MvRock Framework

An Explanation of the Classes and Functions

Summer 2015

Author: Kenneth Liang

Table of Contents

Introduction

This document will explore all the classes in the MvRock framework. Each variable and function will be explained. This introduction will cover each package in the framework and its general purpose.

model

The models package holds the classes that represent MvRock objects. These objects range from users and buddies to song lists. There are three types of song lists: You May Like, You Liked, and Station. The MvRockModel is a class used to hold all these objects in one static location for easy access. Programmatically most model objects are derived from the abstract class MvRockModelObject.

thread

In the threads package are all the thread classes. These threads start ALL network communications to either fetch data from the MvRock database or to store data in the database. Threads are the objects used to interface with the MvRock backend database.

uicomponent

The uicomponent package holds all the Android UI component objects. Such objects include ListViews, ImageViews, etc. The uicomponent package can be split into smaller packages: drawer, player, playlist, and station. The drawer package holds the classes for the sliding drawer layout used to navigate between the different playlists. The player package holds the YouTubePlayerFragment and the user buttons (like, dislike, share, etc.) components of the interface. The playlist holds all the ListViews and Adapter classes. The station package holds the classes responsible for station searching and displaying station songs. The

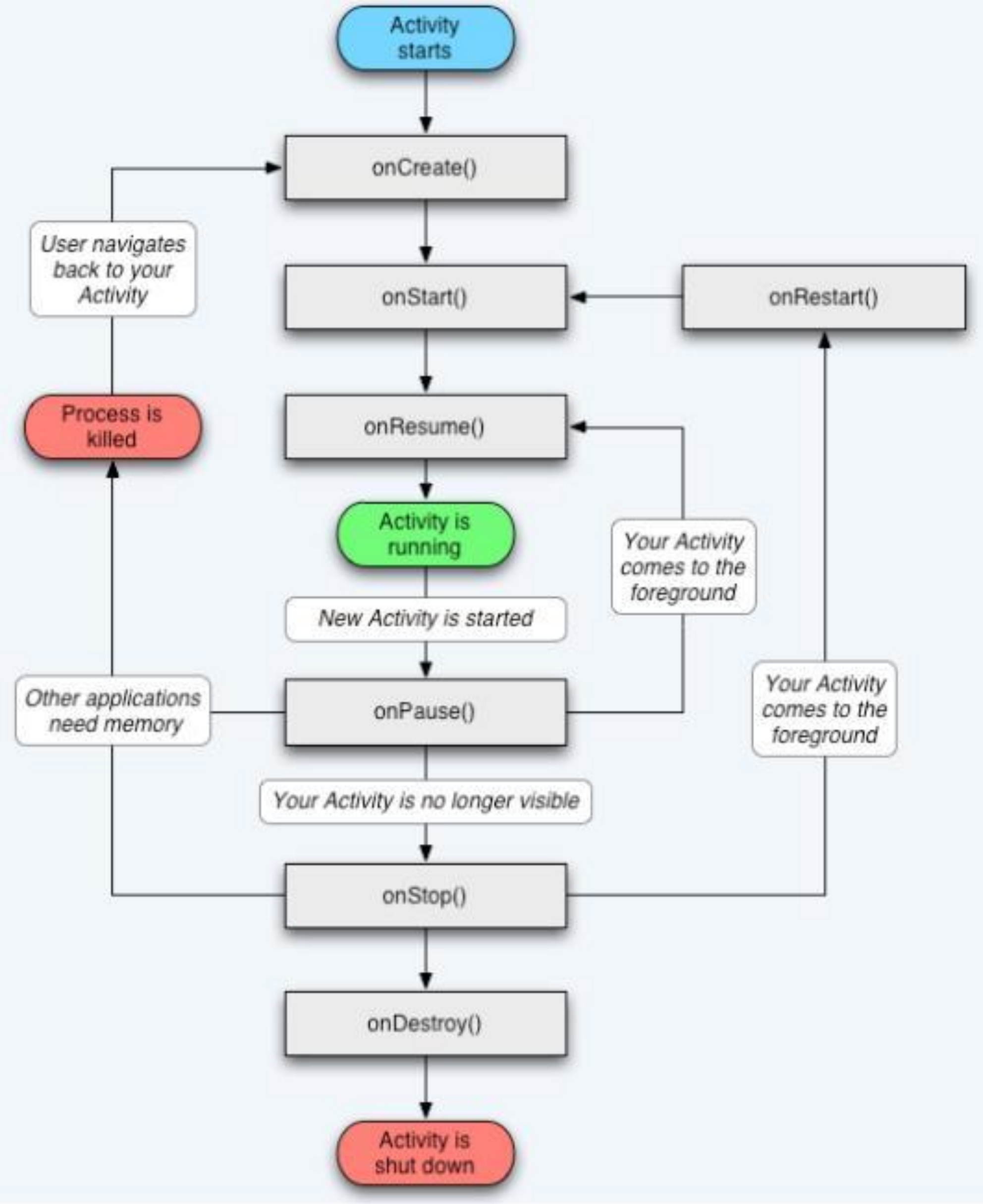
MvRockUiComponent is a class used to hold all these UI components in one static location for easier access. The MvRockUiComponentObject is an abstract class used as template for the UI components and is the parent for most UI components.

view

In the view package are the fragments. A fragment is similar to an activity window. The two main fragments used are the MvRockFragment and the FbLoginFragment. The YouTubePlayerFragment is located in the uicomponent.player package. The playlist fragments are used in the right floating menu. MainActivity is responsible for selecting the right fragment to show. MvRockView is used to hold static instances of the fragments and MainActivity for easier access.

view.MainActivity

This is the main and only activity of the MvRock. This activity controls the transitioning between the FbLoginFragment and MvRockFragment. An activity follows the lifecycle show below:


Activity starts
onCreate()
User navigates back to your Activity
onStart()
onRestart()
Process is killed
onResume()
Activity is running
Your Activity comes to the foreground
New Activity is started
Other applications need memory
onPause()
Your Activity comes to the foreground
Your Activity is no longer visible
onStop()
onDestroy()
Activity is shut down

Fields

String	USER_SKIPPED_LOGIN_KEY	the key for whether the user skipped

		login
String	TAG	for logging
long	TIMEOUT	the time frame for the user to double-press back and exit the app
long	backPressTime	used to hold the time value for double-press back exit
boolean	userSkippedLogin	Boolean whether the user skipped login
AccessTokenTracker	accessTokenTracker	Facebook class that provides the session access token
ProfileTracker	profileTracker	Facebook class that provides the user’s profile
CallbackManager	callbackManager	Facebook class that handles callbacks for certain events, such as onActivityResult

Functions

void	**onCreate(Bundle savedInstanceState)** savedInstanceState – the Bundle used to restore the activity state Called to initialize MainActivity and set its layout.
void	**onResume()** Called when the activity is showing again. Activates FaceBook’s AppEventsLogger.
void	**onPause()** Called when the activity is partially covered. Saves the cache’s data to phone memory. Deactivates FaceBook’s AppEventsLogger.
void	**onDestroy()** Called when the activity is being destroyed (when the app is exiting). Stops the

	accessTokenTracker and profileTracker.
void	**onActivityResult(int requestCode, int resultCode, Intent data)** requestCode – the code used to start the activity resultCode – the outcome of the request data – the data attached to the request This is called when the user wants to login or logout of Facebook. CallbackManager handles the login and logout procedure here.
void	**onBackPressed()** This handles the double-press back to exit logic.
void	**showFragment(int fragmentIndex, boolean addToBackStack)** fragmentIndex – the index of the fragment to show addToBackStack – whether to add the fragment transaction to the back stack This function calls the FragmentManager and begins the transaction to switch fragments.

view.DeveloperKey

This class simply holds the developer key for the YouTube player.

