DailyLife Android应用

开发工具：Android studio  
开发语言：Java

开发框架：MVC

后端数据提供：Bmob云数据库

Bmob云数据库：

Bmob 提供了实时数据与文件存储功能，轻松实现应用“云与端”的数据连通。数据存储除了常规应用文本信息的存储，还可以存储图片、视频、音频、地理位置等信息。

使用 Bmob 几乎能免去所有服务器端编码工作量。

集成教程文档：<http://doc.bmobapp.com/data/android/index.html>

BaseActivity/BaseFragment：每一个Activity和Fragment都是很多模板代码的，为了减少模板代码，封装进Base类其实是一种比较方便且可行的选择.

当创建新的Activity或者Fragment的时候需要想想有没有逻辑可以复用，就去找Base类，或许写Base类的人不同，发现一个项目中可能会存在多个Base类，甚至Base类仍然有多个Base子类实现不同逻辑。这个时候就需要去查看分析每个Base类分别实现了什么功能，决定继承哪个。

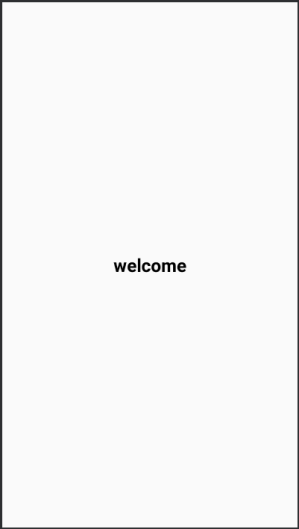
如果一个项目中只有一个Base类的话，仍需要看看Base类实现了什么逻辑，没有实现什么逻辑，防止重复写样板代码。

当出现Base类实现了的，而自己本身并不想需要，例如不想监听广播或者不想用ViewModel，对于不想监听广播的情况就要特殊做适配，例如往Base类加标志位。对于不想用ViewModel但是由于泛型限制，还是只能传进去，不然没法继承。

当发现自己集成Base类出BUG了，就要考虑改子类还是改Base类，由于大量的类都集成了Base类，显然改Base类比较麻烦，于是改自己比较方便。

如果一个Activity中展示了多个Fragment，可能会有业务逻辑的重复，其实只需要一个就好了。

欢迎页面——SplashActivity



作为App的启动页面，并设置Activity为全屏

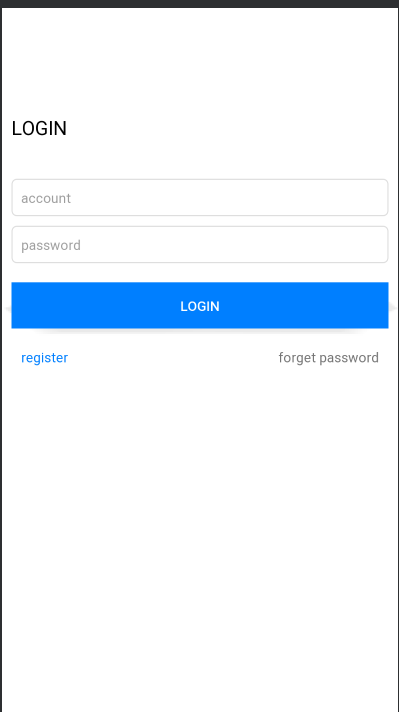
setContentView(R.layout.xxx)之前调用：

requestWindowFeature(Window.FEATURE\_NO\_TITLE);  
getWindow().setFlags(WindowManager.LayoutParams.FLAG\_FULLSCREEN, WindowManager.LayoutParams.FLAG\_FULLSCREEN);  
if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.KITKAT) {  
 getWindow().addFlags(WindowManager.LayoutParams.FLAG\_TRANSLUCENT\_STATUS);  
 getWindow().addFlags(WindowManager.LayoutParams.FLAG\_TRANSLUCENT\_NAVIGATION);  
}

并使用handler进行延时1秒跳转至登录页面LoginActivity

new Handler().postDelayed(new Runnable() {  
 @Override  
 public void run() {  
 startActivity(new Intent(SplashActivity.this,LoginActivity.class));  
 }  
},1000);

**LoginActiv登录界面：**



用户可通过输入手机号和密码登录，手机号采用11位校验，并提供注册和忘记密码重设功能。

监听用户点击登录按钮事件：

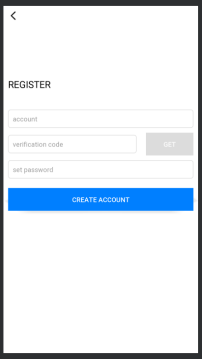
String phone = accountEdit.getText().toString();//获取用户输入的号  
String pwd = pwdEdit.getText().toString();//获取用户输入的密码  
  
if (TextUtils.isEmpty(phone)) {//判断输入的手机号是否为空  
 showToast("Please input phone");//提示用户手机号不能为空  
 return;  
}  
  
if (TextUtils.isEmpty(pwd)) {//判断输入的密码是否为空  
 showToast("Please input password");  
 return;  
}

