

MICROOH 麦可网

Android-从程序员到架构师之路

出品人：Sundy

讲师：高焕堂（台湾）

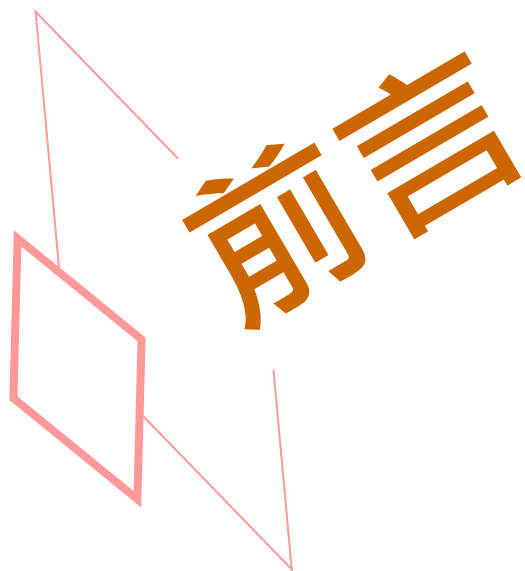
<http://www.microoh.com>

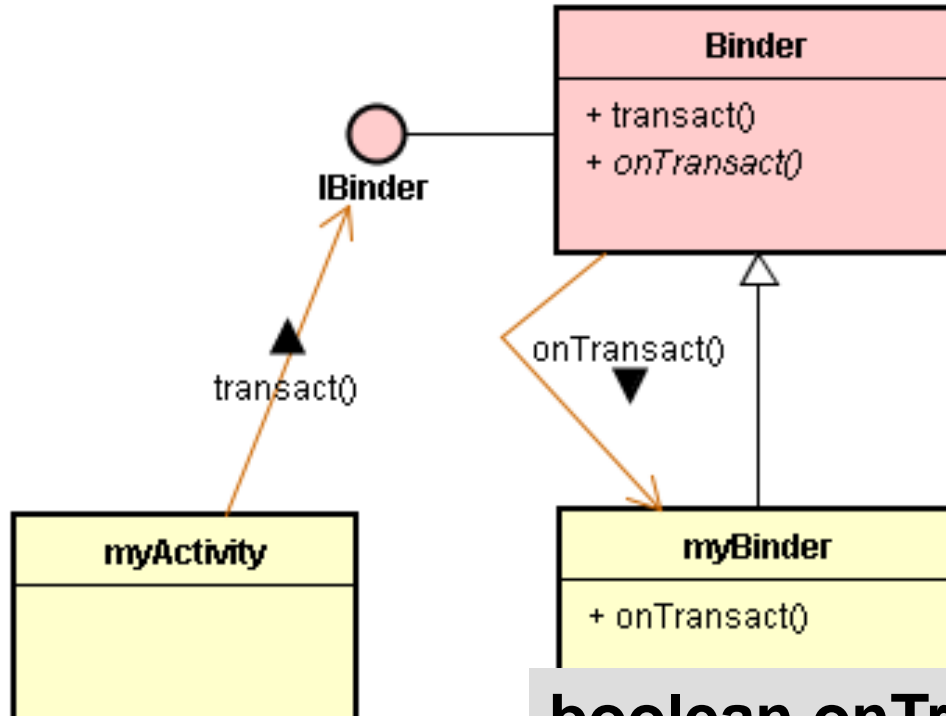
B05_b

IPC的Proxy-Stub设计模式(b)