Fields

String	DEVELOPER_KEY	the key for the YouTube player

view.MvRockView

The class is used to hold some constants and static variables used in the functions of MainActivity and other various classes.

Fields

YouMayLikePlayListFragment	YouMayLikePlayListFragment	holds an instance
YouLikedPlayListFragment	YouLikedPlayListFragment	holds an instance
StationPlayListFragment	StationPlayListFragment	holds an instance
StationListFragment	StationListFragment	holds an instance
MainActivity	MainActivity	holds an instance
int	FBLOGIN_FRAG	index given to the FbLoginFragment in FragmentList
int	MVROCK_FRAG	index given to the MvRockFragment in FragmentList
int	FB_LOGOUT	value used to determine if the user is logging out of FaceBook
int	NOW_SHOWING_POSITION	index for now showing in drawer
int	MY_STATION_POSITION	index for my stations in drawer
int	LOGOUT_POSITION	index for logout in drawer

view.fragment.FacebookLoginFragment

This is the fragment used to handle Facebook login.

Fields

String	TAG	for logging

Functions

View	**onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)** inflater – the LayoutInflater to create the layout container – the ViewGroup the view belongs to savedInstanceState – the Bundle used to restore the fragment's view states Returns View – the layout view of the fragment This function is called to create the fragment layout and the Facebook login button. The button is set the required permissions and initiates the logging in process.

view.fragment.MvRockFragment

This fragment holds the main MvRock view. This is where all the buttons, list views, adapters, and threads will be used. The user will be spending most of his time here.

Fields

String	TAG	for logging

Functions

	MvRockFragment() This is the default constructor. It calls the default constructor of each of the UI components.
View	**onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)** inflater – the LayoutInflater to create the view container – the parent ViewGroup savedInstanceState – the Bundle to restore the view states Returns View – the view of the fragment Initializes all the MvRock UI components and hides the ActionBar.
boolean	**onOptionsItemSelected(MenuItem item)** item – the MenuItem selected Returns boolean – whether the selection is handled This is called when a menu item is selected. It is used to invite friends to MvRock or logout of Facebook.
void	**onCreateOptionsMenu(Menu menu, MenuInflater inflater)** menu – the menu to create inflater – the MenuInflater to create the menu

	Creates the menu and initializes the StationSearchView.
void	**onConfigurationChanged(Configuration newConfig)** newConfig – the new Configuration Sends the new configuration to the left drawer toggle.
void	**onSaveInstanceState(Bundle state)** state – the Bundle to save the fragment's state This is called when the fragment is saving its state. Saves the current time of the currently playing video to be used when the app is restored.
void	**onViewStateRestored(Bundle state)** state – the Bundle to restore the state Called to restore the currently playing song's time.

view.fragment.StationListFragment

This fragment is used to initialize the station list fragment.

Fields

String	TAG	for logging
TextView	title	the TextView to hold the title
ImageView	refreshButton	the ImageView to refresh the list

Functions

	StationListFragment() Creates the station's views.
View	**onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)** inflater – the LayoutInflater to create the view container – the parent ViewGroup savedInstanceState – the Bundle to restore the view states Returns View – the right drawer view Initializes the station views.

view.fragment.StationPlayListFragment

This fragment is used to initialize the station play list fragment.

Fields

TextView	title	the TextView to hold the title
ImageView	cancelButton	the ImageView to cancel the station playlist and return to the station list

Functions

	StationPlayListFragment() Initializes the StationPlayListView.
View	**onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)** inflater – the LayoutInflater to create the view container – the parent ViewGroup savedInstanceState – the Bundle to restore the view states Returns View – the right drawer view Initializes the station playlist views.

view.fragment.YouLikedPlayListFragment

This fragment is used to initialize the you liked playlist fragment.

Fields

String	TAG	for logging
TextView	title	the TextView to hold the title
ImageView	refreshButton	the ImageView to refresh the list

Functions

	YouLikedPlayListFragment() Creates the YouLikedPlayListView.
View	**onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)** inflater – the LayoutInflater to create the view container – the parent ViewGroup savedInstanceState – the Bundle to restore the view states Returns View – the right drawer view Initializes the you liked playlist views.
void	**onResume()** Called when the fragment resumes. Refreshes the YouLikePlayListView.

view.fragment.YouMayLikePlayListFragment

This fragment is used to initialize the you may like playlist fragment.

Fields

String	TAG	for logging
TextView	title	the TextView to hold the title
LanguageButton	languageButton	the button to change the you may like videos' language

Functions

	YouMayLikePlayListFragment() Creates the YouMayLikePlayListView and LanguageButton.
View	**onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)** inflater – the LayoutInflater to create the view container – the parent ViewGroup savedInstanceState – the Bundle to restore the view states Returns View – the right drawer view Initializes the you may like playlist views and the language button.
void	**onResume()** Called when the fragment resumes. Refreshes the YouMayLikePlayListView.
void	**update()** Used to update the image of the language button when the language is changed.

uicomponent.MvRockUIComponent

This class holds the UI components of the MvRockFragment in static variables for easy access.

Fields

YouMayLikePlayListView	YouMayLikePlayListView	holds an instance
YouLikedPlayListView	YouLikedPlayListView	holds an instance
StationPlayListView	StationPlayListView	holds an instance
MvRockYoutubePlayerFragment	MvRockYoutubePlayer	holds an instance
MvRockTabHost	MvRockTabHost	holds an instance
LeftDrawerToggle	LeftDrawerToggle	holds an instance
StationCancelButton	StationCancelButton	holds an instance
StationListView	StationListView	holds an instance
StationSearchView	StationSearchView	holds an instance
SearchStationListView	SearchStationListView	holds an instance
RightFloatingMenu	RightFloatingMenu	holds an instance
MvRockDrawer	MvRockDrawer	holds an instance
SongView	songView	holds an instance
ArtistView	artistView	holds an instance
ToolbarView	toolbarView	holds an instance
CommentView	commentView	holds an instance

uicomponent.MvRockUiComponentObject

This abstract class is a template for the UI components in MvRockFragment. All children must implement the function Init() which is used to initialize the component.

Fields

String	TAG	for logging

Functions

	MvRockUiComponentObject() Appends UiComponent. to TAG
void	**Init()** This is an abstract function implements by the descendants to initialize themselves.

uicomponent.drawer.ExpandableListAdapter

This adapter is used to populate the data in the drawer by creating the views necessary.

