PROGRAMOWANIE OBIEKTOWE I GUI

dr inż. Michał Tomaszewski

katedra Metod Programowania
Polsko-Japońska Akademia Technik Komputerowych

WYKŁAD 03

Poruszyliśmy zagadnienia:

- Typy generyczne

WYKŁAD 03

Poruszyliśmy zagadnienia:

- Typy generyczne
- Wykorzystanie interfejsów

WYKŁAD 03

Poruszyliśmy zagadnienia:

- Typy generyczne
- Wykorzystanie interfejsów
- Typy wylicznikowe

Plan wykładu:

- Kolekcje

Plan wykładu:

- Kolekcje
- Strumienie

KOLEKCJE

Czym są struktury danych?

Czym są struktury danych?
jest to sposób przechowywania danych w pamięci komputera

STRUKTURY DANYCH

Czym są struktury danych?
jest to sposób przechowywania danych w pamięci komputera

Jakie struktury danych znamy?

Czym są struktury danych?
jest to sposób przechowywania danych w pamięci komputera

Jakie struktury danych znamy?

- tablice

Czym są struktury danych?
jest to sposób przechowywania danych w pamięci komputera

Jakie struktury danych znamy?

- tablice
- listy

Przykład

Czym są kolekcje?

Kolekcje to zestaw implementacji struktur danych

Czym są kolekcje?

Kolekcje to zestaw implementacji struktur danych oraz mechanizmów odwołania się do elementów z których się składają.

Czym są kolekcje?

Kolekcje to zestaw implementacji struktur danych oraz mechanizmów odwołania się do elementów z których się składają. Jakie metody dostępu przewidziano?

Czym są kolekcje?

Kolekcje to zestaw implementacji struktur danych oraz mechanizmów odwołania się do elementów z których się składają. Jakie metody dostępu przewidziano?

- list

Czym są kolekcje?

Kolekcje to zestaw implementacji struktur danych oraz mechanizmów odwołania się do elementów z których się składają. Jakie metody dostępu przewidziano?

- list
- zbiór

Czym są kolekcje?

Kolekcje to zestaw implementacji struktur danych oraz mechanizmów odwołania się do elementów z których się składają. Jakie metody dostępu przewidziano?

- list
- zbiór
- map

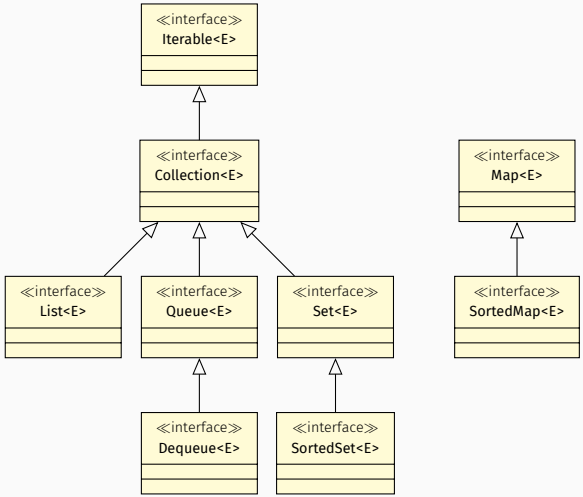
Czym są kolekcje?

Kolekcje to zestaw implementacji struktur danych oraz mechanizmów odwołania się do elementów z których się składają. Jakie metody dostępu przewidziano?