By 高煥堂

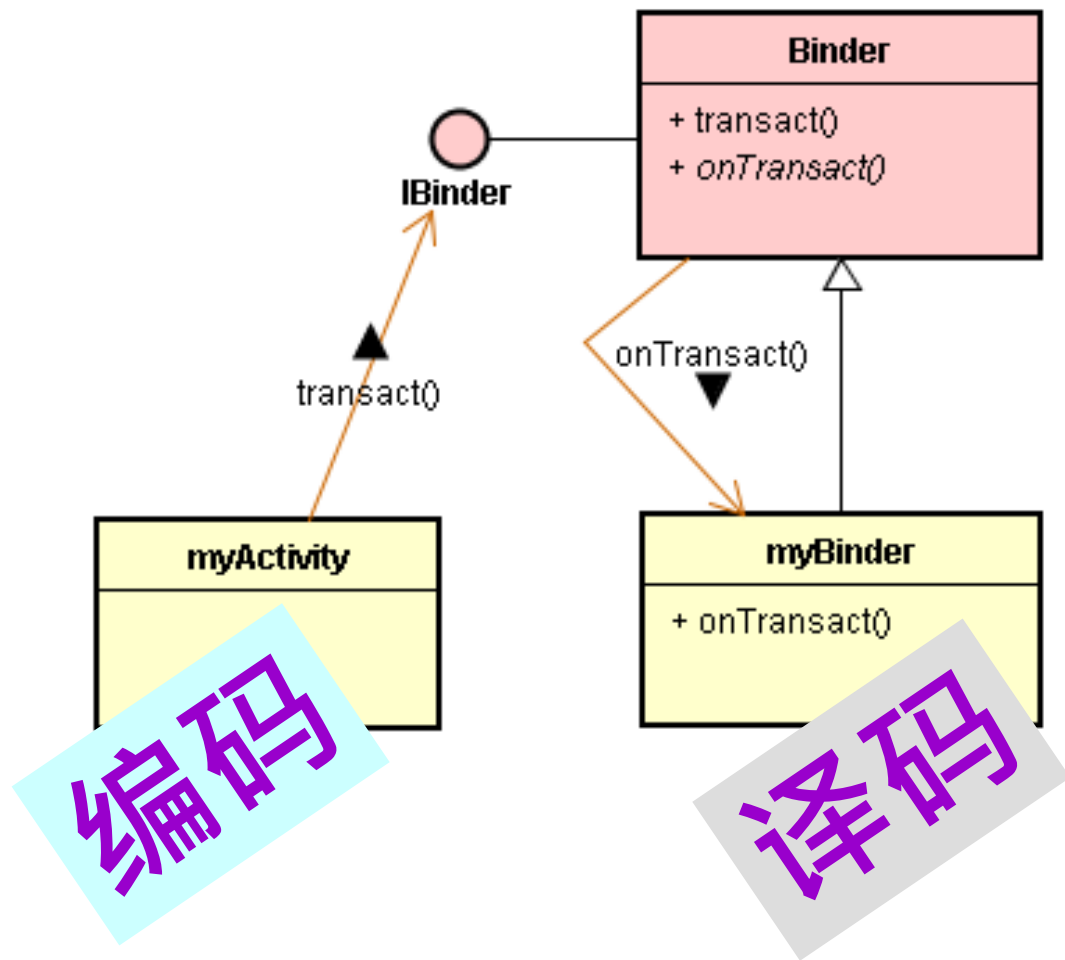
2、IBinder接口的一般用途

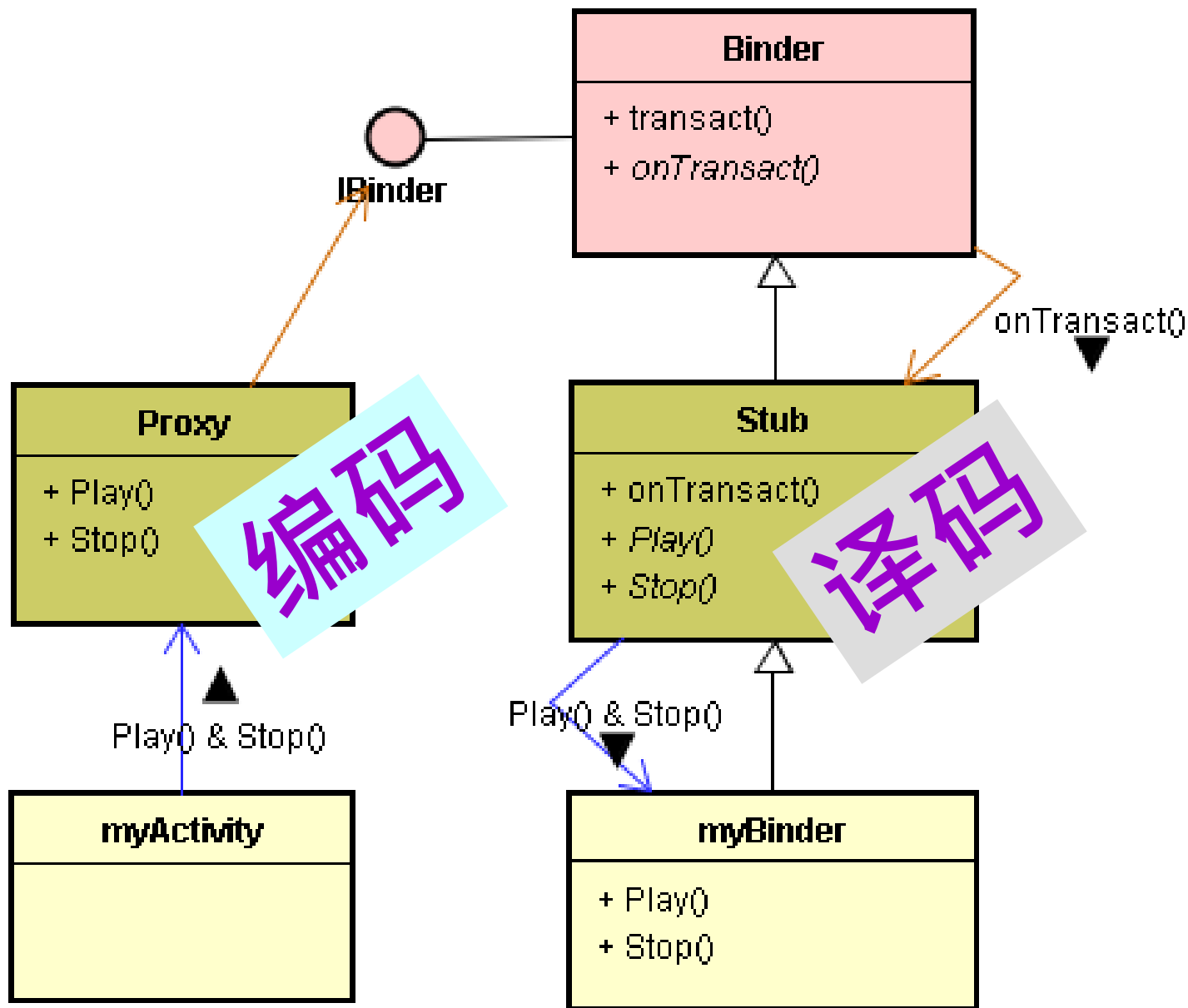


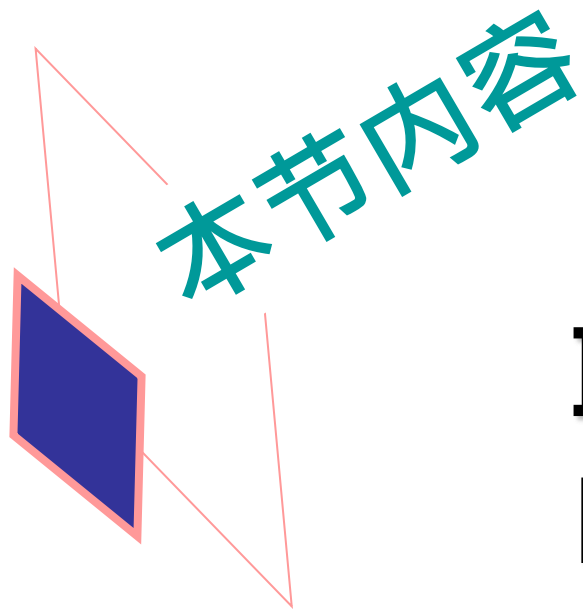


```
ib.transact(1,
            data, reply, 0);
// .....
ib.transact(2,
            data, reply, 0);
```

```
boolean onTransact( int code,
                    Parcel data, Parcel reply, int
                    flags) {
    switch( code ){
        case 1: // call Play()
        case 2: // call Stop()
    }
}
```