Fields

Context	_context	holds a Context instance
List<String>	_listDataHeader	holds a list of strings that represent the titles of each view in the expandable list view
HashMap<String, List<String>>	_listDataChild	holds the list of strings for each group view listed in _listDataHeader

Functions

	ExpandableListAdapter(Context context, List<String> listDataHeader, HashMap<String, List<String>> listChildData) context – a Context instance listDataHeader – the String names for the groups listChildData – the String names for the children in the groups The constructor. Stores the parameters in fields.
Object	**getChild(int groupPosition, int childPosition)** groupPosition – the group to select childPosition – the child to select Returns Object – the child to return Returns the selected child, which in this case is a String object.
long	**getChildId(int arg0, int arg1)** arg0– the group to select arg1– the child to select Returns long – the child Id

	Returns the selected child Id.
View	**getChildView(int groupPosition, int childPosition, boolean isLastChild, View convertView, ViewGroup parent)** groupPosition – the group to select childPosition – the child to select isLastChild – whether this is the last child view convertView – the view to use for this child parent – the parent ViewGroup Returns View – the child view Returns the selected child view and if it does not exist, creates it.
int	**getChildrenCount(int arg0)** arg0 – the group to select Returns int – the number of children Returns the number of children in the group.
Object	**getGroup(int arg0)** arg0 – the group to select Returns Object – the selected group Returns the group view.
int	**getGroupCount()** Returns int – the number of groups Returns the number of groups.
long	**getGroupId(int arg0)** arg0 – the group to select Returns long – the group Id Returns the selected group Id
View	**getGroupView(int groupPosition, boolean isExpanded, View convertView,**

	ViewGroup parent) groupPosition – the group to select isExpanded – whether the selected group is expanded convertView – the view to use for this group parent – the parent ViewGroup Returns View – the group view Returns the selected group view and if it does not exist, creates it.
boolean	**hasStableIds()** Returns whether the views in this adapter has stable Ids (non-changing). Returns false.
boolean	**isChildSelectable(int arg0, int arg1)** arg0 – the group to select arg1 – the child to select Return whether the given child can be selected. Returns true.

uicomponent.drawer.LeftDrawerToggle

This class handles the toggling action of the left drawer in MvRockFragment.

Fields

String	TAG	for logging
Activity	activity	holds a reference to MainActivity

Functions

	LeftDrawerToggle(Activity activity, DrawerLayout drawerLayout, int drawerImageRes, int openDrawerContentDescRes, int closeDrawerContentDescRes) activity – the MainActivity drawerLayout – the DrawerLayout for this toggle drawerImageRes – the image resource Id openDrawerContentDescRes – the resource Id for the content description closeDrawerContentDescRes – the resource Id for the content description Initializes the drawer toggle. Stores an instance of MainActivity in a field.
void	**onDrawerClosed(View view)** view – the drawer view Called when the drawer is closed.
void	**onDrawerOpened(View drawerView)** drawerView – the drawer view Called when the drawer is opened. Sets the action bar title to MvRock.

uicomponent.drawer.LeftTopDrawer

The class represents the left drawer in MvRockFragment. This is used to access what is showing, the user's stations, and the logout button.

Fields

DrawerLayout	leftDrawerLayout	holds an instance of the left drawer
ArrayList<String>	leftDrawerDirectory	holds the list of group names to be used in the adapter
HashMap<String, List<String>>	leftDrawerChildDirectory	holds the list of child names to be used in the adapter
ExpandableListAdapter	leftDrawerListAdapter	the list adapter used to populate the views in the drawer
ExpandableListView	leftDrawerListview	the expandable list view in the left drawer

Functions

	LeftTopDrawer() Appends LeftTopDrawer to TAG for logging.
void	**Init()** Initializes all fields. Sets onClick listeners to views and sets the LeftDrawerToggle.
void	**FBLogoutByThread()** Starts the thread to logout of FaceBook.
void	**AddStationList()** Sets the children of the user's stations and creates and sets the adapter to populate the expandable list view of the drawer.

uicomponent.drawer.MvRockDrawer

This class models the MvRockDrawer. This is the right drawer in the MvRockFragment. This drawer holds the different play lists: you may like, you liked, and station. This class handles the drawer slide which slides the main content view when either the left or right drawer slides.

Fields

DrawerLayout	mDrawerLayout	this holds the right drawer layout
ActionBarDrawerToggle	mDrawerToggle	this holds the right drawer toggle to handle the drawer actions
FrameLayout	frame	The main view of the MvRockFragment
FrameLayout	leftFragment	The left fragment of the left drawer
FrameLayout	rightFragment	The right fragment of the right drawer
float	lastTranslate	used to hold float value used in the animation of sliding the main content view

Functions

	MvRockDrawer() The default constructor. Does nothing.
void	**Init()** Initializes the drawer toggle and sets the drawer layout.
void	**DrawerSlide(View drawerView, float slideOffset)** drawerView – the drawer view that is sliding slideOffset – how much the drawer is sliding Creates the sliding animation for the main MvRock view as the drawer slides.

uicomponent.player.LanguageButton

This class models the language button to change the you may like videos' language.

Fields

ImageView	languageButton	the ImageView holding the language button

Functions

	LanguageButton() The default constructor. Appends to TAG.
void	**Init()** Sets the onClickListener to show a dialog to change the language.
void	**changeLanguageByThread(int lang)** lang – the index of the language to change to Starts the thread to change the language of the you may like videos.

uicomponent.player.MvRockYoutubePlayerFragment

This fragment holds the YouTube player used to play videos. When videos starts, information about the video is gathered from the database and the thumbs up and thumbs down buttons are updated. When videos end, the next video is played. When the player is first initialized, the first video is played.

Fields

String	TAG	for logging
YouTubePlayer	YouTubePlayer	the YouTube player used for YouTube video playback

Functions

void	**Init()** Initializes the YouTube player fragment with the developer key and assigns the PlayerStateChangeListener to handle video playback events.
void	**updateCurrentSong()** Called when a new video is loaded. This updates all fields in CurrentSong with information from the new video. It also updates the views to reflect the new information and starts the GetNewSongDataThread to retrieve the new video information. Hides or shows the recommendation views as necessary.

uicomponent.player.NextSongButton

This class holds the ImageView for the next song button. Clicking on this ImageView will play the next song in the play list. This has the same function for when a video ends.

Fields

ImageView	nextSongImage	view containing the next song image

Functions

	NextSongButton() Appends NextSongButton to TAG for logging
void	**Init()** Initializes the ImageView by setting the onClickListener to play the next video

uicomponent.player.PlayerControlButton

This is an abstract class that models a player control button.

Functions

void	**PostRatingByThread(int flag)** flag – the rating; 1 for liked, 0 for neutral, and -1 for disliked Creates and starts the thread to set the user's rating on the current video and stores the rating in the database.
void	**playNextSongAfterRemovedASongFromYoulikedList()** Plays the next song after removing a video from the you liked list.

uicomponent.player.ReportButton

This class models the report button and is used to report a video for inappropriate content.

Fields

ImageView	reportSongImage	view holding the report image

Functions

	ReportButton() Appends ReportButton to TAG for logging.
void	**Init()** Initializes the report button by setting the onClickListener to open the report dialog.
void	**showReportDialog()** Creates and shows the report AlertDialog. Sends the message to the MvRock dialog for storage and subsequently sends an notification email of the report.

uicomponent.player.SendSongButton

This class models the send song button to send the video to selected Facebook friends.

Fields

ImageView	sendSongImage	view holding the send song image

Functions

	SendSongButton() Appends SendSongButton to TAG for logging.
void	**Init()** Initializes the send song button by setting the onClickListener to start the Facebook message dialog to send the video to selected friends.

uicomponent.player.ShareButton

This class models the share button used to share the current video.

Fields

ImageView	shareSongImage	view holding the share image

Functions

	ShareButton() Appends ShareButton to TAG for logging.
void	**Init()** Initializes the share button by setting the onClickListener to start the Facebook share dialog and to start the share thread for MvRock.
void	**shareMvRockByThread()** Creates and starts the thread to share the video in MvRock by storing it in the database.

uicomponent.player.ThumbDownButton

This class models the thumbs down button used to dislike the video. When the button is clicked, the rating is sent to the database and the thumbs up and thumbs down Drawables are updated to reflect the new rating. If the song removed is from the you liked list, the next song is played.

