/\*\*  
 \* 移动开发学习系统启动页面  
 \*/

public class SplashActivity extends Activity {

@Override

protected void onCreate(Bundle savedInstanceState) {

//淡入淡出效果

overridePendingTransition(android.R.anim.*fade\_in*, android.R.anim.*fade\_out*);

super.onCreate(savedInstanceState);

setContentView(R.layout.*activity\_splash*);

RelativeLayout rl\_splash = findViewById(R.id.*rl\_splash*);

//透明动画

AlphaAnimation alphaAnimation = new AlphaAnimation(0.3f, 1.0f);

alphaAnimation.setDuration(1000);

alphaAnimation.setAnimationListener(new Animation.AnimationListener() {

@Override

public void onAnimationStart(Animation animation) {

}

@Override

public void onAnimationEnd(Animation animation) {

startActivity(new Intent(SplashActivity.this, MainActivity.class));

finish();

overridePendingTransition(android.R.anim.*fade\_in*,android.R.anim.*fade\_out*);

}

@Override

public void onAnimationRepeat(Animation animation) {

}

});

rl\_splash.startAnimation(alphaAnimation);

}

}

/\*\*

\*移动开发学习系统布局文件

\*/

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:id="@+id/rl\_splash"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical">

<TextView

android:id="@+id/tv\_splash"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignParentBottom="true"

android:layout\_centerHorizontal="true"

android:layout\_marginBottom="30dp"

android:textSize="18sp"

android:textColor="@color/black"

android:text=" @string/welcome\_word

"/>

</RelativeLayout>

public abstract class AbstractSimpleActivity extends SupportActivity {  
  
 private Unbinder unbinder;  
 private static final String *TAG* = "AbstractSimpleActivity";  
  
 @Override  
 protected void onCreate(@Nullable Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(getLayoutId());  
 //沉浸式状态栏  
 ImmersionBar.*with*(this)  
 .statusBarView(findViewById(R.id.*status\_bar*))  
 .statusBarDarkFont(true)  
 .keyboardEnable(true)  
 .init();  
 unbinder = ButterKnife.*bind*(this);  
 Log.*d*(*TAG*, "onCreate: ");  
 onViewCreate();  
 initToolbar();  
 initView();  
 initEventAndData();  
 }  
  
 protected abstract int getLayoutId();  
 protected abstract void onViewCreate();  
 protected abstract void initToolbar();  
 protected abstract void initView();  
 protected abstract void initEventAndData();  
  
 @Override  
 protected void onDestroy() {  
 super.onDestroy();  
 ImmersionBar.*with*(this).destroy();  
 if (unbinder != null && unbinder != Unbinder.*EMPTY*) {  
 unbinder.unbind();  
 unbinder = null;  
 }  
 }

}

/\*\*  
 \* 每个Activity都需要继承此类  
 \* @param<T>  
 \*/

public abstract class BaseActivity<T extends BasePresenter> extends AbstractSimpleActivity  
 implements HasSupportFragmentInjector, BaseView {  
 private static final String *TAG* = "BaseActivity";  
  
 @Inject  
 protected T mPresenter;  
 @Inject  
 DispatchingAndroidInjector<Fragment> dispatchingAndroidInjector;  
  
 @Override  
 protected void onCreate(@Nullable Bundle savedInstanceState) {  
 AndroidInjection.*inject*(this); //实现 Activity 的注入  
 super.onCreate(savedInstanceState);  
 Log.*d*(*TAG*, "onCreate: ");  
 }  
 /\*\*  
 \* 父类onCreate调用onViewCreate  
 \* 先执行父类onCreate，在执行子类实现的 onViewCreate  
 \* 最后执行子类的 onCreate  
 \*/@Override  
 protected void onViewCreate() {  
 Log.*d*(*TAG*, "onViewCreate: onViewCreate");  
 if (mPresenter != null) {  
 mPresenter.attachView(this);  
 }  
 }  
  
 @Override  
 public AndroidInjector<Fragment> supportFragmentInjector() {  
 return dispatchingAndroidInjector;  
 }  
  
