package com.wt.yc.oto.oto.activity

import android.app.AlertDialog

import android.os.Bundle

import android.os.Message

import android.support.design.widget.BottomNavigationView

import android.support.design.widget.NavigationView

import android.support.v4.app.Fragment

import android.support.v4.app.FragmentPagerAdapter

import android.util.Log

import android.view.KeyEvent

import android.view.MenuItem

import com.wt.yc.oto.oto.App

import com.wt.yc.oto.oto.R

import com.wt.yc.oto.oto.fragment.\*

import com.wt.yc.oto.oto.net.Config

import com.wt.yc.oto.oto.net.Share

import com.xin.lv.yang.utils.utils.HttpUtils

import kotlinx.android.synthetic.main.activity\_main.\*

import org.json.JSONObject

class MainActivity : ProActivity() {

override fun handler(msg: Message) {

when (msg.what) {

Config.GET\_TOKEN\_CODE -> {

val str = msg.obj as String

Log.i("result", "获取token------$str")

if (str != "" && str.startsWith("{")) {

val json = JSONObject(str)

val code = json.optInt("code")

if (code == 200) {

token = json.optString("token")

Share().saveToken(this, token)

}

}

}

}

}

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.activity\_main)

initViewPager()

disableShiftMode(bottom\_view)

bottom\_view.setOnNavigationItemSelectedListener(object : NavigationView.OnNavigationItemSelectedListener,

BottomNavigationView.OnNavigationItemSelectedListener {

override fun onNavigationItemSelected(item: MenuItem): Boolean {

when (item.itemId) {

R.id.menu1 -> {

view\_pager.currentItem = 0

Share().saveIndexPosition(this@MainActivity, 0)

}

R.id.menu2 -> {

view\_pager.currentItem = 1

Share().saveIndexPosition(this@MainActivity, 1)

(list[1] as TaskFragment).onResume()

}

R.id.menu3 -> {

view\_pager.currentItem = 2

Share().saveIndexPosition(this@MainActivity, 2)

}

R.id.menu4 -> {

view\_pager.currentItem = 3

Share().saveIndexPosition(this@MainActivity, 3)

}

}

return true

}

})

getToken()

}

/\*\*

\* 获取token

\*/

private fun getToken() {

val json = JSONObject()

json.put("account", "kuaipai2018")

json.put("password", "kuaipai\_123456")

HttpUtils.getInstance().postJsonWithHeader(Config.GET\_TOKEN, json.toString(), Config.GET\_TOKEN\_CODE, "header-token", handler)

}

val list: ArrayList<Fragment> = arrayListOf(MainFragment(), TaskFragment(), MessageFragment(), MainUserFragment())

private fun initViewPager() {

view\_pager.currentItem = intent.getIntExtra("code", 0)

view\_pager.adapter = object : FragmentPagerAdapter(supportFragmentManager) {

override fun getItem(position: Int): Fragment {

val fragment = list[position]

return fragment

}

override fun getCount(): Int {

return list.size

}

}

}

override fun onKeyDown(keyCode: Int, event: KeyEvent?): Boolean {

if (keyCode == KeyEvent.KEYCODE\_BACK && event!!.action == KeyEvent.ACTION\_DOWN) {

val builder = AlertDialog.Builder(this)

builder.setMessage("是否退出?").setPositiveButton("退出") { \_, \_ ->

Share().saveIndexPosition(this, 0)

App.exit()

}.setNegativeButton("取消") { \_, \_ ->

}.show()

}

return true

}

}

package com.wt.yc.oto.oto.fragment

import android.Manifest

import android.app.Activity

import android.app.AlertDialog

import android.content.Intent

import android.content.pm.PackageManager

import android.os.Bundle

import android.os.Handler

import android.os.Looper

import android.os.Message

import android.support.design.widget.TabLayout

import android.support.v4.app.ActivityCompat

import android.support.v4.widget.NestedScrollView

import android.support.v7.widget.LinearLayoutManager

import android.support.v7.widget.RecyclerView

import android.util.Log

import android.view.LayoutInflater

import android.view.View

import android.view.ViewGroup

import android.widget.FrameLayout

import android.widget.ImageView

import android.widget.LinearLayout

import android.widget.TextView

import butterknife.ButterKnife

import com.amap.api.location.AMapLocationClient

import com.amap.api.location.AMapLocationClientOption

import com.amap.api.maps.model.LatLng

import com.google.gson.reflect.TypeToken

import com.wt.yc.oto.oto.R

import com.wt.yc.oto.oto.activity.\*

import com.wt.yc.oto.oto.adapter.ItemClickListener

import com.wt.yc.oto.oto.adapter.Shop1Adapter

import com.wt.yc.oto.oto.adapter.Shop2Adapter

import com.wt.yc.oto.oto.info.PicText

import com.wt.yc.oto.oto.info.ShopInfo

import com.wt.yc.oto.oto.model.CityModel

import com.wt.yc.oto.oto.net.Config

import com.wt.yc.oto.oto.net.ConfigNet

import com.wt.yc.oto.oto.net.Share

import com.wt.yc.oto.oto.wxl.SendServiceActivity

import com.wt.yc.oto.oto.wxl.SendSuccessActivity

import com.wt.yc.oto.oto.wxl.ServiceChooseActivity

import com.wt.yc.oto.oto.wxl.ServiceTypeActivity

import com.wt.yc.oto.oto.yc.activity.AllClassifyActivity

import com.wt.yc.oto.oto.yc.activity.CityChooseActivity

import com.xin.lv.yang.utils.line.DividerItemDecoration

import com.xin.lv.yang.utils.utils.HttpUtils

import com.xin.lv.yang.utils.view.CustomSwipeRefreshView

import kotlinx.android.synthetic.main.main\_fragment.\*

import org.json.JSONObject

class MainFragment : ProV4Fragment() {

val WRITE\_COARSE\_LOCATION\_REQUEST\_CODE = 12

lateinit var login\_in\_view: View

lateinit var login\_frame\_layout: FrameLayout

lateinit var service\_recycler\_view: RecyclerView

lateinit var tab\_layout: TabLayout

lateinit var tab\_recycler: RecyclerView

lateinit var tv\_send: TextView

lateinit var jp\_shop\_linear: LinearLayout

lateinit var text\_index: TextView

lateinit var image\_view: ImageView

lateinit var search\_linear: LinearLayout

lateinit var nest\_scroll\_view: NestedScrollView

var isLoadFirst: Boolean = true

/\*\*

\* 登录后展示

\*/

var isLoginFirst: Boolean = true

var saveCity: String = ""

override fun createView(inflater: LayoutInflater, container: ViewGroup?, savedInstanceState: Bundle?): View {

val view = inflater.inflate(R.layout.main\_fragment, container, false)

unbinder = ButterKnife.bind(this, view)

initView(view)

saveCity = Share().getCity(activity!!)

text\_index.text = saveCity

initFragment(1)

if (Share().getUid(activity!!) != 0) {

login\_in\_view.visibility = View.VISIBLE

login\_frame\_layout.visibility = View.VISIBLE

initFragment(2)

} else {

login\_in\_view.visibility = View.GONE

login\_frame\_layout.visibility = View.GONE

}

val code = ActivityCompat.checkSelfPermission(activity!!, Manifest.permission.ACCESS\_COARSE\_LOCATION)

if (code != PackageManager.PERMISSION\_GRANTED) {

ActivityCompat.requestPermissions(activity!!, arrayOf(Manifest.permission.ACCESS\_COARSE\_LOCATION),

WRITE\_COARSE\_LOCATION\_REQUEST\_CODE);

} else {

initPosition()

}

getOneTabLayout()

initJpAdapter(arrayListOf())

initServceAdapter(arrayListOf())

tuiNum = 1

getTuiServiceShop()

return view

}

override fun onActivityCreated(savedInstanceState: Bundle?) {

super.onActivityCreated(savedInstanceState)

initClick()

customSwipeRefreshView.setOnRefreshListener {

handler.postDelayed({

customSwipeRefreshView.isRefreshing = false

}, 3000)

}

customSwipeRefreshView.onLoadByNestScrollView(nest\_scroll\_view) {

log("执行到最底部------")

getTuiServiceShop()

showLoadDialog(activity!!)

}

}

/\*\*

\* 获取TabLayout 数据

\*/

private fun getOneTabLayout() {

HttpUtils.getInstance().postJsonWithHeader(ConfigNet.GET\_MAIN\_INDEX\_URL, "", ConfigNet.GET\_SHOP\_CODE, Share().getToken(activity!!), handler)

}

override fun onRequestPermissionsResult(requestCode: Int, permissions: Array<out String>, grantResults: IntArray) {

super.onRequestPermissionsResult(requestCode, permissions, grantResults)

when (requestCode) {

WRITE\_COARSE\_LOCATION\_REQUEST\_CODE -> {

if (grantResults[0] == PackageManager.PERMISSION\_GRANTED) {

initPosition()

} else {

showToastShort(activity!!, "请检查权限")

}

}

}

}

var option: AMapLocationClientOption? = null

var location: AMapLocationClient? = null

var province = ""

private fun initPosition() {

location = AMapLocationClient(activity)

option = AMapLocationClientOption()

option!!.locationMode = AMapLocationClientOption.AMapLocationMode.Hight\_Accuracy

option!!.interval = 2000

option!!.isNeedAddress = true

option!!.isOnceLocation = true

option!!.isOnceLocationLatest = true

location!!.setLocationOption(option)

location!!.startLocation()

location!!.setLocationListener { p0 ->

val lat = p0!!.latitude

val lon = p0.longitude

val city = p0.city

province = p0.province

Share().saveLatLng(activity!!, LatLng(lat, lon))

saveLatLng(lat, lon)

if (city != "" && saveCity != "" && saveCity != city) {

showTi(city)

} else {

text\_index.text = saveCity

}

log("定位显示---$lat----$lon----$city")

}

}

/\*\*

\* 保存经纬度

\*/

private fun saveLatLng(lat: Double, lon: Double) {

val sb: StringBuilder = StringBuilder()

sb.append(lon).append(",").append(lat)

val json = JSONObject()

json.put("start", sb.toString())

json.put("uid", uid)

HttpUtils.getInstance().postJsonWithHeader(ConfigNet.SAVE\_LAT\_LNG\_URL, json.toString(), ConfigNet.SAVE\_LAT\_LNG, token, handler)

}

private fun showTi(city: String?) {

val build: AlertDialog.Builder = AlertDialog.Builder(activity)

build.setMessage("是否切换到当前城市?").setPositiveButton("切换") { dialog, p ->

dialog.dismiss()

this.saveCity = city!!

text\_index.text = city

Share().saveCity(activity!!, city)

initFragment(1)

}.setNegativeButton("取消") { dialog, p ->

}.show()

}

private fun getTabLayoutData(arr: ArrayList<PicText>) {

getJingPingTabLayout(arr[0].id!!)

initTabLayout(arr)

}

/\*\*

\* 初始化控件

\*/

private fun initView(view: View) {

login\_in\_view = view.findViewById(R.id.login\_in\_view)

login\_frame\_layout = view.findViewById(R.id.login\_frame\_layout)

service\_recycler\_view = view.findViewById(R.id.service\_recycler\_view)

tab\_layout = view.findViewById(R.id.tab\_layout)

tab\_recycler = view.findViewById(R.id.tab\_recycler)

tv\_send = view.findViewById(R.id.tv\_send)

jp\_shop\_linear = view.findViewById(R.id.jp\_shop\_linear)

text\_index = view.findViewById(R.id.text\_index)

image\_view = view.findViewById(R.id.image\_view)

search\_linear = view.findViewById(R.id.search\_linear)

nest\_scroll\_view = view.findViewById(R.id.nest\_scroll\_view)

}

private fun getJingPingTabLayout(id: Int) {

val json = JSONObject()

json.put("id", id)

json.put("num", 1)

HttpUtils.getInstance().postJsonWithHeader(ConfigNet.GET\_INDEX\_JP\_URL, json.toString(),

ConfigNet.GET\_INDEX\_JP\_CODE, token, handler)

}

var tuiNum = 0

fun getTuiServiceShop() {

val json = JSONObject()

json.put("limit", 8)

json.put("page", tuiNum)

HttpUtils.getInstance().postJsonWithHeader(ConfigNet.GET\_MAIN\_INDEX\_URL, json.toString(),

ConfigNet.GET\_TUI\_SERVICE, token, handler)

}

private fun initFragment(code: Int) {

val manager = childFragmentManager

val tran = manager.beginTransaction()

when (code) {

1 -> {

val loadFragment = LoadFragment()

LoadFragment.setCode(loadFragment, 1)

LoadFragment.setCity(loadFragment, saveCity)

if (isLoadFirst) {

tran.add(R.id.load\_frame\_layout, loadFragment)

isLoadFirst = false

} else {

tran.replace(R.id.load\_frame\_layout, loadFragment)

}

}

2 -> {

if (isLoginFirst) {

tran.add(R.id.login\_frame\_layout, LoginInFragment())

isLoginFirst = false

} else {

tran.replace(R.id.login\_frame\_layout, LoginInFragment())

}

}

}

tran.commitAllowingStateLoss()

}

var shop1Adapter: Shop1Adapter? = null

var shop2adapter: Shop2Adapter? = null

private fun initServceAdapter(shopList: ArrayList<ShopInfo>) {

service\_recycler\_view.isNestedScrollingEnabled = false

service\_recycler\_view.addItemDecoration(DividerItemDecoration(activity, DividerItemDecoration.VERTICAL\_LIST, R.drawable.line))

service\_recycler\_view.layoutManager = LinearLayoutManager(activity)

shop2adapter = Shop2Adapter(activity!!, shopList)

service\_recycler\_view.adapter = shop2adapter

shop2adapter!!.getListener(object : ItemClickListener {

override fun onItemClick(position: Int) {

val intent = Intent(activity, ServiceDetailsActivity::class.java)

intent.putExtra("id", shopList[position].id)

intent.putExtra("latlng", shopList[position].scope)

startActivity(intent)

}

override fun onLongClick(position: Int) {

}

})

}

private fun initTabLayout(textList: ArrayList<PicText>) {

for (temp in textList) {

tab\_layout.addTab(tab\_layout.newTab().setText(temp.title))

}

tab\_layout.addOnTabSelectedListener(object : TabLayout.OnTabSelectedListener {

override fun onTabReselected(tab: TabLayout.Tab?) {

}

override fun onTabUnselected(tab: TabLayout.Tab?) {

}

override fun onTabSelected(tab: TabLayout.Tab?) {

val position = tab!!.position

val picText = textList.get(position)

getJingPingTabLayout(picText.id!!)

