GENERICS

Generics

Обобщенные:

- типы (generic types)
  - классы (без enum, исключений, анонимных!)
  - интерфейсы (без аннотаций!)
- методы (generic methods)
- конструкторы (generic constructors)

Пример generics – Collection framework из состава Java core.

Предназначение

- Контейнеры
  - гетерогенные
  - гомогенные
- Параметризация
  - типов
  - методов
  - конструкторов

Преимущества

- Определение ошибок использования типов на этапе компиляции.
- Использование параметризованных
  - типов
  - методов
  - конструкторов
- Использование wildcards
  - с ограничениями
  - без ограничений

Терминология

Generic:

class A<T> {…} класс

public <T> A() {…} конструктор

public <T> T m(T t) {…} метод

T – параметр generic-а

Параметризация

Подстановка вместо параметра конкретного значения **при использовании** generic:

Параметризованный класс:

```
A<String> a = new A<String>();
```

Байт код параметризованных generic-ов

Один байт код на все варианты конкретных значений параметра.

Сырой тип (raw type)

Использование generic без значения параметра

Generic класс:

```
class A<T> {…}
```

Raw класс:

```
A a = new A();
```

Wildcard параметризованные типы

Использование generic с указанием wildcards:

? extends <== *extends* wildcard

? super <== *super* wildcard

? <== *unbounded* wildcard

Generic класс:

```
class A<T> {…}
```

Wildcard параметризованный класс:

```
A<?> a = new A<String>();
A<? extends Number> a = new A<Integer>();
A<? super Integer> a = new A<Number>();
```

Параметры generic с ограничениями

При объявлении можно наложить ограничения на значение параметра generic.

class A <T extends B & J1 & J2> {…}

J1, J2 - интерфейсы

При использовании класса A

- если B – интерфейс
  - T – интерфейс, обязан наследовать B, J1, J2,
- если B – класс
  - T – класс, обязан наследовать B, реализовывать J1, J2

Ограничения использования параметра generics

Внутри generic нельзя:

1) создать объект этого типа
2) создать массив такого типа
3) использовать в секции catch
4) использовать в статическом контексте
5) использовать в instanceof (справа)
6) наследовать

Замечание: информация о конкретном значении параметра во время выполнения **не доступна**.

Преобразования типов

- raw type ⇔ parameterized types
- parameterized types ⇔ parameterized types

Преобразования типов

Начиная с 7й версии JSE язык включает т. н. daemond оператор:

```
A<String> a = new A<>();
```

Компилятор автоматически подставит подходящее (исходя из контекста) значение параметра generic.

Массивы параметризованных generic

Объявить переменную типа массив параметризованных типов можно:

```
List<String>[] ar;
```

Создать массив нельзя:

```
ar = new ArrayList<String>[]; // compile time error!
```

Исключение (*допустимо только для параметризованного wildcard без ограничений*):

```
List<?>[] x = new ArrayList<?>[3]; // all ok
```