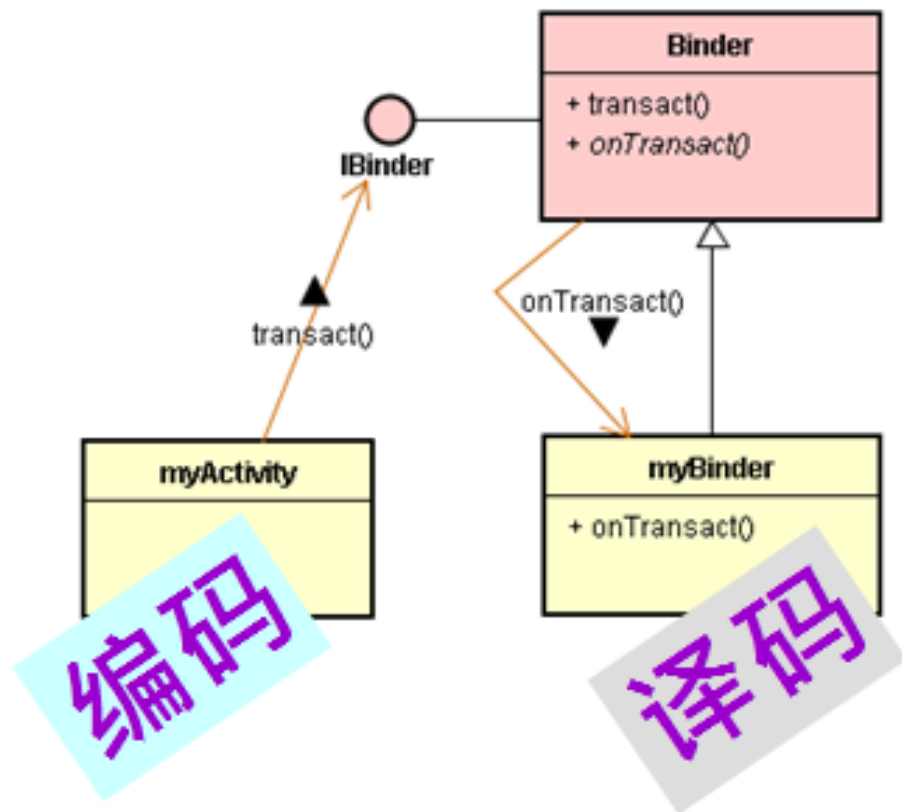




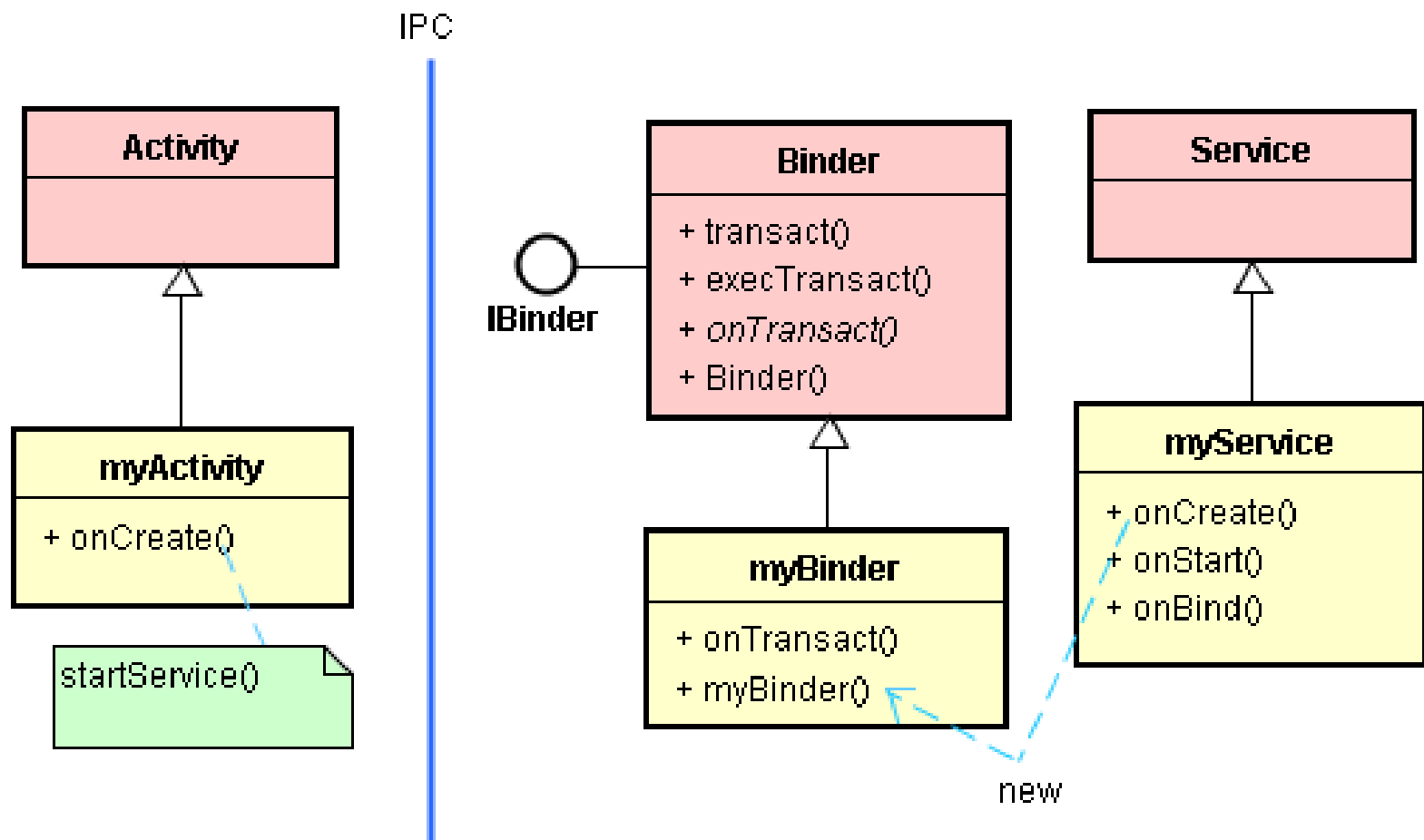


IBinder接口 的一般用途