}

})

}

private fun initJpAdapter(shopList: ArrayList<ShopInfo>) {

val layoutManager = LinearLayoutManager(activity)

layoutManager.orientation = LinearLayoutManager.HORIZONTAL

tab\_recycler.layoutManager = layoutManager

shop1Adapter = Shop1Adapter(activity!!, shopList)

tab\_recycler.adapter = shop1Adapter

shop1Adapter!!.getListener(object : ItemClickListener {

override fun onItemClick(position: Int) {

val intent = Intent(activity, StoreActivity::class.java)

startActivity(intent)

}

override fun onLongClick(position: Int) {

}

})

}

private fun initClick() {

// 发布

tv\_send.setOnClickListener {

if (Share().getUid(activity!!) != 0) {

getUserInfo()

} else {

startActivity(Intent(activity!!, LoginActivity::class.java))

}

}

// 定位

text\_index.setOnClickListener {

val intent = Intent(activity, CityChooseActivity::class.java)

intent.putExtra("city", text\_index.text)

intent.putExtra("province", province)

intent.putExtra("type", "0")

startActivityForResult(intent, 444)

}

jp\_shop\_linear.setOnClickListener {

uid = Share().getUid(activity!!)

if (uid != 0) {

val intent: Intent = Intent(activity, ShopListActivity::class.java)

intent.putExtra("code", 1)

startActivity(intent)

} else {

startActivity(Intent(activity, LoginActivity::class.java))

}

}

image\_view.setOnClickListener {

startActivity(Intent(activity, AllClassifyActivity::class.java))

}

search\_linear.setOnClickListener {

if (Share().getUid(activity!!) != 0) {

startActivity(Intent(activity, SearchActivity::class.java))

} else {

showToastShort(activity!!, "请登录")

startActivity(Intent(activity, LoginActivity::class.java))

}

}

}

private fun getUserInfo() {

val json = JSONObject()

json.put("uid", Share().getUid(activity!!))

HttpUtils.getInstance().postJsonWithHeader(Config.GET\_USER\_URL, json.toString(), Config.GET\_USER, Share().getToken(activity!!), handler)

showLoadDialog(activity!!)

}

private var handler: Handler = object : Handler(Looper.getMainLooper()) {

override fun handleMessage(msg: Message?) {

super.handleMessage(msg)

if (msg != null) {

val str = msg.obj as String

when (msg.what) {

ConfigNet.GET\_INDEX\_JP\_CODE -> {

if (str != "") {

val json = JSONObject(str)

val code = json.optInt("code")

if (code == 200) {

val resultJson = json.optString("data")

val shopList: ArrayList<ShopInfo> = gson!!.fromJson(resultJson, object : TypeToken<ArrayList<ShopInfo>>() {}.type)

if (shop1Adapter != null) {

shop1Adapter!!.updateClear(shopList)

} else {

initJpAdapter(shopList)

}

}

}

}

ConfigNet.GET\_TUI\_SERVICE -> {

removeLoadDialog()

if (str != "") {

val json = JSONObject(str)

val code = json.optInt("code")

if (code == 200) {

val jsonObject = json.optJSONObject("data")

val resultJson = jsonObject.optString("recommend")

if (resultJson != null && resultJson != "null") {

val shopList: ArrayList<ShopInfo> = gson!!.fromJson(resultJson, object : TypeToken<ArrayList<ShopInfo>>() {}.type)

if (shop2adapter != null) {

shop2adapter!!.updataData(shopList)

} else {

initServceAdapter(shopList)

}

tuiNum++

} else {

showToastShort(activity!!, "无更多数据")

}

}

}

}

ConfigNet.GET\_SHOP\_CODE -> {

Log.i("result", "返回数据导航TabLayout-----$str")

if (str != "") {

val json = JSONObject(str)

val code = json.optInt("code")

if (code == 200) {

val resultJson = json.optJSONObject("data")

if (resultJson != null) {

val result = resultJson.optString("nav")

if (result != "") {

val resultArr: ArrayList<PicText> = gson!!.fromJson(result, object : TypeToken<ArrayList<PicText>>() {}.type)

getTabLayoutData(resultArr)

}

}

}

}

}

ConfigNet.SAVE\_LAT\_LNG -> {

}

Config.GET\_USER -> {

removeLoadDialog()

val json = JSONObject(str)

val code = json.optInt("code")

if (code == 200) {

val jsonObject = json.optJSONObject("result")

val status = jsonObject.optInt("status")

if (status != 0) {

val type: Int = jsonObject.optInt("type1")

val type1 = jsonObject.optString("p\_type1")

Share().saveRealType(activity!!, type)

if (type1 != null && type1 != "") {

val intent = Intent(context, ServiceTypeActivity::class.java)

intent.putExtra("id", type1.toInt())

intent.putExtra("shop\_id", type1)

startActivity(intent)

} else {

val intent = Intent(activity, ServiceChooseActivity::class.java)

intent.putExtra("shop\_id", "")

startActivity(intent)

}

} else {

showToastShort(activity!!, "账号正在审核，请耐心等待")

}

}

}

}

}

}

}

override fun onActivityResult(requestCode: Int, resultCode: Int, data: Intent?) {

super.onActivityResult(requestCode, resultCode, data)

if (resultCode == Activity.RESULT\_OK) {

when (requestCode) {

444 -> {

val cityMode = data!!.getParcelableExtra<CityModel>(Intent.EXTRA\_RETURN\_RESULT)

text\_index.text = cityMode.city

this.saveCity = cityMode.city

initFragment(1)

}

}

}

}

}

package com.wt.yc.oto.oto.fragment

import android.annotation.SuppressLint

import android.content.Intent

import android.os.Bundle

import android.os.Handler

import android.os.Looper

import android.os.Message

import android.view.LayoutInflater

import android.view.View

import android.view.ViewGroup

import android.widget.\*

import butterknife.BindView

import butterknife.ButterKnife

import com.bumptech.glide.Glide

import com.wt.yc.oto.oto.R

import com.wt.yc.oto.oto.activity.\*

import com.wt.yc.oto.oto.net.Config

import com.wt.yc.oto.oto.net.Share

import com.wt.yc.oto.oto.wxl.\*

import com.wt.yc.oto.oto.wxl.parttimecat.PartListActivity

import com.xin.lv.yang.utils.utils.HttpUtils

import com.xin.lv.yang.utils.utils.ImageUtil

import org.json.JSONObject

class UserFragment : ProV4Fragment(), View.OnClickListener {

@BindView(R.id.text\_user\_balance)

lateinit var textUserBalance: TextView

@BindView(R.id.text\_user\_mobile)

lateinit var textUserMobile: TextView

@BindView(R.id.text\_user\_nickname)

lateinit var textUserNickname: TextView

@BindView(R.id.image\_my\_head)

lateinit var imageMyHead: ImageView

private lateinit var btnUserLogin: Button

private lateinit var linearUserContent: LinearLayout

@BindView(R.id.text\_pay\_number)

lateinit var text\_pay\_number: TextView

@BindView(R.id.text\_no\_pai)

lateinit var text\_no\_pai: TextView

@BindView(R.id.text\_no\_service)

lateinit var text\_no\_service: TextView

@BindView(R.id.text\_no\_sure)

lateinit var text\_no\_sure: TextView

@BindView(R.id.text\_no\_comment)

lateinit var text\_no\_comment: TextView

@BindView(R.id.linear\_user\_part)

lateinit var linear\_user\_part: LinearLayout

@BindView(R.id.text\_user\_collection)

lateinit var text\_user\_collection: TextView

@BindView(R.id.text\_user\_coupon)

lateinit var text\_user\_coupon: TextView

var username: String = ""

var image: String = ""

var phone: String = ""

var mobile: String = ""

var about\_us: String = ""

var result = ""

override fun createView(inflater: LayoutInflater, container: ViewGroup?, savedInstanceState: Bundle?): View {

val view = View.inflate(activity, R.layout.fragment\_user, null)

ButterKnife.bind(this, view)

btnUserLogin = view.findViewById(R.id.btn\_user\_login)

btnUserLogin.setOnClickListener(this)

val linearUserAddress: LinearLayout = view.findViewById(R.id.linear\_user\_address)

linearUserAddress.setOnClickListener(this)

linearUserContent = view.findViewById(R.id.linear\_user\_content)

linearUserContent.setOnClickListener(this)

val linearUserWallet: LinearLayout = view.findViewById(R.id.linear\_user\_wallet)

linearUserWallet.setOnClickListener(this)

val linearUserSafe: LinearLayout = view.findViewById(R.id.linear\_user\_safe)

linearUserSafe.setOnClickListener(this)

val linearUserAbout: LinearLayout = view.findViewById(R.id.linear\_user\_about);

linearUserAbout.setOnClickListener(this)

val linearUserJoin: LinearLayout = view.findViewById(R.id.linear\_user\_join);

linearUserJoin.setOnClickListener(this)

val linearUserCoupon: LinearLayout = view.findViewById(R.id.linear\_user\_coupon);

linearUserCoupon.setOnClickListener(this)

val linearUserCollection: LinearLayout = view.findViewById(R.id.linear\_user\_collection);

linearUserCollection.setOnClickListener(this)

val linearUserComment: LinearLayout = view.findViewById(R.id.linear\_user\_comment);

linearUserComment.setOnClickListener(this)

val linearUserSure: LinearLayout = view.findViewById(R.id.linear\_user\_sure);

linearUserSure.setOnClickListener(this)

val linearUserService: LinearLayout = view.findViewById(R.id.linear\_user\_service);

linearUserService.setOnClickListener(this)

val linearUserZhiPai: LinearLayout = view.findViewById(R.id.linear\_user\_zhipai);

linearUserZhiPai.setOnClickListener(this)

val linearUserPay: LinearLayout = view.findViewById(R.id.linear\_user\_pay);

linearUserPay.setOnClickListener(this)

val relativeUserOrder: RelativeLayout = view.findViewById(R.id.relative\_user\_order);

relativeUserOrder.setOnClickListener(this)

val linearUserDComment: LinearLayout = view.findViewById(R.id.linear\_user\_dcomment);

linearUserDComment.setOnClickListener(this)

val linearUserFunding: LinearLayout = view.findViewById(R.id.linear\_user\_funding);

linearUserFunding.setOnClickListener(this)

linear\_user\_part.setOnClickListener(this)

imageMyHead.setOnClickListener(this)

if (Share().getUid(activity!!) != 0) {

btnUserLogin.visibility = View.GONE

linearUserContent.visibility = View.VISIBLE

} else {

btnUserLogin.visibility = View.VISIBLE

linearUserContent.visibility = View.GONE

}

return view

}

override fun onResume() {

super.onResume()

if (Share().getUid(activity!!) != 0) {

// 用户信息存在

btnUserLogin.visibility = View.GONE

linearUserContent.visibility = View.VISIBLE

showLoadDialog(activity!!)

getOrder()

} else {

btnUserLogin.visibility = View.VISIBLE

linearUserContent.visibility = View.GONE

}

}

private fun getOrder() {

val json = JSONObject()

json.put("uid", Share().getUid(activity!!))

HttpUtils.getInstance().postJsonWithHeader(Config.GET\_USER\_URL, json.toString(), Config.GET\_USER, Share().getToken(activity!!), handler)

}

private val handler: Handler = object : Handler(Looper.getMainLooper()) {

@SuppressLint("SetTextI18n")

override fun handleMessage(msg: Message?) {

super.handleMessage(msg)

val str: String? = msg!!.obj as String

when (msg.what) {

Config.GET\_USER -> {

removeLoadDialog()

if (str != null && str != "") {

val json = JSONObject(str)

val code = json.optInt("code")

val message = json.optString("msg")

if (code == 200) {

showUser(json)

} else {

showToastShort(activity!!, message)

}

}

}

}

}

}

private fun showUser(json: JSONObject) {

val result = json.getJSONObject("result")

username = result.optString("username")

mobile = result.optString("mobile")

val balance = result.optString("balance")

image = result.optString("image")

val collectNum = result.optString("collect\_num")

val ticketNum = result.optString("ticket\_num")

val noPay = result.optString("no\_pay")

val noPai = result.optString("no\_pai")

val noService = result.optString("no\_servie")

val noComment = result.optString("no\_comment")

val noSure = result.optString("no\_sure")

about\_us = result.optString("about\_us")

phone = mobile.substring(0, 3) + "\*\*\*\*" + mobile.substring(7, mobile.length)

textUserBalance.text = ("￥$balance")

textUserMobile.text = phone

if (collectNum != "0" && collectNum != "null") {

text\_user\_collection.text = collectNum

} else {

text\_user\_collection.text = "0"

}

if (ticketNum != "0" && ticketNum != "null") {

text\_user\_coupon.text = ticketNum

} else {

text\_user\_coupon.text = "0"

}

if (noPay != "0" && noPay != "") {

text\_pay\_number.visibility = View.VISIBLE

text\_pay\_number.text = noPay

}

if (noSure != "0" && noSure != "") {

text\_no\_sure.visibility = View.VISIBLE

text\_no\_sure.text = noSure

}

if (noComment != "0" && noComment != "") {

text\_no\_comment.visibility = View.VISIBLE

text\_no\_comment.text = noComment

}

if (noService != "0" && noService != "") {

text\_no\_service.visibility = View.VISIBLE

text\_no\_service.text = noService

}

if (noPai != "0" && noPai != "") {

text\_no\_pai.visibility = View.VISIBLE

text\_no\_pai.text = noPai

}

if (username != "null") {

textUserNickname.text = username

}

ImageUtil.getInstance().loadImageNoTransformation(activity!!, imageMyHead, R.drawable.my\_head, Config.IP + image)

}

override fun onClick(view: View) {

val id = view.id

when (id) {

R.id.btn\_user\_login -> {

//立即登录

val intent = Intent()

//获取intent对象

intent.setClass(activity, LoginActivity::class.java)

// 获取class是使用::反射

startActivity(intent)

}

R.id.linear\_user\_collection -> {

if (Share().getUid(activity!!) != 0) {

//我的收藏

val intent = Intent()

//获取intent对象

intent.setClass(activity, CollectionActivity::class.java)

// 获取class是使用::反射

startActivity(intent)

} else {

val intent = Intent()

intent.setClass(activity, LoginActivity::class.java)

startActivity(intent)

}

}

R.id.linear\_user\_address -> {

if (Share().getUid(activity!!) != 0) {

//常用地址

val intent = Intent()

//获取intent对象

intent.setClass(activity, AddressActivity::class.java)

intent.putExtra("type", 0)

// 获取class是使用::反射

startActivity(intent)

} else {

val intent = Intent()

intent.setClass(activity, LoginActivity::class.java)

startActivity(intent)

}

}

R.id.linear\_user\_content -> {

//个人资料

val intent = Intent()

intent.setClass(activity, SettingActivity::class.java)

intent.putExtra("username", username)

intent.putExtra("image", image)

intent.putExtra("phone", phone)

startActivity(intent)

}

R.id.image\_my\_head -> {

val intent = Intent()

intent.setClass(activity, SettingActivity::class.java)

intent.putExtra("username", username)

intent.putExtra("image", image)

intent.putExtra("phone", phone)

startActivity(intent)

}

R.id.linear\_user\_wallet -> {

if (Share().getUid(activity!!) != 0) {

//钱包

val intent = Intent()

//获取intent对象

intent.setClass(activity, WalletActivity::class.java)

intent.putExtra("type", "2")