 @Override  
 public void showErrorMessage(String message) {  
  
 }  
  
 @Override  
 public void showLoading() {  
  
 }  
  
 @Override  
 public void hideLoading() {  
  
 }  
  
 @Override  
 public void showError() {  
  
 }  
  
 @Override  
 public void showNoNetwork() {  
  
 }  
  
 @Override  
 public void showEmpty() {  
  
 }  
  
 @Override  
 public void showContent() {  
  
 }  
  
 @Override  
 public void handleLoginSuccess() {  
  
 }  
  
 @Override  
 public void handleLogoutSuccess() {  
  
 }  
}

/\*\*  
 \* View 基类接口  
 \*/public interface BaseView {  
 void showErrorMessage(String message);  
 void showLoading();  
 void hideLoading();  
 void showError();  
  
 void showNoNetwork();  
 void showEmpty();  
 void showContent();  
 void handleLoginSuccess();  
  
 void handleLogoutSuccess();  
}

/\*\*  
 \* Presenter 基类接口  
 \* @param <T>  
 \*/public interface BasePresenter<T extends BaseView> {  
  
 void attachView(T view);  
 void detachView();  
 void reload();  
 void registerEventBus();  
 void unregisterEventBus();  
  
 void setLoginStatus(boolean loginStatus);  
 boolean getLoginStatus();  
 String getLoginAccount();  
 void setLoginAccount(String account);  
  
}

/\*\*  
 \* 主页面  
 \*/public class MainActivity extends BaseActivity<MainPresenter> implements MainContract.View {  
  
 private static final String *TAG* = "MainActivity";  
 @BindView(R.id.*drawer\_layout*)  
 DrawerLayout drawer\_layout;  
 @BindView(R.id.*toolbar*)  
 Toolbar toolbar;  
 @BindView(R.id.*toolbar\_title*)  
 TextView toolbar\_title;  
 @BindView(R.id.*navigation\_view*)  
 NavigationView navigation\_view;  
 @BindView(R.id.*bottom\_navigation\_view*)  
 BottomNavigationView bottom\_navigation\_view;  
  
 private int currentFgIndex = 0;  
 private int lastFgIndex = -1;  
  
 private HomePageFragment homePageFragment;  
 private NavigationFragment navigationFragment;  
 private KnowledgeFragment knowledgeFragment;  
 private PublicNumFragment publicNumFragment;  
 private ProjectFragment projectFragment;  
 private TextView tv\_welcome;  
 private TextView tv\_login\_account;  
 private long clickTime;  
  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 Log.*d*(*TAG*, "onCreate: ");  
 }  
  
 protected int getLayoutId() {  
 return R.layout.*activity\_main*;  
 }  
  
 /\*\*  
 \* 初始化 Toolbar  
 \*/@Override  
 protected void initToolbar() {  
 setSupportActionBar(toolbar);  
 ActionBar actionBar = getSupportActionBar();  
 if (actionBar != null) {  
 actionBar.setDisplayShowTitleEnabled(false);  
 toolbar\_title.setText(R.string.*home\_page*);  
 }

getWindow().setNavigationBarColor(getResources().getColor(R.colo r.*navigation\_bar*));

}  
  
 @Override  
 protected void initView() {  
 initDrawerLayout();  
 showFragment(currentFgIndex);  
 initNavigationView();  
 initBottomNavigationView();  
  
 }  
  
 /\*\*  
 \* 初始化底部导航栏  
 \*/private void initBottomNavigationView() {  
 bottom\_navigation\_view.setOnNavigationItemSelectedListener(new

BottomNavigationView.OnNavigationItemSelectedListener() {  
 @Override  
 public boolean onNavigationItemSelected(@NonNull MenuItem

menuItem) {  
 switch (menuItem.getItemId()) {  
 case R.id.*tab\_home\_pager*:  
 showFragment(Constants.*FRAGMENT\_HOME\_PAGE*);  
 break;  
 case R.id.*tab\_knowledge*:  
 showFragment(Constants.*FRAGMENT\_KNOWLEDGE*);  
 break;  
 case R.id.*tab\_navigation*:  
 showFragment(Constants.*FRAGMENT\_NAIVIGATION*);  
 break;  
 case R.id.*tab\_project*:  
 showFragment(Constants.*FRAGMENT\_PRPJECT*);  
 break;  
 case R.id.*tab\_public\_num*:  
 showFragment(Constants.*FRAGMENT\_PUBLIC\_NUM*);  
 break;  
 }  
 return true;  
 }  
 });  
 }  
  
 private void initNavigationView() {  
 navigation\_view.setNavigationItemSelectedListener(new