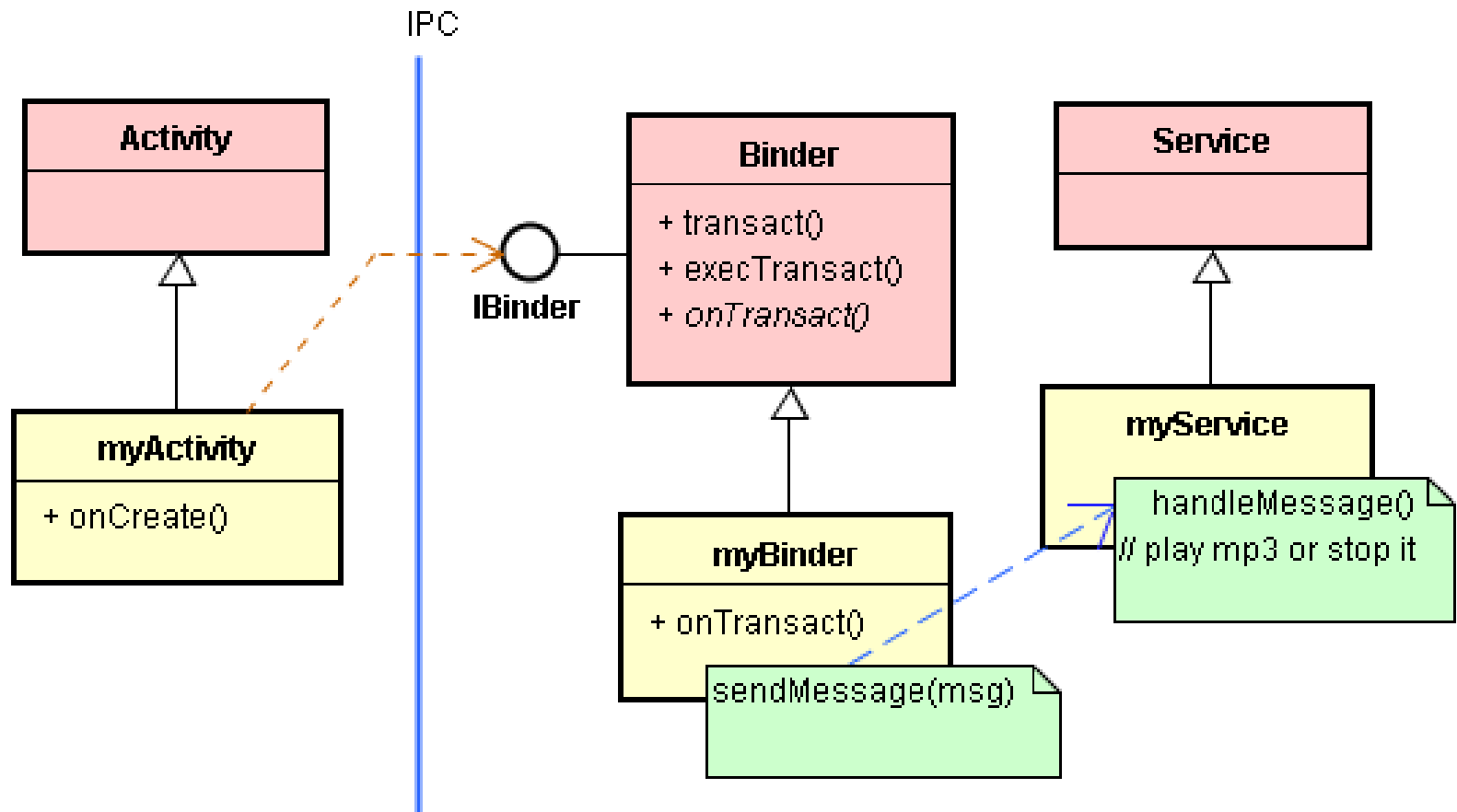
一般用途



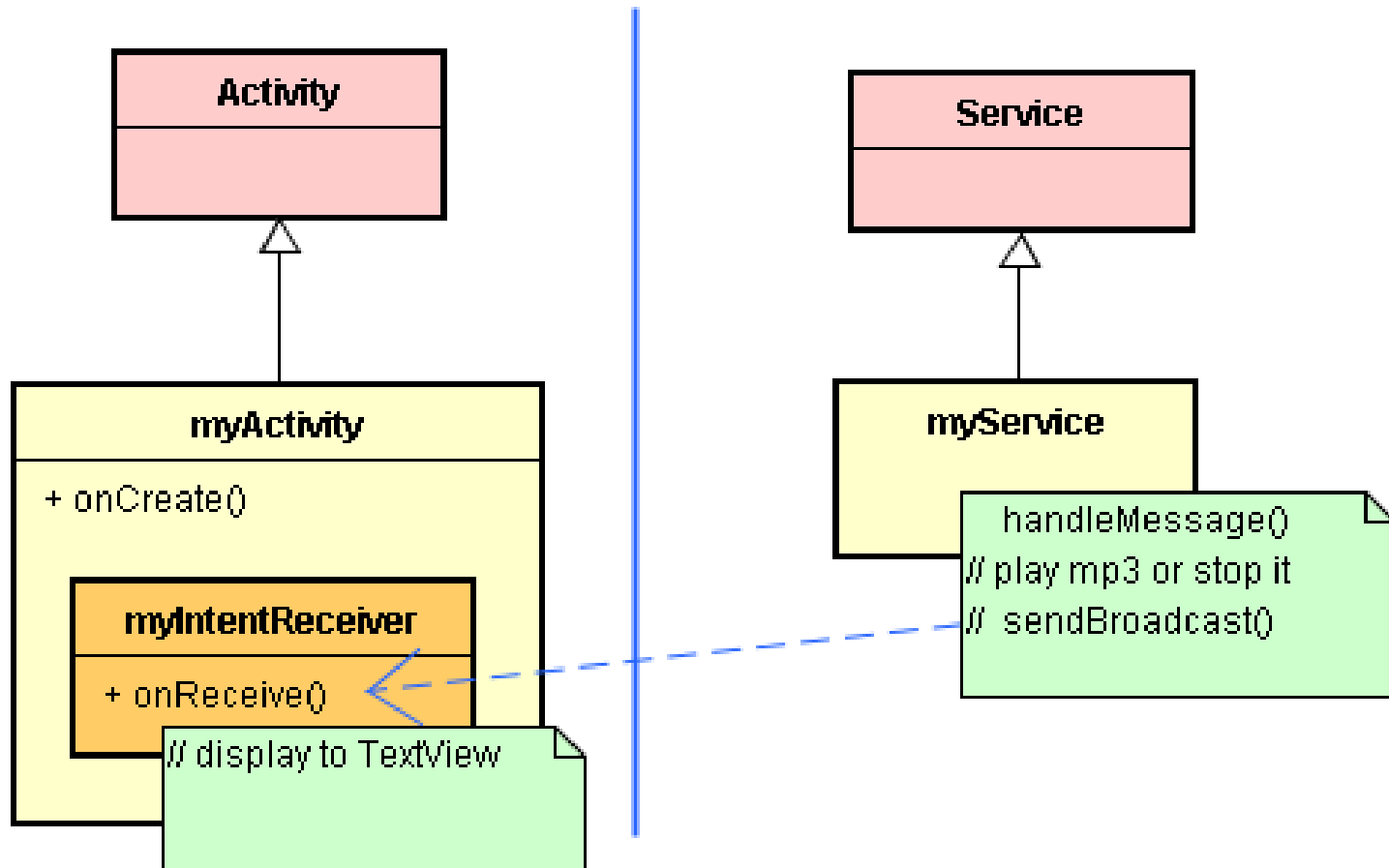
- Android的IPC框架仰赖单一的IBinder接口。此时Client端调用IBinder接口的transact()函数，透过IPC机制而调用到远方(Remote)的onTransact()函数。
- 在Java层框架里，IBinder接口实现于Binder基类，如下图：