// 获取class是使用::反射

startActivity(intent)

} else {

val intent = Intent()

intent.setClass(activity, LoginActivity::class.java)

startActivity(intent)

}

}

R.id.linear\_user\_safe -> {

if (Share().getUid(activity!!) != 0) {

//安全中心

val intent = Intent()

//获取intent对象

intent.setClass(activity, SafeActivity::class.java)

intent.putExtra("phone", phone)

intent.putExtra("mobile", mobile)

// 获取class是使用::反射

startActivity(intent)

} else {

val intent = Intent()

intent.setClass(activity, LoginActivity::class.java)

startActivity(intent)

}

}

R.id.linear\_user\_about -> {

val intent = Intent()

//获取intent对象

intent.setClass(activity, AboutActivity::class.java)

intent.putExtra("about\_us", about\_us)

// 获取class是使用::反射

startActivity(intent)

}

R.id.linear\_user\_join -> {

if (Share().getUid(activity!!) != 0) {

val intent = Intent()

//获取intent对象

intent.setClass(activity, JoinActivity::class.java)

// 获取class是使用::反射

startActivity(intent)

} else {

val intent = Intent()

intent.setClass(activity, LoginActivity::class.java)

startActivity(intent)

}

}

R.id.linear\_user\_coupon -> {

if (Share().getUid(activity!!) != 0) {

val intent = Intent()

//获取intent对象

intent.setClass(activity, CouponActivity::class.java)

// 获取class是使用::反射

startActivity(intent)

} else {

val intent = Intent()

intent.setClass(activity, LoginActivity::class.java)

startActivity(intent)

}

}

R.id.linear\_user\_comment -> {

if (Share().getUid(activity!!) != 0) {

val intent = Intent()

intent.setClass(activity, CommentActivity::class.java)

startActivity(intent)

} else {

val intent = Intent()

intent.setClass(activity, LoginActivity::class.java)

startActivity(intent)

}

}

R.id.linear\_user\_funding -> {

if (Share().getUid(activity!!) != 0) {

val intent = Intent()

intent.setClass(activity, OrderActivity::class.java)

startActivity(intent)

} else {

val intent = Intent()

intent.setClass(activity, LoginActivity::class.java)

startActivity(intent)

}

}

R.id.linear\_user\_dcomment -> {

if (Share().getUid(activity!!) != 0) {

val intent = Intent()

intent.setClass(activity, GoodsActivity::class.java)

intent.putExtra("position", 4)

startActivity(intent)

} else {

val intent = Intent()

intent.setClass(activity, LoginActivity::class.java)

startActivity(intent)

}

}

R.id.linear\_user\_sure -> {

if (Share().getUid(activity!!) != 0) {

val intent = Intent()

intent.setClass(activity, GoodsActivity::class.java)

intent.putExtra("position", 3)

startActivity(intent)

} else {

val intent = Intent()

intent.setClass(activity, LoginActivity::class.java)

startActivity(intent)

}

}

R.id.linear\_user\_service -> {

if (Share().getUid(activity!!) != 0) {

val intent = Intent()

intent.setClass(activity, GoodsActivity::class.java)

intent.putExtra("position", 2)

startActivity(intent)

} else {

val intent = Intent()

intent.setClass(activity, LoginActivity::class.java)

startActivity(intent)

}

}

R.id.linear\_user\_zhipai -> {

if (Share().getUid(activity!!) != 0) {

val intent = Intent()

intent.setClass(activity, GoodsActivity::class.java)

intent.putExtra("position", 1)

startActivity(intent)

} else {

val intent = Intent()

intent.setClass(activity, LoginActivity::class.java)

startActivity(intent)

}

}

R.id.linear\_user\_pay -> {

if (Share().getUid(activity!!) != 0) {

val intent = Intent()

intent.setClass(activity, GoodsActivity::class.java)

intent.putExtra("position", 0)

startActivity(intent)

} else {

val intent = Intent()

intent.setClass(activity, LoginActivity::class.java)

startActivity(intent)

}

}

R.id.relative\_user\_order -> {

if (Share().getUid(activity!!) != 0) {

val intent = Intent()

intent.setClass(activity, GoodsActivity::class.java)

intent.putExtra("position", 5)

startActivity(intent)

} else {

val intent = Intent()

intent.setClass(activity, LoginActivity::class.java)

startActivity(intent)

}

}

R.id.linear\_user\_part -> {

val intent = Intent()

intent.setClass(activity, PartListActivity::class.java)

intent.putExtra("position", 4)

startActivity(intent)

}

else -> {

}

}

}

}

package com.wt.yc.oto.oto.activity

import android.Manifest

import android.annotation.SuppressLint

import android.content.Intent

import android.content.pm.PackageManager

import android.net.Uri

import android.os.Bundle

import android.os.Message

import android.support.v4.app.ActivityCompat

import android.support.v7.widget.GridLayoutManager

import android.util.Log

import android.view.MotionEvent

import android.view.View

import android.widget.CompoundButton

import android.widget.TextView

import com.google.gson.reflect.TypeToken

import com.wt.yc.oto.oto.R

import com.wt.yc.oto.oto.adapter.ItemClickListener

import com.wt.yc.oto.oto.adapter.Shop1Adapter

import com.wt.yc.oto.oto.adapter.Shop2Adapter

import com.wt.yc.oto.oto.info.CommentInfo

import com.wt.yc.oto.oto.info.ShopInfo

import com.wt.yc.oto.oto.net.Config

import com.wt.yc.oto.oto.net.ConfigNet

import com.wt.yc.oto.oto.net.Share

import com.wt.yc.oto.oto.yc.activity.PlaceAnOrderActivity

import com.xin.lv.yang.utils.utils.BitmapUtil

import com.xin.lv.yang.utils.utils.HttpUtils

import com.xin.lv.yang.utils.utils.ImageUtil

import kotlinx.android.synthetic.main.comment\_item.\*

import kotlinx.android.synthetic.main.item\_top\_no\_bg.\*

import kotlinx.android.synthetic.main.shop\_details\_layout.\*

import org.json.JSONObject

class ServiceDetailsActivity : ProActivity() {

var shopResult = ""

override fun handler(msg: Message) {

val str = msg.obj as String

when (msg.what) {

ConfigNet.GET\_SHOP -> {

Log.i("toby", "str------>$str")

if (str != "" && str.startsWith("{")) {

val json = JSONObject(str)

val code = json.optInt("code")

if (code == 200) {

val jsonObject = json.optJSONObject("data")

shopResult = jsonObject.optString("shop")

val shopInfo: ShopInfo = gson!!.fromJson(shopResult, ShopInfo::class.java)

showShopInfo(shopInfo)

val commentJson = jsonObject.optJSONArray("label")

if (commentJson != null) {

val len = commentJson.length()

val commentArr: ArrayList<String> = arrayListOf()

for (i in 0 until len) {

val commentResult = commentJson.optString(i)

commentArr.add(commentResult)

}

initFlex(commentArr)

}

val explainJson = jsonObject.optJSONObject("explain")

val image = explainJson.optString("imge")

val text = explainJson.optString("text")

dt\_tv\_jj.text = text

ImageUtil.getInstance().loadImageNoTransformation(this, dt\_iv\_jj\_pic, R.drawable.banner, Config.IP + image)

val commentArr = jsonObject.optString("comment")

if (commentArr != null && commentArr != "" && commentArr != "null") {

include\_linear.visibility = View.VISIBLE

details\_find.visibility = View.VISIBLE

val commentInfo: CommentInfo = gson!!.fromJson(commentArr, CommentInfo::class.java)

showCommment(arrayListOf(commentInfo))

} else {

include\_linear.visibility = View.GONE

details\_find.visibility = View.GONE

}

val goodsShop: String = jsonObject.optString("goods")

if (goodsShop != "null" && goodsShop != "") {

val moreList: ArrayList<ShopInfo> = gson!!.fromJson(goodsShop, object : TypeToken<ArrayList<ShopInfo>>() {}.type)

initMoreAdapter(moreList)

}

}

}

}

ConfigNet.GET\_SHOP\_COMMENT\_CODE -> {

val json = JSONObject(str)

val code = json.optInt("code")

if (code == 200) {

val result = json.optString("data")

val arr: ArrayList<CommentInfo> = gson!!.fromJson(result, object : TypeToken<ArrayList<CommentInfo>>() {}.type)

showCommment(arr)

}

}

ConfigNet.COLLECT\_CODE -> {

val json = JSONObject(str)

val code = json.optInt("code")

if (code == 200) {

} else {

if (collCode == 1) {

dt\_collect.isChecked = false

} else if (collCode == 2) {

dt\_collect.isChecked = true

}

}

showToastShort(json.optString("msg"))

}

}

}

private fun initMoreAdapter(moreList: ArrayList<ShopInfo>) {

val adapter = Shop1Adapter(this, moreList)

adapter.shopCode = 1

dt\_more\_recycler.adapter = adapter

adapter.getListener(object : ItemClickListener {

override fun onItemClick(position: Int) {

val shopInfo = moreList[position]

}

override fun onLongClick(position: Int) {

}

})

}

var shopInfo: ShopInfo? = null

/\*\*

private fun showShopInfo(shopInfo: ShopInfo) {

this.shopInfo = shopInfo

phone = shopInfo.mobile

initViewPager(details\_pic\_view\_pager, arrayListOf(Config.IP + shopInfo.img))

details\_money.text = "￥${shopInfo.money}"

details\_tv\_name.text = shopInfo.shop\_name

val starNum = shopInfo.star\_sun

if (starNum == 0) {

details\_fen.text = "5"

details\_rating.rating = 5f

} else {

details\_rating.rating = starNum.toFloat()

details\_fen.text = starNum.toString()

}

dt\_com\_name.text = shopInfo.s\_name

details\_jd.text = ": ${shopInfo.up\_order\_num}"

details\_sale.text = ": ${shopInfo.sun}"

if (shopInfo.s\_loge.contains("mr.jpg")) {

ImageUtil.getInstance().loadImageNoTransformation(this, dt\_com\_head, R.mipmap.logo, "")

} else {

ImageUtil.getInstance().loadImageNoTransformation(this, dt\_com\_head, R.mipmap.logo, Config.IP + shopInfo.s\_loge)

}

val late = shopInfo.late

if (late == 1) {

details\_service\_type2.visibility = View.VISIBLE

} else {

details\_service\_type2.visibility = View.GONE

}

details\_good.text = shopInfo.describe

dt\_tv\_all.text = ": ${shopInfo.shop\_count}"

dt\_tv\_service\_num.text = ": ${shopInfo.s\_comment\_sun}"

if (shopInfo.comment\_sun != 0) {

dt\_good\_num.text = ": ${shopInfo.good / shopInfo.comment\_sun}"

}

details\_good.text = "${shopInfo.good}好评"

details\_comment\_num.text = "${shopInfo.comment\_sun}条"

}

private fun showCommment(arr: ArrayList<CommentInfo>) {

val commentInfo = arr[0]

ImageUtil.getInstance().loadImageNoTransformation(this, comm\_user\_head, R.mipmap.logo, Config.IP + commentInfo.user\_img)

comm\_user\_phone.text = commentInfo.mobile

comm\_user\_rating.rating = commentInfo.star.toFloat()

comm\_user\_time.text = BitmapUtil.longToTime(commentInfo.create\_time, "yyyy-MM-dd")

comm\_content.text = commentInfo.content

comm\_type.text = commentInfo.label

}

var phone: String = ""

var shopId = 0

/\*\*

\* 是否点击CheckBox

\*/

var isCheckBox = false

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.shop\_details\_layout)

image\_back.setOnClickListener { finish() }

text\_top.text = "详情"

shopId = intent.getIntExtra("id", 0)

dt\_more\_recycler.layoutManager = GridLayoutManager(this, 2)

dt\_more\_recycler.isNestedScrollingEnabled = false

getShopDetails(shopId)

initComment()

initClick()

}

private fun initClick() {

dt\_bt\_in\_store.setOnClickListener {

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

intent.putExtra("id", shopInfo!!.uid)

startActivity(intent)

}

details\_find.setOnClickListener {

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

intent.putExtra("id", shopId)

startActivity(intent)

}

dt\_store.setOnClickListener {

contactStore()

}

dt\_lx\_store.setOnClickListener {

contactStore()

}

dt\_tv\_store.setOnClickListener {

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

intent.putExtra("id", shopInfo!!.uid)

startActivity(intent)

}

dt\_collect.setOnTouchListener { p0, p1 ->

isCheckBox = true

false

}

dt\_collect.setOnCheckedChangeListener { p0, p1 ->

if (isCheckBox) {

if (uid != 0) {

if (p1) {

// 收藏

collectShop(1)

} else {

// 取消收藏

collectShop(2)

}

} else {

dt\_collect.isChecked = false

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

startActivity(intent)

}

}

}

dt\_bt\_yuyue.setOnClickListener {

if (Share().getUid(this) != 0) {

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

intent.putExtra("shop", shopResult)

startActivity(intent)

} else {

startActivity(Intent(this, LoginActivity::class.java))

showToastShort("请登录")

}

}

dt\_tv\_jb.setOnClickListener {

/// 举报

}

}

fun contactStore() {

val code = ActivityCompat.checkSelfPermission(this, android.Manifest.permission.CALL\_PHONE)

if (code == PackageManager.PERMISSION\_GRANTED) {

val intent = Intent(Intent.ACTION\_CALL)

intent.flags = Intent.FLAG\_ACTIVITY\_NEW\_TASK

val uri: Uri = Uri.parse("tel:$phone")

intent.data = uri

startActivity(intent)

} else {

ActivityCompat.requestPermissions(this, arrayOf(Manifest.permission.CALL\_PHONE), 1234)

}

}

var collCode: Int = 0

private fun collectShop(code: Int) {

this.collCode = code

val json = JSONObject()

json.put("uid", uid)

json.put("goods\_id", shopId)

json.put("type", 2)

json.put("coll\_type", code)

HttpUtils.getInstance().postJsonWithHeader(ConfigNet.COLLECT\_URL, json.toString(), ConfigNet.COLLECT\_CODE, token, handler)

isCheckBox = false

}

private fun getShopDetails(id: Int) {

val json = JSONObject()

json.put("id", id)

// json.put("type", 1)

HttpUtils.getInstance().postJsonWithHeader(ConfigNet.GET\_SHOP\_URL, json.toString(), ConfigNet.GET\_SHOP, token, handler)

}

private fun initComment() {

val json = JSONObject()

json.put("shop\_id", shopId)

json.put("num", 1)

HttpUtils.getInstance().postJsonWithHeader(ConfigNet.GET\_SHOP\_COMMENT\_URL, json.toString(),

ConfigNet.GET\_SHOP\_COMMENT\_CODE, token, handler)