Fields

ImageView	dislikeSongImage	view holding the dislike image

Functions

	ThumbDownButton () Appends ThumbDownButton to TAG for logging.
void	**Init ()** Initializes the thumbs down button by setting the onClickListener. The listener starts the thread to store the rating in the database and updates the thumbs up and down Drawables. It also removes the video from the you liked list.

uicomponent.player.ThumbUpButton

This class models the thumbs up button used to like videos. When the button is clicked, the rating is sent to the database and the thumbs up and thumbs down drawables are updated to reflect the new rating. The song is then added to the you liked list.

Fields

ImageView	likeSongImage	view holding the like image

Functions

	ThumbUpButton () Appends ThumbUpButton to TAG for logging.
void	**Init ()** Initializes the thumbs up button by setting the onClickListener. The listener starts the thread to store the rating in the database and updates the thumbs up and down Drawables. It also adds the video to the you liked list.
void	**getOneRecSongByThread()** Starts the function to get one recommended song after liking a song on the you may like list.

uicomponent.playlist.ArtistView

This models the artist section in MvRockFragment. It displays the artist image and the artist name.

Fields

TextView	artistNameView	the TextView holding the artist name
ImageView	artistImageView	the ImageView holding the artist image
Button	subscribeButton	the Button used to subscribe to the artist

Functions

void	**Init ()** Sets the onClickListener for the subscribe button.
void	**update ()** Updates artistNameView, artistImageView, and subscribeButton with CurrentSong information.

uicomponent.playlist.CommentListAdapter

This is the adapter used to populate the NonScrollListView used for the comments.

Fields

ArrayList<Drawable>	authorPicture	the list holding the commenter's image
int	VIEW_TYPE	1; constant used for distinguishing a view
int	TYPE_1	0; constant used for distinguishing a view
LayoutInflater	inflater	used to inflate a view

Functions

void	**CommentListAdapter(ArrayList<String> userID)** userID – the list of the commenters' user ids Initializes inflater and grabs the commenters' images from userID and stores them in authorPicture
int	**getCount()** Returns the number of comments
Object	**getItem(int position)** position – the index to retrieve the object Returns Object – the list object at the specified position
long	**getItemId(int position)** position – the index to retrieve from Returns long – the item's id
View	**getView(int position, View convertView, ViewGroup parent)** position – the index to retrieve from convertView – the view to use parent – the parent ViewGroup Returns View – the view at the position specified; creates it if necessary

int	**getViewTypeCount()** Returns int – the VIEW_TYPE constant
int	**getItemViewType(int position)** position – the position to retrieve from Returns int – the TYPE_1
ArrayList<Drawable>	**getAuthorAvatar(ArrayList<String> userID)** userID – the list of the commenters’ user ids Returns ArrayList<Drawable> – the list of profile pictures Starts the thread to retrieve the profile pictures of the user ids.

uicomponent.playlist.CommentView

This models the comments section in the MvRockFragment. It displays the comments on the video and handles sending new comments to the database.

Fields

ArrayAdapter<String>	userNameArray	the array holding the usernames of the commenters
NonScrollListView	commentList	the comment list
TextView	commentNumber	the TextView holding the number of comments
ImageView	userAvatar	the ImageView holding the user's profile picture
MultiAutoCompleteTextView	textInput	the view that allows the user to type a new comment

Functions

	CommentView() Initializes some fields.
void	**Init()** Initializes by settings the user's profile picture to userAvatar, setting the number of comments to commentNumber, and assigning listeners.
void	**setNewComment(String str)** str – the new comment Starts the thread to send the comment.
void	**showComment()**

	Function that shows the comments.
void	**getTheNewInfoAfterSetComment()** Creates and starts the GetNewSongDataThread to retrieve the newly updated comments after submitting a comment.
void	**update()** Updates the number of comments TextView.

uicomponent.playlist.MvRockTabHost

This models the TabHost object used to switch between play lists. There are two tabs: You May Like and You Liked. The You May Like tab can turn into a Stations tab when a station is selected.

Fields

TabHost	TabHost	stores the TabHost instance

Functions

	MvRockTabHost () The default constructor appends MvRockTabHost to TAG and store a new TabHost instance in the TabHost field.
void	**Init ()** Does nothing. Commented out.

uicomponent.playlist.NonScrollListView

This models the list view used for the comments.

Functions

	NonScrollListView(Context context) context – a Context instance Passes on to parent constructor.
	NonScrollListView(Context context, AttributeSet attrs) context – a Context instance attrs – the attributes retrieved from xml Passes on to parent constructor.
	NonScrollListView(Context context, AttributeSet attrs, int defStyle) context – a Context instance attrs – the attributes retrieved from xml defStyle – the default style Passes on to parent constructor.
void	**onMeasure(int widthMeasureSpec, int heightMeasureSpec)** widthMeasureSpec – horizontal space requirements as imposed by the parent heightMeasureSpec – vertical space requirements as imposed by the parent Function called when measuring the size of the list view. Used to implement non-scrolling in this case.

uicomponent.playlist.PlaylistAdapter

This adapter is used to populate the list views for a song list. It creates the views for each song and displays them in the list.

Functions

	PlaylistAdapter (Context context, ArrayList<Map<String, String>> song_info, String[] from, int[] to) context – a Context instance song_info – list of all the song's information from – array of the map keys to select data from to – array of view Ids to assign the data to Appends UIComponent. to TAG and calls the parent constructor.
View	**getView (int position, View convertView, ViewGroup parent)** position – the position of the view in the ListView convertView – the view to use parent – the parent ViewGroup Returns View – the view at the position in the ListView Inflates the view if it does not exist and sets the song's name and artist's name. Returns the view.

uicomponent.playlist.PlayListView

This abstract class models a playlist list view. The purpose of this of this class is to specify common fields and functions in all playlists to create coherency.

Fields

ListView	playListview	the list view to hold the playlist
Context	context	holds an instance of Context

Functions

	PlayListView(Context context) context – a Context instance Initializes the playlist and set the TAG to UIComponent. for logging.
boolean	**isAvailable()** Returns boolean – whether the playlist is available Returns whether the playlist is available.
void	**setAvailable()** Sets the playlist to be available.
void	**RequestPlayListByThread()** Creates and starts the thread to fetch a playlist from the database.
void	**RefreshListView()** Refreshes the list view to show changes.
void	**Init()** Initializes the playlist list view.

ArrayList<Drawable>	**RequestImageListByThread(List<Map<String, String>> song_info)** song_info – the list of song information Starts the thread to retrieve the video images from YouTube and returns the list of the images when done.

uicomponent.playlist.RightFloatingMenu

This models the 3rd party library CircularFloatingActionMenu. This “menu” handles the right drawer for the MvRockFragment.

Fields

FloatingActionButton	actionButton	the floating button on the bottom right of the screen; it used to slide open the right drawer and to switch between the different play lists
ImageView	youMayLikePlayListButton	the image view used for the content of the SubActionButton for the YouMayLike button; holds the YouMayLike drawable
ImageView	youLikedPlayListButton	the image view used for the content of the SubActionButton for the YouLiked button; holds the YouLiked drawable
ImageView	stationPlayListButton	the image view used for the content of the SubActionButton for the Station button; holds the Station drawable
FloatingActionMenu	actionMenu	the action menu that holds the three SubActionButton and attached to actionButton to facilitate play list switching

Functions

	RightFloatingMenu() Creates an ImageView and assigns it to RightDrawerControlButton.
void	**Init()**

	Initializes all the fields; attaches an onStateChangeListener to actionMenu to open the right drawer when the action button is pressed; attaches onClickListeners to the three SubActionButton buttons to switch between the three playlists.

uicomponent.playlist.SongView

This models the song section in MvRockFragment. It displays the song name and the recommendation reason.

