

CSDN

博客学院下载GitChat论坛

写博客

发Chat

登录注册

未来很长，但我会努力的走下去。

知道自己想做什么的人,比什么都想做的人更容易成功!

Android 开源项目源码解析 -->公共技术点之 Java 注解 Annotation(四)

标签：源码 android 动画

2016年09月27日 11:47:55

224人阅读 评论(0)

分类：Android 开源项目源码解析 (17)

目录(?)

不少开源库都用到了注解的方式来简化代码提高开发效率。本文简单介绍下 Annotation 示例、概念及作用、分类、自定义、解析，并对几个 Android 开源库 Annotation 原理进行简析。

1. Annotation 示例

Override Annotation

```
@Override
public void onCreate(Bundle savedInstanceState);
```

Retrofit Annotation

```
@GET("/users/{username}")
User getUser(@Path("username") String username);
```

Butter Knife Annotation

```
@InjectView(R.id.user) EditText username;
```

ActiveAndroid Annotation

```
@Column(name = "Name") public String name;
```

Retrofit 为符合 RESTful 规范的网络请求框架
Butter Knife 为 View 及事件等依赖注入框架
Active Android 为 ORM 框架
更多见：Android 开源项目汇总

2. Annotation 概念及作用

2.1 概念

An annotation is a form of metadata, that can be added to Java source code. Classes, methods, variables, parameters and packages may be annotated. Annotations have no direct effect on the operation of the code they annotate.

能够添加到 Java 源代码的语法元数据。类、方法、变量、参数、包都可以被注解，可用来将信息元数据与程序元素进行关联。Annotation 中文常译为“注解”。

2.2 作用

a. 标记，用于告诉编译器一些信息
b. 编译时动态处理，如动态生成代码
c. 运行时动态处理，如得到注解信息

这里的三个作用实际对应着后面自定义 Annotation 时说的 @Retention 三种值分别表示的 Annotati

public class Person {

个人资料

Wei_Leng

博客专家

原创536

粉丝440

喜欢460

评论235

等级：博客 7

访问量：70万+

积分：1万+

排名：1261

友情链接

张鸿洋的博客
郭霖的博客
任玉刚的博客
开斌的博客
敬佩的孔老师
foruok的订阅号程序视界
OpenCV大神shiter
专为Android程序员的导航
泡在网上的日子
逗比小明的博客

文章搜索

博客专栏

Android 性能调优和开源代码

文章：26篇
阅读：25061

第1页 共9页

2018/3/22 下午4:18



Android 基础总结篇

文章：9篇

阅读：9204



Android 学习进阶

文章：28篇

阅读：37340

文章分类

-----软件开发----- (1)

Android 日常记录 (133)

Android 开发常见错误 (8)

Android 基础总结篇 (10)

android 自定义组件 (7)

Android 或Java理论 (6)

Android Git (5)

Android 翻译 (1)

Android Github (2)

Android 个人理解 (5)

Android 开发DEBUG (3)

Android 反编译|混淆|adb|通知 (10)

Android studio (21)

Android 动画|特效|新技术 (5)

Android TV 机顶盒开发 (5)

Android 自定义控件进阶 (344)

Android View系统源码解析 (13)

Android Fragment (4)

Android 不错文章转载 (182)

Android 杂七杂八 (16)

书籍|工具类 (10)

Android 开源代码 (15)

Android性能调优 (13)

Android 开源项目源码解析 (18)

Android Studio 插件类 (5)

Android 最新内容 (1)

阅读排行

android 基于ijkplayer项目进... (26984)

TwinklingRefreshLayout 支持... (10384)

AndPermission 运行时权限 (8922)

setOnFocusChangeListener的... (6426)

在线安装eclipse中html/jsp/x... (6408)

android studio关于命令行打... (6200)

android 傻瓜式 MultiDex 插件... (5278)

如何评价 APICloud ? (5199)

andriod 实现新浪、QQ空间、... (4943)

