : void
#onDraw(canvas : Canvas) : void
#onAttachedToWindow() : void
+onTouchEvent(event : MotionEvent) : boolean
-resetDrawableState() : void
-checkMenuPressed(event : MotionEvent) : void
-updateDrawableState(event : MotionEvent) : void
-startReverseExtraAnimation() : void
-isButtonPressed(event : MotionEvent) : boolean
+setOnMenuListener(listener : MenuListener) : void
<<Interface>>
MenuListener
+rightPressed() : void
+leftPressed() : void
+topPressed() : void
+bottomPressed() : void

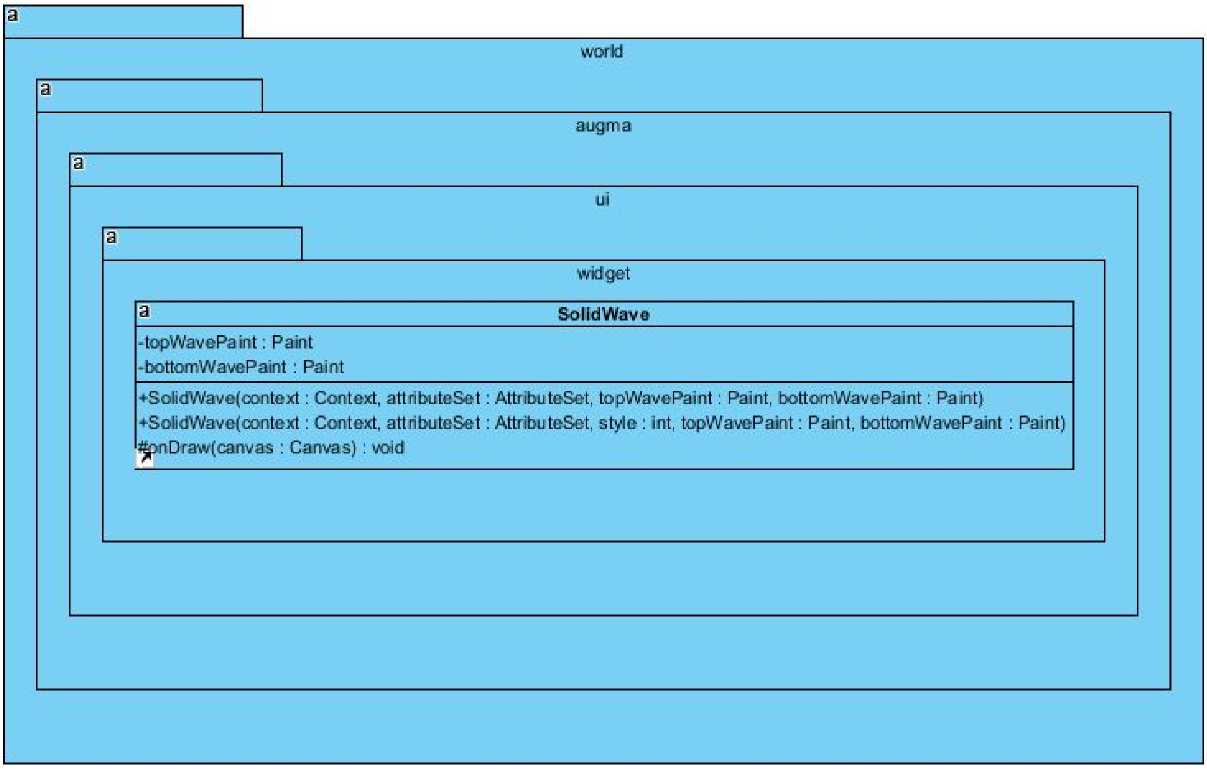

world
augma
ui
widget
CircleSelectionListener
+onClick(v : View) : void
CircleCanvas
-circleCreation : Animation
-circleGrow : Animation
+CircleCanvas(context : Context, attrs : AttributeSet)
+CircleCanvas(context : Context, attrs : AttributeSet, defStyleAttr : int)
+init(circleList : List<Circle>) : void

