Dagger 2: Step To Step

文／iamwent（简书作者）
原文链接：http://www.jianshu.com/p/7505d92d7748

> 假设你已经了解 **依赖注入** 这一概念，只是在如何使用 Dagger 时遇到了一些困扰，因为 Dagger 其实是一个上手难度颇高的库。我试图通过这篇文章解决如何上手这一问题。

目前 Dagger 有两个分支，一个由 Square 维护，一个为 Google 在前者的基础上开出的分支，即 Dagger 2 。关于二者的比较，点击此处 。

本文写作过程中参考了不少优秀的 Dagger 文章，列在文章末尾。

在此一并感谢他们的工作！

Dagger

在引入 Dagger 之前，我们需要了解一些基础概念。Dagger 主要分三块：

- **@Inject**：需要注入依赖的地方，Dagger 会构造一个该类的实例并满足它所需要的依赖；
- **@Module**：依赖的提供者，Module 类中的方法专门提供依赖，并用 **@Provides** 注解标记；
- **@Component**：依赖的注入者，是 **@Inject** 和 **@Module** 的桥梁，它从 **@Module** 中获取依赖并注入给 **@Inject**。

对于以上关系，一句话解释就是：**模块（Module）负责提供依赖，组件（Component）负责注入依赖。**

Sourcecode

源码放在 Github: DaggerDemo

Gradle

1. project/build.gradle

```
buildscript {
    ...

    dependencies {
        ...

        classpath 'com.neenbedankt.gradle.plugins:android-apt:1.8'
    }
}
```

2. project/app/build.gradle

```
apply plugin: 'com.android.application'
apply plugin: 'com.neenbedankt.android-apt'

android {
    ...
}

dependencies {
    ...

    //Required by Dagger2
    apt 'com.google.dagger:dagger-compiler:2.0.2'
    compile 'com.google.dagger:dagger:2.0.2'
    // Dagger 2 中会用到 @Generated 注解，而 Android SDK 中没有 javax.anotation.Generated.class，所以在 Android 项目要添加此句
    provided 'org.glassfish:javax.annotation:10.0-b28'
```

我的参与

最新评论

我的标签

更多链接

最新随笔

1. android viewpager change adapter －－－在使用viewpager设置新的adapter的时候发现页面还是显示旧的adapter中的值

2. 数据库升级总结

3. 异常情况下的Activity生命周期分析

4. 知识总结： Activity的四种启动模式

5. 自定义NavigationView's item 的高度

6. 图片缩放时java.lang.IllegalArgumentException: pointerIndex out of range解决方案

7. 你所不知道的string.xml

8. 滑动RecyclerView时出现异常： java.lang.IndexOutOfBoundsException: Inconsistency detected. Invalid item position 6(offset:6).state:30

9. 图片url中包含中文导致网络请求404

10. RecyclerView一个奇怪的npe异常

我的标签

线程安全(1)

随笔分类(247)

android Training(9)

android知识点复习与总结(75)

JAVA 知识点复习与总结(2)

RxJava(9)

读书笔记(8)

开源项目学习(37)

设计模式(14)

算法与数据结构(2)

项目经验谈(73)

性能优化(12)

学习大牛(6)

```
}
```

Example

Demo 实现一个简单的 ListView 显示字符串列表

1. 创建 UserAdapter 类，并在构造函数前添加 **@Inject** 注解。这样，Dagger 就会在需要获取 UserAdapter 对象时，调用这个被标记的构造函数，从而生成一个 UserAdapter 对象。

```
public class UserAdapter extends BaseAdapter {
    private LayoutInflater inflater;
    private List<String> users;

    @Inject
    public UserAdapter(Context ctx, List<String> users) {
        this.inflater = LayoutInflater.from(ctx);
        this.users = users;
    }

    ...
}
```

需要注意的是：如果构造函数含有参数，Dagger 会在调用构造对象的时候先去获取这些参数(不然谁来传参？)，所以你要保证它的参数也提供可被 Dagger 调用到的生成函数。Dagger 可调用的对象生成方式有两种：一种是用 **@Inject** 修饰的构造函数，上面就是这种方式。另外一种是用 **@Provides** 修饰的函数，下面会讲到。参考：Dagger 源码解析

2. 构建 Module，提供 Context 和 List<String> 依赖，如此， Dagger 生成 UserAdapter 时所需要的依赖就从这里获取。

```
@Module
public class UserModule {
    private static final int COUNT = 10;

    private final Context context;

    public UserModule(Context context) {
        this.context = context;
    }

    @Provides
    @ActivityScope
    Context provideActivityContext() {
        return context;
    }

    @Provides
    @ActivityScope
    List<String> provideUsers() {
        List<String> users = new ArrayList<>(COUNT);

        for (int i = 0; i < COUNT; i++) {
            users.add("item " + i);
        }

        return users;
    }
}
```

@ActivityScope 是一个自定义的范围注解，作用是允许对象被记录在正确的组件中，当然这些对象的生命周期应该遵循 Activity 的生命周期。

```
import java.lang.annotation.Retention;
import static java.lang.annotation.RetentionPolicy.RUNTIME;

import javax.inject.Scope;

@Scope
@Retention(RUNTIME)
public @interface ActivityScope {

}
```

3. 构建 Component，负责注入依赖

```
@ActivityScope
@Component(modules = {UserModule.class})
```

随笔档案(241)

2016年3月 (21)

2016年2月 (29)

2016年1月 (5)

2015年12月 (21)

2015年11月 (8)