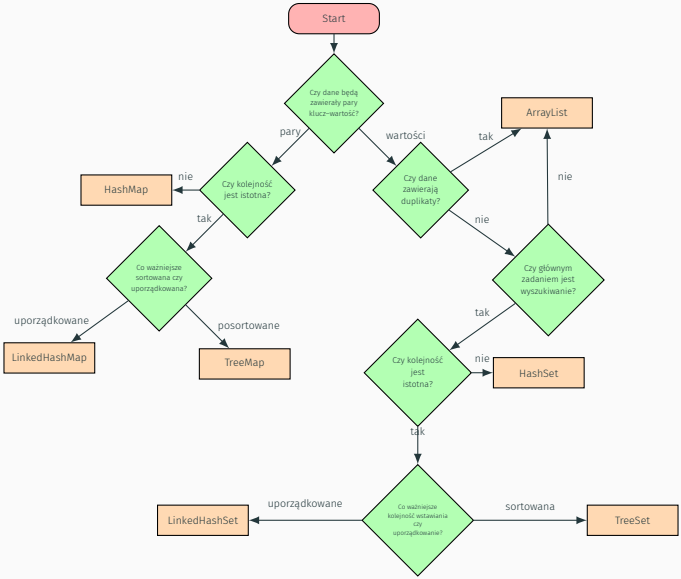
- list
- zbiór
- map
- kolejek

Przykład

HIERARCHIA MECHANIZMÓW DOSTĘPU



DOBÓR KOLEKCJI

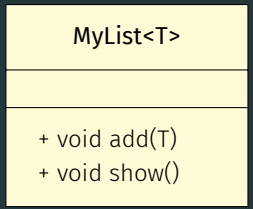


Iterator

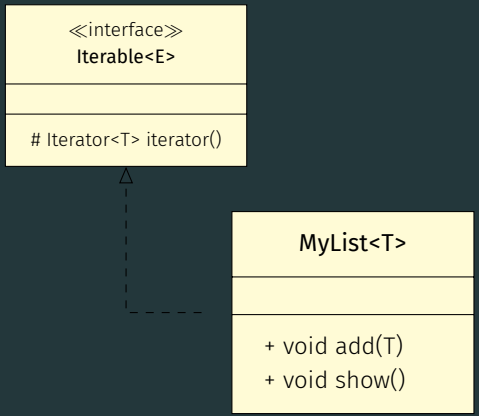
ITERATOR

Obiekt pozwalający na sekwencyjny dostęp do elementów zawartych w innym obiekcie.

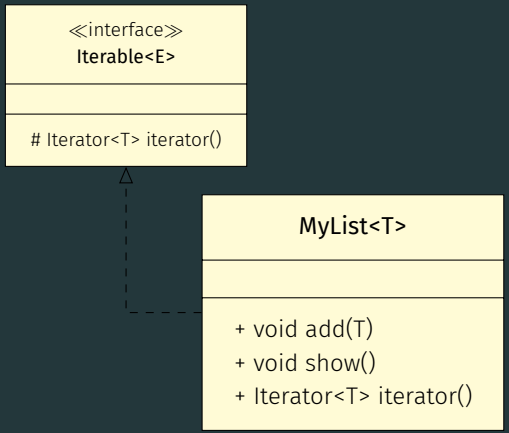
Przykład



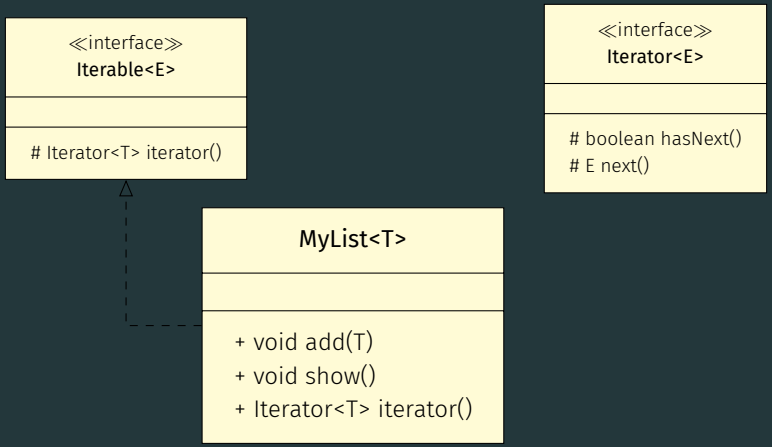
Przykład


≪interface≫
Iterable<E>
Iterator<T> iterator()
MyList<T>
+ void add(T)
+ void show()

Przykład


≪interface≫
Iterable<E>
Iterator<T> iterator()
MyList<T>
+ void add(T)
+ void show()
+ Iterator<T> iterator()

Przykład


≪interface≫
Iterable<E>
Iterator<T> iterator()
≪interface≫
Iterator<E>
boolean hasNext()
E next()
MyList<T>
+ void add(T)
+ void show()
+ Iterator<T> iterator()

WŁAŚCIWE PRZYGOTOWANIE KLAS

Teoretycznie w obiekcie danej kolekcji można umieścić dowolny obiekt, jednak może to doprowadzić do niepoprawnego funkcjonowania kolekcji.

WŁAŚCIWE PRZYGOTOWANIE KLAS

Teoretycznie w obiekcie danej kolekcji można umieścić dowolny obiekt, jednak może to doprowadzić do niepoprawnego funkcjonowania kolekcji. Dlatego aby klasa była akceptowalnym obiektem w kolekcji, musi spełniać kilka właściwości:

WŁAŚCIWE PRZYGOTOWANIE KLAS

Teoretycznie w obiekcie danej kolekcji można umieścić dowolny obiekt, jednak może to doprowadzić do niepoprawnego funkcjonowania kolekcji. Dlatego aby klasa była akceptowalnym obiektem w kolekcji, musi spełniać kilka właściwości:

- implementacja interfejsu `java.lang.Comparable` lub `java.util.Comparator`;