Retrofit2 结合 Rxjava 解决返... (4884)

```
private int    id;
private String name;


public Person(int id, String name) {
    this.id = id;
    this.name = name;
}

public boolean equals(Person person) {
    return person.id == id;
}


public int hashCode() {
    return id;
}


public static void main(String[] args) {


    Set<Person> set = new HashSet<Person>();
    for (int i = 0; i < 10; i++) {
        set.add(new Person(i, "Jim"));
    }
    System.out.println(set.size());
}
}
```





0











上面的运行结果是多少？

3. Annotation 分类

3.1 标准 Annotation , Override, Deprecated, SuppressWarnings

标准 Annotation 是指 Java 自带的几个 Annotation , 上面三个分别表示重写函数 , 不鼓励使用(有更好方式、使用有风险或已不在维护) , 忽略某项 Warning

3.2 元 Annotation , @Retention, @Target, @Inherited, @Documented

元 Annotation 是指用来定义 Annotation 的 Annotation , 在后面 Annotation 自定义部分会详细介绍含义

3.3 自定义 Annotation

自定义 Annotation 表示自己根据需要定义的 Annotation , 定义时需要用到上面的元 Annotation 这里是一种分类而已 , 也可以根据作用域分为源码时、编译时、运行时 Annotation , 后面在自定义 Annotation 时会具体介绍

4. Annotation 自定义

4.1 调用

```
public class App {

    @MethodInfo(
        author = "trinea.cn+android@gmail.com",
        date = "2014/02/14",
        version = 2)
    public String getAppName() {
        return "trinea";
    }
}
```

这里是调用自定义 Annotation——MethodInfo 的示例。

MethodInfo Annotation 作用为给方法添加相关信息 , 包括 author、date、version。

4.2 定义

```
@Documented
@Retention(RetentionPolicy.RUNTIME)
@Target(ElementType.METHOD)
@Inherited
public @interface MethodInfo {
```



Unable

The Proxy was unable to connect to the server responding to requests. If you feel you have please submit a ticket via the link provided t

URL: <http://pos.baidu.com/s?hei=250&wid=250&2Fblog.csdn.net%2Fu014608640%2Farticle/details/526...>

联系我们



请扫描二维码联系客服

✉ webmaster@csdn.net

☎ 400-660-0108

💬 QQ客服 🗨 客服论坛

关于 招聘 广告服务 百度

©1999-2018 CSDN版权所有

京ICP证09002463号

经营性网站备案信息

网络110报警服务

中国互联网举报中心

北京互联网违法和不良信息举报中心

```
String author() default "trinea@gmail.com";

String date();

int version() default 1;

}
```

这里是 MethodInfo 的实现部分

(1). 通过 @interface 定义，注解名即为自定义注解名

(2). 注解配置参数名为注解类的方法名，且：

- 所有方法没有方法体，没有参数没有修饰符，实际只允许 public & abstract 1 个，默认为 public，不允许抛异常
- 方法返回值只能是基本类型，String, Class, annotation, enumeration 或者是他们的 数组
- 若只有一个默认属性，可直接用 value() 函数。一个属性都没有表示该 Annotation 为 Mark Annotation

(3). 可以加 default 表示默认值

4.3 元 Annotation

@Documented 是否会保存到 Javadoc 文档中

@Retention 保留时间，可选值 SOURCE (源码时)，CLASS (编译时)，RUNTIME (运行时)，默认为 CLASS，SOURCE 大都为 Mark Annotation，这类 Annotation 大都用来校验，比如 Override, SuppressWarnings

@Target 可以用来修饰哪些程序元素，如 TYPE, METHOD, CONSTRUCTOR, FIELD, PARAMETER 等，未标注则表示可修饰所有

@Inherited 是否可以被继承，默认为 false

5. Annotation 解析

5.1 运行时 Annotation 解析

(1) 运行时 Annotation 指 @Retention 为 RUNTIME 的 Annotation，可手动调用下面常用 API 解析

```
method.getAnnotation(AnnotationName.class);
method.getAnnotations();
method.isAnnotationPresent(AnnotationName.class);
```

