

## Аннотации, как создавать

Java Collections  
8 уровень, 9 лекция

ОТКРЫТА

— Так вот, давай-ка сейчас создадим парочку аннотаций и используем их.

Например, мы пишем движок для игры. При этом у нас в игре очень много персонажей, которые делятся на три категории: эльфы, охрана дворца и злодей.

В процессе разработки игры, могут добавляться новые персонажи, и при этом будет меняться баланс игры. Поэтому было бы очень удобно приписать каждому «классу персонажа» свою аннотацию с описанием его физических характеристик.

Тогда можно было бы очень просто симулировать бои между различными персонажами и/или быстро просчитать баланс игры.

— [Согласен](#) – это хорошая идея.

— Давай создадим аннотацию **@Person**, где будем хранить: жизнь, силу, магию, а также параметры атаки и защиты. Вот как бы выглядела такая аннотация:

Пример	
1	<b>@interface</b> Person
2	{
3	String <b>name</b> () <b>default</b> "";
4	<b>int</b> <b>live</b> ();
5	<b>int</b> <b>strength</b> ();
6	<b>int</b> <b>magic</b> () <b>default</b> 0;
7	<b>int</b> <b>attack</b> () <b>default</b> 0;
8	<b>int</b> <b>defense</b> ();
9	}

А вот так выглядело бы описание, например, лесного эльфа-мага:

Пример	
1	<b>@Person</b> (live=100, strength=10, magic=5, attack=20, defense=20)
2	<b>class</b> Elf
3	{
4	...
5	}

А вот так выглядело бы описание главного злодея:

Пример	
1	<b>@Person</b> (live=1000, strength=150, magic=250, attack=99, defense=99)
2	<b>class</b> EvilMaster
3	{
4	...
5	}

— Да, только, во-первых, не приходится ничего наследовать, во вторых, в аннотациях можно хранить дополнительную информацию.

Есть еще несколько аннотаций, которыми помечаются аннотации. Вот они:

Аннотация `@Retention` указывает, где будет видна наша аннотация: только в исходном коде, еще и после компиляции, будет доступна даже во время исполнения программы.

Аннотация `@Target` указывает, что именно можно пометить этой аннотацией: класс, поле, метод, параметр метода и т.д.

Если мы хотим, чтобы наша аннотация действовала не только на отмеченный ей класс, но и на его наследников, то надо пометить ее `@Inherited`.

Вот как будет выглядеть наша аннотация `@Person`.

Пример	
1	<code>@Target(value=ElementType.TYPE)</code>
2	<code>@Retention(value=RetentionPolicy.RUNTIME)</code>
3	<code>@interface</code> Person
4	{
5	String name() default "";
6	int live();
7	int strength();
8	int magic() default 0;
9	int attack() default 0;
10	int defense();
11	}

— Это было очень интересно, спасибо, Риша.

А как работать с этими аннотациями в программе? Как их использовать? Как прочитать их значения?

— Для этого принято использовать **Reflection**.

Вот как выглядело бы определение того, какой из персонажей сильнее:

Пример	
1	<code>public boolean</code> fight(Class first, Class second)
2	{
3	<code>if</code> (!first.isAnnotationPresent(Person.class))
4	<code>throw new</code> RuntimeException("first param is not game person");
5	<code>if</code> (!second.isAnnotationPresent(Person.class))
6	<code>throw new</code> RuntimeException("second param is not game person");
7	
8	<code>Person</code> firstPerson = (Person) first.getAnnotation(Person.class);
9	<code>Person</code> secondPerson = (Person) second.getAnnotation(Person.class);
10	
11	<code>int</code> firstAttack = firstPerson.attack() * firstPerson.strength() + firstPerson.magic();
12	<code>int</code> firstPower = firstPerson.live() * firstPerson.defense() * firstAttack;
13	
14	<code>int</code> secondAttack = secondPerson.attack() * secondPerson.strength() + secondPerson.magic();
15	<code>int</code> secondPower = secondPerson.live() * secondPerson.defense() * secondAttack;
16	
17	<code>return</code> firstPower > secondPower;
18	}

Методы		Описание
1	<code>isAnnotationPresent(Annotation.class)</code>	Проверяет, если ли у класса нужная аннотация
1	<code>getAnnotation(Annotation.class)</code>	Возвращает объект-аннотацию, если такая у класса есть.
1	<code>Annotation[] getAnnotations()</code>	Возвращает массив всех аннотаций класса

— Отлично. А я и не думал, что получить аннотацию так просто.

— Ага. Просто вызвал метод `getAnnotation` у объекта класса, и передал туда нужный тебе тип аннотации.

На этом на сегодня все.

— Спасибо, Риша, это была очень интересная лекция. И теперь я уже не боюсь аннотаций как воды.