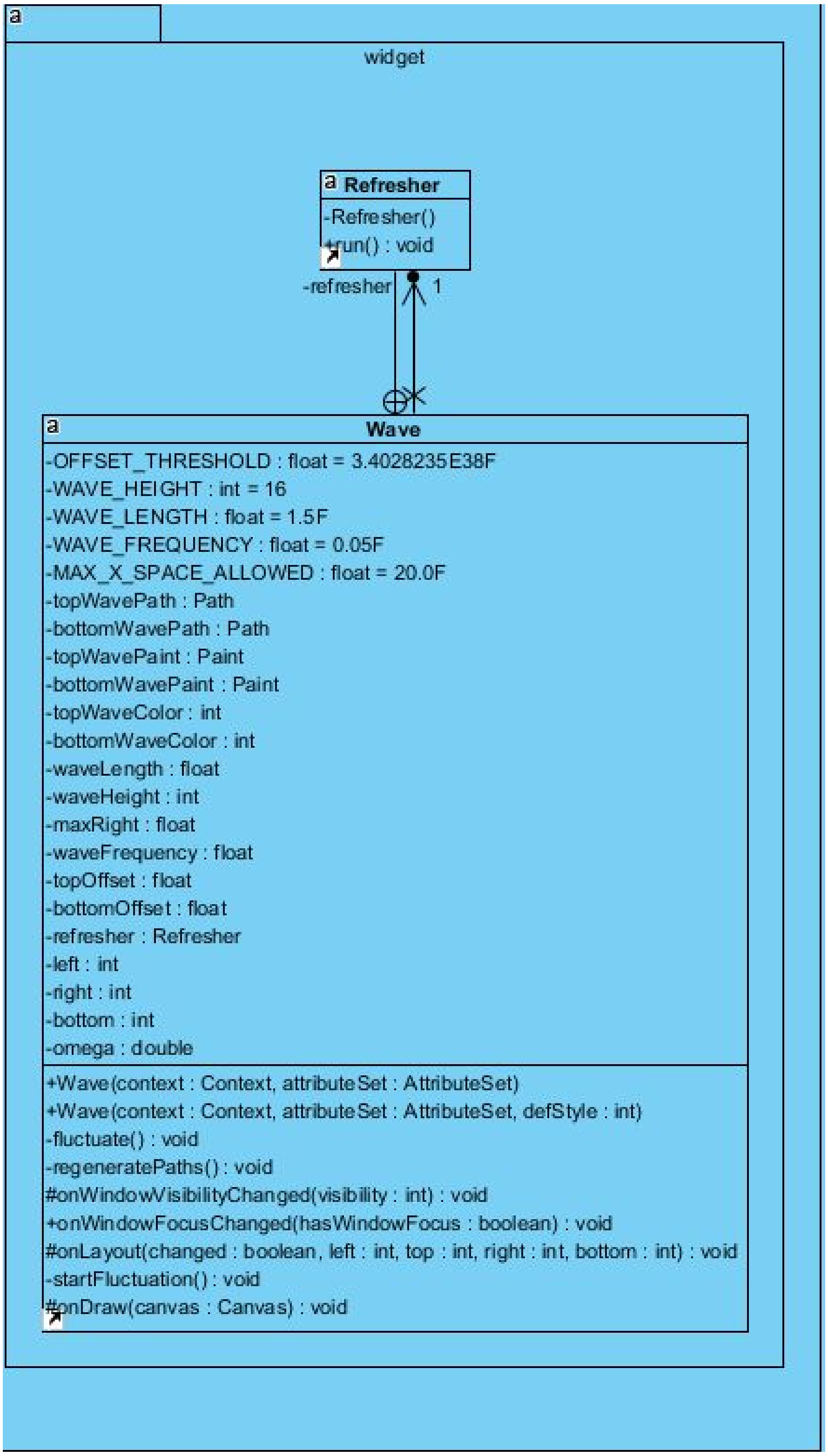

world
augma
ui
widget
<<Interface>>
CircleMenuListener
+left() : void
+right() : void
+top() : void
+bottom() : void