WŁAŚCIWE PRZYGOTOWANIE KLAS

Teoretycznie w obiekcie danej kolekcji można umieścić dowolny obiekt, jednak może to doprowadzić do niepoprawnego funkcjonowania kolekcji. Dlatego aby klasa była akceptowalnym obiektem w kolekcji, musi spełniać kilka właściwości:

- implementacja interfejsu `java.lang.Comparable` lub `java.util.Comparator`;
- implementacja interfejsu `java.io.Serializable`;

WŁAŚCIWE PRZYGOTOWANIE KLAS

Teoretycznie w obiekcie danej kolekcji można umieścić dowolny obiekt, jednak może to doprowadzić do niepoprawnego funkcjonowania kolekcji. Dlatego aby klasa była akceptowalnym obiektem w kolekcji, musi spełniać kilka właściwości:

- implementacja interfejsu `java.lang.Comparable` lub `java.util.Comparator`;
- implementacja interfejsu `java.io.Serializable`;
- definiowanie metody `equals`;

WŁAŚCIWE PRZYGOTOWANIE KLAS

Teoretycznie w obiekcie danej kolekcji można umieścić dowolny obiekt, jednak może to doprowadzić do niepoprawnego funkcjonowania kolekcji. Dlatego aby klasa była akceptowalnym obiektem w kolekcji, musi spełniać kilka właściwości:

- implementacja interfejsu `java.lang.Comparable` lub `java.util.Comparator`;
- implementacja interfejsu `java.io.Serializable`;
- definiowanie metody `equals`;
- definiowanie metody `hashCode`;

WŁAŚCIWE PRZYGOTOWANIE KLAS

Teoretycznie w obiekcie danej kolekcji można umieścić dowolny obiekt, jednak może to doprowadzić do niepoprawnego funkcjonowania kolekcji. Dlatego aby klasa była akceptowalnym obiektem w kolekcji, musi spełniać kilka właściwości:

- implementacja interfejsu `java.lang.Comparable` lub `java.util.Comparator`;
- implementacja interfejsu `java.io.Serializable`;
- definiowanie metody `equals`;
- definiowanie metody `hashCode`;
- implementacja konstruktora bezargumentowego;

WŁAŚCIWE PRZYGOTOWANIE KLAS

Teoretycznie w obiekcie danej kolekcji można umieścić dowolny obiekt, jednak może to doprowadzić do niepoprawnego funkcjonowania kolekcji. Dlatego aby klasa była akceptowalnym obiektem w kolekcji, musi spełniać kilka właściwości:

- implementacja interfejsu `java.lang.Comparable` lub `java.util.Comparator`;
- implementacja interfejsu `java.io.Serializable`;
- definiowanie metody `equals`;
- definiowanie metody `hashCode`;
- implementacja konstruktora bezargumentowego;
- implementacja interfejsu `java.lang.Iterable`;

STRUMIENIE

`Stream API` nie ma nic wspólnego z strumieniami plikowymi!

STRUMIENIE

`Stream API` nie ma nic wspólnego z strumieniami plikowymi!

Strumienie są tworem pochodzącym z programowania funkcyjnego.

`Stream API` nie ma nic wspólnego z strumieniami plikowymi!

Strumienie są tworem pochodzącym z programowania funkcyjnego.

Implementują wzorzec programistyczny Monad, który przedstawia mechanizm powiązania metod w ten sposób że rezultat jednej staje się wejściem drugiej.

Stream reprezentują zbiór danych, na którym wykonuje się sekwencje operacji rozpoczynających się od wypełnienia, przez szereg modyfikacji, a kończącą się na dostarczeniu wyniku.

Stream reprezentują zbiór danych, na którym wykonuje się sekwencje operacji rozpoczynających się od wypełnienia, przez szereg modyfikacji, a kończącą się na dostarczeniu wyniku.

Operacje na strumienach są:

Stream reprezentują zbiór danych, na którym wykonuje się sekwencje operacji rozpoczynających się od wypełnienia, przez szereg modyfikacji, a kończącą się na dostarczeniu wyniku.

Operacje na strumienach są:

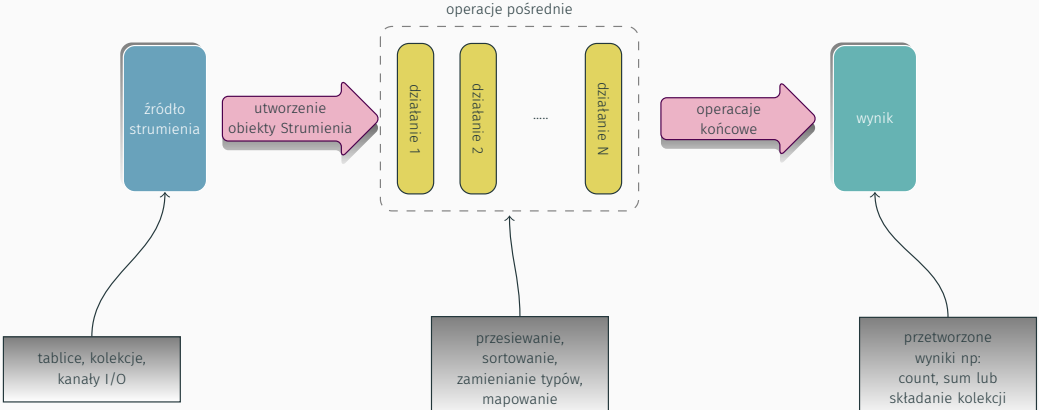
- pośrednie,

Stream reprezentują zbiór danych, na którym wykonuje się sekwencje operacji rozpoczynających się od wypełnienia, przez szereg modyfikacji, a kończącą się na dostarczeniu wyniku.

Operacje na strumienach są:

- pośrednie,
- końcowe.

Stream reprezentują zbiór danych, na którym wykonuje się sekwencje operacji rozpoczynających się od wypełnienia, przez szereg modyfikacji, a kończącą się na dostarczeniu wyniku.



WŁAŚCIWOŚCI

- Strumień nie przechowuje elementów na których działa.

WŁAŚCIWOŚCI

- Strumień nie przechowuje elementów na których działa.
- Operacje strumieniowe nie modyfikują swojego źródła.

WŁAŚCIWOŚCI

- Strumień nie przechowuje elementów na których działa.
- Operacje strumieniowe nie modyfikują swojego źródła.
- Operacje strumieniowe są „leniwe” (lazy) kiedy tylko jest to możliwe.

TWORZENIE STRUMIENI

Strumienie tworzy się:

TWORZENIE STRUMIENI

Strumienie tworzy się:

- na podstawie istniejących kolekcji – metoda `of`,

Strumienie tworzy się:

- na podstawie istniejących kolekcji – metoda `of`,

```
Stream<String> ryhme = Stream.of(
    "Ene", "due", "rabe",
    "chinczyk", "polknal" "zabe",
    "a", "zaba", "chinczyka",
    "co", "z", "tego", "wynika"
);
```

TWORZENIE STRUMIENI

Strumienie tworzy się:

- na podstawie istniejących kolekcji – metoda `of`,
- bez elementów – metoda `empty`,

Strumienie tworzy się:

- na podstawie istniejących kolekcji – metoda `of`,
- bez elementów – metoda `empty`,

```
Stream<Integer> ints = Stream.empty();
```

TWORZENIE STRUMIENI

Strumienie tworzy się:

- na podstawie istniejących kolekcji – metoda `of`,
- bez elementów – metoda `empty`,
- produkując zestaw elementów – metoda `generate`,
- produkując zakres elementów – metoda `iterate`.

TWORZENIE STRUMIENI

Strumienie tworzy się:

- na podstawie istniejących kolekcji – metoda `of`,
- bez elementów – metoda `empty`,
- produkując zestaw elementów – metoda `generate`,
- produkując zakres elementów – metoda `iterate`.

```
Stream<Double> doubles = Stream.generate(Math::random);
```

TWORZENIE STRUMIENI

Strumienie tworzy się:

- na podstawie istniejących kolekcji – metoda `of`,
- bez elementów – metoda `empty`,
- produkując zestaw elementów – metoda `generate`,
- produkując zakres elementów – metoda `iterate`.

TWORZENIE STRUMIENI

Strumienie tworzy się:

- na podstawie istniejących kolekcji – metoda `of`,
- bez elementów – metoda `empty`,
- produkując zestaw elementów – metoda `generate`,
- produkując zakres elementów – metoda `iterate`.

```
Stream<BigInteger> integers = Stream.iterate(
    BigInteger.ZERO, n->n.add(BigInteger.ONE)
);
```

Przykład

Strumienie można zawęzić:

Strumienie można zawęzić:

- filter,

Strumienie można zawęzić:

- filter,
- map,

Strumienie można zawęzić:

- filter,
- map,
- flatMap.

Przykład

ZBIERANIE WYNIKÓW

Aby zebrać efekt pracy poszczególnych operacji strumienia można użyć interfejsu Collector.

ZBIERANIE WYNIKÓW

Aby zebrać efekt pracy poszczególnych operacji strumienia można użyć interfejsu Collector.

- filter,

Aby zebrać efekt pracy poszczególnych operacji strumienia można użyć interfejsu Collector.

- filter,
- map,

ZBIERANIE WYNIKÓW

Aby zebrać efekt pracy poszczególnych operacji strumienia można użyć interfejsu Collector.

- filter,
- map,
- flatMap.

Przykład