其他 @Target 如 Field, Class 方法类似

getAnnotation(AnnotationName.class) 表示得到该 Target 某个 Annotation 的信息，因为一个 Target 可以被多个 Annotation 修饰

getAnnotations() 则表示得到该 Target 所有 Annotation

isAnnotationPresent(AnnotationName.class) 表示该 Target 是否被某个 Annotation 修饰

(2) 解析示例如下：

```
public static void main(String[] args) {
    try {
        Class cls = Class.forName("cn.trinea.java.test.annotation.App");
        for (Method method : cls.getMethods()) {
            MethodInfo methodInfo = method.getAnnotation(
                MethodInfo.class);
            if (methodInfo != null) {
                System.out.println("method name:" + method.getName());
                System.out.println("method author:" + methodInfo.author());
                System.out.println("method version:" + methodInfo.version());
                System.out.println("method date:" + methodInfo.date());
            }
        }
    } catch (ClassNotFoundException e) {
        e.printStackTrace();
    }
}
```

以之前自定义的 MethodInfo 为例，利用 Target (这里是 Method) getAnnotation 函数得到 Annotation 信息，然后就可以调用 Annotation 的方法得到响应属性值

5.2 编译时 Annotation 解析

(1) 编译时 Annotation 指 @Retention 为 CLASS 的 Annotation，由编译器自动解析。需要做的

- 自定义类集成自 AbstractProcessor

b. 重写其中的 process 函数

这块很多同学不理解，实际是编译器在编译时自动查找所有继承自 AbstractProcessor 的类，然后调用他们的 process 方法去处理

(2) 假设 MethodInfo 的 @Retention 为 CLASS，解析示例如下：

```
@SupportedAnnotationTypes({ "cn.trinea.java.test.annotation.MethodInfo" })
public class MethodInfoProcessor extends AbstractProcessor {

    @Override
    public boolean process(Set<? extends TypeElement> annotations, RoundEnvironment env) {
        HashMap<String, String> map = new HashMap<String, String>();
        for (TypeElement te : annotations) {
            for (Element element : env.getElementsAnnotatedWith(te)) {
                MethodInfo methodInfo = element.getAnnotation(MethodInfo.class);
                map.put(element.getEnclosingElement().toString(), methodInfo.author());
            }
        }
        return false;
    }
}
```

SupportedAnnotationTypes 表示这个 Processor 要处理的 Annotation 名字。

process 函数中参数 annotations 表示待处理的 Annotations，参数 env 表示当前或是之前的运行环境 process 函数返回值表示这组 annotations 是否被这个 Processor 接受，如果接受后续子 processor 不会再对这个 Annotations 进行处理

6. 几个 Android 开源库 Annotation 原理简析

6.1 Annotation — Retrofit

(1) 调用

```
@GET("/users/{username}")
User getUser(@Path("username") String username);
```

(2) 定义

```
@Documented
@Target(METHOD)
@Retention(RUNTIME)
@RestMethod("GET")
public @interface GET {
    String value();
}
```

从定义可看出 Retrofit 的 Get Annotation 是运行时 Annotation，并且只能用于修饰 Method

(3) 原理

```
private void parseMethodAnnotations() {
    for (Annotation methodAnnotation : method.getAnnotations()) {
        Class<? extends Annotation> annotationType = methodAnnotation.annotationType();
        ;

        RestMethod methodInfo = null;

        for (Annotation innerAnnotation : annotationType.getAnnotations()) {
            if (RestMethod.class == innerAnnotation.annotationType()) {
                methodInfo = (RestMethod) innerAnnotation;
                break;
            }
        }
        .....
    }
}
```

[RestMethodInfo.java](#) 的 parseMethodAnnotations 方法如上，会检查每个方法的每个 Annotation，看

是否被 RestMethod 这个 Annotation 修饰的 Annotation 修饰，这个有点绕，就是是否被 GET、DELETE、POST、PUT、HEAD、PATCH 这些 Annotation 修饰，然后得到 Annotation 信息，在对接接口进行动态代理时会掉用到这些 Annotation 信息从而完成调用。

Retrofit 原理涉及到[动态代理](#)，这里原理都只介绍 Annotation，具体原理分析请见[Android 开源项目实现原理解析](#)