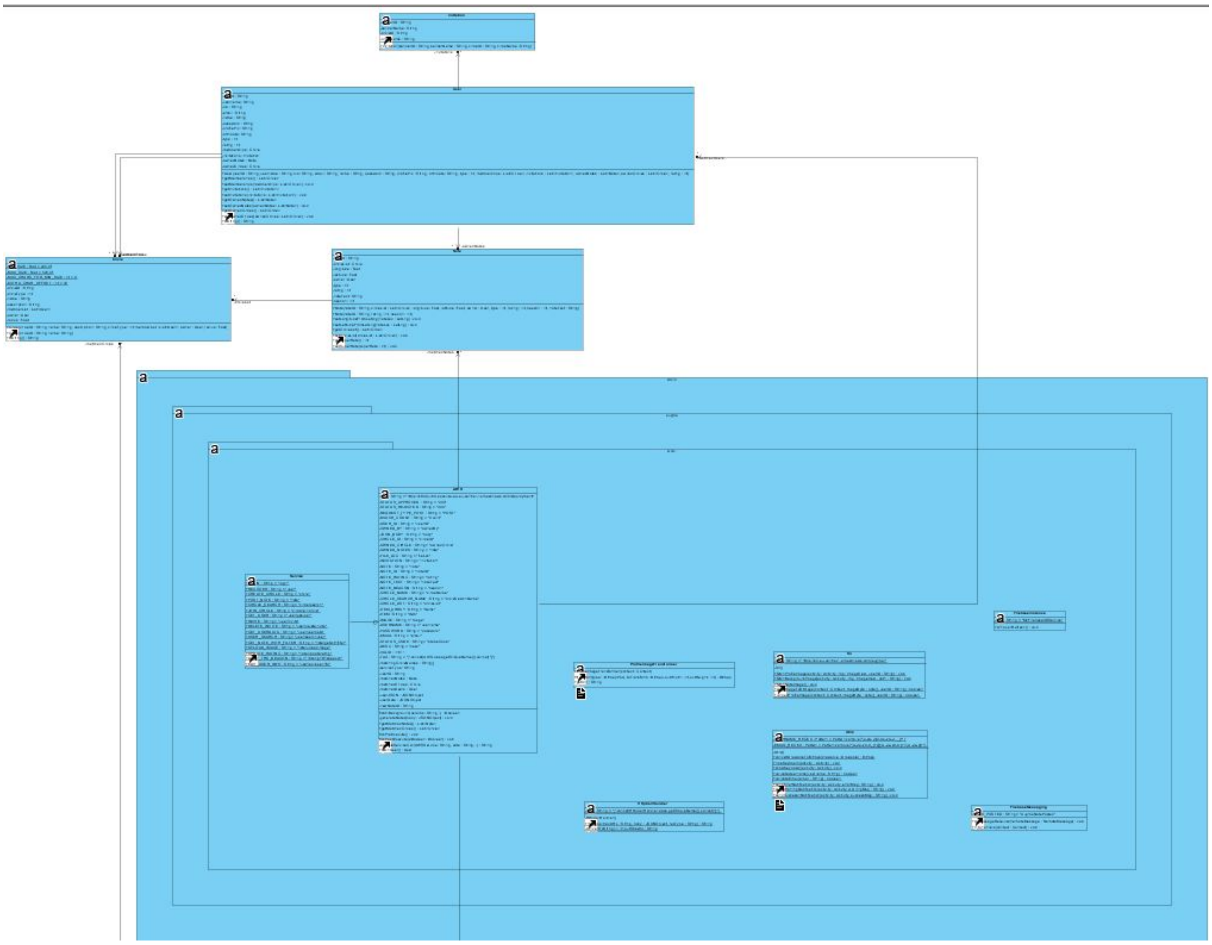
```
<<Interface>>
MenuListener
+rightPressed() : void
+leftPressed() : void
+topPressed() : void
+bottomPressed() : void
```