Fields

TextView	songNameView	the TextView holding the song name
TextView	recommendationTitleView	the TextView holding the recommendation title
TextView	recommendationReasonView	the TextView holding the recommendation reason

Functions

void	**Init ()** Does nothing.
void	**update ()** Updates songNameView and recommendationReasonView with CurrentSong information.
void	**hideRecommendation()** Hides the recommendation title and recommendation reason views.
void	**showRecommendation()** Shows the recommendation title and recommendation reason views.

uicomponent.playlist.StationPlayListAdapter

This adapter creates the views necessary for the station playlist. The station playlist uses the same list view as the you may like list view.

Functions

	StationPlayListAdapter(Context context, String[] from, int[] to) context – a Context instance from – the map keys to get information from to – the view Ids to put the information to Initiates the adapter by setting the TAG and calls the parent constructor.
View	**getView(int position, View convertView, ViewGroup parent)** position – the position of the view in the ListView convertView – the view to use parent – the parent ViewGroup Returns View – the view at the specified position Sets the song image and returns the view.

uicomponent.playlist.StationPlayListView

This class models the station playlist list view. Functions include requesting the playlist from the database and refreshing the list view for new content.

Functions

	StationPlayListView() Appends to TAG for logging and passes MainActivity to parent.
boolean	**isAvailable()** Returns boolean – whether the list view is selected Returns whether the station list view is selected.
void	**setAvailable()** Sets station list view as selected.
void	**RequestPlayListByThread()** Creates and starts the thread for fetching the station song list from the database.
void	**RefreshListView()** Starts the thread to download the playlist images. Creates an adapter to populate the list view.
void	**Init()** Initializes the station playlist list view by calling RefreshListView. Sets the onItemClickListener to play the selected video when an item on the list view is pressed.

uicomponent.playlist.ToolbarView

This class models the video toolbar.

Fields

TextView	thumbUpNumber	the TextView holding the number of likes
TextView	thumbDownNumber	the TextView holding the number of dislikes
NextSongButton	nextSongButton	the next song button
ThumbUpButton	thumbUpButton	the thumb up button
ThumbDownButton	thumbDownButton	the thumb down button
ShareButton	shareButton	the share button
SendSongButton	sendSongButton	the send song button
ReportButton	reportButton	the report button

Functions

	ToolbarView() Appends to TAG for logging and instantiates the fields.
void	**Init()** Initializes the fields.
void	**update()** Updates the number of likes and dislikes TextViews. Updates the buttons to either the colored or non-colored versions.

uicomponent.playlist.YouLikedPlayListAdapter

This adapter populates the you liked playlist list view.

Functions

	YouLikedPlayListAdapter (Context context, String[] from, int[] to) context – a Context instance from – the map keys to get information from to – the view Ids to put the information to Initiates the adapter by setting the TAG and calls parent constructor.
View	**getView(int position, View convertView, ViewGroup parent)** position – the position of the view in the ListView convertView – the view to use parent – the parent ViewGroup Returns View – the view at the specified position Sets the song image and returns the view.

uicomponent.playlist.YouLikedPlayListView

This class models the you liked playlist list view. Functions include requesting the playlist from the database and refreshing the list view for new content.

Functions

	YouLikedPlayListView() Appends to TAG for logging and passes MainActivity to parent.
boolean	**isAvailable()** Returns boolean – whether the list view is selected Returns whether the you liked list view is selected.
void	**setAvailable()** Sets you liked list view as selected.
void	**RequestPlayListByThread()** Creates and starts the thread for fetching the you liked song list from the database.
void	**RefreshListView()** Starts the thread to download the playlist images. Creates an adapter to populate the list view.
void	**Init()** Initializes the you liked playlist list view by calling RefreshListView. Sets the onItemClickListener to play the selected video when an item on the list view is pressed.

uicomponent.playlist.YouMayLikePlayListAdapter

This adapter populates the you may like playlist list view.

Functions

	YouMayLikePlayListAdapter(Context context, String[] from, int[] to) context – a Context instance from – the map keys to get information from to – the view Ids to put the information to Initiates the adapter by setting the TAG and calls the parent constructor.
View	**getView(int position, View convertView, ViewGroup parent)** position – the position of the view in the ListView convertView – the view to use parent – the parent ViewGroup Returns View – the view at the specified position Sets the song image and returns the view.

uicomponent.playlist.YouMayLikePlayListView

This class models the you may like playlist list view. Functions include requesting the playlist from the database and refreshing the list view for new content.

Functions

	YouMayLikePlayListView() Appends to TAG for logging and passes MainActivity to parent.
boolean	**isAvailable()** Returns boolean – whether the list view is selected Returns whether the you may like list view is selected.
void	**setAvailable()** Sets you may like list view as selected.
void	**RequestPlayListByThread()** Creates and starts the thread for fetching the you may like song list from the database.
void	**RefreshListView()** Starts the thread to download the playlist images. Creates an adapter to populate the list view.
void	**Init()** Initializes the you may like playlist list view by calling RefreshListView. Sets the onItemClickListener to play the selected video when an item on the list view is pressed.

uicomponent.station.SearchStationListAdapter

This is the adapter used to populate the search station list.

Fields

String	TAG	for logging

Functions

	StationListAdapter (Context context, ArrayList<Map<String, String>> station_info, String[] from, int[] to) context – a Context instance station_info – list of all the stations' information from – array of the map keys to select data from to – array of view Ids to assign the data to Appends UIComponent. to TAG and calls the parent constructor.
View	**getView (int position, View convertView, ViewGroup parent)** position – the position of the view in the ListView convertView – the view to use parent – the parent ViewGroup Returns View – the view at the position in the ListView Inflates the view if it does not exist and sets the station's name and the subscribe button to the views. The view is then returned.

uicomponent.station.SearchStationListView

This class holds the list view for the station search results.

Fields

ListView	SearchStationListview	the list view for the station search results
TextView	noSearchResults	the TextView to show when there are no results
boolean	hasResults	whether there are any results

Functions

	SearchStationListView() Appends to TAG for logging.
void	**Init()** Logs Init() and initially hides the search station view.
void	**RefreshListView()** Creates the adapter to populate the station search results. If there are no results, the no results text view is shown.

uicomponent.station.StationCancelButton

This class models a station cancel button. This cancels a station playing and turns the first tab in the tab host back into You May Like.

Fields

ImageView	stationCancelImage	the image view used to hold the cancel station image

Functions

	StationCancelButton() Appends to TAG for logging.
void	**Init()** Initializes the station cancel button and sets the onClickListener to convert the tab host's first tab back into the You May Like playlist.

uicomponent.station.StationListAdapter

This is the adapter used to populate the station list.

Fields

String	TAG	for logging

Functions

	StationListAdapter (Context context, ArrayList<Map<String, String>> station_info, String[] from, int[] to) context – a Context instance station_info – list of all the stations' information from – array of the map keys to select data from to – array of view Ids to assign the data to Appends UIComponent. to TAG and calls the parent constructor.
View	**getView (int position, View convertView, ViewGroup parent)** position – the position of the view in the ListView convertView – the view to use parent – the parent ViewGroup Returns View – the view at the position in the ListView Inflates the view if it does not exist and sets station's image. The view is then returned.

uicomponent.station.StationListView

This class models the station list view.