 +42 

Комментарии (36)

популярные

новые

старые

JavaCoder

Введите текст комментария

Igor Petrashevsky Уровень 47

28 августа, 18:05 

какая халтура

Ответить


 0 

Макс Дудин Уровень 40, Калининград, Россия

14 июля, 11:34 

а вот [ту](#) зачем это всё и как может использоваться..  
дальше есть одна задача где может понадобится

Ответить

 +5  

Andrey Karelin Уровень 41, Sumy, Украина

14 мая, 23:57 

И все-таки как-то не раскрыта тема.  
Много внимания уделяется области действия, значениям, как получить и проверить.  
Но в последней лекции и задачах Аннотациями, по сути, заменили структуру наследования классов (парент-чайлд, или абстрактный - реализация).  
И если в наследовании все понятно и, собственно, в поведении обычных объектов с полями и методами, то что дает то, что вместо класса, мы все поля засунули в аннотации и оставили классы пустыми? Зачем нужно именно так....какие практические преимущества это дает, какие задачи решает?  
  
P.S. честно говоря, объяснения материала на основе ассоциаций с эльфами, орками и подобными игровыми персонажами, лично мне, не "игроману", понятно с легким туманом. Уж лучше на примере автомобилей, пицерий, заводов и фабрик).

Ответить

 +2 

**LuneFox** Уровень 41, Москва, Россия EXPERT

10 марта, 14:34 

НАЧАТЬ ОБУЧЕНИЕ

А кому нужна такая аннотация, если вдруг какой-то эльф или подвид эльфа будет в программе в будущем невероятно сильным с параметрами больше 9000? Зачем считывать аннотацию, если можно просто получить какое-нибудь значение непосредственно из поля класса? Или это частный случай, когда параметры эльфа чётко определены и никогда не изменятся?

Ответить

−

+2

+

**Anton Stezhkin** Уровень 41 30 августа 2021, 17:49 

...

"в игре очень много персонажей, которые делятся на три категории: эльфы, охрана дворца и злодей" - Где-то вдалеке послышалось мычание корованов...

Ответить

−

+5

+

**Лунёв** Уровень 41, Москва, Россия EXPERT 10 марта, 10:59 

...

Я джва года ждал такой комментарий

Ответить

−

+2

+

**Алексей** Уровень 41, Чебоксары, Россия 1 февраля 2021, 15:22 

...

О мои глаза! Как это прочитать и понять?!

"Аннотация @Retention указывает, где будет видна наша аннотация: только в исходном коде, еще и после компиляции, будет доступна даже во время исполнения программ"

Ответить

−

+6

+

**Илья** Backend Developer в СберТех 18 февраля 2021, 16:35 

...

согласен, так себе написание

Ответить

−

0

+

**Anton Stezhkin** Уровень 41 30 августа 2021, 20:08 

...

Может тут лучше? [Руководство по аннотациям](#)

Ответить

−

+5

+

**Igor Petrashevsky** Уровень 47 28 августа, 18:04 

...

прям , очень хорошо. на 2 головы лучше JR

Ответить

−

0

+

**Dmitriy** Уровень 41 25 декабря 2020, 00:03 

...

К слову, у Class (этот тот резидент java.lang, который от Object наследуется) есть метод getDeclaredAnnotation. И еще несколько в продолжение этой же темы.

Ответить

−

+1

+

**Dmitriy** Уровень 41 24 декабря 2020, 23:40 

...

Про типы аннотаций Target, если интересно, очень коротко - <http://www.seostella.com/ru/article/2012/05/20/annotacii-v-java-target.html>.

Ответить

−

+3

+

**wan-derer.ru** Уровень 40, Москва, Россия 19 декабря 2020, 16:24 

...

Непонятно... Что это?

**Person firstPerson = (Person) first.getAnnotation(Person.class);**

Person это аннотация. Значит у нас получилась битва аннотаций?

Ответить

−

+2

+

**Dmitriy** Уровень 41 24 декабря 2020, 23:49 

...

Думаю, сначала для first берется из аннотаций и выполняется инициализация полей, затем - приведение к Person. Некая замена инициализации через конструктор или геттер.

Ответить

−

+1

+

**Сергей** Уровень 38 26 января 2021, 19:46 

...

Таким образом получают объект аннотации Person у класса, переданного через параметр first. Класс или другой объект могут быть аннотированы несколькими аннотациями.

Ответить

−

0

+

**Interstellar** Java Developer в EPAM EXPERT 17 августа 2020, 15:53 

...

Свойства, которые мы задаём в классе аннотации Person, это ведь по сути методы, а не поля, поскольку имеют скобки. Правильно я понимаю? А к методу после скобок мы ещё и прикручиваем слово default. Непривычно.

Ответить

−

0

+

↺ Показать еще комментарии

ОБУЧЕНИЕ

- Курсы программирования
- Курс Java
- Помощь по задачам
- Подписки
- Задачи-игры

СООБЩЕСТВО

- Пользователи
- Статьи
- Форум
- Чат
- Истории успеха
- Активности

КОМПАНИЯ

- О нас
- Контакты
- Отзывы
- FAQ
- Поддержка



JavaRush — это интерактивный онлайн-курс по изучению Java-программирования с нуля. Он содержит 1200 практических задач с проверкой решения в один клик, необходимый минимум теории по основам Java и мотивирующие фишки, которые помогут пройти курс до конца: игры, опросы, интересные проекты и статьи об эффективном обучении и карьере Java-девелопера.

ПОДПИСЫВАЙТЕСЬ

ЯЗЫК ИНТЕРФЕЙСА

 Русский

▼

СКАЧИВАЙТЕ НАШИ ПРИЛОЖЕНИЯ