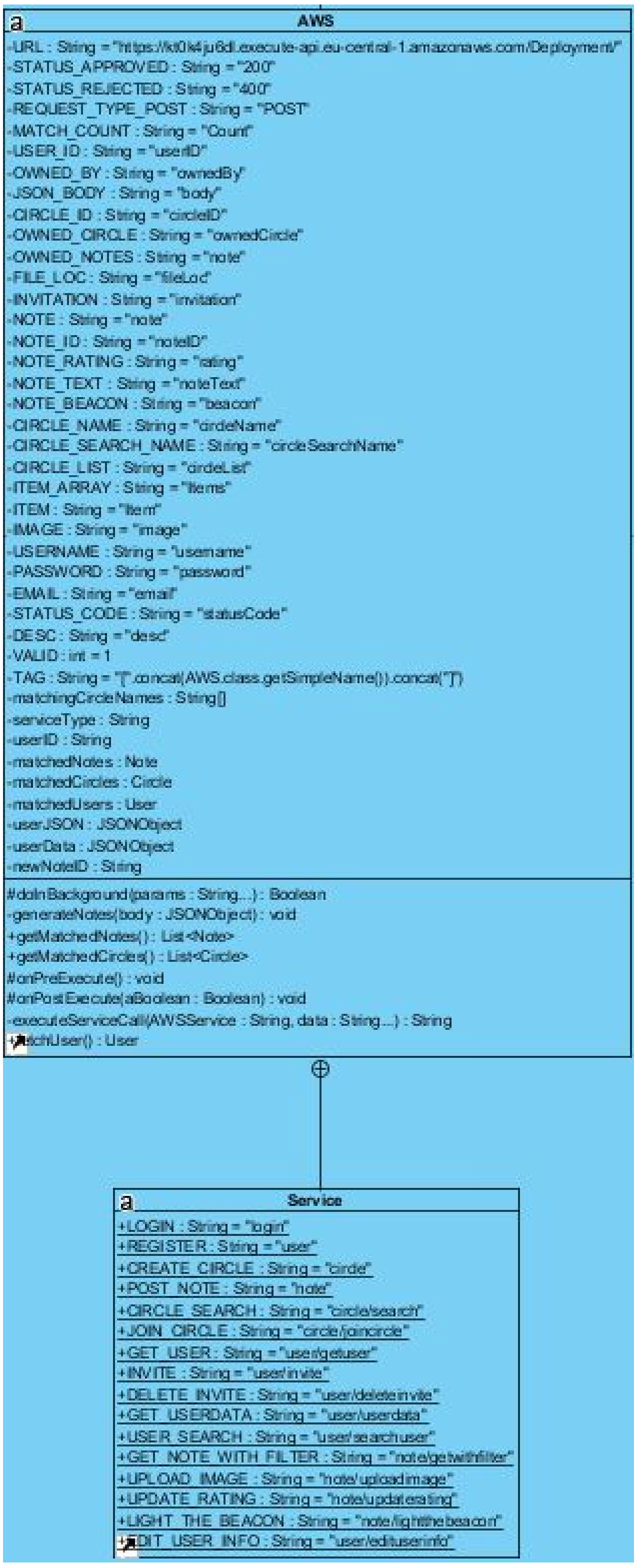
```
NestedCircleMenu
-ANIMATION_DURATION : int = 100
-DISTANCE_CONSTANT : int = 100
-mCenterY : int
-mCenterX : int
-mCirclePaint : Paint
-mSmallRadius : int
-mBigRadius : int
-mRadius : float
+STATE_ACTIVE : int[] = {android.R.attr.state_enabled, android.R.attr.state_active}
+STATE_PRESSED : int[] = {android.R.attr.state_enabled, -android.R.attr.state_active,
              android.R.attr.state_pressed}
-mCenterDrawable : Drawable
-mLeftDrawable : Drawable
-mRightDrawable : Drawable
-mTopDrawable : Drawable
-mBottomDrawable : Drawable
-mDrawableHeight : int = 0
-mDrawableWidth : int = 0
-isPressed : boolean = false
-mExpendAnimation : ValueAnimator
-mCompressAnimation : ValueAnimator
-mListener : MenuListener
-mExtraExpendAnimation : ValueAnimator
-mReverserExtraAnimation : ValueAnimator
-isExtraAnimate : boolean
-mUpdateListener : AnimatorUpdateListener = new ValueAnimator.AnimatorUpdateListener() {
      @Override
      public void onAnimationUpdate(ValueAnimator valueAnimator) {
         mRadius = (float) valueAnimator.getAnimatedValue();
         invalidate();
      }
  }
+NestedCircleMenu(context : Context)
+NestedCircleMenu(context : Context, attrs : AttributeSet)
+NestedCircleMenu(context : Context, attrs : AttributeSet, defStyleAttr : int)
+NestedCircleMenu(context : Context, attrs : AttributeSet, defStyleAttr : int, defStyleRes : int)
-initView(context : Context, attrs : AttributeSet) : void
-setDrawableBound() : void
-getDrawable(resId : int) : Drawable
-updateDrawableSize(width : int, height : int) : void
#onMeasure(widthMeasureSpec : int, heightMeasureSpec : int) : void
#onSizeChanged(w : int, h : int, oldw : int, oldh : int) : void
#onDraw(canvas : Canvas) : void
#onAttachedToWindow() : void
+onTouchEvent(event : MotionEvent) : boolean
-resetDrawableState() : void
-checkMenuPressed(event : MotionEvent) : void
-updateDrawableState(event : MotionEvent) : void
-startReverserExtraAnimation() : void
-isButtonPressed(event : MotionEvent) : boolean
+setOnMenuListener(listener : MenuListener) : void
```


world
augma
ui
widget
SolidWave
-topWavePaint : Paint
-bottomWavePaint : Paint
+SolidWave(context : Context, attributeSet : AttributeSet, topWavePaint : Paint, bottomWavePaint : Paint)
+SolidWave(context : Context, attributeSet : AttributeSet, style : int, topWavePaint : Paint, bottomWavePaint : Paint)
#onDraw(canvas : Canvas) : void