- myActivity调用IBinder接口，执行myBinder的onTransact()函数，可送信息给myService去播放mp3音乐，如下图：





- myService也能送Broadcast信息给 myActivity，将字符串显示于画面上：



3G   5:47 PM

Rx005

play 

stop 

exit

Playing

// myActivity.java

//

```
public class myActivity extends Activity implements OnClickListener {  
    private final int WC = LinearLayout.LayoutParams.WRAP_CONTENT;  
    private final int FP = LinearLayout.LayoutParams.FILL_PARENT;  
    private Button btn, btn2, btn3;  
    public TextView tv;  
    private IBinder ib = null;  
    private final String MY_S_EVENT =  
        new String("com.misoo.pk01.myService.MY_S_EVENT");  
    protected final IntentFilter filter=new IntentFilter(MY_S_EVENT);  
    private BroadcastReceiver receiver=new myIntentReceiver();
```

```
public void onCreate(Bundle icle) {  
    super.onCreate(icle);  
    LinearLayout layout = new LinearLayout(this);  
    layout.setOrientation(LinearLayout.VERTICAL);  
    btn = new Button(this); btn.setId(101); btn.setText("play");  
    btn.setBackgroundResource(R.drawable.heart);  
    btn.setOnClickListener(this);  
    LinearLayout.LayoutParams param =  
        new LinearLayout.LayoutParams(80, 50);  
    param.topMargin = 10; layout.addView(btn, param);  
  
    btn2 = new Button(this);  
    btn2.setId(102); btn2.setText("stop");  
    btn2.setBackgroundResource(R.drawable.heart);  
    btn2.setOnClickListener(this);  
    layout.addView(btn2, param);  
}
```

```
btn3 = new Button(this);  
btn3.setId(103); btn3.setText("exit");  
btn3.setBackgroundResource(R.drawable.cloud);  
btn3.setOnClickListener(this);  
layout.addView(btn3, param);  
tv = new TextView(this); tv.setText("Ready");
```

```
LinearLayout.LayoutParams param2 = new  
LinearLayout.LayoutParams(FP, WC);  
param2.topMargin = 10;  
layout.addView(tv, param2);  
setContentView(layout);  
//-----  
registerReceiver(receiver, filter);  
//-----  
bindService( new  
    Intent("com.misoo.pk01.REMOTE_SERVICE"),  
    mConnection, Context.BIND_AUTO_CREATE);  
}
```

```
btn3 = new Button(this);  
btn3.setId(103); btn3.setText("exit");  
btn3.setBackgroundResource(R.drawable.cloud);  
btn3.setOnClickListener(this);  
layout.addView(btn3, param);  
tv = new TextView(this); tv.setText("Ready");
```

```
LinearLayout.LayoutParams param2 = new  
LinearLayout.LayoutParams(FP, WC);  
param2.topMargin = 10;  
layout.addView(tv, param2);  
setContentView(layout);  
//-----  
registerReceiver(receiver, filter);  
//-----  
bindService( new Intent("com.misoo.pk01.REMOTE_SERVICE"),  
            mConnection, Context.BIND_AUTO_CREATE );  
}
```

```
private ServiceConnection mConnection =  
    new ServiceConnection() {  
        @Override public void  
            onServiceConnected(ComponentName className,  
                               IBinder ibinder) {  
                ib = ibinder;  
            }  
        @Override public void  
            onServiceDisconnected(ComponentName name) {  
            }  
    };
```

```
public void onClick(View v) {  
    switch (v.getId()) {  
        case 101: // Play Button  
            Parcel data = Parcel.obtain();  
            Parcel reply = Parcel.obtain();  
            try { ib.transact(1, data, reply, 0);  
            } catch (Exception e) { e.printStackTrace(); }  
            break;  
        case 102: // Stop Button  
            data = Parcel.obtain(); reply = Parcel.obtain();  
            try { ib.transact(2, data, reply, 0);  
            } catch (Exception e) { e.printStackTrace(); }  
            break;  
        case 103: finish(); break;  
    }  
}
```

- 其中的代码：

```
case 101: // Play Button
    //.....
    ib.transact(1, data, reply, 0);
case 102: // Stop Button
    // .....
    ib.transact(2, data, reply, 0);
```



- 就是对<Play>和<Stop>两个功能进行
“ 编码 ” 的动作。

- 编好码之后，就将这编码值当做第1个参数传给IBinder接口的transact()函数。
- 于是编码值就跨进程地传递到myBinder类里的onTransact()函数了。


```
class myIntentReceiver extends BroadcastReceiver {  
    @Override  
    public void onReceive(Context context, Intent intent) {  
        int bn = intent.getIntExtra("key",-1);  
        if(bn == 0)  
            tv.setText("Playing");  
        else  
            tv.setText("Stop.");  
        }  
    }  
}
```

```
// myService.java
```

```
// .....
```

```
public class myService extends Service implements Runnable {  
    private IBinder mBinder = null;  
    private Thread th1;  
    public static Handler h;  
    private MediaPlayer mPlayer = null;  
    public static Context ctx;  
    private final String MY_S_EVENT =  
        new String("com.misoo.pk01.myService.MY_S_EVENT");  
    @Override public void onCreate() {  
        super.onCreate(); ctx = this;  
        mBinder = new myBinder();  
        // 诞生一个子线程及其MQ ; 等待Message  
        th1 = new Thread(this);  
        th1.start();  
    }  
}
```

@Override

```
public IBinder onBind(Intent intent) { return mBinder; }
```

```
public void run() {
```

```
    Looper.prepare();
```

```
    h = new EventHandler(Looper.myLooper());
```

```
    Looper.loop();
```

```
}
```

```
//-----
```

```
class EventHandler extends Handler {
```

```
    public EventHandler(Looper looper) { super(looper); }
```

```
    public void handleMessage(Message msg) {
```

```
        String obj = (String)msg.obj;
```

```
        if(obj.contains("play")) {
```

```
            if(mPlayer != null) return;
```

```
            //-----
```

```
            Intent in = new Intent(MY_S_EVENT);
```

```
            in.putExtra("key", 0);
```