//查询方法，先查询该手机号是否注册  
BmobQuery<User> query = new BmobQuery<>();  
query.addWhereEqualTo("phone", phone);//设置用户输入的手机  
query.findObjects(new FindListener<User>() {  
 @Override  
 public void done(List<User> list, BmobException e) {

//查询方法回调事件  
 if (e == null) {  
 if (list == null || list.size() == 0){//账号不存在即手机号未注册  
 Toast.makeText(mActivity, "The account does not exist", Toast.LENGTH\_SHORT).show();  
 }else {  
 User user = list.get(0);//账号存在 则判断密码是否输入正确  
 if (user != null){  
 if (pwd.equals(user.getPwd())){  
 Toast.makeText(mActivity, "Login success", Toast.LENGTH\_SHORT).show();  
 MyApplication.getInstance().setLoginUser(user);  
 SharedPreferenceUtil.putString(mContext,Constants.PHONE,phone);  
 startActivity(new Intent(mActivity,MainActivity.class));//登录成功跳转至主页  
 finish();  
 }else {

//提示密码输入错误  
 Toast.makeText(mActivity, "Password entered incorrectly", Toast.LENGTH\_SHORT).show();  
 }  
 }  
 }  
 } else {  
 Toast.makeText(mActivity, mActivity.getResources().getString(R.string.unknown\_error), Toast.LENGTH\_SHORT).show();  
 }  
 }  
});

**RegisterActivity注册页面:**



注册页面和重设密码页面公用一个页面，通过变量type判断，type等于0是注册页面，type等于1为重设密码。

通过用户输入的手机号获取短信验证码校验。

点击获取验证码按钮：

BmobSMS.requestSMSCode(phone, "DailyLife", new QueryListener<Integer>() {  
 @Override  
 public void done(Integer smsId, BmobException e) {  
 if (e == null) {

//获取成功，并开启倒计时功能  
 countDownTimer.start();  
 getCodeBtn.setBackgroundColor(mActivity.getResources().getColor(R.color.dcdcdc));  
 getCodeBtn.setEnabled(false);//倒计时中 获取验证码按钮不可用  
 if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.JELLY\_BEAN) {  
 getCodeBtn.setBackground(null);  
 }  
 Toast.makeText(mActivity, "The verification code is successfully obtained", Toast.LENGTH\_SHORT).show();  
 Log.e(TAG, "done: " + "ID：" + smsId + "\n");  
 } else {  
 Toast.makeText(mActivity, "Failed to obtain the verification code.", Toast.LENGTH\_SHORT).show();  
 Log.e(TAG, "done: " + "Failed：" + e.getErrorCode() + "-" + e.getMessage() + "\n");  
 }  
 }  
});

监听事件：

if (TextUtils.isEmpty(phone)) {  
 showToast("Please input phone");  
 return;  
}  
if (TextUtils.isEmpty(code)) {  
 showToast("Please input code");  
 return;  
}  
if (TextUtils.isEmpty(password)) {  
 showToast("Please input password");  
 return;  
}

//先查询该手机号是否已经注册  
BmobQuery<User> query = new BmobQuery<>();  
query.addWhereEqualTo("phone", phone);  
query.findObjects(new FindListener<User>() {  
 @Override  
 public void done(List<User> list, BmobException e) {  
 if (e == null) {  
 if (type == 1) {//如果为重设密码状态  
 if (list == null || list.size() == 0) { //重设密码时 账号不存在  
 Toast.makeText(mActivity, "The account does not exist", Toast.LENGTH\_SHORT).show();  
 } else {  
 User user = list.get(0);//重设密码 账号存在  
 if (user != null) {  
 user.setPwd(password);//重新设置密码

//调用更新方法 进行更新密码操作  
 user.update(new UpdateListener() {  
 @Override  
 public void done(BmobException e) {  
 if (e == null) {  
 Toast.makeText(mActivity, "Succeeded in resetting the password.", Toast.LENGTH\_SHORT).show();  
 SharedPreferenceUtil.putString(mActivity, Constants.PHONE, phone);  
 finish();  
 } else {  
 Toast.makeText(mActivity, mActivity.getResources().getString(R.string.unknown\_error), Toast.LENGTH\_SHORT).show();  
 }  
 }  
 });  
 }  
 }  
 } else {  
 if (list == null || list.size() == 0) {  
 CheckVerificationCode(phone, password, code);  
 } else {  
 Toast.makeText(mActivity, "The phone number has been registered", Toast.LENGTH\_SHORT).show();  
 }  
 }  
 } else {  
 Toast.makeText(mActivity, mActivity.getResources().getString(R.string.unknown\_error), Toast.LENGTH\_SHORT).show();  
 }  
 }  
});

**验证短信验证码是否正确**

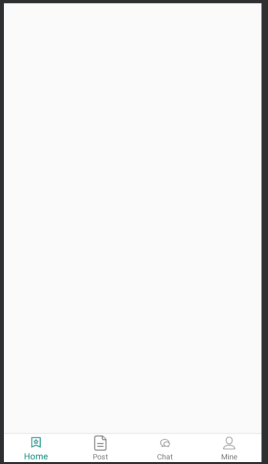
private void CheckVerificationCode(String phone, String password, String code,User user) {

