河北师范大学软件学院
Software College of Hebei Normal University

EventBus

智能设备教研室

- EventBus简介

- EventBus配置
- EventBus基本用法

传值并返回的情况

```
btnTurn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Intent intent = new Intent();
        intent.setClass(getApplicationContext(), ThirdActivity.class);
        startActivityForResult(intent,  requestCode: 1);
    }
});
```

传值并返回的情况

```
btnSend.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        String msg = "我是返回的数据";
        Intent intent = new Intent();
        intent.setClass(getApplicationContext(), MainActivity.class);
        intent.putExtra( name: "msg", msg);
        setResult( resultCode: 2, intent);
        finish();
    }
});
```

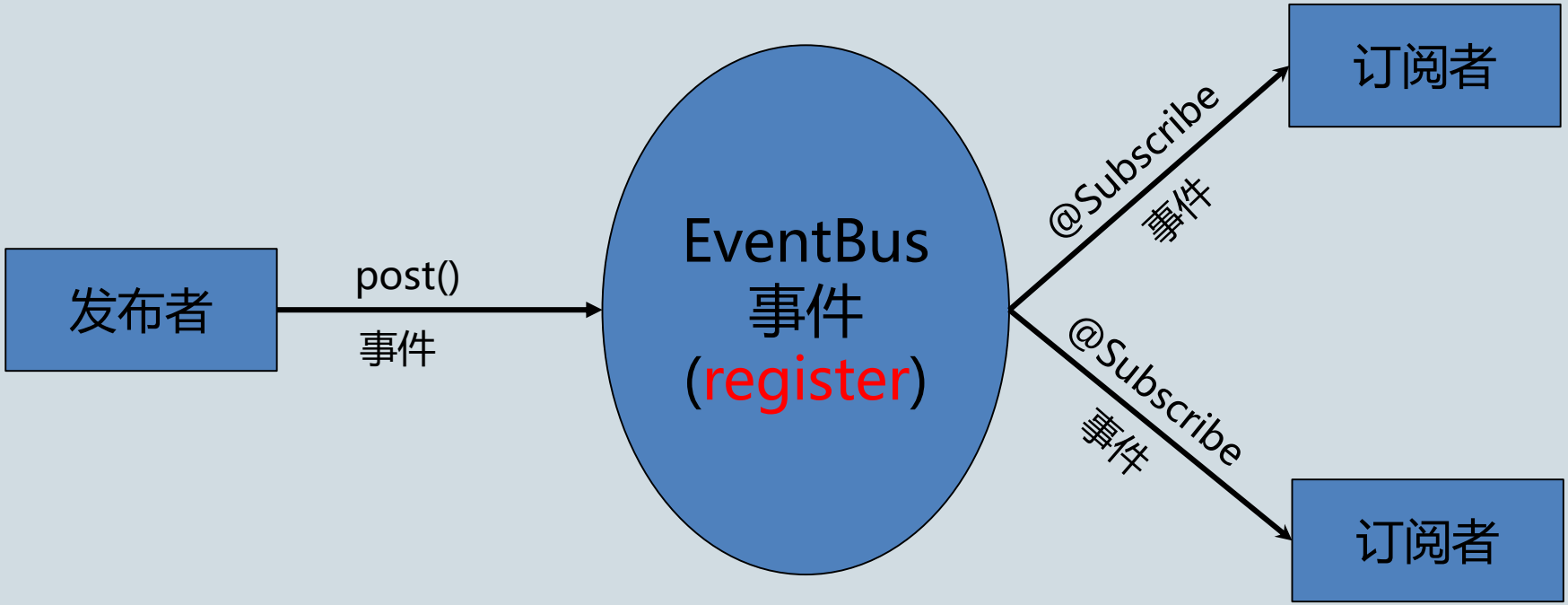
传值并返回的情况

```
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if(resultCode == 2){
        String msg = data.getStringExtra( name: "msg");
        welcome.setText(msg);
    }
}
```

逻辑复杂，代码繁琐，如何简化？

EventBus

- 类似于发布者/订阅者模式



EventBus

- 最新版本 3.1.1
- 三要素
  - 事件
  - 事件订阅者
  - 事件发布者
- 注意
  - EventBus事件必须注册才有效

EventBus简介

- 优点
  - 简化组件之间的通信
  - 解耦
  - 速度快
  - 结构清晰，使用简单
  - 包小（大概50K）

- EventBus简介
- EventBus配置
- 基本用法

EventBus配置

- 官网地址：http://greenrobot.org/eventbus
- 源码地址：https://github.com/greenrobot/EventBus

EventBus配置

- 在Module的gradle文件中添加依赖

```
dependencies {
    //EventBus
    implementation 'org.greenrobot:eventbus:3.1.1'
    ……
}
```

- EventBus简介
- EventBus配置
- EventBus基本用法

EventBus基本用法

- 使用步骤
  - 定义事件（可以是普通JavaBean的形式）
  - 订阅事件（方法名称可以自定义）

```
@Subscribe(threadMode = ThreadMode.MAIN)
public void onMessageEvent(MessageEvent event) {
/* Do something */
};
```

  - 发布事件

```
EventBus.getDefault().post(new MessageEvent());
```

EventBus基本用法

- 必须要注册EventBus才有效
  - 如：在Activity或Fragment的声明周期中

```
@Override
public void onStart() {
      super.onStart();
      EventBus.getDefault().register(this);
 }
```

EventBus基本用法

- 取消注册EventBus
  - 如：在Activity或Fragment的声明周期中

```
@Override
public void onDestroy() {
      super. onDestroy();
      EventBus.getDefault().unregister(this);
 }
```

EventBus基本用法

- 五种线程模型

```
@Subscribe(threadMode = ThreadMode.MAIN)//主线程模型
public void onMessageEvent(MessageEvent event) {
    /* Do something */
}
```

EventBus基本用法

- 五种线程模型
  - POSTING（默认）：订阅者在发布事件的同一个线程中调用
  - MAIN：订阅者在主线程（UI）中调用，不能进行耗时操作
  - MAIN_ORDERED：订阅者在Android主线程中排队等待被调用
  - BACKGROUND：订阅者将在后台线程中调用
  - ASYNC：订阅者方法在单独的线程中调用，始终独立于发布线程和主线程，适用于事件处理程序是耗时的操作，如网络访问

参考：http://greenrobot.org/eventbus/documentation/delivery-threads-threadmode/

扩展

- 粘性事件
  - 如果先发布了事件，然后有订阅者订阅了该事件，那么除非再次发布该事件，否则订阅者将永远接收不到该事件

```
//发布粘性事件
EventBus.getDefault().postSticky(new MessageEvent(“……”));
// 移除指定的粘性事件
removeStickyEvent(Object event);
// 移除所有的粘性事件
removeAllStickyEvents();
// 移除指定类型的粘性事件
removeStickyEvent(Class<T> eventType);
```

Thank You!