6.2 Annotation — Butter Knife

(1) 调用

```
@InjectView(R.id.user)
EditText username;
```



0



(2) 定义

```
@Retention(CLASS)
@Target(FIELD)
public @interface InjectView {
    int value();
}
```



可看出 Butter Knife 的 InjectView Annotation 是编译时 Annotation，并且只能用于修饰属性

(3) 原理

```
@Override
public boolean process(Set<? extends TypeElement> elements, RoundEnvironment env)
{
    Map<TypeElement, ViewInjector> targetClassMap = findAndParseTargets(env);

    for (Map.Entry<TypeElement, ViewInjector> entry : targetClassMap.entrySet()) {
        TypeElement typeElement = entry.getKey();
        ViewInjector viewInjector = entry.getValue();

        try {
            JavaFileObject jfo = filer.createSourceFile(viewInjector.getFqcn(), typeElement);
            Writer writer = jfo.openWriter();
            writer.write(viewInjector.brewJava());
            writer.flush();
            writer.close();
        } catch (IOException e) {
            error(typeElement, "Unable to write injector for type %s: %s", typeElement, e.getMessage());
        }
    }

    return true;
}
```

[ButterKnifeProcessor.java](#) 的 process 方法如上，编译时，在此方法中过滤 InjectView 这个 Annotation 到 targetClassMap 后，会根据 targetClassMap 中元素生成不同的 class 文件到最终的 APK 中，然后在运行时调用 ButterKnife.inject(x) 函数时会到之前编译时生成的类中去找。这里原理都只介绍 Annotation，具体原理分析请见[Android 开源项目实现原理解析](#)

6.3 Annotation — ActiveAndroid

(1) 调用

```
@Column(name = "Name")
public String name;
```

(2) 定义

```
@Target(ElementType.FIELD)
@Retention(RetentionPolicy.RUNTIME)
public @interface Column {
    .....
}
```

可看出 ActiveAndroid 的 Column Annotation 是运行时 Annotation，并且只能用于修饰属性。
(3) 原理

```
Field idField = getIdField(type);
mColumnNames.put(idField, mIdName);

List<Field> fields = new LinkedList<Field>(ReflectionUtils.getDeclaredColumnFields(
type));
Collections.reverse(fields);

for (Field field : fields) {
    if (field.isAnnotationPresent(Column.class)) {
        final Column columnAnnotation = field.getAnnotation(Column.class);
        String columnName = columnAnnotation.name();
        if (TextUtils.isEmpty(columnName)) {
            columnName = field.getName();
        }

        mColumnNames.put(field, columnName);
    }
}
```




TableInfo.java 的构造函数如上，运行时，得到所有行信息并存储起来用来构件表信息。

- 上一篇 Android 开源项目源码解析 -->公共技术点之 View 绘制流程(三)
- 下一篇 Android 开源项目源码解析 -->公共技术点之 Java 动态代理(五)



您还没有登录,请[登录](#)或[注册](#)


公共技术点之 Java 注解 Annotation

 oHuijia1 2015年10月13日 10:42 251

不少开源库都用到了注解的方式来简化代码提高开发效率。本文简单介绍下 Annotation 示例、概念及作用、分类、自定义、解析，并对几个 Android 开源库 Annotation 原理进行简析...


Android 开源项目源码解析 -->公共技术点之依赖注入(二)

Android 开源项目源码解析 -->公共技术点之依赖注入(二)


 u014608640 2016年09月27日 11:36 231

开源项目源码解析-Java 注解 Annotation

Java 注解 Annotation 本文为 Android 开源项目源码解析 公共技术点中的 注解 部分分析者：Trinea，校对者：Trinea，校对状态：完成不少开源库都用...


 ljx19900116 2015年02月04日 09:43 990

Java Annotation及注解原理简析

 u012227846 2015年01月15日 10:41 4007

参考资料：http://www.trinea.cn/android/java-annotation-android-open-source-analysis/ ...


Android Annotation注解学习笔记

 lintcgirl 2016年09月07日 17:35 1111


今天讲下注解吧，现在遇到的用注解的开源库越来越多，虽然知道怎么用，但是其原理，怎么写都|楚。学习了一些网站和资料，先共享下我的学习资料：Java Annotation 及几个常用开源项目注解原...