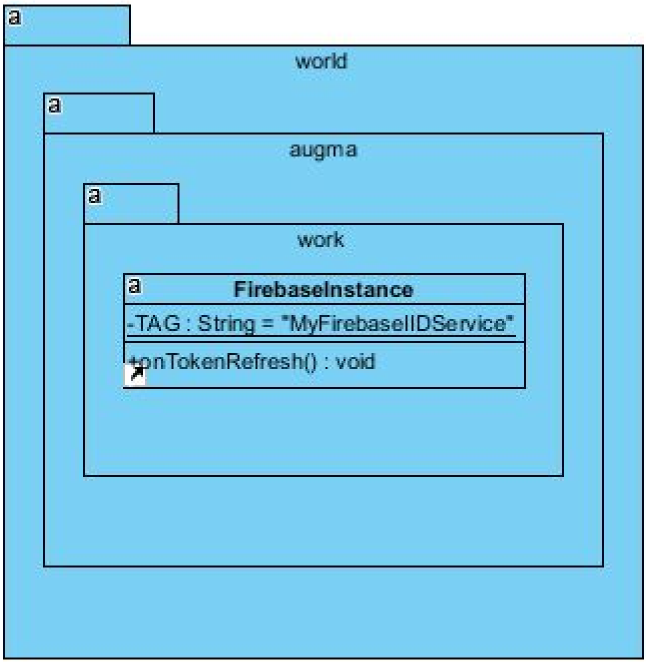

widget
Refresher
-Refresher()
+run() : void
-refresher
1
Wave
-OFFSET_THRESHOLD : float = 3.4028235E38F
-WAVE_HEIGHT : int = 16
-WAVE_LENGTH : float = 1.5F
-WAVE_FREQUENCY : float = 0.05F
-MAX_X_SPACE_ALLOWED : float = 20.0F
-topWavePath : Path
-bottomWavePath : Path
-topWavePaint : Paint
-bottomWavePaint : Paint
-topWaveColor : int
-bottomWaveColor : int
-waveLength : float
-waveHeight : int
-maxRight : float
-waveFrequency : float
-topOffset : float
-bottomOffset : float
-refresher : Refresher
-left : int
-right : int
-bottom : int
-omega : double
+Wave(context : Context, attributeSet : AttributeSet)
+Wave(context : Context, attributeSet : AttributeSet, defStyle : int)
-fluctuate() : void
-regeneratePaths() : void
#onWindowVisibilityChanged(visibility : int) : void
+onWindowFocusChanged(hasWindowFocus : boolean) : void
#onLayout(changed : boolean, left : int, top : int, right : int, bottom : int) : void
-startFluctuation() : void
#onDraw(canvas : Canvas) : void