}

private fun initFlex(arr: ArrayList<String>) {

for (temp in arr) {

val view: View = layoutInflater.inflate(R.layout.text\_item\_layout, null)

val textView = view.findViewById(R.id.text\_item) as TextView

textView.text = temp

textView.textSize = 12f

textView.setBackgroundResource(R.drawable.details\_kuang)

view.setOnClickListener {

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

intent.putExtra("id", shopId)

startActivity(intent)

}

details\_flex\_layout.addView(view)

}

}

}

package com.wt.yc.oto.oto.wxl.fragment

import android.content.Intent

import android.os.Bundle

import android.os.Handler

import android.os.Looper

import android.os.Message

import android.support.v7.widget.LinearLayoutManager

import android.support.v7.widget.OrientationHelper

import android.support.v7.widget.RecyclerView

import android.view.LayoutInflater

import android.view.View

import android.view.ViewGroup

import butterknife.BindView

import butterknife.ButterKnife

import com.google.gson.reflect.TypeToken

import com.wt.yc.oto.oto.R

import com.wt.yc.oto.oto.adapter.ItemClickListener

import com.wt.yc.oto.oto.fragment.ProV4Fragment

import com.wt.yc.oto.oto.model.GoodsModel

import com.wt.yc.oto.oto.net.Config

import com.wt.yc.oto.oto.net.Share

import com.wt.yc.oto.oto.wxl.CycleOrderItemActivity

import com.wt.yc.oto.oto.wxl.GoodsDetailActivity

import com.wt.yc.oto.oto.wxl.adapter.GoodsAdapter

import com.xin.lv.yang.utils.utils.HttpUtils

import com.xin.lv.yang.utils.view.CustomSwipeRefreshView

import kotlinx.android.synthetic.main.fragment\_order.\*

import org.json.JSONObject

import java.util.\*

/\*\*

\* 我的订单列表

\*/

class GoodsFragment() : ProV4Fragment(), CustomSwipeRefreshView.OnLoadListener {

override fun onLoad() {}

@BindView(R.id.recycler\_order)

lateinit var recyclerOrder: RecyclerView

@BindView(R.id.refresh\_layout)

lateinit var refreshLayout: CustomSwipeRefreshView

private var adapter: GoodsAdapter? = null

private var arrayList: ArrayList<GoodsModel>? = null

var classifyId: String = ""

var order\_type = ""

var shop\_type = ""

var num: String = "16"

var page: Int = 1

override fun createView(inflater: LayoutInflater, container: ViewGroup?, savedInstanceState: Bundle?): View {

val view = inflater.inflate(R.layout.fragment\_order, container, false)

unbinder = ButterKnife.bind(this, view)

//设置固定大小

recyclerOrder.setHasFixedSize(true)

arrayList = ArrayList()

adapter = GoodsAdapter(activity!!, arrayList!!, classifyId, order\_type)

//创建线性布局

val mLayoutManager = LinearLayoutManager(activity)

//垂直方向

mLayoutManager.orientation = OrientationHelper.VERTICAL

//给RecyclerView设置布局管理器

recyclerOrder.layoutManager = mLayoutManager

recyclerOrder.adapter = adapter

adapter!!.getListener(object : ItemClickListener {

override fun onItemClick(position: Int) {

val info = arrayList!![position]

if (info.mode == "2") {

val intent = Intent(activity, CycleOrderItemActivity::class.java)

intent.putExtra("orderId", info.id)

startActivity(intent)

} else {

val intent = Intent(activity, GoodsDetailActivity::class.java)

val orderId = info.order\_number.toString()

intent.putExtra("order\_number", orderId)

intent.putExtra("orderType", order\_type)

startActivity(intent)

}

}

override fun onLongClick(position: Int) {

}

})

refreshLayout.setOnRefreshListener {

log("执行获取信息---")

page = 1

arrayList!!.clear()

if (adapter != null) {

adapter!!.notifyDataSetChanged()

}

getOrder()

}

refreshLayout.onLoadMore(recyclerOrder) {

if (arrayList!!.size > (num.toInt() - 1)) {

page++

getOrder()

}

}

return view

}

override fun onResume() {

super.onResume()

page = 1

arrayList!!.clear()

if (adapter != null) {

adapter!!.notifyDataSetChanged()

}

getOrder()

}

override fun onActivityCreated(savedInstanceState: Bundle?) {

super.onActivityCreated(savedInstanceState)

text\_order\_more.visibility = View.GONE

}

private fun getOrder() {

val json = JSONObject()

json.put("uid", Share().getUid(activity!!))

json.put("type", 1)

if (classifyId != "") {

json.put("status", classifyId)

}

json.put("num", num)

json.put("page", page)

json.put("shop\_type", shop\_type)

if (order\_type == "4") {

json.put("shop\_pid", "1")

json.put("order\_type", "1")

} else {

json.put("order\_type", order\_type)

}

HttpUtils.getInstance().postJsonWithHeader(Config.GET\_ORDER\_URL, json.toString(), Config.GET\_ORDER, Share().getToken(activity!!), handler)

showLoadDialog(activity!!, "获取中")

}

private val handler: Handler = object : Handler(Looper.getMainLooper()) {

override fun handleMessage(msg: Message?) {

super.handleMessage(msg)

val str: String? = msg!!.obj as String

when (msg.what) {

Config.GET\_ORDER -> {

removeLoadDialog()

refreshLayout.isRefreshing = false

jsonDate(str)

}

}

}

}

private fun jsonDate(str: String?) {

if (str != null && str != "") {

val json = JSONObject(str)

val code = json.optInt("code")

val message = json.getString("msg")

if (code == 200) {

val resultJson = json.optString("data")

if (resultJson != null && resultJson != "") {

val resultArr: ArrayList<GoodsModel> = gson!!.fromJson(resultJson, object : TypeToken<ArrayList<GoodsModel>>() {}.type)

arrayList!!.addAll(resultArr)

adapter!!.notifyDataSetChanged()

}

}

}

}

}

package com.wt.yc.oto.oto.wxl.adapter

import android.app.AlertDialog

import android.app.Dialog

import android.content.Context

import android.content.Intent

import android.graphics.BitmapFactory

import android.os.Handler

import android.os.Looper

import android.os.Message

import android.support.v7.widget.RecyclerView

import android.util.Log

import android.view.View

import android.view.ViewGroup

import butterknife.BindView

import butterknife.ButterKnife

import com.wt.yc.oto.oto.R

import com.wt.yc.oto.oto.adapter.ProBaseAdapter

import com.wt.yc.oto.oto.wxl.AddCommentActivity

import com.wt.yc.oto.oto.wxl.AfterSaleActivity

import android.view.LayoutInflater

import android.widget.\*

import com.amap.api.maps.\*

import com.amap.api.maps.model.\*

import com.amap.api.navi.view.AmapCameraOverlay

import com.lljjcoder.citylist.Toast.ToastUtils

import com.wt.yc.oto.oto.model.GoodsModel

import com.wt.yc.oto.oto.net.Config

import com.wt.yc.oto.oto.net.Share

import com.wt.yc.oto.oto.wxl.GoodsActivity

import com.wt.yc.oto.oto.yc.activity.ActualTimeActivity

import com.wt.yc.oto.oto.yc.activity.PayOrderActivity

import com.wt.yc.oto.oto.yc.mopao.MoPaoSendActivity

import com.xin.lv.yang.utils.utils.HttpUtils

import com.xin.lv.yang.utils.utils.ImageUtil

import com.xin.lv.yang.utils.utils.TextUtils

import org.json.JSONObject

class GoodsAdapter(context: Context, list: ArrayList<GoodsModel>, val type: String, val order\_type: String)

: ProBaseAdapter<GoodsModel>(context, list) {

lateinit var dialog\_tips: Dialog

var num: Int = 0

var ids = ""

override fun onBindViewHolder(holder: RecyclerView.ViewHolder, position: Int) {

val vh = holder as VH

val array = list[position]

val status = array.status

val payType = array.pay\_type

var pay = array.pay

var mode = array.mode

val distance = array.distance

when (order\_type) {

"1" -> {

vh.text\_goods\_name.text = if (array.worker\_name != null && array.worker\_name != "") {

array.worker\_name

} else {

}

when (mode) {

"1" -> {

val startTime = array.visit\_start\_time

if (startTime != null && startTime != "") {

val time = TextUtils().longTimeToString(startTime.toLong() \* 1000, "yyyy-MM-dd")

val hour = TextUtils().longTimeToString(startTime.toLong() \* 1000, "HH:mm:ss")

vh.text\_goods\_time.text = ("(单次服务) " + time + "(" + TextUtils().getWeekByDateStr(time) + ") " + hour)

} else {

vh.text\_goods\_time.text = ""

}

if (payType == "0") {

if (status == "6") {

vh.textGoodsType.text = ("￥ ${array.pay\_price}" + array.logo\_img + " 已完成 ")

} else {

vh.textGoodsType.text = ("￥ ${array.pay\_price}" + " 待支付 " + array.logo\_img)

vh.textGoodsPay.visibility = View.VISIBLE

}

vh.linearGoodsMap.visibility = View.GONE

vh.mapView.visibility = View.GONE

vh.linearGoodsContent.visibility = View.VISIBLE

vh.linearGoodsPay.visibility = View.VISIBLE

} else {

vh.textGoodsType.text = ("￥${array.pay\_price} ${array.logo\_img}")

vh.linearGoodsMap.visibility = View.GONE

vh.mapView.visibility = View.GONE

vh.linearGoodsContent.visibility = View.VISIBLE

}

}

"2" -> {

val startTime = array.week\_time

if (startTime != null && startTime != "") {

vh.text\_goods\_time.text = ("() $startTime")

} else {

vh.text\_goods\_time.text = ""

}

if (payType == "0") {

if (status == "6") {

vh.textGoodsType.text = ("￥ ${array.pay\_price}" + array.logo\_img + " 已完成 ")

} else {

vh.textGoodsType.text = ("￥ ${array.pay\_price}" + " 待支付 " + array.logo\_img)

vh.textGoodsPay.visibility = View.VISIBLE

}

vh.linearGoodsMap.visibility = View.GONE

vh.mapView.visibility = View.GONE

vh.linearGoodsContent.visibility = View.VISIBLE

vh.linearGoodsPay.visibility = View.VISIBLE

vh.textGoodsPay.visibility = View.GONE

} else {

vh.textGoodsType.text = ("￥${array.pay\_price} ${array.logo\_img}")

vh.linearGoodsMap.visibility = View.GONE

vh.mapView.visibility = View.GONE

vh.linearGoodsContent.visibility = View.VISIBLE

}

}

// 实时上门

"3" -> {

val startTime = array.create\_time

if (startTime != null && startTime != "") {

val time = TextUtils().longTimeToString(startTime.toLong() \* 1000, "yyyy-MM-dd")

val hour = TextUtils().longTimeToString(startTime.toLong() \* 1000, "HH:mm:ss")

vh.text\_goods\_time.text = ("(实时上门) " + time + "(" + TextUtils().getWeekByDateStr(time) + ") " + hour)

} else {

vh.text\_goods\_time.text = ""

}

if (distance != null && distance != "" && distance != "null") {

vh.tvYuTime.text = "${(distance.toFloat() / 80 / 60).toInt()}"

vh.linearFen.visibility = View.VISIBLE

} else {

vh.linearFen.visibility = View.GONE

vh.tvYuTime.text = ""

}

if (payType == "0") {

if (status == "6") {

vh.textGoodsType.text = ("￥ ${array.pay\_price}" + array.logo\_img + " 已完成 ")

} else {

vh.textGoodsType.text = ("￥ ${array.pay\_price}" + " 待支付 " + array.logo\_img)

vh.textGoodsPay.visibility = View.VISIBLE

}

vh.linearGoodsMap.visibility = View.VISIBLE

vh.mapView.visibility = View.VISIBLE

vh.linearGoodsContent.visibility = View.VISIBLE

} else {

vh.textGoodsType.text = ("￥${array.pay\_price} ${array.logo\_img}")

vh.mapView.visibility = View.VISIBLE

vh.linearGoodsMap.visibility = View.VISIBLE

vh.linearGoodsContent.visibility = View.VISIBLE

}

}

}

showStatus(vh, mode, status, payType)

if (array.end\_address == "" || array.end\_address == null) {

vh.text\_goods\_address.text = (array.city + array.address).trim()

} else {

vh.text\_goods\_address.text = (array.address + "-${array.end\_address}").trim()

}

}

"2" -> {

vh.text\_goods\_service.visibility = View.GONE

vh.linearFen.visibility = View.GONE

vh.text\_goods\_services.visibility = View.GONE

vh.fenRelative.visibility = View.GONE

vh.text\_goods\_name.text = if (array.worker\_name != null && array.worker\_name != "") {

array.worker\_name

} else {

}

vh.mapView.visibility = View.VISIBLE

vh.linearGoodsMap.visibility = View.VISIBLE

val createTime = array.create\_time

vh.textGoodsType.text = ("￥${array.pay\_price} ${array.logo\_img}")

if (createTime != null && createTime != "") {

val time = TextUtils().longTimeToString(createTime.toLong() \* 1000, "yyyy-MM-dd")

val hour = TextUtils().longTimeToString(createTime.toLong() \* 1000, "HH:mm:ss")

vh.text\_goods\_time.text = ("(摩跑服务)" + time + "(" + TextUtils().getWeekByDateStr(time) + ") " + hour)

} else {

vh.text\_goods\_time.text = ""

}

if (array.go\_address == "" || array.go\_address == null) {

vh.text\_goods\_address.text = array.address.trim()

} else {

vh.text\_goods\_address.text = (array.address + "—${array.go\_address}").trim()

}

if (distance != null && distance != "" && distance != "null") {

vh.tvYuTime.text = "${(distance.toFloat() / 80 / 60).toInt()}"

vh.linearFen.visibility = View.VISIBLE

} else {

vh.linearFen.visibility = View.GONE

vh.tvYuTime.text = ""

}

if (mode == null || mode == "") {

mode = "0"

}

showStatus(vh, mode, status, payType)

}

"3" -> {

}

}

vh.text\_goods\_service.text = array.title

vh.text\_goods\_services.text = array.title

ImageUtil.getInstance().loadImageNoTransformation(context, vh.image\_goods\_head, R.drawable.comment\_good, Config.IP + array.loge)

if (array.end\_address != "") {

vh.textGoodsPay.visibility = View.GONE

}

if (vh.mapView.visibility == View.VISIBLE) {

vh.mapView.onCreate(null)

val aMap = vh.mapView.map

val mCameraUpdate = CameraUpdateFactory.zoomTo(17f)

aMap!!.moveCamera(mCameraUpdate)

val lat: String = array.latitude

val lng: String = array.longitude

val goLat = array.go\_latitude

val goLng = array.go\_longitude

Log.i("result", "------" + lat + "------" + lng)

if (lat != "" && lng != "") {

val latLng: LatLng = LatLng(lat.toDouble(), lng.toDouble())

aMap.uiSettings.isZoomControlsEnabled = false

aMap.uiSettings.isScrollGesturesEnabled = true

aMap.uiSettings.isScaleControlsEnabled = true

if (array.uid.toInt() != Share().getUid(context)) {

/// 默认窗口

val marker = aMap.addMarker(MarkerOptions().position(latLng)

