package com.sharehub.sharehub.sharehub

import android.content.Intent

import android.os.Bundle

import android.widget.Toast

import com.loopj.android.http.AsyncHttpClient

import com.loopj.android.http.JsonHttpResponseHandler

import com.loopj.android.http.RequestParams

import cz.msebera.android.httpclient.Header

import org.json.JSONObject

import java.io.Serializable

class CommunityLauncher : ShareHubActivityBase() {

class CommunityData : Serializable {

var name: String? = null

var uniqueName: String? = null

var users: Array<String>? = null

var owner: String? = null

var admins: Array<String>? = null

var description: String? = null

var boxKey: Array<String>? = null

var boxLocation: Array<String>? = null

var boxName: Array<String>? = null

}

companion object {

val communityDataMap: HashMap<String, CommunityData> = HashMap()

fun loadCommunity(communityKey: String, context: ShareHubActivityBase, callback: (profileData: CommunityData) -> Unit) {

if (communityDataMap.containsKey(communityKey)) {

callback(communityDataMap[communityKey]!!)

return

}

val client = AsyncHttpClient()

val params = RequestParams()

params.add("userKey", Statics.UserKey)

params.add("uniqueName", communityKey)

client.post(Constants.CommunityGateway.GetCommunity, params, object : JsonHttpResponseHandler() {

override fun onSuccess(statusCode: Int, headers: Array<out Header>?, response: JSONObject) {

try {

if (response["success"] as Boolean) {

val communityJson = response.getJSONObject("community")

val communityData = CommunityData()

communityData.name = communityJson.cast("name")

communityData.uniqueName = communityJson.cast("uniqueName")

communityData.users = communityJson.castArray("users")

communityData.owner = communityJson.cast("owner")

communityData.admins = communityJson.castArray("admins")

communityData.description = communityJson.cast("description")

communityData.boxKey = communityJson.castArray("boxKey")

communityData.boxLocation = communityJson.castArray("boxLocation")

communityData.boxName = communityJson.castArray("boxName")

communityDataMap[communityKey] = communityData

callback(communityData)

}

else {

Toast.makeText(context, "you have been logged out", Toast.LENGTH\_LONG).show()

Statics.logout(context)

context.replaceActivity(Intent(context, Login::class.java))

}

}

catch (e: Exception) {

Toast.makeText(context, "an exception occurred: " + e.message, Toast.LENGTH\_LONG).show()

}

}

override fun onFailure(statusCode: Int, headers: Array<out Header>?, responseString: String?, throwable: Throwable?) {

Toast.makeText(context, "3\_http request error: " + throwable?.message, Toast.LENGTH\_LONG).show()

}

})

}

}

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.community\_launcher)

if (Statics.UserKey == null) {

replaceActivity(Intent(this, Login::class.java))

return

}

val community = intent.getStringExtra("community")

val reload = intent.getBooleanExtra("reload", false)

if(reload){

communityDataMap.remove(community)

}

if (community == null) {

val profileIntent = Intent(this, ProfileLauncher::class.java)

profileIntent.putExtra("user", Statics.Username)

replaceActivity(intent)

}

else {

loadCommunity(community, this, { communityData ->

run {

val intent = Intent(this, Community::class.java)

val bundle = Bundle()

bundle.putSerializable("communityData", communityData)

intent.putExtras(bundle)

replaceActivity(intent)

}

})

}

}

}