NavigationView.OnNavigationItemSelectedListener() {  
 @Override  
 public boolean onNavigationItemSelected(@NonNull MenuItem

menuItem) {  
 switch (menuItem.getItemId()) {  
 case R.id.*item\_login*:  
 startLoginActivity();  
 break;  
 case R.id.*item\_collect*:  
 if (mPresenter.getLoginStatus()) {   
 Intent intent = new

Intent(MainActivity.this, ItemActivity.class);  
 intent.putExtra("Type",

Constants.*TYPE\_COLLECT*);  
 startActivity(intent);  
 } else {  
 startLoginActivity(); //未登录跳转登录页  
 Toast.*makeText*(MainActivity.this,

getString(R.string.*please\_login*),  
 Toast.*LENGTH\_SHORT*).show();  
 }  
 break;  
 case R.id.*item\_about*:  
 Intent intentAbout = new

Intent(MainActivity.this, ItemActivity.class);  
 intentAbout.putExtra("Type",

Constants.*TYPE\_ABOUT*);  
 startActivity(intentAbout);  
 break;  
 case R.id.*item\_logout*:  
 mPresenter.logout();  
 break;  
 }  
 return true;  
 }  
 });  
 tv\_welcome = navigation\_view.getHeaderView(0)

.findViewById(R.id.*tv\_welcome*);  
tv\_login\_account = navigation\_view.getHeaderView(0)

.findViewById(R.id.*tv\_login\_account*);

if (mPresenter.getLoginStatus()) {  
 tv\_welcome.setVisibility(View.*VISIBLE*);  
 tv\_login\_account.setText(mPresenter.getLoginAccount());

navigation\_view.getMenu().findItem(R.id.*item\_login*)

.setVisible(false); navigation\_view.getMenu().findItem(R.id.*item\_logout*)

.setVisible(true);  
} else {  
 tv\_welcome.setVisibility(View.*INVISIBLE*);  
 tv\_login\_account.setText("");

navigation\_view.getMenu().findItem(R.id.*item\_login*)

.setVisible(true); navigation\_view.getMenu().findItem(R.id.*item\_logout*)

.setVisible(false);

}

}  
  
 @Override  
 public boolean onCreateOptionsMenu(Menu menu) {  
 getMenuInflater().inflate(R.menu.*toolbar\_menu*, menu);  
 return super.onCreateOptionsMenu(menu);  
 }  
  
 /\*\*  
 \* Toolbar 两个按钮  
 \* @param item  
 \* @return\*/@Override  
 public boolean onOptionsItemSelected(MenuItem item) {  
 Intent intent;  
 switch (item.getItemId()) {  
 case R.id.*action\_usage*:  
 intent=new Intent(MainActivity.this, ItemActivity.class);  
 intent.putExtra("Type", Constants.*TYPE\_WEBSITE*);  
 startActivity(intent);  
 break;  
 case R.id.*action\_search*:  
 intent = new Intent(MainActivity.this, SearchActivity.class);  
 startActivity(intent);  
 break;  
 default:  
 break;  
 }  
 return true;  
 }  
  
 */\*\*  
 \* 展示对应 Fragment  
 \** ***@param*** *index  
 \*/* private void showFragment(int index) {  
 currentFgIndex = index;  
 FragmentTransaction transaction = getSupportFragmentManager().beginTransaction();  
 hideFragment(transaction);  
 lastFgIndex = index;  
 switch (index) {  
 case Constants.*FRAGMENT\_HOME\_PAGE*:  
 toolbar\_title.setText(R.string.*home\_page*);  
 if (homePageFragment == null) {  
 homePageFragment = HomePageFragment.*newInstance*();  
 transaction.add(R.id.*fragment\_root*, this.homePageFragment);  
 }  
 transaction.show(this.homePageFragment);  
 break;  
 case Constants.*FRAGMENT\_PUBLIC\_NUM*:  
 toolbar\_title.setText(R.string.*public\_num*);  
 if (publicNumFragment == null) {  
 publicNumFragment = PublicNumFragment.*newInstance*();  
 transaction.add(R.id.*fragment\_root*, this.publicNumFragment);  
 }  
 transaction.show(this.publicNumFragment);  
 break;  
 case Constants.*FRAGMENT\_KNOWLEDGE*:  
 toolbar\_title.setText(R.string.*knowledge*);  
 if (knowledgeFragment == null) {  
 knowledgeFragment = KnowledgeFragment.*newInstance*();  
 transaction.add(R.id.*fragment\_root*, this.knowledgeFragment);  
 }  
 transaction.show(this.knowledgeFragment);  
 break;  
 case Constants.*FRAGMENT\_PRPJECT*:  
 toolbar\_title.setText(R.string.*project*);  
 if (projectFragment == null) {  
 projectFragment = ProjectFragment.*newInstance*();  
 transaction.add(R.id.*fragment\_root*, this.projectFragment);  
 }  
 transaction.show(this.projectFragment);  
 break;  
 case Constants.*FRAGMENT\_NAIVIGATION*:  
 toolbar\_title.setText(R.string.*navigation*);  
 if (navigationFragment == null) {  
 navigationFragment = NavigationFragment.*newInstance*();  
 transaction.add(R.id.*fragment\_root*, this.navigationFragment);  
 }  
 transaction.show(this.navigationFragment);  
 break;  
 default:  
 break;  
 }  
 transaction.commit();  
 }  
  