Fields

ListView	StationListView	the list view used to hold the station search results
TextView	noStations	the TextView to show if there are no stations

Functions

	StationListView() Appends to TAG for logging.
void	**Init()** Initializes the station search list view and sets the onItemClickListener. The listener calls the RequestPlayListByThread to get the station songs and shows the StationPlayListFragment. The first song is then played.
void	**RefreshListView()** Creates the adapter if necessary for the station list view and refreshes the list view.
void	**RequestStationByThread()** Creates and starts the thread to fetch the selected station.
void	**CreateStationByThread(String stationName)** stationName – the station name Creates and starts the thread to create the station.
ArrayList<Drawable>	**RequestStationImageListByThread(List<Map<String, String>> song_info)** song_info – the list of song information

	Starts GetStationImageListThread to retrieve the station images and returns them in a map.

uicomponent.station.StationSearchView

This class models the search view in the action bar that is used to search for stations.

Fields

SearchView	topSearchView	the SearchView view used for searching stations

Functions

	StationSearchView() Appends to TAG for logging.
void	**Init()** Initializes the station search view. Sets an onQueryTextListener to listen for when the search button is submitted. Also sets the setOnCloseListener to reshow the station list when searching is exited.
void	**RequestSearchStationResultByThread ()** Creates and starts the thread responsible for searching for stations by retrieving data from the database.

thread.ChangeLanguageThread

This thread changes the language of the video selection algorithm. ALL means both English and Chinese videos. ENG means English videos. CHN means Chinese videos.

Functions

	ChangeLanguageThread() Appends to TAG for logging and sets the Url.
void	**setParams()** Sets the parameters for post when the thread connects to the PHP file.
void	**setResponse()** Implemented from parent abstract class. Not used.

thread.CreateStationThread

This thread creates the station chosen from the station search view.

Functions

	CreateStationThread() Appends to TAG for logging and sets the Url.
void	**setParams()** Sets the parameters for post when the thread connects to the PHP file.
void	**setResponse()** Implemented from parent abstract class. Not used.

thread.FacebookLogoutThread

This thread logs the user out of FaceBook. THIS THREAD IS PROBABLY NOT NECESSARY ANYMORE.

```
LoginManager loginManager = LoginManager.getInstance();

loginManger.logOut();
```

The above code should be sufficient for logging out.

Fields

String	TAG	for logging

Functions

	FacebookLogoutThread() Does nothing.
void	**run()** The function that is run when the thread starts. Simply logs the user out of FaceBook.

thread.GetArtistImageThread

This thread retrieves all the images in a song list and stores the Drawables.

Fields

String	TAG	for logging
ArrayList<Drawable>	artistImages	the list of artist images
ArrayList<Map<String, String>>	artistsList	the list of maps holding the artist images' url

Functions

	GetArtistImageThread(ArrayList<Drawable> artistImages, JSONArray imageUrls) artistImages – the ArrayList to store the download images imageUrls – the JSONArray holding all the URLs to retrieve the images Stores the parameters in fields. Initializes the artistsList.
void	**run()** Calls the cache's getImageFromCache function and stores the downloaded images in artistImages.

thread.GetBuddyFeedThread

This thread retrieves the buddy feed JSON strResponse.

Functions

	GetBuddyFeedThread(String User_id, String Extra) User_id – the user’s id Extra – nothing Stores the user’s id and appends to TAG and Url.
void	**setParams()** Sets the parameters for post when the thread connects to the PHP file.
void	**setResponse()** Sets the HTTP strResponse to the BuddyFeed class.

thread.GetImageListThread

This Runnable is used in a thread to gather the images for a song list and stores the Drawable images in a map.

Fields

String	TAG	for logging
ArrayList<Drawable>	ImageView_List	used to store the Drawable images for the song list
List<Map<String, String>>	song_info	the song list information

Functions

	GetImageListThread() Initializes the thread.
ArrayList<Drawable>	**getImageView_List()** Returns ArrayList<Drawable> - the map holding all the video Drawables Returns the list of video images.
void	**run()** Gets the image from the cache and stores them in ImageView_List.

thread.GetMusicBuddyThread

This thread retrieves the music buddy JSON strResponse.

Functions

	GetMusicBuddyThread(String User_id, String Extra) User_id – the user's id Extra – nothing Stores the user's id and appends to TAG and Url.
void	**setParams()** Sets the parameters for post when the thread connects to the PHP file.
void	**setResponse()** Sets the HTTP strResponse to the MusicBuddy class.

thread.GetNewSongDataThread

This thread retrieves information for the currently playing song, such as the user's rating for the song.

Functions

	GetNewSongDataThread() Appends to TAG for logging and sets the Url.
void	**setParams()** Sets the parameters for post when the thread connects to the PHP file.
void	**setResponse()** Sets the HTTP strResponse to the CurrentSong class.

thread.GetOneRecSongThread

This thread retrieves one recommended song from the database. Used when the user likes a song on the you may like list.

Functions

	GetOneRecSongThread(String userId, String url) userId – the user Id url – the url Calls the parent constructor and appends to TAG and Url.
void	**run()** Retrieves the song data from the database, sets the response, and converts the response into data. Allows this thread to be run asynchronously.
void	**setResponse()** Sets the HTTP strResponse to the YouMayLikeSongList class.
void	**setParams()** Sets the parameters for post when the thread connects to the PHP file.

thread.GetProfilePicThread

This thread retrieves the user's Facebook profile picture.

Fields

String	TAG	for logging

Functions

void	**run()** Retrieves the user's profile picture from Facebook.

thread.GetRecBuddyThread

This thread retrieves the recommended buddy JSON strResponse.

Functions

	GetRecBuddyThread(String User_id, String Extra) User_id – the user's id Extra – nothing Stores the user's id and appends to TAG and Url.
void	**setParams()** Sets the parameters for post when the thread connects to the PHP file.
void	**setResponse()** Sets the HTTP strResponse to the RecBuddy class.

thread.GetSearchStationThread

This thread retrieves the station search results from the database.

Functions

	GetSearchStationThread(String User_id, String searchStr) User_id – the user Id searchStr – the search query Calls the parent constructor and appends to TAG and Url.
void	**setResponse()** Sets the HTTP strResponse to the SearchStationList class.
void	**setParams()** Sets the parameters for post when the thread connects to the PHP file.

thread.GetStationImageListThread

This thread retrieves a station’s images.

Fields

String	TAG	for logging
ArrayList<Drawable>	ImageView_List	the map holding all the images
List<Map<String, String>>	song_info	the list of all song information

Functions

	GetStationImageListThread(List<Map<String, String>> song_info, Context context) song_info – the list of song information context – a Context instance Initializes fields and stores parameters in fields.
ArrayList<Drawable>	**getImageView_List()** Returns ArrayList<Drawable> - the list of station images Returns ImageView_List.
void	**run()** Gets the images from the cache and stores them in ImageView_List.

thread.GetStationSongsThread

This thread retrieves a station’s songs.

Functions

	GetStationSongsThread() Appends to TAG for logging and sets the Url.
void	**setParams()** Sets the parameters for post when the thread connects to the PHP file.
void	**setResponse()** Sets the HTTP strResponse to the StationSongList class.

thread.GetStationThread

This thread retrieves the matching stations for the search query in the station search view.

