关于Fragment重复添加的异常处理

异常详情

应用界面需要弹出Loading加载框，由于我们的加载框是基于DialogFragment的，本质上还是一个Fragment，所以在会出现重复add的问题。

测试描述：连续在不同状态的TAB页上快速切换点击必然会闪退。

处理措施：在代码中加上判断，先利用 FragmentManager 寻找，未找到再执行 add 操作，但是还是未能完全解决。

测试

测试代码一

```
TestFragment fragment = new TestFragment();
FragmentManager supportFragmentManager = getSupportFragmentManager();
FragmentTransaction fragmentTransaction = supportFragmentManager.beginTransaction();
fragmentTransaction.add(fragment, "test");
fragmentTransaction.commit();
Fragment t = supportFragmentManager.findFragmentByTag("t");
ToastUtil.showMessage(t == null ? "null" : t.getClass().getName());
```

运行测试代码，toast弹出 “null”。

测试代码二

```
TestFragment fragment = new TestFragment();
FragmentManager supportFragmentManager = getSupportFragmentManager();
FragmentTransaction fragmentTransaction = supportFragmentManager.beginTransaction();
fragmentTransaction.add(fragment, "test");
fragmentTransaction.commit();
ToastUtil.showMessage(fragment.isAdded() + "");
```

运行测试代码，Toast弹出 “false”。

源码查看

查看add方法源码：

```
    @Override
    public FragmentTransaction add(Fragment fragment, String tag) {
```

```
        doAddOp(0, fragment, tag, OP_ADD);
        return this;
    }

    private void doAddOp(int containerViewId, Fragment fragment, String tag, int opcmd)
{
        ...

        addOp(new Op(opcmd, fragment));
    }

    void addOp(Op op) {
        mOps.add(op);
        op.enterAnim = mEnterAnim;
        op.exitAnim = mExitAnim;
        op.popEnterAnim = mPopEnterAnim;
        op.popExitAnim = mPopExitAnim;
    }
```

事务的add等方法都是将操作缓存起来，等到执行commit的时候再执行。

查看 commit方法：

```
    @Override
    public int commit() {
        return commitInternal(false);
    }

    int commitInternal(boolean allowStateLoss) {
        ...
        // 队列，可疑
        mManager.enqueueAction(this, allowStateLoss);
        return mIndex;
    }

    public void enqueueAction(OpGenerator action, boolean allowStateLoss) {
        ...
        synchronized (this) {
            ...
            // schedule，调度，可疑
            scheduleCommit();
        }
    }

    private void scheduleCommit() {
        synchronized (this) {
            ...
            if (postponeReady || pendingReady) {
                mHost.getHandler().removeCallbacks(mExecCommit);
                // 这里它使用post方法，来执行commit操作
                mHost.getHandler().post(mExecCommit);
            }
        }
```

```
    }
```

因为Android的刷新机制，post的代码至少要等到下一帧才能执行，有一定时间间隔。

测试代码三：

```
Log.e("time", "onClick2: " + System.currentTimeMillis());
new Handler().post(new Runnable() {
    @Override
    public void run() {
        Log.e("time", "onClick3: " + System.currentTimeMillis());
    }
});
```

输出如下：

```
2018-11-29 10:50:17.412 27664-27664/com.sf.trtms.sample E/time: onClick2: 1543459817412
2018-11-29 10:50:17.420 27664-27664/com.sf.trtms.sample E/time: onClick3: 1543459817420
```

解决方法

由于Fragment的添加是基于Handler内部队列的，我们也可以使用队列，如下：

```
TestFragment fragment = new TestFragment();
FragmentManager supportFragmentManager = getSupportFragmentManager();
FragmentTransaction fragmentTransaction = supportFragmentManager.beginTransaction();
fragmentTransaction.add(fragment, "test");
fragmentTransaction.commit();
new Handler().post(new Runnable() {
    @Override
    public void run() {
        Fragment t = supportFragmentManager.findFragmentByTag("t");
        ToastUtil.showMessage(t == null ? "null" : t.getClass().getSimpleName());
    }
});
```

执行代码，输出“TestFragment”。