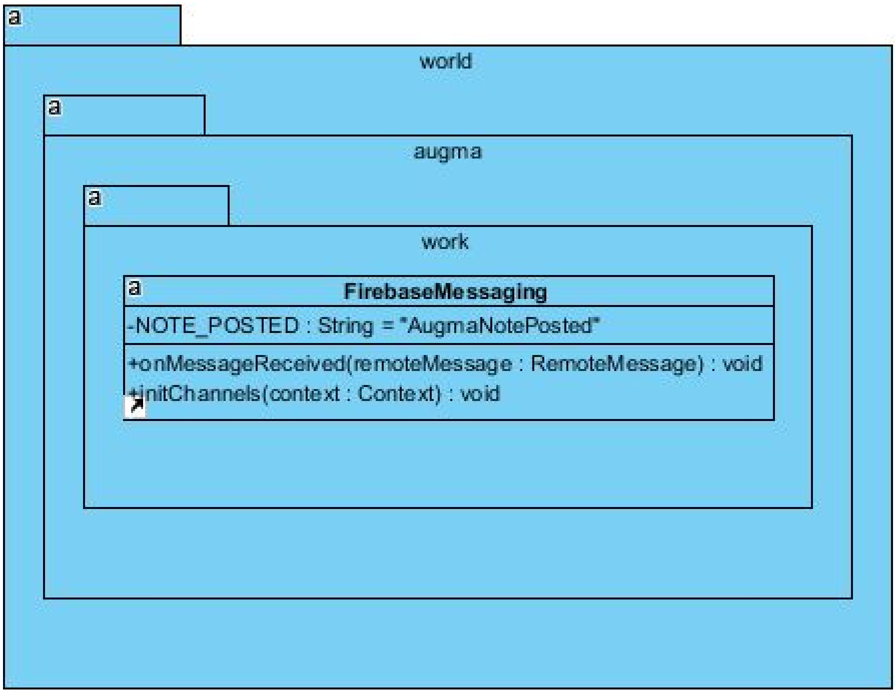
Overview of Works Package

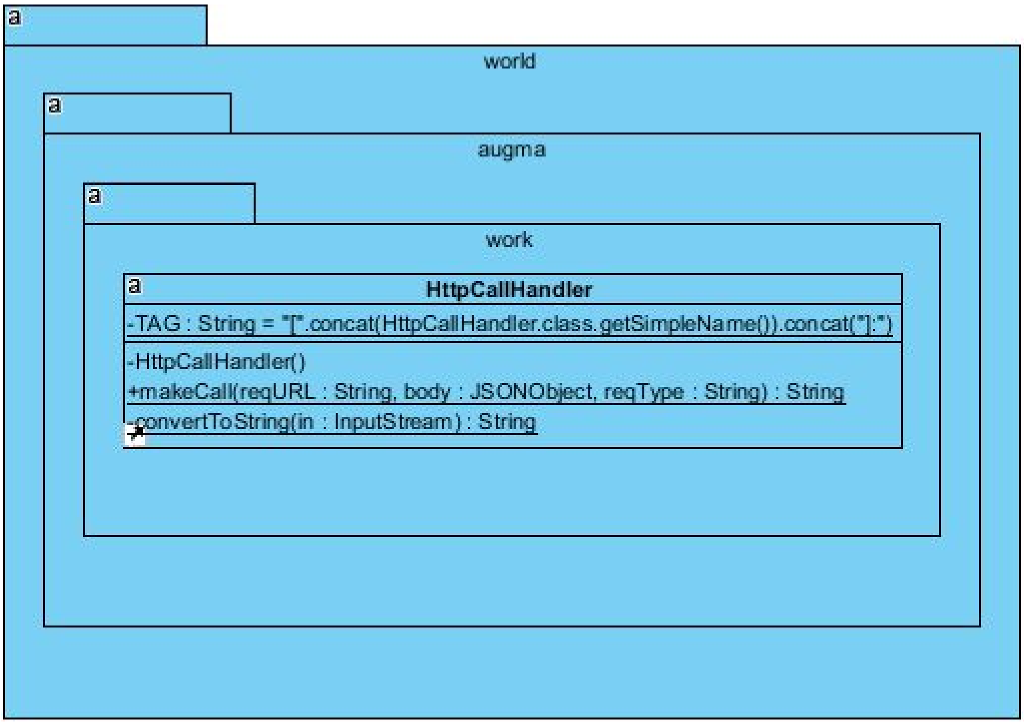


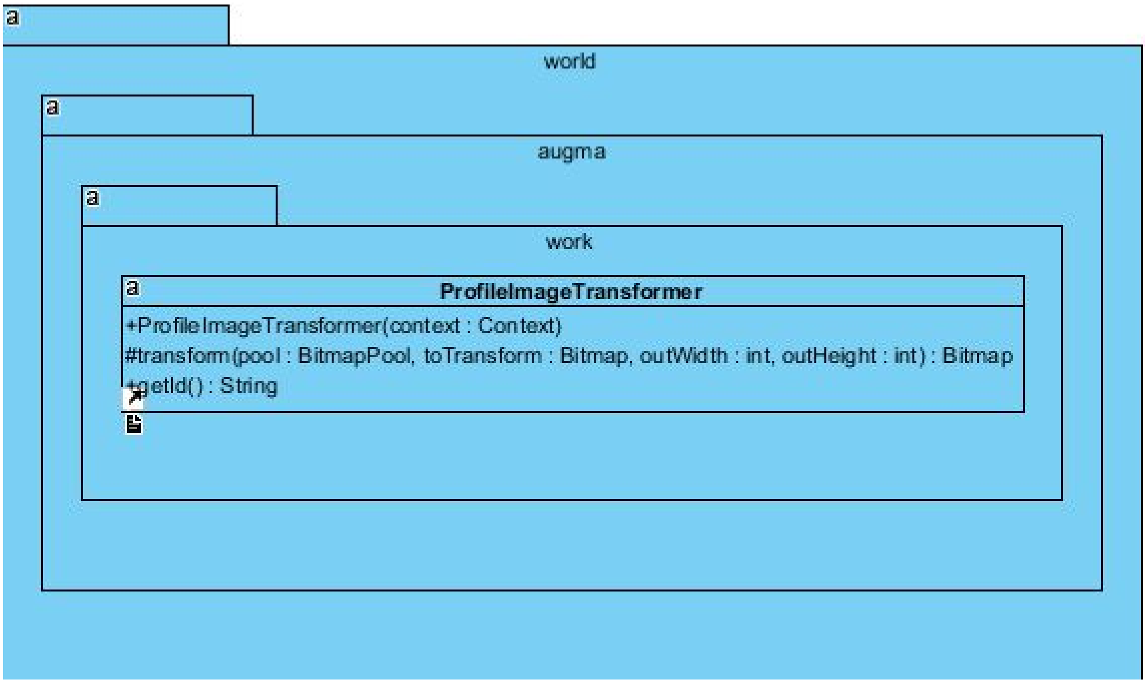


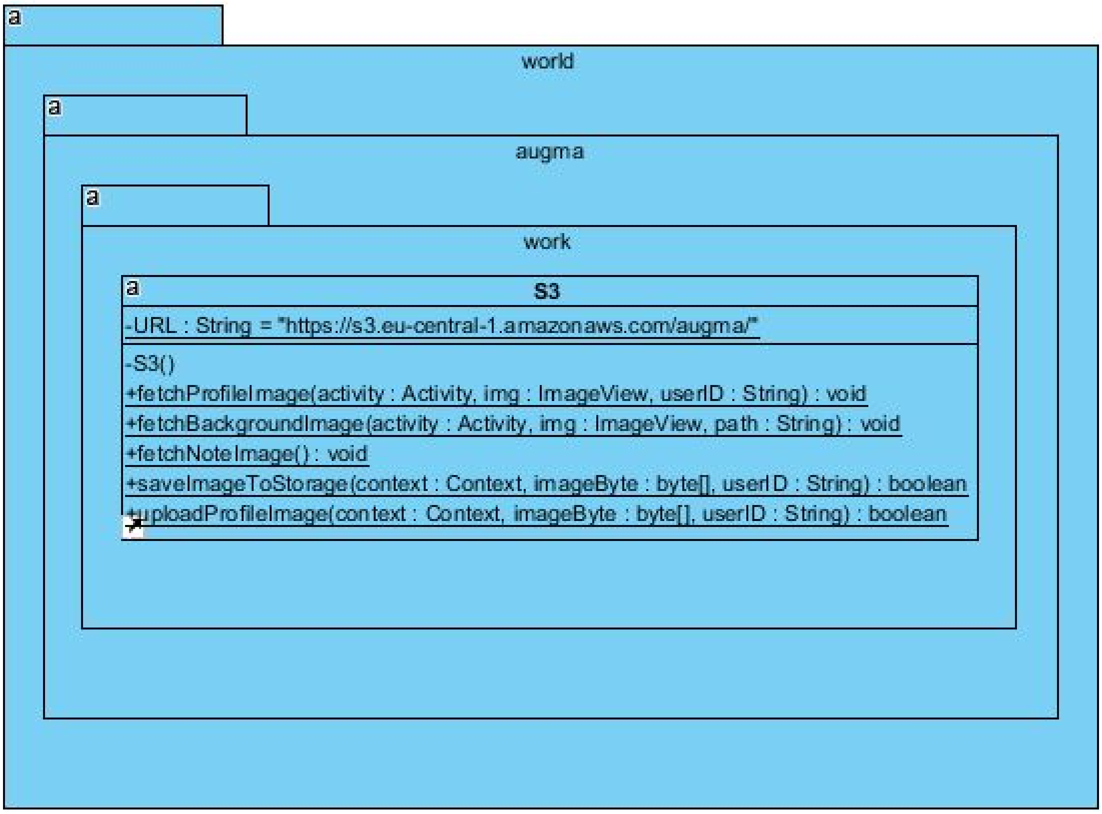
```
AWS
-URL : String = "https://kt0k4ju6dl.execute-api.eu-central-1.amazonaws.com/Deployment/"
-STATUS_APPROVED : String = "200"
-STATUS_REJECTED : String = "400"
-REQUEST_TYPE_POST : String = "POST"
-MATCH_COUNT : String = "Count"
-USER_ID : String = "userID"
-OWNED_BY : String = "ownedBy"
-JSON_BODY : String = "body"
-CIRCLE_ID : String = "circleID"
-OWNED_CIRCLE : String = "ownedCircle"
-OWNED_NOTES : String = "note"
-FILE_LOC : String = "fileLoc"
-INVITATION : String = "invitation"
-NOTE : String = "note"
-NOTE_ID : String = "noteID"
-NOTE_RATING : String = "rating"
-NOTE_TEXT : String = "noteText"
-NOTE_BEACON : String = "beacon"
-CIRCLE_NAME : String = "circleName"
-CIRCLE_SEARCH_NAME : String = "circleSearchName"
-CIRCLE_LIST : String = "circleList"
-ITEM_ARRAY : String = "Items"
-ITEM : String = "Item"
-IMAGE : String = "image"
-USERNAME : String = "username"
-PASSWORD : String = "password"
-EMAIL : String = "email"
-STATUS_CODE : String = "statusCode"
-DESC : String = "desc"
-VALID : int = 1
-TAG : String = "[".concat(AWS.class.getSimpleName()).concat("]")
-matchingCircleNames : String[]
-serviceType : String
-userID : String
-matchedNotes : Note
-matchedCircles : Circle
-matchedUsers : User
-userJSON : JSONObject
-userData : JSONObject
-newNoteID : String
#doInBackground(params : String...) : Boolean
-generateNotes(body : JSONObject) : void
+getMatchedNotes() : List<Note>