Functions

	GetStationThread() Appends to TAG for logging and sets the Url.
void	**setParams()** Sets the parameters for post when the thread connects to the PHP file.
void	**setResponse()** Sets the HTTP strResponse to the StationList class.

thread.GetUserProfilePicture

This thread retrieves the profile pictures from a list of user ids.

Fields

String	TAG	for logging
ArrayList<Drawable>	userPicture	the list to store the user profile pictures
List<Map<String, String>>	userIDMap	the list of maps to use for the cache image retrieval function

Functions

	GetUserProfilePicture(ArrayList<Drawable> userPicture, ArrayList<String> userID) userPicture – the list to store the images in ArrayList<String> - the list of user ids Appends to TAG and stores the parameters in the fields.
void	**run()** Grabs the profile pictures from the cache.

thread.GetYouLikedSongAndUserDataThread

This thread retrieves the user’s you liked songs.

Functions

	GetYouLikedSongAndUserDataThread() Appends to TAG for logging and sets the Url.
void	**setParams()** Sets the parameters for post when the thread connects to the PHP file.
void	**setResponse()** Sets the HTTP strResponse to the YouLikedSongList class.

thread.GetYoumaylikePlayListThread

This thread retrieves the data for the you may like play list.

Functions

	GetYoumaylikePlayListThread() Appends to TAG for logging and sets the Url.
void	**setParams()** Sets the parameters for post when the thread connects to the PHP file.
void	**setResponse()** Sets the HTTP strResponse to the YouMayLikeSongList class.

thread.MvRockThreadObject

This abstract class is the model for most of the MvRock threads. Depending on the connected PHP file, this thread either stores data in the database or retrieves information from it as a String which can be used to create a JSONArray.

Fields

String	HOST	the Poly wanlab url
String	PATH	the directoy in the Poly wanlab to use
String	TAG	for logging
List<NameValuePair>	params	the parameters used to connect to the PHP file
String	User_id	the user's Id
String	Extra	a field to store extra information
String	Url	used to store the path to the PHP and later concatenated into the full url
String	strResponse	the information retrieved from the database; can be used to create a JSONArray

Functions

	MvRockThreadObject() Initializes the fields.
void	**run()** Creates the full url from HOST, PATH, and Url and connects to it. Stores the response if any to strResponse.
void	**setParams()** Abstract function that must be implemented by children. Used to set the parameters needed for post.

void	**setResponse()** Abstract function that must be implemented by children. Used to set the HTTP response to the correct to be turned into a JSONArray later.

thread.RemoveStationThread

This thread removes the station.

Functions

	RemoveStationThread(String stationName, String User_id) stationName – the station’s name User_id – the user’s name Calls parent constructor and appends to TAG and Url.
void	**setParams()** Sets the parameters for post when the thread connects to the PHP file.
void	**setResponse()** Does nothing.

thread.ReportThread

This thread reports the video. The PHP file subsequently sends an email notification to the MvRock email address.

Functions

	RemoveStationThread(String userId, String songUrl, String comment) userId – the user Id songUrl – the YouTube video Id comment – the user's comment on the problem Calls parent constructor and appends to TAG and Url. Stores parameters.
void	**setParams()** Sets the parameters for post when the thread connects to the PHP file.
void	**setResponse()** Does nothing.

thread.SetCommentThread

This thread stores the user's new comment in the database.

Fields

String	songUrl	the YouTube video Id
String	replyTo	the person being replied to

Functions

	SetCommendThread(String User_id, String Extra, String songUrl, String replyTo) User_id – the user Id Extra – the comment songUrl – the YouTube video Id replyTo – the person being replied to Stores parameters in fields.
void	**run()** Calls parent run.
void	**setResponse()** Does nothing.
void	**setParams()** Sets the parameters for post when the thread connects to the PHP file.

thread.SetRatingThread

This thread stores the user's rating in the database when they press either the thumbs up or thumbs down button.

Fields

int	rating	the rating to set; neutral is 0; liked is 1; disliked is -1

Functions

	SetRatingThread(String User_id, String Song_Url, int rating) User_id – the user Id Song_Url – the YouTube video Id rating – the rating Appends to TAG for logging and sets the Url. Stores parameters.
void	**setParams()** Sets the parameters for post when the thread connects to the PHP file.
void	**setResponse()** Not used.

thread.SetShareThread

This thread shares the video by setting it to be shared in the database.

Fields

String	fromChannel	The index where the video came from

Functions

	SetShareThread(String User_id, String Song_Url, String fromChannel) User_id – the user Id Song_Url – the YouTube video Id fromChannel – the index where the video came from Appends to TAG for logging and stores the parameters.
void	**setParams()** Sets the parameters for post when the thread connects to the PHP file.
void	**setResponse()** Not used.

model.buddy.BuddyFeed

This class models the buddy feed. It holds all the user’s buddies.

Fields

Set<String>	musicBuddy	the set holding all the user buddies’ usernames

Functions

	BuddyFeed() Initializes the musicBuddy field and appends to TAG.
void	**convertData()** Retrieves the user buddies’ usernames from the JSON strResponse and adds them to the set

model.buddy.MusicBuddy

This class models a music buddy (MvRock friend).

Fields

ArrayList<String>	userName	the list holding all the user buddies' usernames
ArrayList<String>	uID	the list holding all the user's buddies' user ids
ArrayList<String>	vip	the list holding all the user's buddies' vip data

Functions

	MusicBuddy() Initializes the fields and appends to TAG.
void	**convertData()** Retrieves the user's buddies' information from the JSON strResponse and stores them in the fields

model.buddy.RecBuddy

This class models a recommended buddy.

Fields

ArrayList<String>	buddyName	the list holding the buddies’ names
ArrayList<String>	buddyUid	the list holding the buddies’ user ids

Functions

	RecBuddy() Initializes the fields and appends to TAG.
void	**convertData()** Converts information from strResponse and stores them in the fields.

model.buddy.User

This class models a user. It is used to hold user information.

Fields

AccessToken	accessToken	the session's access token
Profile	profile	the user's profile
String	User_Id	holds the user id
String	User_Name	holds the username
Drawable	User_Profile_pic	holds the user's profile picture
int	skin	the index of the skin; not used
int	language	the index of the you may like videos' language
int	accuPoint	the index of the accuPoint; not used

Functions

	User() Initializes all the User fields.
void	**getProfilePicByThread()** Creates and starts the thread that retrieves the user's profile picture from Facebook.
void	**getUserDataByThread()** Creates and starts the thread that retrieves the user's information from the database
void	**convertData()** Converts the strResponse from getUserDataByThread into skin, language, and accuPoint.

model.song.CurrentSong

This class holds information for the current song. It holds whether or not the user liked or disliked the current song.

Fields

boolean	isLikedIconPressed	whether the user liked the video
boolean	isDislikedIconPressed	whether the user disliked the video
boolean	isShared	whether the user shared the video
boolean	hasSentSong	whether the user sent this song recently
boolean	isReported	whether the user reported the video recently
boolean	isChanged	if the current song changed
boolean	isArtistSubscribed	whether the user has subscribed to the artist
int	currentMVIndex	the current song index
String	url	the current video YouTube Id
String	songName	the current video's name
String	artistName	the current artist's name
String	rootShareUserId	the user that originally shared the video
int	currentTime	the current time of video playback
ReasonOption	reason	the reason the video is recommended; used for you may like videos
int	numberOfComments	the number of comments for the video
int	numLikes	the number of likes for the video
int	numDislikes	the number of dislikes for the video
int	songId	the song id of the video in the MvRock database
ArrayList<String>	commentAuthor	the list holding all the comments' usernames
ArrayList<String>	commentContent	the list holding all the comments' content
ArrayList<String>	commentTime	the list holding all the comments' time
ArrayList<String>	authorID	the list holding all the comments' user ids
Drawable	artistImage	the current artist's image

Functions

	CurrentSong() Initializes all the CurrentSong fields.
void	**convertData()** Extracts the data from strResponse. Gathers the number of comments, the comments information, the rating, the number of likes, the number of dislikes, whether the user has shared the video, comment information, etc.

model.songlist.SearchStationList

This class models the station search results.