 private void hideFragment(FragmentTransaction transaction) {  
 switch (lastFgIndex) {  
 case Constants.*FRAGMENT\_HOME\_PAGE*:  
 if (homePageFragment != null) {  
 transaction.hide(homePageFragment);  
 }  
 break;  
 case Constants.*FRAGMENT\_NAIVIGATION*:  
 if (navigationFragment != null) {  
 transaction.hide(navigationFragment);  
 }  
 break;  
 case Constants.*FRAGMENT\_KNOWLEDGE*:  
 if (knowledgeFragment != null) {  
 transaction.hide(knowledgeFragment);  
 }  
 break;  
 case Constants.*FRAGMENT\_PUBLIC\_NUM*:  
 if (publicNumFragment != null) {  
 transaction.hide(publicNumFragment);  
 }  
 break;  
 case Constants.*FRAGMENT\_PRPJECT*:  
 if (projectFragment != null) {  
 transaction.hide(projectFragment);  
 }  
 break;  
 default:  
 break;  
 }  
 }  
  
 private void initDrawerLayout() {  
 ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(this, drawer\_layout, toolbar,  
 R.string.*open\_drawer*, R.string.*close\_drawer*);  
 toggle.syncState();  
 drawer\_layout.addDrawerListener(toggle);  
 }  
  
 private void startLoginActivity() {  
 Intent intent = new Intent(this, LoginActivity.class);  
 startActivity(intent);  
 }  
  
  
 @Override  
 protected void initEventAndData() {  
  
 }  
  
  
 @Override  
 public void showErrorMessage(String message) {  
  
 }  
  
 @Override  
 public void showLoading() {  
  
 }  
  
 @Override  
 public void hideLoading() {  
  
 }  
  
 @Override  
 public void showError() {  
  
 }  
  
 @Override  
 public void showNoNetwork() {  
  
 }  
  
 @Override  
 public void showEmpty() {  
  
 }  
  
 @Override  
 public void showContent() {  
  
 }  
  
 @Override  
 public void handleLoginSuccess() {  
 tv\_welcome.setVisibility(View.*VISIBLE*);  
 tv\_login\_account.setText(mPresenter.getLoginAccount());  
 //隐藏登录按钮  
 navigation\_view.getMenu().findItem(R.id.*item\_login*).setVisible(false);  
 //显示注销按钮  
 navigation\_view.getMenu().findItem(R.id.*item\_logout*).setVisible(true);  
 }  
  
 @Override  
 public void handleLogoutSuccess() {  
 tv\_welcome.setVisibility(View.*INVISIBLE*);  
 tv\_login\_account.setText("");  
 navigation\_view.getMenu().findItem(R.id.*item\_login*).setVisible(true);  
 navigation\_view.getMenu().findItem(R.id.*item\_logout*).setVisible(false);  
 Toast.*makeText*(this,getString(R.string.*logout\_success*),Toast.*LENGTH\_SHORT*).show();  
 }  
 */\*\*  
 \* 处理回退事件  
 \*/* @Override  
 public void onBackPressedSupport() {  
 if (getSupportFragmentManager().getBackStackEntryCount() > 1) {  
 pop();  
 } else {  
 long currentTime = System.*currentTimeMillis*();  
 if ((currentTime - clickTime) > Constants.*DOUBLE\_INTERVAL\_TIME*) {  
 Toast.*makeText*(MainActivity.this, getString(R.string.*double\_click\_exit\_toast*),Toast.*LENGTH\_SHORT*).show();  
 clickTime = System.*currentTimeMillis*();  
 } else {  
 finish();  
 }  
 }  
 }  
}