.visible(true)

.draggable(true)

.icon(BitmapDescriptorFactory.fromBitmap(BitmapFactory.decodeResource(context.resources, R.drawable.end))))

aMap.setInfoWindowAdapter(MyInfoWindow(LatLng(goLat.toDouble(), goLng.toDouble()), latLng))

marker.showInfoWindow()

aMap.moveCamera(CameraUpdateFactory.newLatLng(latLng))

aMap.isTrafficEnabled = false

} else {

val marker = aMap.addMarker(MarkerOptions().position(latLng)

.visible(true)

.draggable(true)

.icon(BitmapDescriptorFactory.fromBitmap(BitmapFactory.decodeResource(context.resources,

R.drawable.end))))

aMap.setInfoWindowAdapter(AddressInfoWindow(array.address))

marker.showInfoWindow()

aMap.moveCamera(CameraUpdateFactory.newLatLng(latLng))

}

aMap.setOnMapClickListener {

when (order\_type) {

"2" -> {

val intent = Intent(context, MoPaoSendActivity::class.java)

intent.putExtra("startLat", latLng)

intent.putExtra("endLat", LatLng(goLat.toDouble(), goLng.toDouble()))

intent.putExtra("code", 2)

intent.putExtra("worker\_id", array.worker\_id)

intent.putExtra("sendUid", array.uid)

intent.putExtra("orderNum", array.order\_number)

context.startActivity(intent)

}

else -> {

val intent = Intent(context, ActualTimeActivity::class.java)

intent.putExtra("startLat", Share().getLatLng(context))

intent.putExtra("endLat", latLng)

intent.putExtra("orderNum", array.order\_number)

intent.putExtra("sendUid", array.uid)

intent.putExtra("take", array.take)

intent.putExtra("distance", distance)

intent.putExtra("status", status)

context.startActivity(intent)

}

}

vh.mapView.onDestroy()

}

} else {

}

}

vh.textGoodsSale.setOnClickListener {

val intent = Intent()

intent.setClass(context, AfterSaleActivity::class.java)

context.startActivity(intent)

}

vh.textGoodsComment.setOnClickListener {

val intent = Intent()

intent.setClass(context, AddCommentActivity::class.java)

intent.putExtra("id", array.id)

intent.putExtra("order\_type", order\_type)

intent.putExtra("code", 1)

context.startActivity(intent)

}

/// 订单确认

vh.text\_order\_sure.setOnClickListener {

num = position

ids = array.id

getSure(array.order\_number)

}

vh.textGoodsPay.setOnClickListener {

val intent = Intent()

intent.setClass(context, PayOrderActivity::class.java)

intent.putExtra("payId", array.order\_number)

if (array.worker\_name == null || array.worker\_name == "") {

if (array.shop\_name != null && array.shop\_name != "") {

intent.putExtra("worker\_name", array.shop\_name)

} else {

intent.putExtra("worker\_name", "保洁人员")

}

} else {

intent.putExtra("worker\_name", array.worker\_name)

}

intent.putExtra("position", 0)

intent.putExtra("code", 1)

intent.putExtra("money", array.pay\_price)

context.startActivity(intent)

}

vh.textGoodsCancel.setOnClickListener {

num = position

dialogTip(list[position].id)

}

}

private fun showStatus(vh: VH, mode: String, status: String?, payType: String) {

when (status) {

"0" -> {

vh.linearGoodsPay.visibility = View.VISIBLE

vh.linearGoodsOrder.visibility = View.GONE

vh.linearGoodsComment.visibility = View.GONE

if (mode == "2") {

vh.textGoodsPay.visibility = View.GONE

} else {

vh.textGoodsPay.visibility = View.VISIBLE

}

}

"1" -> {

vh.linearGoodsOrder.visibility = View.GONE

vh.linearGoodsPay.visibility = View.VISIBLE

vh.linearGoodsComment.visibility = View.GONE

}

"2" -> {

vh.linearGoodsOrder.visibility = View.GONE

vh.linearGoodsPay.visibility = View.VISIBLE

if (payType != "0") {

vh.textGoodsPay.visibility = View.GONE

} else {

if (mode == "2") {

vh.textGoodsPay.visibility = View.GONE

} else {

vh.textGoodsPay.visibility = View.VISIBLE

}

}

vh.linearGoodsComment.visibility = View.GONE

}

"3" -> {

vh.linearGoodsOrder.visibility = View.GONE

vh.linearGoodsPay.visibility = View.GONE

vh.textGoodsPay.visibility = View.GONE

vh.linearGoodsComment.visibility = View.GONE

}

"4" -> {

vh.linearGoodsOrder.visibility = View.VISIBLE

vh.linearGoodsPay.visibility = View.VISIBLE

vh.linearGoodsComment.visibility = View.GONE

if (mode == "2") {

vh.textGoodsCancel.visibility = View.VISIBLE

vh.textGoodsPay.visibility = View.GONE

} else {

vh.textGoodsCancel.visibility = View.GONE

vh.textGoodsPay.visibility = View.GONE

}

}

"5" -> {

vh.linearGoodsOrder.visibility = View.GONE

vh.linearGoodsPay.visibility = View.GONE

vh.linearGoodsComment.visibility = View.VISIBLE

}

"6" -> {

vh.linearGoodsOrder.visibility = View.GONE

vh.linearGoodsPay.visibility = View.GONE

vh.linearGoodsComment.visibility = View.GONE

vh.linearGoodsMap.visibility = View.GONE

}

}

}

override fun onCreateView(parent: ViewGroup, viewType: Int): RecyclerView.ViewHolder {

val view: View = inflater.inflate(R.layout.item\_goods, parent, false)

return VH(view)

}

private fun dialogTip(ids: String) {

val inflater\_type = LayoutInflater.from(context)

val layout\_type = inflater\_type.inflate(R.layout.pop\_order\_cancel, null)

val record\_type = AlertDialog.Builder(context)

dialog\_tips = record\_type.create()

dialog\_tips.show()

dialog\_tips.setContentView(layout\_type)

dialog\_tips.setCanceledOnTouchOutside(true)

val imageCancelClose: ImageView = layout\_type.findViewById(R.id.image\_cancel\_close)

var textCancelTitle: TextView = layout\_type.findViewById(R.id.text\_cancel\_title)

val textCancelSure: TextView = layout\_type.findViewById(R.id.text\_cancel\_sure)

val textCancelNot: TextView = layout\_type.findViewById(R.id.text\_cancel\_not)

textCancelSure.setOnClickListener {

dialog\_tips.dismiss()

getOrder(ids)

}

imageCancelClose.setOnClickListener {

dialog\_tips.dismiss()

}

textCancelNot.setOnClickListener {

dialog\_tips.dismiss()

}

}

private fun getOrder(ids: String) {

val json = JSONObject()

json.put("uid", Share().getUid(context))

json.put("type", "2")

json.put("id", ids)

HttpUtils.getInstance().postJsonWithHeader(Config.GET\_ORDER\_URL, json.toString(), Config.GET\_ORDER, Share().getToken(context!!), handler)

}

/\*\*

\* 确认完工 先要确认支付

\*/

private fun getSure(ids: String) {

val json = JSONObject()

json.put("uid", Share().getUid(context))

json.put("status", "5")

json.put("order\_number", ids)

json.put("type", "1")

HttpUtils.getInstance().postJsonWithHeader(Config.GET\_SHOP\_STATUS\_URL, json.toString(), Config.GET\_SHOP\_STATUS, Share().getToken(context!!), handler)

(context as GoodsActivity).showLoadDialog("操作中")

}

private val handler: Handler = object : Handler(Looper.getMainLooper()) {

override fun handleMessage(msg: Message?) {

super.handleMessage(msg)

val str: String? = msg!!.obj as String

when (msg.what) {

Config.GET\_ORDER -> {

if (str != null && str != "") {

Log.i("toby", "str>>>>>$str")

val json = JSONObject(str)

val code = json.optInt("code")

val msg = json.getString("msg")

if (code == 200) {

list.removeAt(num)

notifyDataSetChanged()

} else {

}

}

}

Config.GET\_SHOP\_STATUS -> {

(context as GoodsActivity).removeLoadDialog()

if (str != null && str != "") {

Log.i("toby", "str>>>>>$str")

val json = JSONObject(str)

val code = json.optInt("code")

val message = json.getString("msg")

ToastUtils.showLongToast(context, message)

if (code == 200) {

/// 移除数据

list.removeAt(num)

notifyDataSetChanged()

}

}

}

}

}

}

inner class AddressInfoWindow(val address: String) : AMap.InfoWindowAdapter {

override fun getInfoContents(p0: Marker?): View {

val view = inflater.inflate(R.layout.map\_info\_custom, null)

val textView = view.findViewById<TextView>(R.id.tvTextView)

textView.text = address.trim()

return view

}

override fun getInfoWindow(p0: Marker?): View {

val view = inflater.inflate(R.layout.map\_info\_custom, null)

val textView = view.findViewById<TextView>(R.id.tvTextView)

textView.text = address.trim()

return view

}

}

inner class MyInfoWindow(startLatLng: LatLng, endLatLng: LatLng) : AMap.InfoWindowAdapter {

val dd = AMapUtils.calculateLineDistance(LatLng(startLatLng.latitude, startLatLng.longitude), LatLng(endLatLng.latitude, endLatLng.longitude))

override fun getInfoContents(p0: Marker?): View {

val view = inflater.inflate(R.layout.map\_info\_custom, null)

val textView = view.findViewById<TextView>(R.id.tvTextView)

textView.text = "距离我${dd.toInt() / 1000}千米"

return view

}

override fun getInfoWindow(p0: Marker?): View {

val view = inflater.inflate(R.layout.map\_info\_custom, null)

val textView = view.findViewById<TextView>(R.id.tvTextView)

textView.text = "距离我${dd.toInt() / 1000}千米"

return view

}

}

override fun onUpdateHolder(holder: RecyclerView.ViewHolder, position: Int, payloads: MutableList<Any>) {

}

class VH(view: View) : RecyclerView.ViewHolder(view) {

init {

ButterKnife.bind(this, view)

}

@BindView(R.id.linear\_goods\_order)

lateinit var linearGoodsOrder: LinearLayout

@BindView(R.id.text\_goods\_pay)

lateinit var textGoodsPay: TextView

@BindView(R.id.text\_goods\_type)

lateinit var textGoodsType: TextView

@BindView(R.id.linear\_goods\_pay)

lateinit var linearGoodsPay: LinearLayout

@BindView(R.id.linear\_goods\_comment)

lateinit var linearGoodsComment: LinearLayout

@BindView(R.id.text\_goods\_sale)

lateinit var textGoodsSale: TextView

@BindView(R.id.text\_goods\_comment)

lateinit var textGoodsComment: TextView

@BindView(R.id.text\_goods\_cancel)

lateinit var textGoodsCancel: TextView

@BindView(R.id.linear\_goods\_map)

lateinit var linearGoodsMap: LinearLayout

@BindView(R.id.linear\_goods\_content)

lateinit var linearGoodsContent: LinearLayout

@BindView(R.id.map\_view)

lateinit var mapView: MapView

@BindView(R.id.text\_goods\_name)

lateinit var text\_goods\_name: TextView

@BindView(R.id.text\_goods\_service)

lateinit var text\_goods\_service: TextView

@BindView(R.id.text\_goods\_services)

lateinit var text\_goods\_services: TextView

@BindView(R.id.text\_goods\_address)

lateinit var text\_goods\_address: TextView

@BindView(R.id.text\_goods\_time)

lateinit var text\_goods\_time: TextView

@BindView(R.id.image\_goods\_head)

lateinit var image\_goods\_head: ImageView

@BindView(R.id.text\_order\_sure)

lateinit var text\_order\_sure: TextView

@BindView(R.id.tvYuTime)

lateinit var tvYuTime: TextView

@BindView(R.id.linearFen)

lateinit var linearFen: LinearLayout

@BindView(R.id.fenRelative)

lateinit var fenRelative: RelativeLayout

}

}

package com.wt.yc.oto.oto.yc.activity

import android.app.Activity

import android.content.Intent

import android.os.Bundle

import android.os.Message

import android.util.Log

import android.view.View

import com.google.gson.Gson

import com.google.gson.JsonArray

import com.wt.yc.oto.oto.R

import com.wt.yc.oto.oto.activity.AddressActivity

import com.wt.yc.oto.oto.activity.ProActivity

import com.wt.yc.oto.oto.info.OrderInfo

import com.wt.yc.oto.oto.info.PersonInfo

import com.wt.yc.oto.oto.info.PicText

import com.wt.yc.oto.oto.info.ShopInfo

import com.wt.yc.oto.oto.model.AddressModel

import com.wt.yc.oto.oto.model.CouponModel

import com.wt.yc.oto.oto.net.Config

import com.wt.yc.oto.oto.net.Share

import com.wt.yc.oto.oto.wxl.GoodsActivity

import com.xin.lv.yang.utils.info.HttpInfo

import com.xin.lv.yang.utils.utils.HttpUtils

import kotlinx.android.synthetic.main.item\_top.\*

import kotlinx.android.synthetic.main.make\_order\_layout.\*

import org.json.JSONArray

import org.json.JSONObject

class MakeOrderActivity : ProActivity() {

var address\_id: String = ""

var visit\_start\_time = ""

var visit\_end\_time = ""

var mode: Int = 0

var shopInfo = ""

var shopId = ""

var coupon: CouponModel? = null

var time = ""

var money = ""

var content: PicText? = null

var dataStr: PicText? = null

var personInfo: PersonInfo? = null

var serviceJson: String = ""

var qingMoney = "50"

override fun handler(msg: Message) {

when (msg.what) {

Config.GET\_ORDER\_PRODUCE -> {

removeLoadDialog()

Log.i("toby", "sssss-------->${msg.obj}")

val json = JSONObject(msg.obj.toString())

val code = json.optInt("code")

if (code == 200) {

val data = json.getJSONObject("data")

val order = data.optString("order")

if (order != null && order != "" && order != "null") {

when (mode) {

3 -> {

val orderIntent = Intent(this, GoodsActivity::class.java)

orderIntent.putExtra("position", 5)

startActivity(orderIntent)

finish()

}

else -> {

val orderInfo: OrderInfo = gson!!.fromJson(order, OrderInfo::class.java)

val payIntent = Intent(this, PayDetailsActivity::class.java)

payIntent.putExtra("data", orderInfo)

val address = addressModel!!.city + addressModel!!.address

payIntent.putExtra("address", address)

payIntent.putExtra("code", mode)

val yearTime = dataStr!!.year + "年" + dataStr!!.month + "月" + dataStr!!.day + "日"

val title = content!!.text!!.split("\n")[0]

payIntent.putExtra("time", "$yearTime $title")

payIntent.putExtra("workerInfo", personInfo)

payIntent.putExtra("alltime", time)

payIntent.putExtra("oneMoney", "25")

payIntent.putExtra("service", jsonArr.toString())

startActivity(payIntent)