//  
 BmobSMS.verifySmsCode(phone, code, new UpdateListener() {  
 @Override  
 public void done(BmobException e) {  
 if (e == null) {  
 if (type == 1){   
 if (user != null) {  
 user.setPwd(password);//重新设置密码  
 //调用更新方法  
 user.update(new UpdateListener() {  
 @Override  
 public void done(BmobException e) {  
 if (e == null) {  
 Toast.makeText(mActivity, "Succeeded in resetting the password.", Toast.LENGTH\_SHORT).show();  
 SharedPreferenceUtil.putString(mActivity, Constants.PHONE, phone);  
 finish();  
 } else {  
 Toast.makeText(mActivity, mActivity.getResources().getString(R.string.unknown\_error), Toast.LENGTH\_SHORT).show();  
 }  
 }  
 });  
 }  
 }else {

//创建新用户  
 User user = new User();  
 user.setPhone(phone);  
 user.setPwd(password);

//调用保存方法  
 user.save(new SaveListener<String>() {  
 @Override  
 public void done(String s, BmobException e) {  
 if (e == null) {  
 Toast.makeText(mActivity, "Successful registration", Toast.LENGTH\_SHORT).show();  
 SharedPreferenceUtil.putString(mActivity, Constants.PHONE, phone);  
 finish();  
 } else {  
 Toast.makeText(mActivity, "Registration failure", Toast.LENGTH\_SHORT).show();  
 }  
 }  
 });  
 }  
  
   
 } else {  
 Toast.makeText(mActivity, "Verification code error", Toast.LENGTH\_SHORT).show();  
 //mTvInfo.append("Verification code error" + e.getErrorCode() + "-" + e.getMessage() + "\n");  
 }  
 }  
 });  
}

**MainActivity 主页面：**

  
主页面采用viewpager+bottomnavigation实现底部导航栏和左右页面联动。

MainActivity.java代码：

public class MainActivity extends BaseActivity {  
  
 @BindView(R.id.view\_pager)  
 ViewPager viewPager;  
 @BindView(R.id.bottom\_navigation)  
 BottomNavigationView bottomNavigation;  
  
 private HomeFragment homeFragment;  
 private PostFragment postFragment;  
 private ChatFragment chatFragment;  
 private MineFragment mineFragment;  
  
 private List<Fragment> mFragments;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 //setContentView(R.layout.activity\_main);  
 }  
  
 @Override  
 protected int setLayout() {  
 return R.layout.activity\_main;  
 }  
  
 @Override  
 protected void initView() {  
 if (isDark) {  
 ImmersionBar.with(mActivity).statusBarColor(R.color.white).statusBarDarkFont(false).init();  
 } else {  
 ImmersionBar.with(mActivity).statusBarColor(R.color.white).statusBarDarkFont(true).init();  
 }

//实例化四个fragment  
 homeFragment = new HomeFragment();  
 postFragment = new PostFragment();  
 chatFragment = new ChatFragment();  
 mineFragment = new MineFragment();  
 mFragments = new ArrayList<>();//初始化fragment容器

//将四个fragment添加到容器中  
 mFragments.add(homeFragment);  
 mFragments.add(postFragment);  
 mFragments.add(chatFragment);  
 mFragments.add(mineFragment);  
 //viewpager绑定适配器  
 viewPager.setAdapter(new FragmentsPagerAdapter(getSupportFragmentManager()));  
 // navigationView点击事件  
 bottomNavigation.setOnNavigationItemSelectedListener(menuItem -> {  
 // menu文件夹中bottom\_nav\_menu.xml里加的android:orderInCategory属性就是下面item.getOrder()取的值  
 viewPager.setCurrentItem(menuItem.getOrder());//与viewpager绑定  
 return true;  
  
 });  
  
 /\*\*  
 \* 监听ViewPager页面变化事件  
 \* \*/  
 viewPager.addOnPageChangeListener(new ViewPager.OnPageChangeListener() {  
 @Override  
 public void onPageScrolled(int position, float positionOffset, int positionOffsetPixels) {  
  
 }  
  
 @Override  
 public void onPageSelected(int position) {  
 bottomNavigation.getMenu().getItem(position).setChecked(true);//与底部导航绑定  
 }  
  
 @Override  
 public void onPageScrollStateChanged(int state) {  
  
 }  
 });  
 }  
  
 public class FragmentsPagerAdapter extends FragmentPagerAdapter{  
 private FragmentManager mFragmentManager;  
 public FragmentsPagerAdapter(FragmentManager fm) {  
 super(fm);  
 mFragmentManager = fm;  
 }  
 @Override  
 public Fragment getItem(int position) {  
 return mFragments.get(position);  
 }  
 @Override  
 public int getCount() {  
 return mFragments.size();  
 }  
 }  
  
  
}