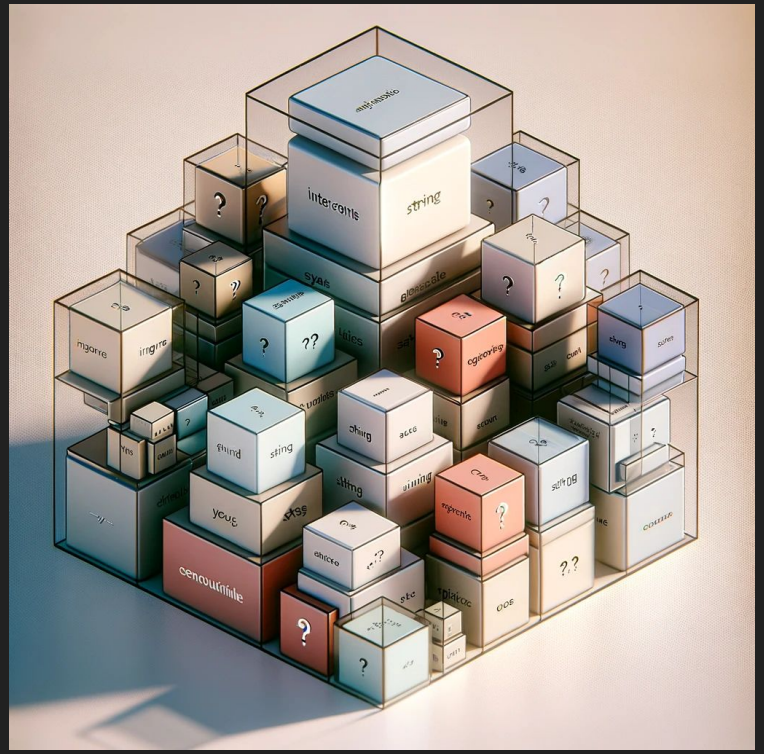
Generics

Generics в Java



Generics (обобщения) в Java позволяют определять классы, интерфейсы и методы с использованием типов-параметров.

Они обеспечивают строгую типизацию во время компиляции и повышают переиспользуемость кода.

Синтаксис Generics

```java
package l33.slides.ex1;

// Объявление обобщенного класса Box с
// типом-параметром T
class Box<T> {
    // Поле "content" типа T
    private T content;

    // Метод для установки значения поля "content"
    public void set(T content) {
        this.content = content;
    }

    // Метод для получения значения поля "content"
    public T get() {
        return content;
    }
}
```

Здесь Box<T> - обобщенный класс, где T - тип-параметр. Этот класс может быть использован с любым типом данных.

Принятые наименования (принято не значит что вы обязаны так называть):

- E - Element (used extensively by the Java Collections Framework)
- K - Key
- N - Number
- T - Type
- V - Value
- S,U,V etc. - 2nd, 3rd, 4th types

Примеры использования Generics

```
package l33.slides.ex1;

class Main {
   public static void main(String[] args) {
       // Создание экземпляра обобщенного
       // класса Box для String
       Box<String> stringBox =  new Box<>();
       // Установка значения
       stringBox.set("Hello Generics");
       // Выводит "Hello Generics"
       System.out.println(stringBox.get());
   }
}
```

В данном случае используем созданный класс как Box<String>, то есть параметризированный тип - String

Стирание типов

```java
package l33.slides.ex1;

class Main {
    public static void main(String[] args) {
        // Создание экземпляра обобщенного
        // класса Box для String
        Box<String> stringBox = new Box<>();
        // Установка значения
        stringBox.set("Hello Generics");
        // Выводит "Hello Generics"
        System.out.println(stringBox.get());
    }
}
```

Стирание типов - это процесс, при котором компилятор удаляет информацию о типах-параметрах во время компиляции и заменяет на Object и ограничения по типам.

То есть во время выполнения программы информации о generics нет как таковой, есть только ограничения по типам и преобразования там, где они нужны (процесс замены generics происходит во время компиляции).

Это делается для обеспечения совместимости с кодом, написанным до введения Generics в Java 5.

Обобщенные методы

```java
package l33.slides.ex2;

import l33.slides.ex1.Box;

// Объявление класса Util с обобщенным методом isEqual
class Util {
   // Обобщенный метод isEqual с типом-параметром T
   public static <T> boolean isEqual(Box<T> b1, Box<T> b2)
   {
       // Сравнение содержимого двух Box объектов
       return b1.get().equals(b2.get());
   }
}

class Main {
   public static void main(String[] args) {
       // Создание двух Box объектов для Integer
       Box<Integer> b1 = new Box<>();
       box1.set(10); // Установка значения для первого Box

       Box<Integer> b2 = new Box<>();
       box2.set(10); // Установка значения для второго Box

       // Вызов обобщенного метода isEqual для сравнения
       // двух Box объектов, возвращает true, если равны
       boolean isEqual = Util.<Integer>isEqual(b1, b2);
   }
}
```

Обобщенные методы или типизированные методы

Обобщенные методы позволяют указывать типы-параметры непосредственно в объявлении метода, делая методы более гибкими.

Метод isEqual может сравнивать содержимое двух Box объектов любого типа.

Оберточные типы

```java
package l33.slides.ex1;

class Main {
   public static void main(String[] args) {
       // Создание экземпляра обобщенного
       // класса Box для String
       Box<Integer> integerBox =  new Box<>();
       // Установка значения
       integerBox.set(38292893);
       // Выводит 38292893
       System.out.println(integerBox.get());
   }
}
```

Generics работают только с объектными типами, не с примитивными типами. Для работы с примитивными типами используются оберточные классы (Integer, Double, Character и т.д.).

Здесь примитивный тип int автоматически упаковывается в оберточный тип Integer, благодаря чему мы можем использовать int с обобщениями.

Контрольная точка

- Понятно ли?

Если все ясно, ставим плюсы, иначе - задаем вопросы.

Домашнее задание

Материалы:

- ссылка №1
- ссылка №2

Задачи минимум:

- почитать про generic wildcards в java
- написать свой Box и реализовать в нем не статические методы: T get(), void set(T data), boolean isValueEqual(Object value), static boolean isEqual(box1, box2)

The end