Fields

ArrayList<Map<String, String>>	searchStationArrayList	holds all station results
int[]	subscribeList	holds whether the corresponding station in searchStationArrayList is subscribed or not; 0 is not subscribed, 1 is subscribed

Functions

	SearchStationList() Initializes all fields.
void	**convertData()** Extracts all the stations search results from strResponse. Also fills subscribeList with 0s and 1s depending on the subscribe status of each station.

model.songlist.SongList

This abstract class models a song list. It holds song information for a playlist and the images for the songs.

Fields

ArrayList<Drawable>	imageViewList	holds the songs Drawable images
ArrayList<Map<String, String>>	songArrayList	holds all the song information
ArrayList<Drawable>	artistImages	holds the artist images

Functions

	SongList() Initializes all the SongList fields.
void	**convertData()** Abstract method for all children to implement. It is used to extract information from strResponse, which is usually a JSONObject or JSONArray.

model.songlist.StationList

This class models a station list. This holds all the user’s stations.

Fields

ArrayList<Map<String, String>>	stationArrayList	the list holding all the user’s stations
ArrayList<Drawable>	stationImageArrayList	the list holding all the station images

Functions

	StationList() Appends to TAG and initializes the fields.
void	**convertData()** Extracts the user’s stations and station image URLs from strResponse.
void	**createStationByThread(String stationName)** stationName - station’s name Starts the thread to create the station.
void	**removeStationByThread(String stationName)** stationName - station’s name Starts the thread to remove the station.
boolean	**isSubscribed(String station)** station- station’s name Returns whether the user has subscribed to the station.

model.songlist.StationSongList

This class models a station song list. This holds a stations list of songs.

Functions

	StationSongList() Appends to TAG.
void	**convertData()** Extracts all the station song information from strResponse and stores them in songArrayList.

model.songlist.YouLikedSongList

This class models a you liked song list. This holds all the user's liked songs.

Functions

	YouLikedSongList() Appends to TAG.
void	**convertData()** Extracts all the you liked song information from strResponse and stores them in songArrayList. Starts the thread to get all the artist images.

model.songlist.YouMayLikeSongList

This class models a you may like song list. This holds song recommendations for MvRock.

Functions

	YouMayLikeSongList() Appends to TAG.
void	**convertData()** Extracts all the you may like song information from strResponse and stores them in songArrayList. Starts the thread to get all the artist images.
void	**convertOneRecSongData()** Extracts the one recommended song data from strReponse and adds it the you may like list.

model.Cache

This class models the cache. It is used to retrieve cached song image and artist images. If the images are not in the cache, they will be downloaded and then cached.

Fields

DiskLruCache	DiskLruCache	the cache
String	TAG	for logging
long	timeDifference	the time difference to check before updating the image; if difference between the current time and the last modified time of image is greater than this, the image will be redownloaded
long	testTimeDifference	the time difference to use during testing and debugging

Functions

	Cache() Creates the DiskLruCache.
void	**getImageFromCache(List<Drawable> ImageView_List, List<Map<String, String>> song_info, String prefix, String postfix)** ImageView_List – the list to store the Drawable image in song_info – the list to retrieve the image url from prefix – the prefix of the url postfix – the postfix of the url Retrieves the images from the cache and stores them in ImageView_List. Download the images if necessary
boolean	**imageNeedUpdate(String key)**

	key – the key of the image in the cache Returns whether the image needs to be updated.
Drawable	**getDrawable(DiskLruCache.Snapshot snapshot)** snapshot – the cache’s snapshot of the image Returns the Drawable image from the cache’s snapshot. If there is an error, an image failed Drawable will be returned instead.
File	**getDiskCacheDir(Context context, String uniqueName)** context – a Context instance uniqueName - a uniqueName for the cache directory Returns a directory for the cache.
int	**getAppVersion(Context context)** context – a Context instance Returns the app version.
boolean	**download(String imageUrl, OutputStream outputStream)** imageUrl – the url to download the image outputStream – the stream to output the image data to Downloads the image. Returns whether the download was successful.
String	**hashKeyForDisk(String key)** key – the song url Returns a MD5 key for the songUrl to be used in the cache as a key/value pair.
String	**bytesToHexString(byte[] bytes)** bytes – the array of bytes Returns the hex of the array of bytes.

model.DataInitialization

This class starts some threads concurrently for faster startup time.

Fields

String	TAG	for logging

Functions

	DataInitialization() Sets the Tag and calls init().
void	**Init()** Starts GetStationThread, GetYouLikedSongAndUserDataThread, GetYoumaylikePlayListThread, GetStationSongsThread, GetMusicBuddyThread, GetRecBuddyThread, and GetBuddyFeedThread concurrently. Also sets their strResponses and converts their data.

model.LanguageOption

This class holds the integer values corresponding to the you may like videos' language options.

Fields

int	ALL	(3) – value for all languages
int	ENG	(1) – value for English
int	CHN	(2) – value for Chinese

model.MvRockModel

This class holds all the static MvRock model objects for easy access.

Fields

DataInitialization	dataInitialization	holds an instance
PlayListOption	playListOption	holds an instance
YouLikedSongList	YouLikedSongList	holds an instance
YouMayLikeSongList	YouMayLikeSongList	holds an instance
StationSongList	StationSongList	holds an instance
String	CurrentStation	holds the current station name
StationList	StationList	holds an instance
SearchStationList	SearchStationList	holds an instance
CurrentSong	CurrentSong	holds an instance
MusicBuddy	MusicBuddy	holds an instance
Cache	cache	holds an instance
RecBuddy	RecBuddy	holds an instance
BuddyFeed	BuddyFeed	holds an instance
User	User	holds an instance

model.MvRockModelObject

This abstract class is a template for MvRock models. Children must implement convertData to convert strResponse into useful data.

Fields

String	TAG	for logging
String	strResponse	the HTTP response to create a JSON object to extract information

Functions

	MvRockModelObject() Appends to TAG and initializes fields.
void	**setHttpResponse(String strResponse)** strResponse – the HTTP response Sets the HTTP response to strResponse.
void	**convertData()** Abstract method for the children to implement. It will be used to convert strResponse to a JSON object and extract information from it.

model.PlayListOption

This enum stores the type of playlists. It is used for isAvailable and setAvailable in playlist list view classes.

Types

YOU_MAY_LIKE_LIST	0
YOU_LIKED_LIST	1
STATION_LIST	2

Functions

	PlayListOption(int option) option – the PlayListOption Stores option in the option field. Unused.
String	**toString()** Returns String – the enum as a String Returns the enum as a String.

model.ReasonOption

This enum represents the reason for why MvRock recommends the video in the you may like list.

Types

None	0
Random	1
YouLikedBefore	2
Personalized	3

Functions

	ReasonOption(String option) option – the ReasonOption Stores option in the option field. Unused.
String	**toString()** Returns String – the enum as a String Returns the enum as a String.