+getMatchedCircles() : List<Circle>
#onPreExecute() : void
#onPostExecute(aBoolean : Boolean) : void
-executeServiceCall(AWSService : String, data : String...) : String
+tchUser() : User

Service
+LOGIN : String = "login"
+REGISTER : String = "user"
+CREATE_CIRCLE : String = "circle"
+POST_NOTE : String = "note"
+CIRCLE_SEARCH : String = "circle/search"
+JOIN_CIRCLE : String = "circle/joincircle"
+GET_USER : String = "user/getuser"
+INVITE : String = "user/invite"
+DELETE_INVITE : String = "user/deleteinvite"
+GET_USERDATA : String = "user/userdata"
+USER_SEARCH : String = "user/searchuser"
+GET_NOTE_WITH_FILTER : String = "note/getwithfilter"
+UPLOAD_IMAGE : String = "note/uploadimage"
+UPDATE_RATING : String = "note/updaterating"
+LIGHT_THE_BEACON : String = "note/lightthebeacon"
+EDIT_USER_INFO : String = "user/edituserinfo"
```


world
augma
work
FirebaseInstance
-TAG : String = "MyFirebaseIIDService"
+onTokenRefresh() : void


world
augma
work
FirebaseMessaging
-NOTE_POSTED : String = "AugmaNotePosted"
+onMessageReceived(remoteMessage : RemoteMessage) : void
+initChannels(context : Context) : void


world
augma
work
HttpCallHandler
-TAG : String = "[".concat(HttpCallHandler.class.getSimpleName()).concat("]:")
-HttpCallHandler()
+makeCall(reqURL : String, body : JSONObject, reqType : String) : String
-convertToString(in : InputStream) : String


world
augma
work
ProfileImageTransformer
+ProfileImageTransformer(context : Context)
#transform(pool : BitmapPool, toTransform : Bitmap, outWidth : int, outHeight : int) : Bitmap
+getId() : String