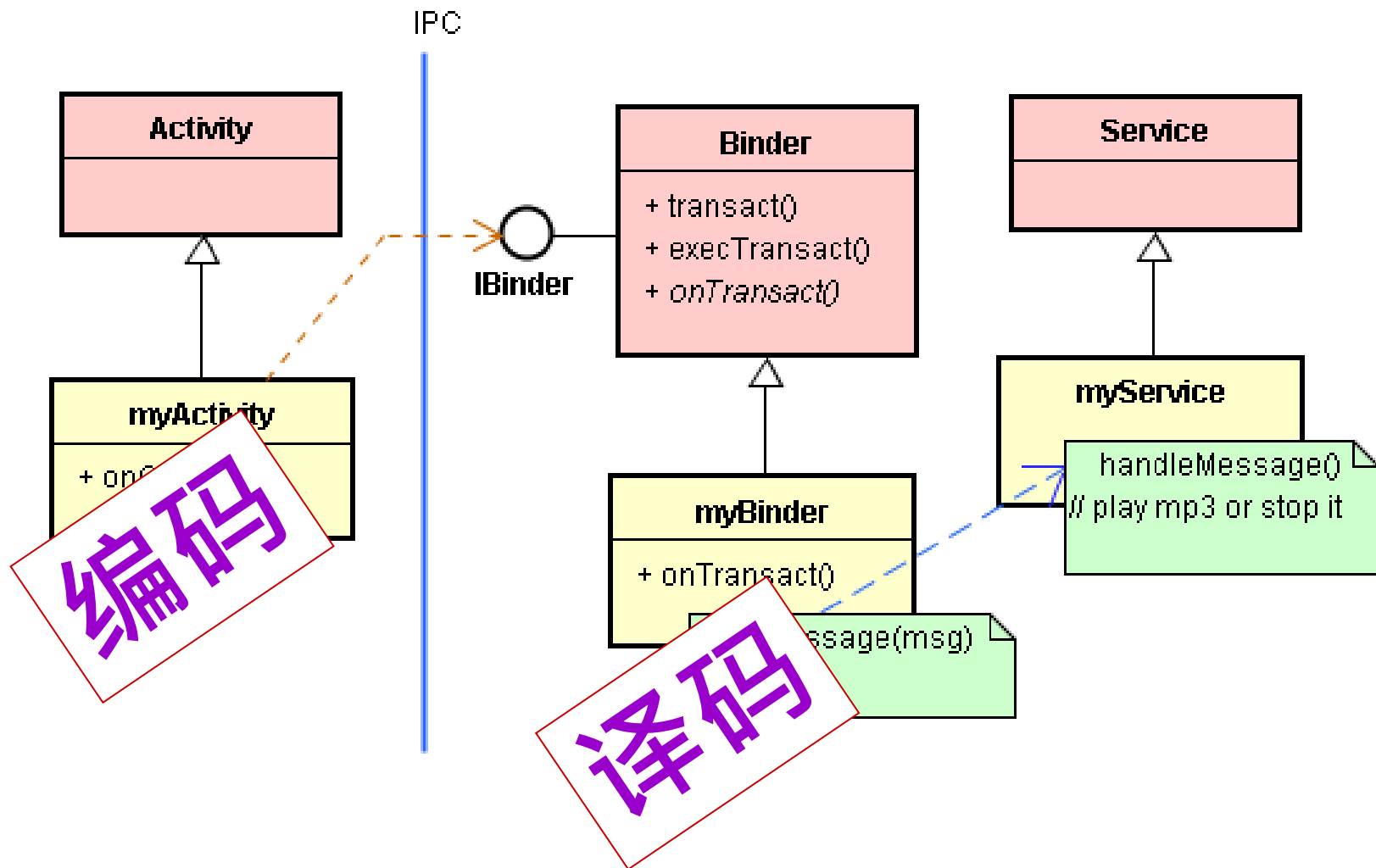
```
            ctx.sendBroadcast(in);
```

```
//-----  
mPlayer = MediaPlayer.create(ctx, R.raw.dreamed);  
try { mPlayer.start();  
    } catch (Exception e) {  
        Log.e("Play", "error: " + e.getMessage(), e);  
    }  
}  
else if(obj.contains("stop")) {  
    if (mPlayer != null) {  
        Intent in = new Intent(MY_S_EVENT);  
        in.putExtra("key", 1);  
        ctx.sendBroadcast(in);  
        //-----  
        mPlayer.stop(); mPlayer.release();  
        mPlayer = null;  
    }  
}  
}
```

```
// myBinder.java
// .....
public class myBinder extends Binder{
    @Override public boolean onTransact( int code, Parcel data,
        Parcel reply, int flags) throws android.os.RemoteException {
        switch( code ){
            case 1:
                // 将Message丢到子线程的MQ to play MP3
                String obj = "play";
                Message msg = myService.h.obtainMessage(1,1,1,obj);
                myService.h.sendMessage(msg);
                break;
            case 2:
                // 将Message丢到子线程的MQ to stop playing
                obj = "stop";
                msg = myService.h.obtainMessage(1,1,1,obj);
                myService.h.sendMessage(msg);
                break;
        }
        return true;
    }
}
```



- 其代码就是对code进行“译码”动作。
- 如果code值为1就执行<Play>动作；如果code值为2就执行<Stop>动作。



大大增加了
App开发者的负担!!



~ Continued ~