2015年10月 (1)

2015年7月 (1)

2015年5月 (1)

2015年4月 (4)

2015年1月 (1)

2014年12月 (2)

2014年10月 (4)

2014年9月 (1)

2014年8月 (3)

2014年5月 (27)

2014年4月 (7)

2014年3月 (32)

2014年2月 (17)

2014年1月 (7)

2013年11月 (2)

2013年10月 (10)

2013年9月 (4)

2013年8月 (28)

2013年7月 (5)

文章分类

android知识点

java 设计模式

java设计模式

积分与排名

积分 - 7676

排名 - 20958

最新评论

1. Re:Github上更新自己Fork的代码

```
public interface UserComponent {
    void inject(MainActivity activity);
}
```

注意：这里必须是真正消耗依赖的类型 MainActivity，而不可以写成其父类，比如 Activity ，否则会导致注入失败。(参考：使用Dagger 2进行依赖注入)

4. 完成依赖注入

```
public class MainActivity extends AppCompatActivity {

    ...

    @Inject
    UserAdapter adapter;

    @Override
    protected void onCreate(Bundle savedInstanceState) {

        ...

        // 完成注入
        DaggerUserComponent.builder()
            .userModule(new UserModule(this))
            .build()
            .inject(this);

        listView.setAdapter(adapter);
    }
}
```

如果找不到 DaggerUserComponent 类，你需要先编译一下整个项目。这是因为 Dagger 是在编译时生成必要的元素，编译时 Dagger 会处理我们的注解，为 @Components 生成实现并命名为 Dagger$${YouComponentClassName}，如 UserComponent -> DaggerUserComponent 。你可以在 app/build/generated/source/apt 下找到相关的类。

实际上，调用 inject 方法最终调用的是以下这样一段代码，更多细节可以查看源码。

```
@Override
public void injectMembers(MainActivity instance) {
  if (instance == null) {
    throw new NullPointerException("Cannot inject members into a null reference");
  }
  supertypeInjector.injectMembers(instance);
  instance.adapter = adapterProvider.get();
}
```

Dagger too

1. @Inject 和 @Provide 两种依赖生成方式的区别：
   - @Inject 用于注入可实例化的类，@Provides 可用于注入所有类
   - @Inject 可用于修饰属性和构造函数，可用于任何非 Module 类，@Provides 只可用于用于修饰非构造函数，并且该函数必须在某个Module内部
   - @Inject 修饰的函数只能是构造函数，@Provides 修饰的函数必须以 provide 开头
2. Dagger 的其他注解：
   - **@Scope**： Dagger 可以通过自定义注解限定注解作用域，参考前面的 **@ActivityScope**。
   - **@Qualifier**：限定符，当类的类型不足以鉴别一个依赖的时候，我们就可以使用这个注解来区分。例如：在 Android 中，我们会需要不同类型的 Context，所以我们可以定义 @Qualifier 注解 @ForApplication 和 @ForActivity，这样当注入一个 Context 的时候，我们就可以告诉 Dagger 我们想要哪种类型的 Context。
   - **@Singleton**：单例模式，依赖的对象只会被初始化一次
3. Dagger 的实际应用：本例只是一个上手教程，辅助理解 Dagger 的原理及使用方式，具体的项目应用可以参考 Reference 中第 3 条的 Avengers 的源码 。

Reference

1. Dagger 2 Official Documentation
2. Tasting Dagger 2 on Android，中文：详解 Dagger 2

可以在线pull request同步

--Michael Jiang

阅读排行榜

1. 自定义滚轮效果选择器spinnerwheel的使用总结(431)
2. Android开源项目发现---ViewPager 、Gallery 篇（持续更新）(222)
3. 实现ImageView中两张图片重叠显示(202)
4. Android用户界面 UI组件--AdapterView及其子类(一) ListView及各种Adapter详解(142)
5. Android开源项目发现---TextView,Button篇（持续更新）(121)

评论排行榜

1. Github上更新自己Fork的代码(1)

3. When the Avengers meet Dagger2, RxJava and Retrofit in a clean way，中文：当复仇者联盟遇上 Dragger2、RxJava 和 Retrofit 的巧妙结合
4. 使用 Dagger 2 进行依赖注入
5. Dagger 源码解析 (PS: 这是 Dagger 1，但是很有参考价值)

分类: 开源项目学习

好文要顶 关注我 收藏该文

Leo的银弹
关注 - 0
粉丝 - 7
+加关注

0 0

(请您对文章做出评价)

« 上一篇：Introducing RecyclerView(二)
» 下一篇：Android：控件WebView显示网页

posted @ 2016-03-10 22:04 Leo的银弹 阅读(12) 评论(0) 编辑 收藏

刷新评论 刷新页面 返回顶部

注册用户登录后才能发表评论，请 登录 或 注册，访问网站首页。

最新IT新闻:
· 苹果支持新购iPhone 5S免费退换iPhone SE？
· 程序bug导致了天大的损失，要枪毙程序猿吗？
· 日本PC出货量连21个月衰退，2月大减23%
· 微软问你愿意用1/10的价格，将你的Xbox数码游戏卖回给微软吗？
· 艾滋病治疗重大突破！科学家成功从人类免疫细胞上移除HIV-1病毒
» 更多新闻...

最新知识库文章:
· 如何运维千台以上游戏云服务器
· 架构漫谈（一）：什么是架构？
· 架构的本质
· 谷歌背后的数学
· Medium开发团队谈架构设计
» 更多知识库文章...