}

}

}

}

showToastShort(json.optString("msg"))

}

}

}

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.make\_order\_layout)

image\_back.setOnClickListener { finish() }

mode = intent.getIntExtra("code", -1)

shopId = intent.getStringExtra("id")

shopInfo = intent.getStringExtra("shopInfo")

when (mode) {

1 -> {

text\_top.text = "单次预约"

bt\_next.text = "提交"

choose\_type\_linear.visibility = View.GONE

tv\_ti.visibility = View.GONE

}

3 -> {

text\_top.text = "实时上门"

bt\_next.text = "生成订单"

choose\_type\_linear.visibility = View.GONE

tv\_ti.visibility = View.VISIBLE

}

else -> {

text\_top.text = "立即预约"

choose\_type\_linear.visibility = View.VISIBLE

tv\_ti.visibility = View.GONE

}

}

choose\_index\_linear.setOnClickListener {

// 选择服务地点

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

intent.putExtra("type", 1)

startActivityForResult(intent, 1)

}

choose\_type\_linear.setOnClickListener {

/// 选择服务类型

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

startActivityForResult(intent, 2)

}

choose\_time\_linear.setOnClickListener {

// 服务时间

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

intent.putExtra("typeid", shopId)

intent.putExtra("shopInfo", shopInfo)

intent.putExtra("code", mode)

startActivityForResult(intent, 3)

}

check\_box.setOnCheckedChangeListener { p0, p1 ->

qingMoney = if (p1) {

"5"

} else {

"0"

}

}

choose\_you\_linear.setOnClickListener {

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

startActivityForResult(intent, 4)

}

bt\_next.setOnClickListener {

/// 下一步

postService()

}

}

val MAP\_URL = "https://restapi.amap.com/v3/geocode/geo"

val MAP\_KEY = "157451a5109276d0edaa88bbacfee7aa"

var mapAddressResult = ""

lateinit var jsonArr: JSONArray

private fun postService() {

jsonArr = JSONArray()

when {

else -> {

showLoadDialog("正在生成")

val json = JSONObject()

json.put("uid", Share().getUid(this))

json.put("type", mode)

json.put("typeid", shopId) // 商品id

json.put("address\_id", address\_id) // 地址 id

json.put("service", jsonArr.put(JSONObject(serviceJson))) // 订单服务

json.put("visit\_start\_time", visit\_start\_time) //上门开始时间

json.put("num", time)

json.put("pre\_price", money) //订单原价

json.put("visit\_end\_time", visit\_end\_time) //上门结束时间

json.put("type1", Share().getOneTypeId(this))

if (mode == 3) {

if (mapAddressResult != null && mapAddressResult != "") {

val jjJson = JSONObject(mapAddressResult)

val infocode = jjJson.optInt("infocode")

if (infocode == 10000) {

val jsonArray = jjJson.optJSONArray("geocodes")

val jsonObject = jsonArray.optJSONObject(0)

if (jsonObject != null) {

val location = jsonObject.optString("location")

val locArr = location.split(",")

json.put("longitude", locArr[0])

json.put("latitude", locArr[1])

}

} else {

json.put("longitude", "")

json.put("latitude", "")

}

} else {

json.put("longitude", "")

json.put("latitude", "")

}

json.put("number", time)

}

if (personInfo != null) {

json.put("worker\_id", personInfo!!.uid)

}

if (coupon != null) {

json.put("ticket\_id", coupon!!.id) // 优惠券id

json.put("pay\_price", (money.toDouble() - coupon!!.money.toDouble())) //支付金额

} else {

json.put("pay\_price", money)

}

Log.i("toby", json.toString())

HttpUtils.getInstance().postJsonWithHeader(Config.GET\_ORDER\_PRODUCE\_URL, json.toString(), Config.GET\_ORDER\_PRODUCE, Share().getToken(this).toString(), handler)

}

}

}

fun addressToLatLng() {

Thread() {

kotlin.run {

val mapList: ArrayList<HttpInfo> = arrayListOf()

val add = addressModel!!.address.toString()

mapList.add(HttpInfo("key", MAP\_KEY))

mapList.add(HttpInfo("address", add))

mapAddressResult = HttpUtils.getInstance().getTong(MAP\_URL, mapList)

log("---------$add-----$mapAddressResult")

}

}.start()

}

var upTime = ""

var addressModel: AddressModel? = null

override fun onActivityResult(requestCode: Int, resultCode: Int, data: Intent?) {

super.onActivityResult(requestCode, resultCode, data)

if (resultCode == Activity.RESULT\_OK) {

when (requestCode) {

1 -> {

val address = data!!.getStringExtra("address")

addressModel = gson!!.fromJson(address, AddressModel::class.java)

address\_id = addressModel!!.id

make\_tv\_address.text = addressModel!!.city.toString() + addressModel!!.address.toString()

if (mode == 3) {

addressToLatLng()

}

}

2 -> {

val shopInfo: ShopInfo = data!!.getParcelableExtra("name")

tv\_make\_type.text = shopInfo.text

}

3 -> {

when (mode) {

3 -> {

time = data!!.getStringExtra("time")

// 所需金额

money = data.getStringExtra("money")

// 服务信息，包括次数 金额 json数组格式

serviceJson = data.getStringExtra("service")

tv\_make\_time.text = "${time}小时 ${money}元"

}

else -> {

// 服务时长

time = data!!.getStringExtra("time")

// 所需金额

money = data.getStringExtra("money")

personInfo = data.getParcelableExtra("person")

content = data.getParcelableExtra("content") as PicText

dataStr = data.getParcelableExtra("data") as PicText

// 服务信息，包括次数 金额 json数组格式

serviceJson = data.getStringExtra("service")

val yearTime = dataStr!!.year + "-" + dataStr!!.month + "-" + dataStr!!.day

val ttt = content!!.text!!

val title = if (ttt.contains("\n")) {

ttt.split("\n")[0]

} else {

ttt

}

val time = title.split("-")[0]

upTime = "$yearTime $time"

val times = title.split("-")

/// 开始时间

val start = times[0]

/// 结束时间

val end = times[1]

visit\_start\_time = "$yearTime $start"

visit\_end\_time = "$yearTime $end"

tv\_make\_time.text = "${personInfo!!.name} ${dataStr!!.text!!.replace("\n", "")}$title ${money}元"

}

}

}

4 -> {

coupon = data!!.getParcelableExtra("data")

}

}

}

}

}

package com.wt.yc.oto.oto.yc.activity

import android.app.Activity

import android.content.Intent

import android.graphics.drawable.BitmapDrawable

import android.os.Bundle

import android.os.Message

import android.support.v7.widget.GridLayoutManager

import android.support.v7.widget.LinearLayoutManager

import android.support.v7.widget.RecyclerView

import android.util.Log

import android.view.Gravity

import android.view.View

import android.widget.PopupWindow

import com.google.gson.reflect.TypeToken

import com.wt.yc.oto.oto.R

import com.wt.yc.oto.oto.activity.ProActivity

import com.wt.yc.oto.oto.adapter.ItemClickListener

import com.wt.yc.oto.oto.info.PersonInfo

import com.wt.yc.oto.oto.info.PicText

import com.wt.yc.oto.oto.info.ShopInfo

import com.wt.yc.oto.oto.model.GoodsModel

import com.wt.yc.oto.oto.net.Config

import com.wt.yc.oto.oto.net.ConfigNet

import com.wt.yc.oto.oto.net.Share

import com.wt.yc.oto.oto.yc.adapter.PeopleAdapter

import com.wt.yc.oto.oto.yc.adapter.TextAdapter

import com.xin.lv.yang.utils.line.DividerItemDecoration

import com.xin.lv.yang.utils.utils.BitmapUtil

import com.xin.lv.yang.utils.utils.HttpUtils

import kotlinx.android.synthetic.main.item\_top.\*

import kotlinx.android.synthetic.main.notification\_template\_part\_time.\*

import kotlinx.android.synthetic.main.yc\_choose\_service\_time.\*

import org.json.JSONArray

import org.json.JSONObject

import java.util.\*

import kotlin.collections.ArrayList

class ServiceTimeActivity : ProActivity() {

var typeid = ""

var shopInfo = ""

override fun handler(msg: Message) {

val str = msg.obj as String

Log.i("result", "str>>>>>----$str")

when (msg.what) {

ConfigNet.GET\_PERSON\_CODE -> {

removeLoadDialog()

Log.i("result", "服务人员返回------$str")

val json = JSONObject(str)

val code = json.optInt("code")

if (code == 200) {

val result = json.optString("data")

if (result != "" && result != "null") {

personArr = gson!!.fromJson(result, object : TypeToken<ArrayList<PersonInfo>>() {}.type)

showChoosePerson(personArr!!)

}

}

showToastShort(json.optString("msg"))

}

Config.GET\_PAI -> {

removeLoadDialog()

Log.i("result", "str>>>>>$str")

if (str != null && str != "") {

val json = JSONObject(str)

val code = json.optInt("code")

val message = json.getString("msg")

if (code == 200) {

val resultJson = json.getJSONArray("data")

val len = resultJson.length()

for (i in 0 until len) {

val model = GoodsModel();

model.name = resultJson[i].toString()

}

} else {

}

}

}

Config.GET\_SERVICE\_PAI -> {

log("排班列表----str>>>>>$str")

/// 工作人员可利用的时间

val json = JSONObject(str)

val code = json.optInt("code")

if (code == 200) {

val result = json.optString("data")

checkDate(result)

}

}

}

}

inner class D {

var visit\_start\_time: String = ""

var visit\_end\_time: String = ""

}

val contentArr = arrayListOf<PicText>()

private fun checkDate(result: String) {

contentArr.clear()

for (temp in titleStr) {

val p = PicText()

p.text = temp

p.isChoose = false

contentArr.add(p)

}

if (result != null && result != "" && result != "null") {

val dateArr: ArrayList<D> = gson!!.fromJson(result, object : TypeToken<ArrayList<D>>() {}.type)

for (temp in dateArr) {

val num = checkList(temp.visit\_start\_time, temp.visit\_end\_time)

if (num != -1) {

val picText = contentArr[num]

picText.isChoose = true

}

}

}

val indexTime = System.currentTimeMillis()

val hhmm = BitmapUtil.longToTime(indexTime / 1000, "HH:mm")

val yearDate = BitmapUtil.longToTime(indexTime / 1000, "yyyy-MM-dd")

/// 当前时间为选择的时间，判断当前时间在哪个时间段

if (yearDate == chooseDay) {

val code: Int = checkFan(hhmm)

log("返回的从的数------$code")

if (code != -1) {

for (i in 0..code) {

val pp = contentArr[i]

pp.isChoose = true

}

}

}

initContentRecycler(contentArr)

}

/\*\*

\* 当前时间在哪个范围内

\*/

private fun checkFan(hhmm: String): Int {

log("------------$hhmm")

val hh = hhmm.split(":")[0].toInt()

val mm = hhmm.split(":")[1].toInt()

for (i in titleStr.indices) {

val str = titleStr[i]

val arr = str.split("-")

val start = arr[0]

val end = arr[1]

val start1 = start.split(":")[0].toInt()

val start2 = start.split(":")[1].toInt()

val end1 = end.split(":")[0].toInt()

if ((hh == start1 && mm < 59) || (hh > start1 && hh <= (start1 + 1))) {

return i

}

}

return -1

}

private fun checkList(visit\_start\_time: String, visit\_end\_time: String): Int {

for (i in titleStr.indices) {

val arr = titleStr[i].split("-")

if (visit\_start\_time == arr[0] && visit\_end\_time == arr[1]) {

return i

}

}

return -1

}

var code: Int = 0

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.yc\_choose\_service\_time)

image\_back.setOnClickListener { finish() }

text\_top.text = "选择服务时间"

typeid = intent.getStringExtra("typeid")

shopInfo = intent.getStringExtra("shopInfo")

code = intent.getIntExtra("code", 0)

if (code == 3) {

choosePersonLinear.visibility = View.GONE

title\_recycler\_view.visibility = View.GONE

content\_recycler\_view.visibility = View.GONE

}

showShop()

initClick()

}

/\*\*

\* 服务时长number

\*/

var count: String = ""

/\*\*

\* 当前选择的位置，默认为 0

\*/

var position = 0

var resultArr: ArrayList<ShopInfo> = ArrayList()

/\*\*

\* 交易总金额

\*/

var price = ""

private fun showShop() {

if (shopInfo != "" && shopInfo != "null") {

val list: ArrayList<ShopInfo> = gson!!.fromJson(shopInfo, object : TypeToken<ArrayList<ShopInfo>>() {}.type)

for (temp in list) {

if (temp.shop\_name != "" && temp.shop\_name != "null") {

resultArr.add(temp)

}

}

tv\_text\_num.text = resultArr[position].shop\_name

tv\_service\_money.text = resultArr[position].money

}

}

private fun initClick() {

tv\_jian.setOnClickListener {

val len = resultArr.size

// 数量减

if (position > 0) {

position--

tv\_text\_num.text = resultArr[position].shop\_name.toString()

tv\_service\_money.text = ("${resultArr[position].money}元").toString()

}

}

tv\_add\_num.setOnClickListener {

val len = resultArr.size

// 数量加

if (position < len - 1) {

position++

tv\_text\_num.text = resultArr[position].shop\_name.toString()

price = resultArr[position].money

tv\_service\_money.text = ("${resultArr[position].money}元").toString()

}

}

bt\_qr\_choose\_time.setOnClickListener {

when (code) {

3 -> {

val money = resultArr[position].money

val str: String = resultArr[position].shop\_name

if (str.contains("小时")) {

count = str.replace("小时", "")

} else {

count = "1"

}

val data = Intent()

data.putExtra("time", count.toString())

data.putExtra("money", money)

data.putExtra("service", buildService())

setResult(Activity.RESULT\_OK, data)

finish()

}

else -> {

if (contentAdapter == null) {

showToastShort("请选择时间")

} else {

// 时间段集合

val arr = contentAdapter!!.getSelector()

when {

choosePersonInfo == null -> showToastShort("请选择人员")

arr.isEmpty() -> showToastShort("请选择时间段")

else -> {

val money = resultArr[position].money

val str: String = resultArr[position].shop\_name

if (str.contains("小时")) {

count = str.replace("小时", "")

} else {

count = "1"

}

val data = Intent()

val content: PicText? = arr[0]

// 星期集合信息

val titleArr = titleAdapter!!.getSelector()

if (titleArr.isNotEmpty()) {

val title: PicText? = titleArr[0]

data.putExtra("data", title)

}

data.putExtra("time", count.toString())

data.putExtra("money", money)

data.putExtra("content", content)

data.putExtra("service", buildService())

data.putExtra("person", choosePersonInfo)

setResult(Activity.RESULT\_OK, data)

finish()

}

}

}

}

}

}

choosePersonLinear.setOnClickListener {

if (personArr != null && personArr!!.isNotEmpty()) {

showChoosePerson(personArr!!)