Java Annotation 及几个常用开源项目注解原理简析

Java Annotation 及几个常用开源项目注解原理简析 http://www.trinea.cn/android/java-annotation-oid-open-so
urce-an...

 doomsj 2015年12月05日 23:47 511


Android公共技术点之一-Java注解

 zhangmiaoping23 2016年10月26日 23:52 181

转：http://yeungeek.com/2016/04/25/Android公共技术点之一-Java注解/ 基础是学习任何技术的必须，接下来会介绍一些Android上用到的一些公...


Android 开源项目源码解析 -->公共技术点之 Java 动态代理(五)

Android 开源项目源码解析 -->公共技术点之 Java 动态代理(五)


 u014608640 2016年09月27日 11:49 290

Android 开源项目源码解析 -->公共技术点之 Java 反射 Reflection(十六)

Android 开源项目源码解析 -->公共技术点之 Java 反射 Reflection(十六)

 u014608640 2016年09月28日 10:12 223

枚举和注解（Enum and Annotation）


 Laneruan 2017年08月15日 22:29 281

Java 1.5发行版本新增了两个引用类型家族：枚举类型（Enumerate类）和注解类型（Annotation接口）。...

福利！免费试听深度学习！
人工智能必备，迈出百万年薪第一步！




Android开源优秀项目源码

 u013986975 2017年07月17日 17:20 3454

BeautifulRefreshLayout-漂亮的美食下拉刷新https://github.com/android-cjj/BeautifulRefreshLayout/tree/Beautiful..
..

深入理解Java注解类型(@Annotation)

 javazejian 2017年05月21日 10:51 2871

【版权申明】未经博主同意，谢绝转载！（请尊重原创，博主保留追究权）http://blog.csdn.net/javazejian/artic
etails/71860633 出自【...

Android 开源项目源码解析 -->公共技术点之 Android 动画基础(十七)

Android 开源项目源码解析 -->公共技术点之 Android 动画基础(十七)


 u014608640

2016年09月28日 10:26


 254

Android —— 注解（ Annotation ） 也被称为元数据(Metadata)

之前博主讲xUtils的时候，介绍过注解，不过是蜻蜓点水，有兴趣的同学可以先移步到xUtils介绍2， 我们就来详细解剖一下Android中注解的使用。 Java注解是附加在代码中的一些元信息， ...


 womengmengyan

2016年09月18日 16:19


 1271

详解JDK 5 Annotation 注解之@Target的用法介绍

前言目前，越来越多的架构设计在使用注解，例如spring3.0、 struts2等框架。让我们先来看看注解的定义。如下是一段使用了JDK 5 Annotation @Target的代码： ...

 snakeMoving

2017年07月04日 23:44

 1918

Java注解（ Annotation ） 原理详解

注解在Java中到底是什么样的东西？具体是如何实现的？我想刚刚接触注解的时候大家都会有这个疑分析测试的代码：
@Target(ElementType.TYPE) @Retention(...



Unable to Connect to Site

Java Annotation注解继承方式说明


有关Annotation的继承说明： 1、JDK文档中的说明是：只有在类上应用的Annotation才能被继承，而实际应用时的结果是：除了类上应用的Annotation能被继承外，没有被重写的方法的...

java annotation(注解) 的优点缺点


注解本质上通过反射来实现的，我们都知道，反射是一种程序的自省机制，其实反射是破坏封装的一种方式，反射的效率很低的，对程序本身访问会造成很多的额外开销。比如你采用Spring注解，@resource标识...

Android注解（ Annotation ） 知识点总结整理

Java提供了Annotations来对代码提供一些注解，以解释、约束代码，也可以称为编程的元数据。到底什么是Annotation呢，举几个大家熟悉的例子如@Override（表示重载）、@Depre...


 clumsypanda

2016年05月18日 15:29


 4110

Android 开源项目源码解析 -->公共技术点之 View 事件传递(一)

Android 开源项目源码解析 -->公共技术点之 View 事件传递(一)

 u014608640

2016年09月27日 11:30

 191