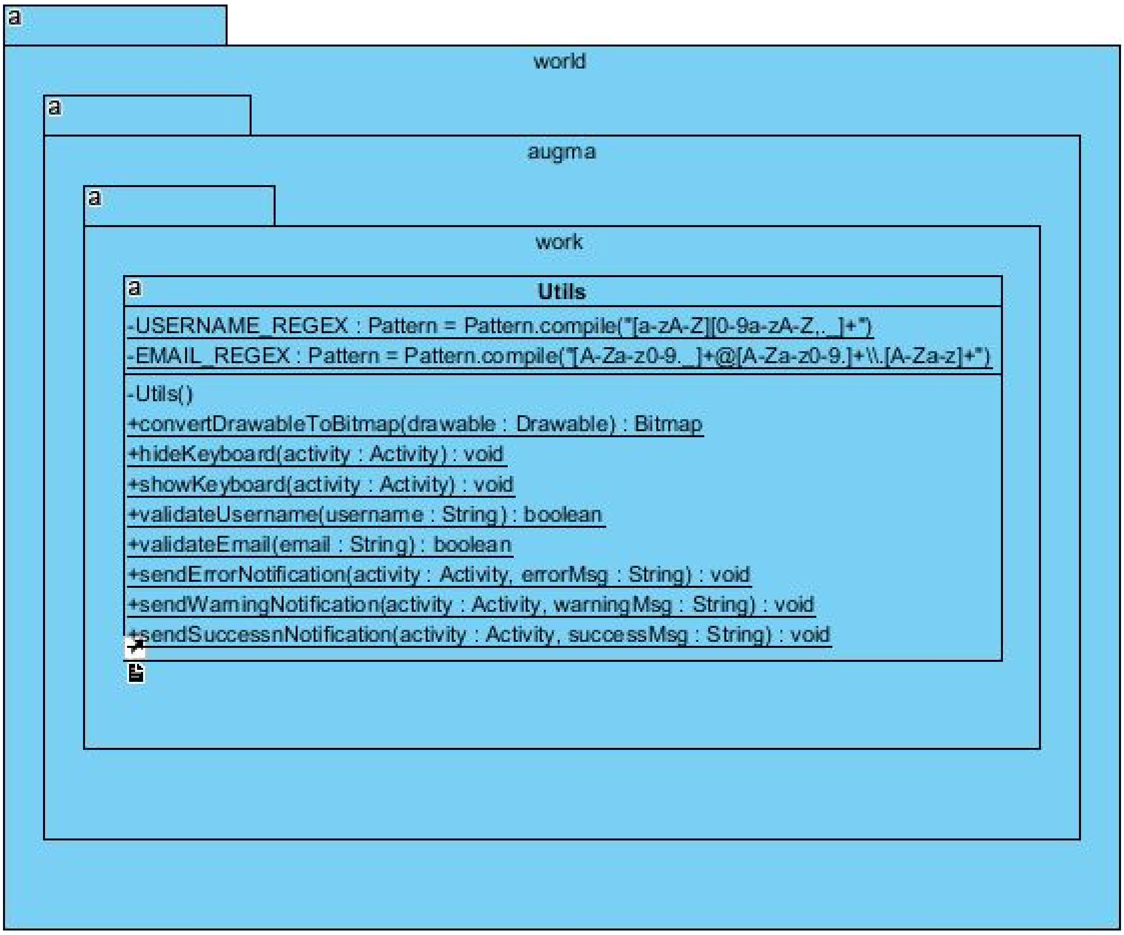
world

augma

work

S3

```
-URL : String = "https://s3.eu-central-1.amazonaws.com/augma/"

-S3()
+fetchProfileImage(activity : Activity, img : ImageView, userID : String) : void
+fetchBackgroundImage(activity : Activity, img : ImageView, path : String) : void
+fetchNoteImage() : void
+saveImageToStorage(context : Context, imageByte : byte[], userID : String) : boolean
+uploadProfileImage(context : Context, imageByte : byte[], userID : String) : boolean
```



world

augma

work

Utils

```
-USERNAME_REGEX : Pattern = Pattern.compile("[a-zA-Z][0-9a-zA-Z,._]+")
-EMAIL_REGEX : Pattern = Pattern.compile("[A-Za-z0-9._]+@[A-Za-z0-9.]+\\.[A-Za-z]+")

-Utils()
+convertDrawableToBitmap(drawable : Drawable) : Bitmap
+hideKeyboard(activity : Activity) : void
+showKeyboard(activity : Activity) : void
+validateUsername(username : String) : boolean
+validateEmail(email : String) : boolean
+sendErrorNotification(activity : Activity, errorMsg : String) : void
+sendWarningNotification(activity : Activity, warningMsg : String) : void
+sendSuccessnNotification(activity : Activity, successMsg : String) : void
```