Annotationen

Annotationen

Annotation ist optionaler Parameter, der

- an (fast) beliebige Sprachkonzepte anheftbar ist,
- im Java-Code statisch gesetzt wird,
- von gesamter Werkzeugkette bis zur Laufzeit auslesbar ist.

```
@Override
public String toString() { ... }

@BugFix(who="Kaspar", date="2013-12-04", level=3,
        bug="class unnecessary and maybe harmful",
        fix="contents of class body removed")
public class Buggy { }
```

Definition von Annotationen

Syntax von Interfaces adaptiert

```
@Retention(RetentionPolicy.RUNTIME)
@Target({ElementType.TYPE})
public @interface BugFix {
    String who() default "me"; // author of bug fix
    String date();             // when was bug fixed
    int level() default 1;     // importance level 1-5
    String bug();              // description of bug
    String fix();              // description of fix
}
```

Einschränkungen in @interface

nur parameterlose Methoden

als Ergebnistypen nur erlaubt:

- alle elementaren Typen (`int`, `double`, ...)
- Aufzählungstypen (`enum`)
- `String`
- `Class`
- andere Annotationen
- sowie eindimensionale Arrays dieser Typen (als „Mengen“)

Methodenname `value` → Name in Annotation optional

Annotationen für Annotationsdefinition

```
@Retention(RUNTIME)                    // @Retention(value=RUNTIME)
@Target(value=ANNOTATION_TYPE)
public @interface Retention {
    RetentionPolicy value();
}
public enum RetentionPolicy { CLASS, RUNTIME, SOURCE }

@Retention(value=RUNTIME)
@Target(value=ANNOTATION_TYPE)
public @interface Target {
    ElementType[] value();
}
public enum ElementType { ANNOTATION_TYPE, CONSTRUCTOR,
    FIELD, LOCAL_VARIABLE, METHOD, PACKAGE, PARAMETER, TYPE
}
```

Annotationen zur Laufzeit

Falls `@Retention(RUNTIME)` wird echtes Interface erzeugt:

```
public interface BugFix
       extends java.lang.annotation.Annotation {
    String who();
    String date();
    int level();
    String bug();
    String fix();
}
```

Zugriff zur Laufzeit (Reflexion)

Annotationen über Reflection zur Laufzeit zugreifbar
(falls `@Retention(RUNTIME)`):

```
String s = "";
BugFix a = Buggy.class.getAnnotation(BugFix.class);
if (a != null) {  // null if no such Annotation
    s += a.who() + " fixed a level " + a.level() + " bug";
}

Annotation[] as = Buggy.class.getAnnotations();  // all
Method[] ms = Buggy.class.getMethods();
// verschiedene analoge Methoden auch auf Method, Field
```