Annotations

Annotation که به عنوان فراداده نیز شناخته شده است روشی رسمی برای اضافه کردن اطلاعات به کد را فراهم می کند و از این طریق می توان از این داده ها بعدا استفاده نمود.

annotationها

- Annotation تا حدودی با انگیزه ترکیبmetadata و فایل های‌فراداده یا توضیح درباره اطلاعات)source-code به، جای نگه داری آن ها در داکیومنت های جدا،ایجاد شده است .
- با استفاده از Annotation در جاوا،می توان metadataها را در سورس کد نگه داشت و از امکاناتی مانند زیر بهره برد:
  - ✓ فراهم کردن اطلاعات برای کامپایلر: توسط کامپایلر برای یافتن خطاها یا اعلام warning ها استفاده می شوند.
  - ✓ پردازش زمان کامپایل (Compile-time) یا زمان deploy: کتابخانه ها و ابزار ها از اطلاعات Annotation ها برای ایجاد کد، فایل های XML و ... استفاده می کنند.
  - ✓ پردازش زمان اجرا (runtime): بعضی از Annotation ها برای چک کردن در زمان اجرا کاربرد دارند.

annotationها

❑قواعد نوشتن annotationها منطقا ساده و عمدتا شامل علامت @ می باشد.در ورژن Java SE5 سه نوع کلی از annotationها وجود دارد که در تعریف شده اند:

✓@Override برای نشان دادن این که تعریف یک متد برای:override کردن متدی در کلاس پایه در نظر گرفته شده است و در صورتی که به صورت تصادفی املای نام متد را اشتباه نوشته باشید و یا یک امضای نامناسب برای متد مشخص کرده باشید،پیغام خطای کامپایل تولید خواهد کرد .

✓@Deprecated برای ایجاد:warning. های کامپایلر،در صورتی که المان استفاده شده باشد

✓@SuppressWarnings برای حذف و خاموش کردن:warning های نامناسب کامپایلر. استفاده از اینannotation مجاز است اما در ورژن های قبل ازJava SE5. پشتیبانی نمی شود

تعریف annotationها

```
package net.mindview.atunit;
import java.lang.annotation.*;

@Target(ElementType.METHOD)
@Retention(RetentionPolicy.RUNTIME)
public @interface Test {

}
```

❑ تعریف یک annotation نیازمند دو جز @Target و @Retention است.قسمت @Target مشخص می کند که در کجا می توان annotation را به کار برد(به عنوان مثال در یک متد یا یک فیلد) و قسمت @Retention مشخص می کند که Annotation چگونه باید ذخیره بشود؛در داخل یک سورس کد (**SOURCE**)،در داخل فایل های کلاس (**CLASS**) یا در زمان اجرا (**RUNTIME**).

تعریف annotationها

- Annotation ها معمولا شامل عناصری(elements) هستند که مقادیرAnnotation را مشخص می کنند و در هنگام پردازشAnnotationهایتان،برنامه یا ابزار مدنظر می تواند از این پارامترهای استفاده کنند.عناصر همانند متدهایinterface. هستند با این تفاوت که می توان مقادیر پیش فرضی را برای آن ها در نظر گرفت

```
import java.lang.annotation.*;

@Target(ElementType.METHOD)
@Retention(RetentionPolicy.RUNTIME)
public @interface UseCase{
   public int id();
   public String description() default "no description";
}
```

تعریف annotationها

```
import java.util.*;
public class PasswordUtils{
    @UseCase(id = 47,
       description = "Passwords must contain at least one numeric")
    public boolean validatePassword(String password){
        return (password.matches("\\w*\\d\\w*"));
    }
    @UseCase(id = 48)
    public String encryptPassword(String password){
        return new StringBuilder(password).reverse().toString();
    }
}
```

@Target	annotation مد نظر در کجا می تواند استفاده شود. آرگومان هایElementType ممکن برای آن عبارتند از : **CONSTRUCTOR** اعلان:CONSTRUCTOR **FIELD** شامل ثوابتاعلان فیلد(enum) **LOCAL_VARIABLE**اعلان متغیرهای محلی : **METHOD**اعلان متد: **PACKAGE** اعلان:Package **PARAMETER** اعلان:Parameter **TYPE**،اعلان کلاس:interface شامل نوع)annotation یا (enum
@Retention	اطلاعات annotation چه مدت نگه داری می شوند.آرگومان های RetentionPolicy ممکن برای آن عبارتند از: **SOURCE** در این حالت:Annotation فقط در سورس نگهداری می شود و به وسیله کامپایلر نادیده گرفته می شود . **CLASS** در زمان کامپایل نگهداری می شود ولی به وسیله:JVM نادیده گرفته می شود **RUNTIME** توسط:JVM نگهداری می شود ومی توان در Runtime. از آن استفاده کند
@Documented	هر زمان از این Annotation استفاده کردیم یعنی باید توسط Javadoc مستند سازی گردد.
@Inherited	به زیر کلاس ها اجازه می دهد تا از annotationهای پدر ارث ببرند.

نوشتن پردازش گرهای annotation

```
public static void trackUseCases(Class<?> cl){
        for(Method m : cl.getDeclaredMethods()){
            UseCase uc = m.getAnnotation(UseCase.class);
            if(uc != null){
                System.out.println("Found Use Case:" + uc.id() +
                                   " " + uc.description());
        }
        }
   }
public static void main(String[] args){
      trackUseCases(useCases, PasswordUtils.class);
}
```

عناصر Annotation

- همه نوع های اولیه (int, float, boolean و غیره)
- String
- Class
- Enumها
- Annotationها
- آرایه ای از هر کدام از موارد بالا