} else {

getPeopleFromUrl()

}

}

}

private fun buildService(): String {

val str: String = resultArr[position].shop\_name

if (str.contains("小时")) {

count = str.replace("小时", "")

}

val json = JSONObject()

try {

json.put("sid", resultArr[position].id)

json.put("number", count)

json.put("price", resultArr[position].money)

json.put("total\_price", resultArr[position].money)

} catch (e: Exception) {

e.printStackTrace()

}

log("json-----" + json.toString())

return json.toString()

}

private fun getPeopleFromUrl() {

val json = JSONObject()

json.put("typeid", typeid)

HttpUtils.getInstance().postJsonWithHeader(ConfigNet.GET\_SERVICE\_PERSON\_URL, json.toString(), ConfigNet.GET\_PERSON\_CODE, Share().getToken(this), handler)

showLoadDialog("获取中")

}

var choosePersonPop: PopupWindow? = null

var personArr: ArrayList<PersonInfo>? = null

private fun showChoosePerson(arr: ArrayList<PersonInfo>) {

val view: View = layoutInflater.inflate(R.layout.recycler\_view\_layout, null)

val recyclerView: RecyclerView = view.findViewById(R.id.recycler\_view)

recyclerView.layoutManager = LinearLayoutManager(this)

recyclerView.isNestedScrollingEnabled = false

recyclerView.addItemDecoration(DividerItemDecoration(this, DividerItemDecoration.VERTICAL\_LIST, R.drawable.line))

initAdapter(recyclerView, arr)

choosePersonPop = PopupWindow(view, getW(), getH() / 2, true)

choosePersonPop!!.isOutsideTouchable = true

choosePersonPop!!.setBackgroundDrawable(BitmapDrawable())

choosePersonPop!!.animationStyle = R.style.bottom\_to\_up\_style

choosePersonPop!!.showAtLocation(view, Gravity.BOTTOM, 0, 0)

setAlpha(0.6f)

choosePersonPop!!.setOnDismissListener {

choosePersonPop!!.dismiss()

setAlpha(1f)

}

}

var choosePersonInfo: PersonInfo? = null

private fun initAdapter(recyclerView: RecyclerView, arrayListOf: ArrayList<PersonInfo>) {

val adapter = PeopleAdapter(this, arrayListOf)

recyclerView.adapter = adapter

adapter.getListener(object : ItemClickListener {

override fun onItemClick(position: Int) {

choosePersonInfo = arrayListOf[position]

showPerson()

choosePersonPop!!.dismiss()

setAlpha(1f)

}

override fun onLongClick(position: Int) {

}

})

}

val titleStr = arrayListOf<String>("08:00-11:00",

"09:00-12:00", "10:00-13:00", "11:00-14:00",

"12:00-15:00", "13:00-16:00", "14:00-17:00",

"15:00-18:00", "16:00-19:00", "17:00-20:00")

private fun showPerson() {

if (choosePersonInfo != null) {

tvChoosePerson.text = choosePersonInfo!!.name

}

val personId = choosePersonInfo!!.uid

val arr = arrayListOf<PicText>()

val calendar = Calendar.getInstance()

for (i in 0..6) {

val month = calendar.get(Calendar.MONTH)

val day = calendar.get(Calendar.DAY\_OF\_MONTH)

val dayOfWeek = calendar.get(Calendar.DAY\_OF\_WEEK)

val year = calendar.get(Calendar.YEAR)

val weekStr: String = when (dayOfWeek) {

1 -> "日"

2 -> "一"

3 -> "二"

4 -> "三"

5 -> "四"

6 -> "五"

7 -> "六"

else -> {

""

}

}

val picText: PicText = PicText()

/// 年

picText.year = year.toString()

/// 月

picText.month = when ((month + 1).toString().length) {

1 -> "0${month + 1}"

2 -> "${month + 1}"

else -> {

""

}

}

/// 日

picText.day = when (day.toString().length) {

1 -> "0$day"

2 -> "$day"

else -> {

""

}

}

picText.text = "${

when (i) {

0 -> "今天"

1 -> "明天"

else -> "周$weekStr"

}}\n${month + 1}月${day}日"

picText.click = i == 0

arr.add(picText)

calendar.set(Calendar.DAY\_OF\_MONTH, day + 1)

}

initTitleRecycler(arr)

val picText = arr[0]

val dd = picText.year + "-" + picText.month + "-" + picText.day

getTimeWithDayFrom(personId.toString(), dd)

}

/\*\*

\* 当前选择的时间

\*/

var chooseDay = ""

/\*\*

\* 获取排班信息

\*/

fun getTimeWithDayFrom(personId: String, day: String) {

chooseDay = day

val json = JSONObject()

json.put("uid", personId)

json.put("date", day)

HttpUtils.getInstance().postJsonWithHeader(Config.GET\_SERVICE\_PAI\_URL, json.toString(),

Config.GET\_SERVICE\_PAI, Share().getToken(this), handler)

}

var contentAdapter: TextAdapter? = null

private fun initContentRecycler(arr: ArrayList<PicText>) {

content\_recycler\_view.layoutManager = GridLayoutManager(this, 4)

contentAdapter = TextAdapter(this, arr, 2)

content\_recycler\_view.adapter = contentAdapter

contentAdapter!!.getListener(object : ItemClickListener {

override fun onItemClick(position: Int) {

val pp = arr[position]

if (pp.isChoose) {

showToastShort("该时间段已被预约，请选择其他时间段")

} else {

contentAdapter!!.updateSelector(position)

}

}

override fun onLongClick(position: Int) {

}

})

}

var titleAdapter: TextAdapter? = null

private fun initTitleRecycler(arr: ArrayList<PicText>) {

val linearManager = LinearLayoutManager(this)

linearManager.orientation = LinearLayoutManager.HORIZONTAL

title\_recycler\_view.layoutManager = linearManager

titleAdapter = TextAdapter(this, arr, 1)

title\_recycler\_view.adapter = titleAdapter

titleAdapter!!.getListener(object : ItemClickListener {

override fun onItemClick(position: Int) {

val info = arr[position]

val day = info.year + "-" + info.month + "-" + info.day

getTimeWithDayFrom(choosePersonInfo!!.uid.toString(), day)

titleAdapter!!.updateSelector(position)

}

override fun onLongClick(position: Int) {

}

})

}

}

**六、我的兼职信息**

package com.wt.yc.oto.oto.yc.parttime

import android.content.Intent

import android.os.Bundle

import android.os.Message

import android.support.v7.widget.LinearLayoutManager

import android.view.View

import com.google.gson.reflect.TypeToken

import com.wt.yc.oto.oto.R

import com.wt.yc.oto.oto.activity.LoginActivity

import com.wt.yc.oto.oto.activity.ProActivity

import com.wt.yc.oto.oto.adapter.ItemClickListener

import com.wt.yc.oto.oto.fragment.LoadFragment

import com.wt.yc.oto.oto.net.Config

import com.wt.yc.oto.oto.net.ConfigNet

import com.wt.yc.oto.oto.net.Share

import com.wt.yc.oto.oto.wxl.parttimecat.AddPartActivity

import com.xin.lv.yang.utils.line.DividerItemDecoration

import com.xin.lv.yang.utils.utils.HttpUtils

import kotlinx.android.synthetic.main.part\_time\_job\_layout.\*

import org.json.JSONObject

class PartTimeJobActivity : ProActivity() {

/\*\*

\* 一级id信息

\*/

var id = 0

override fun handler(msg: Message) {

val str = msg.obj as String

when (msg.what) {

ConfigNet.GET\_PAET\_TIME\_CODE -> {

removeLoadDialog()

val json = JSONObject(str)

val code = json.optInt("code")

if (code == 200) {

val result = json.optString("data")

val list: ArrayList<PartInfo> = gson!!.fromJson(result, object : TypeToken<ArrayList<PartInfo>>() {}.type)

partTimeAdapter!!.updateData(list)

}

}

ConfigNet.BAO\_MING\_CODE -> {

val json = JSONObject(str)

val code = json.optInt("code")

if (code == 200) {

updateData()

}

showToastShort(json.optString("msg"))

}

Config.GET\_USER -> {

removeLoadDialog()

val json = JSONObject(str)

val code = json.optInt("code")

if (code == 200) {

val jsonObject = json.optJSONObject("result")

val status = jsonObject.optInt("status")

if (status != 0) {

val type = jsonObject.optInt("type1")

toWhere(type)

} else {

showToastShort("账号正在审核，请耐心等待")

}

}

}

}

}

private fun updateData() {

val info = partTimeAdapter!!.list[baoPosition]

info.part\_uid = 1

partTimeAdapter!!.notifyDataSetChanged()

}

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.part\_time\_job\_layout)

id = intent.getIntExtra("id", 0)

initClick()

initLoad()

initContentRecyclerView()

getPartTimeList()

}

private fun getPartTimeList() {

val json = JSONObject()

json.put("uid", uid)

json.put("index", 1)

json.put("site", "")

json.put("type", 1)

json.put("sort", 1)

HttpUtils.getInstance().postJsonWithHeader(ConfigNet.GET\_PART\_TIME\_LIST\_URL, json.toString(),

ConfigNet.GET\_PAET\_TIME\_CODE, token, handler)

showLoadDialog("加载中")

}

var partTimeAdapter: PartTimeAdapter? = null

var list = arrayListOf<PartInfo>()

var baoPosition: Int = 0

private fun initContentRecyclerView() {

partTimeRecycler.isNestedScrollingEnabled = false

partTimeRecycler.layoutManager = LinearLayoutManager(this)

partTimeAdapter = PartTimeAdapter(this, list, 1)

partTimeRecycler.addItemDecoration(DividerItemDecoration(this, DividerItemDecoration.VERTICAL\_LIST, R.drawable.line))

partTimeRecycler.adapter = partTimeAdapter

partTimeAdapter!!.getListener(object : ItemClickListener {

override fun onItemClick(position: Int) {

val intent = Intent(this@PartTimeJobActivity, PartTimeDetailsActivity::class.java)

intent.putExtra("id", list[position].id)

intent.putExtra("name", list[position].title)

startActivity(intent)

}

override fun onLongClick(position: Int) {

}

})

partTimeAdapter!!.onBaoListener = object : PartTimeAdapter.OnBaoListener {

override fun onBao(position: Int) {

baoPosition = position

// 报名

baoMing(list[position])

}

}

}

private fun baoMing(partInfo: PartInfo) {

val json = JSONObject()

json.put("uid", uid)

json.put("part\_id", partInfo.id)

HttpUtils.getInstance().postJsonWithHeader(ConfigNet.BAO\_MING\_URL, json.toString(),

ConfigNet.BAO\_MING\_CODE, token, handler)

}

private fun initLoad() {

val partTimeManager = supportFragmentManager

val ta = partTimeManager.beginTransaction()

val loadFragment = LoadFragment()

loadFragment.code = 3

ta.add(R.id.partTimeLoad, loadFragment)

ta.commitAllowingStateLoss()

}

private fun initClick() {

imageBack.setOnClickListener { finish() }

partTimeSend.setOnClickListener {

getUserInfo()

}

}

private fun getUserInfo() {

val json = JSONObject()

json.put("uid", Share().getUid(this))

HttpUtils.getInstance().postJsonWithHeader(Config.GET\_USER\_URL,

json.toString(), Config.GET\_USER, Share().getToken(this), handler)

showLoadDialog()

}

/\*\*

\* 到哪个界面发布兼职

\*/

fun toWhere(type: Int) {

Share().saveRealType(this, type)

if (Share().getUid(this) != 0) {

if (type == 2 || type == 4) {

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

intent.putExtra("id", "")

startActivity(intent)

} else if (type == 1) {

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

intent.putExtra("id", id)

startActivity(intent)

}

} else {

startActivity(Intent(this, LoginActivity::class.java))

showToastShort("请登录")

}

}

}

package com.wt.yc.oto.oto.yc.parttime

import android.Manifest

import android.content.Intent

import android.content.pm.PackageManager

import android.net.Uri

import android.os.Bundle

import android.os.Message

import android.support.v4.app.ActivityCompat

import android.view.View

import com.wt.yc.oto.oto.R

import com.wt.yc.oto.oto.activity.ProActivity

import com.wt.yc.oto.oto.info.PersonInfo

import com.wt.yc.oto.oto.net.ConfigNet

import com.xin.lv.yang.utils.utils.BitmapUtil

import com.xin.lv.yang.utils.utils.HttpUtils

import kotlinx.android.synthetic.main.item\_top.\*

import kotlinx.android.synthetic.main.yc\_part\_time\_details\_layout.\*

import org.json.JSONObject

class PartTimeDetailsActivity : ProActivity() {

override fun handler(msg: Message) {

val str = msg.obj as String

when (msg.what) {

ConfigNet.GET\_PART\_DETAILS\_CODE -> {

val json = JSONObject(str)

val code = json.optInt("code")

if (code == 200) {

val result = json.optString("data")

val partInfo = gson!!.fromJson<PartInfo>(result, PartInfo::class.java)

val jsonObject = json.optJSONObject("data")

val shopUser = jsonObject.optString("shop\_user")

if (shopUser != null && shopUser != "" && shopUser != "null") {

val personInfo: PersonInfo = gson!!.fromJson(shopUser, PersonInfo::class.java)

partInfo.shopUser = personInfo

}

showPartInfo(partInfo)

}

}

}

}

private fun showPartInfo(partInfo: PartInfo) {

phone = partInfo.mobile

partTimeName.text = partInfo.brief

tvPartTimeMoney.text = ("￥ " + partInfo.money + " 元")

val jieWay = when (partInfo.close) {

1 -> "月结"

2 -> "周结"

3 -> "日结"

else -> ""

}

tvPartTimeTime.text = jieWay

tvPartPersonNum.text = ("招聘人数: ${partInfo.num}")

val updateTime = partInfo.update\_time

if (updateTime.toInt() != 0) {

tvUpdate.text = (": ${BitmapUtil.longToTime(updateTime, "yyyy-MM-dd")}")

} else {

tvUpdate.text = (": ${BitmapUtil.longToTime(partInfo.create\_time, "yyyy-MM-dd")}")

}

tvPartBrowse.text = ""

tvPartApplyFor.text = ""

partDescribe.text = partInfo.detail

partWorkTime.text = partInfo.working\_time

val limitStr = partInfo.valid

if (limitStr == "1" || limitStr == "0") {

partWorkLimit.text = ""

} else {

partWorkLimit.text = ("截止到 ${BitmapUtil.longToTime(limitStr.toLong(), "yyyy-MM-dd")}")

}

partWorkAddress.text = partInfo.site

if (partInfo.shopUser != null) {

companyLinear.visibility = View.VISIBLE

companyName.text = partInfo.shopUser!!.s\_name

val shopType = partInfo.shopUser!!.shop\_type

when (shopType) {

1 -> companyType.text = "个人招聘"

2 -> companyType.text = "个体工商招聘"

3 -> companyType.text = "企业招聘"

