Java 自定义注解

作者：laker
原文链接：https://ld246.com/article/1535936247376
来源网站：链滴

定义注解类

```
import java.lang.annotation.ElementType;
import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;
import java.lang.annotation.Target;

@Retention(RetentionPolicy.RUNTIME) // 注解会在class字节码文件中存在，在运行时可以通过反
获取到
@Target({ElementType.FIELD}) //定义注解的作用目标**作用范围字段、枚举的常量/方法
public @interface FieldMeta {
    int len() default 10;//长度，默认为10。
}
```

定义类使用注解

```
public class BillItemHuawei {
    @FieldMeta(len=10)
    private int recordType;//    Integer    4    记录类型。    25：ismpRecord
    @FieldMeta(len=10)
    private int streamNo;//    Integer    4    流水号，由网关生成，最大值为4294967295。
    @FieldMeta(len=10)
    private String timeStamp;//    Octet String    10    时间戳，由发端设备生成，网关收到消息的
间。格式：月日时分秒。
    @FieldMeta(len=20)
    private String msgID;
}
```

利用注解

```
public String getLine(BillItemHuawei item) {
    Class c = item.getClass();
    Field fields[] = c.getDeclaredFields();
    StringBuffer sb = new StringBuffer();
    for(int i=0;i<fields.length;i++){
        sb.append(getPropertyValue(item, fields[i]));
        if(i!=fields.length-1){
            sb.append(PropertiesUtil.getProperties("fieldSeparator"));
        }
    }
    //System.out.println(sb.toString());
    return sb.toString();
}

public String getPropertyValue(Object owner, Field field) {
    FieldMeta meta;
    int len = 10;
    String formatStr = "";
    meta = field.getAnnotation(FieldMeta.class);
    if(meta != null){
        len = meta.len();
        if(field.getType().getName().equals("int")){
```

```
            formatStr = "%0" + len + "d";
        }else{
            formatStr = "%-" + len + "s";
        }
    }
    Object o = getPropertyValue(owner, field.getName());
    return String.format(formatStr, o);
}

public Object getPropertyValue(Object owner, String methodName) {
    Class ownerClass = owner.getClass();
    methodName = methodName.substring(0, 1).toUpperCase()
            + methodName.substring(1);
    Method method = null;
    try {
        method = ownerClass.getMethod("get" + methodName);
        Object o=method.invoke(owner);
        if(o==null)
            return "";
        return o;
    } catch (Exception e) {
        return " can't find 'get" + methodName + "' method";
    }
}
```

原文