}

val nature = partInfo.shopUser!!.nature

val ss = if (nature == "1") {

"私企"

} else {

"国企"

}

companyApprove1.visibility = View.VISIBLE

companyApprove2.visibility = View.VISIBLE

val gui = when (partInfo.shopUser!!.scale) {

"1" -> "50人以下"

"2" -> "50到100 人"

"3" -> "100人以上"

else -> {

"其他"

}

}

companyScale.text = ("公司类型: $ss\n规模: $gui\n行业: ${partInfo.shopUser!!.industry}")

} else {

companyLinear.visibility = View.GONE

}

}

/\*\*

\* onCreate

\*/

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.yc\_part\_time\_details\_layout)

image\_back.setOnClickListener { finish() }

val title = intent.getStringExtra("name")

val id = intent.getIntExtra("id", 0)

if (title != "") {

text\_top.text = title

} else {

text\_top.text = "详情"

}

getPartDetails(id)

initClick()

}

/\*\*

\* 电话号码

\*/

var phone = ""

private fun initClick() {

btPartPhone.setOnClickListener {

if (ActivityCompat.checkSelfPermission(this, Manifest.permission.CALL\_PHONE) == PackageManager.PERMISSION\_GRANTED) {

val intent = Intent(Intent.ACTION\_CALL)

intent.data = Uri.parse("tel:$phone")

intent.flags = Intent.FLAG\_ACTIVITY\_NEW\_TASK

startActivity(intent)

} else {

ActivityCompat.requestPermissions(this, arrayOf(Manifest.permission.CALL\_PHONE), 444)

}

}

}

/\*\*

\* 权限请求返回

\*/

override fun onRequestPermissionsResult(requestCode: Int, permissions: Array<out String>, grantResults: IntArray) {

super.onRequestPermissionsResult(requestCode, permissions, grantResults)

when (requestCode) {

444 -> {

if (grantResults[0] != PackageManager.PERMISSION\_GRANTED) {

showToastShort("请开启手机拨打电话权限")

}

}

}

}

private fun getPartDetails(id: Int) {

val json = JSONObject()

json.put("part\_id", id)

HttpUtils.getInstance().postJsonWithHeader(ConfigNet.GET\_PART\_DETAILS\_URL, json.toString(),

ConfigNet.GET\_PART\_DETAILS\_CODE, token, handler)

}

}

package com.wt.yc.oto.oto.yc.freight

import android.annotation.SuppressLint

import android.app.Activity

import android.content.Intent

import android.os.Bundle

import android.os.Message

import android.support.design.widget.TabLayout

import android.support.v4.app.Fragment

import android.support.v4.app.FragmentPagerAdapter

import android.support.v4.view.PagerAdapter

import android.view.View

import android.view.ViewGroup

import android.widget.ImageView

import android.widget.LinearLayout

import android.widget.PopupWindow

import android.widget.TextView

import com.amap.api.maps.AMapUtils

import com.amap.api.maps.model.LatLng

import com.amap.api.services.core.PoiItem

import com.google.gson.reflect.TypeToken

import com.wt.yc.oto.oto.R

import com.wt.yc.oto.oto.activity.ProActivity

import com.wt.yc.oto.oto.info.CarInfo

import com.wt.yc.oto.oto.info.PicText

import com.wt.yc.oto.oto.model.CityModel

import com.wt.yc.oto.oto.net.Config

import com.wt.yc.oto.oto.net.ConfigNet

import com.wt.yc.oto.oto.overlay.AMapUtil

import com.wt.yc.oto.oto.view.DataTimePop

import com.wt.yc.oto.oto.yc.activity.CityChooseActivity

import com.wt.yc.oto.oto.yc.mopao.MoPaoChooseAddressActivity

import com.xin.lv.yang.utils.utils.HttpUtils

import com.xin.lv.yang.utils.utils.ImageUtil

import kotlinx.android.synthetic.main.item\_top.\*

import kotlinx.android.synthetic.main.yc\_taxi\_layout.\*

import org.json.JSONObject

/\*\*

\* 货运的士

\*/

class FreightTaxiActivity : ProActivity() {

val ADDRESS\_CHOOSE = 3344

val START\_ADDRESS = 5566

val END\_ADDRESS = 7788

override fun handler(msg: Message) {

val str = msg.obj as String

when (msg.what) {

ConfigNet.GET\_CAR\_LIST\_CODE -> {

val json = JSONObject(str)

val code = json.optInt("code")

if (code == 200) {

val result = json.optString("data")

if (result != null && result != "") {

viewPagerContentList = gson!!.fromJson(result, object : TypeToken<ArrayList<CarInfo>>() {}.type)

initTabLayout(viewPagerContentList)

initViewTaxiPager(viewPagerContentList)

}

}

}

}

}

var id: Int = 0

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.yc\_taxi\_layout)

text\_top.text = ""

id = intent.getIntExtra("id", 0)

initClick()

val picText = PicText()

picText.title = "测试"

getCarFromUrl()

}

/\*\*

\* 选中的位置信息

\*/

var indexShowViewPager = 0

var viewPagerContentList: ArrayList<CarInfo> = arrayListOf()

val taxiViewList: ArrayList<View> = arrayListOf()

private fun initViewTaxiPager(contentArr: ArrayList<CarInfo>) {

taxiViewPager.removeAllViews()

taxiViewList.clear()

indexShowViewPager = 0

for (temp in contentArr) {

val view: View = layoutInflater.inflate(R.layout.taxi\_detail, null)

val taxiStartMoney = view.findViewById(R.id.taxiStartMoney) as TextView

val taxiPic = view.findViewById(R.id.taxiPic) as ImageView

val tvIndexMoney = view.findViewById(R.id.tvIndexMoney) as TextView

val tvTvChang = view.findViewById(R.id.tvTvChang) as TextView

ImageUtil.getInstance().loadImageNoTransformation(this, taxiPic, R.drawable.banner, Config.IP + temp.icon)

tvIndexMoney.text = temp.price

taxiStartMoney.text = temp.start\_price

val len = temp.start\_length

val wid = temp.width

val hei = temp.height

if (len != "") {

tvTvChang.text = ("$len\*$wid\*$hei")

} else {

tvTvChang.text = ""

}

taxiViewList.add(view)

}

taxiViewPager.currentItem = indexShowViewPager

taxiViewPager.adapter = object : PagerAdapter() {

override fun isViewFromObject(view: View, `object`: Any): Boolean {

return view == `object`

}

override fun getCount(): Int {

return taxiViewList.size

}

override fun instantiateItem(container: ViewGroup, position: Int): Any {

val view = taxiViewList[position]

container.addView(view)

return view

}

override fun destroyItem(container: ViewGroup, position: Int, `object`: Any) {

container.removeView(taxiViewList[position])

}

}

}

private fun initTabLayout(arr: ArrayList<CarInfo>) {

for (temp in arr) {

taxiTabLayout.addTab(taxiTabLayout.newTab().setText(temp.name))

}

taxiTabLayout.addOnTabSelectedListener(object : TabLayout.OnTabSelectedListener {

override fun onTabReselected(tab: TabLayout.Tab?) {

}

override fun onTabUnselected(tab: TabLayout.Tab?) {

}

override fun onTabSelected(tab: TabLayout.Tab?) {

when (tab!!.position) {

0 -> indexShowViewPager = 0

1 -> indexShowViewPager = 1

}

taxiViewPager.currentItem = indexShowViewPager

}

})

}

private fun getCarFromUrl() {

val json = JSONObject()

HttpUtils.getInstance().postJsonWithHeader(ConfigNet.GET\_CAR\_LIST\_FROM\_URL, json.toString(), ConfigNet.GET\_CAR\_LIST\_CODE, token, handler)

}

private fun initClick() {

image\_back.setOnClickListener {

finish()

}

taxiLeft.setOnClickListener {

if (indexShowViewPager > 0) {

indexShowViewPager--

taxiViewPager.currentItem = indexShowViewPager

showPrice()

}

}

taxiRight.setOnClickListener {

if (indexShowViewPager < taxiViewList.size - 1) {

indexShowViewPager++

taxiViewPager.currentItem = indexShowViewPager

showPrice()

}

}

tvUpdateTime.setOnClickListener {

showChooseTime()

}

tvStartAddress.setOnClickListener {

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

startActivityForResult(intent, START\_ADDRESS)

}

tvTaxiChooseCity.setOnClickListener {

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

intent.putExtra("city", tvTaxiChooseCity.text.toString())

intent.putExtra("province", "")

intent.putExtra("type", "3")

startActivityForResult(intent, ADDRESS\_CHOOSE)

}

moneyRelative.setOnClickListener {

if (money != "") {

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

intent.putExtra("car", viewPagerContentList[taxiViewPager.currentItem])

intent.putExtra("startPoiItem", startPoiItem)

intent.putExtra("endPoiItem", endPoiItem)

intent.putExtra("money", money)

startActivity(intent)

}

}

btNext.setOnClickListener {

val timeStr = tvTaxiTime.text.toString()

val startAddress = tvStartAddress.text.toString()

val endAddress = etEndAddress.text.toString()

when {

timeStr == "" -> showToastShort("请选择时间")

startAddress == "" -> showToastShort("请选择开始地址")

endAddress == "" -> showToastShort("请选择或者输入结束地址")

else -> {

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

intent.putExtra("time", timeStr)

intent.putExtra("startAddress", startPoiItem)

intent.putExtra("endAddress", endPoiItem)

intent.putExtra("car", viewPagerContentList[taxiViewPager.currentItem])

intent.putExtra("money", money)

intent.putExtra("textId", id)

startActivity(intent)

}

}

}

endAddressCarLinear.setOnClickListener {

if (startPoiItem != null) {

startActivityForResult(Intent(this, ChooseFreightAddressActivity::class.java), END\_ADDRESS)

} else {

showToastShort("请选择始发地址")

}

}

}

/\*\*

\* 选择时间

\*/

private fun showChooseTime() {

val chooseTime = DataTimePop(this, getW(), LinearLayout.LayoutParams.WRAP\_CONTENT)

setAlpha(0.6f)

chooseTime.show(object : DataTimePop.OnChooseListener {

override fun onChoose(data: String, time: String) {

tvTaxiTime.text = ("$data $time").toString()

chooseTime.dismiss()

setAlpha(1f)

}

override fun onCancle() {

chooseTime.dismiss()

setAlpha(1f)

}

})

chooseTime.setOnDismissListener { setAlpha(1f) }

}

var startPoiItem: PoiItem? = null

var endPoiItem: PoiItem? = null

var money: String = ""

override fun onActivityResult(requestCode: Int, resultCode: Int, data: Intent?) {

super.onActivityResult(requestCode, resultCode, data)

if (resultCode == Activity.RESULT\_OK) {

when (requestCode) {

ADDRESS\_CHOOSE -> {

val cityMode = data!!.getParcelableExtra<CityModel>(Intent.EXTRA\_RETURN\_RESULT)

tvTaxiChooseCity.text = cityMode.province + cityMode.city + cityMode.street

}

START\_ADDRESS -> {

startPoiItem = data!!.getParcelableExtra("result")

tvStartAddress.text = (startPoiItem!!.provinceName + startPoiItem!!.cityName + startPoiItem!!.title).toString()

}

END\_ADDRESS -> {

endPoiItem = data!!.getParcelableExtra("result")

etEndAddress.text = (endPoiItem!!.provinceName + endPoiItem!!.cityName + endPoiItem!!.title).toString()

showPrice()

}

}

}

}

@SuppressLint("SetTextI18n")

private fun showPrice() {

if (startPoiItem == null || endPoiItem == null) {

showToastShort("请选择开始和结束地址计算价格")

return

}

val distence: Float = AMapUtils.calculateLineDistance(

LatLng(startPoiItem!!.latLonPoint.latitude, startPoiItem!!.latLonPoint.longitude),

LatLng(endPoiItem!!.latLonPoint.latitude, endPoiItem!!.latLonPoint.longitude)) / 1000

val carInfo = viewPagerContentList[taxiViewPager.currentItem]

val num = carInfo.num

money = if (distence <= num) {

carInfo.start\_price

} else {

val yuNum = distence - num

val jiMoney = carInfo.start\_price.toFloat() + carInfo.price.toFloat() \* yuNum

String.format("%.2f", jiMoney)

}

tvTaxiMoney.text = "约 $money 元"

}

}

package com.wt.yc.oto.oto.yc.freight

import android.app.Activity

import android.content.Intent

import android.os.Bundle

import android.os.Message

import com.wt.yc.oto.oto.R

import com.wt.yc.oto.oto.activity.ProActivity

import com.wt.yc.oto.oto.info.CarInfo

import com.wt.yc.oto.oto.net.ConfigNet

import com.xin.lv.yang.utils.utils.HttpUtils

import kotlinx.android.synthetic.main.item\_top.\*

import kotlinx.android.synthetic.main.yc\_driver\_add\_layout.\*

import org.json.JSONObject

class AddDriverActivity : ProActivity() {

override fun handler(msg: Message) {

val str = msg.obj as String

when (msg.what) {

ConfigNet.SEND\_ADD\_CAR\_CODE -> {

val json = JSONObject(str)

val code = json.optInt("code")

if (code == 200) {

startActivity(Intent(this@AddDriverActivity, AddDriverSuccessActivity::class.java))

finish()

}

showToastShort(json.optString("msg"))

}

}

}

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.yc\_driver\_add\_layout)

text\_top.text = ""

image\_back.setOnClickListener {

finish()

}

btAddDriver.setOnClickListener {

tiAddCar()

}

carLinearLayout.setOnClickListener {

startActivityForResult(Intent(this, ChooseCarTypeActivity::class.java), 3456)

}

}

override fun onActivityResult(requestCode: Int, resultCode: Int, data: Intent?) {

super.onActivityResult(requestCode, resultCode, data)

if (resultCode == Activity.RESULT\_OK) {

when (requestCode) {

3456 -> {

pid = data!!.getIntExtra("typeCode", 0)

carInfoResult = data.getParcelableExtra("result") as CarInfo

tvCarTypeDetails.text = (carInfoResult!!.name + " " + when (pid) {

1 -> "不可拆"

2 -> "可拆一排座"

3 -> "可拆两排座"

else -> {

""

}

}).toString()

}

}

}

}

var pid = 0

var carInfoResult: CarInfo? = null

private fun tiAddCar() {

val name = etRealName.text.toString()

val idCard = etIdCard.text.toString()

val phone = etAddPhone.text.toString()

val carNumber = etCarNumber.text.toString()

when {

else -> {

val json = JSONObject()

json.put("uid", uid)

json.put("pid", 55)

json.put("id\_number", idCard)

json.put("name", name)

json.put("car\_number", carNumber)

json.put("car\_id", carInfoResult!!.id)

json.put("mobile", phone)

json.put("car\_type", pid)

HttpUtils.getInstance().postJsonWithHeader(ConfigNet.SEND\_ADD\_CAR\_URL, json.toString(),

ConfigNet.SEND\_ADD\_CAR\_CODE, token, handler)

}

}

}

}