Java Collections Framework

№ урока: 1 **Курс:** Java Advanced

Средства обучения: Компьютер с установленной IntelliJ IDEA.

Обзор, цель и назначение урока

На уроке рассматривается структура Java Collections Framework. Рассматривается применение таких интерфейсов и классов как:

- Comparator (компаратор);
- Iterator (итератор);
- Stack (стек);
- ArrayList (динамический массив);
- LinkedList (двусвязный список);
- Queue (очередь) на примере LinkedList;
- Map (карта) на примере HashMap (ключ-значение).

Изучив материал данного занятия, учащийся сможет:

- Понимать структуру и иерархию интерфейсов и классов Java Collections Framework.
- Работать с коллекциями.
- Создавать свои коллекции.

Содержание урока

1. Интерфейс Comparable. Создание своего компаратора.
2. Использование стандартной реализации Comparable. Отличия от Comparator.
3. Интерфейс Iterator. Создание и использование своего итератора. Стандартный итератор.
4. Создание и применение коллекции Stack.
5. Создание и применение коллекции ArrayList.
6. Создание и применение коллекции LinkedList.
7. Создание очереди на примере LinkedList.
8. Создание и применение карты на примере HashMap.

Резюме

- Java Collection Framework — иерархия интерфейсов и их реализаций, которая является частью JDK и позволяет разработчику пользоваться большим количеством структур данных.
- Интерфейсы Collection и Map разделяют все коллекции, входящие во фреймворк на две части по типу хранения данных: простые последовательные наборы элементов и наборы пар «ключ — значение».
- Collection - интерфейс находится в составе JDK с версии 1.2 и определяет основные методы работы с простыми наборами элементов, которые будут общими для всех его реализаций.
- Map - интерфейс находится в составе JDK с версии 1.2 и предоставляет разработчику базовые методы для работы с данными вида «ключ — значение».
- Компаратор (англ. Comparator – сравнивающее устройство) – принимает на вход два элемента и возвращает результат их сравнения (БОЛЬШЕ, МЕНЬШЕ или РАВНО).
- Компаратор используется для упорядочивания элементов коллекции.
- В интерфейсе Comparator объявлены методы compare(Object o1, Object o2) и equals(Object obj).
- Реализация метода compare(x, y) должен возвращать отрицательное значение если x < y, положительное если x > y и ноль если x = y.
- Реализация метода x.equals(y) возвращает TRUE если объекты эквивалентны. В противном случае возвращает FALSE.

WebCamp
Webcamp 2015
10 Darvina Str., 13 office
Kyiv, Ukraine
t. +380 (44) 235-6117
E-mail: info@webcamp.com.ua
Site: webcamp.com.ua

- В интерфейсе Comparable объявлен один метод compareTo(T o), который имеет туже логику работы что и Comparator.compare(). Типы сравниваемых объектов должны быть совместимы при сравнении.
- Итератор – объект, абстрагирующий за единым интерфейсом доступ к элементам коллекции.
- В интерфейсе Iterator объявлены методы hasNext() – проверяет наличие следующего элемента коллекции и next() – осуществляет переход на следующий элемент коллекции.

- Hashtable — реализация такой структуры данных, как хэш-таблица. Она не позволяет использовать null в качестве значения или ключа. Hashtable является синхронизированной (почти все методы помечены как synchronized). В большинстве случаев рекомендуется использовать другие реализации интерфейса Map ввиду отсутствия у них синхронизации.
- HashMap — коллекция является альтернативой Hashtable. Двумя основными отличиями от Hashtable являются то, что HashMap не синхронизирована и HashMap позволяет использовать null как в качестве ключа, так и значения. Так же как и Hashtable, данная коллекция не является упорядоченной: порядок хранения элементов зависит от хэш-функции. Добавление элемента выполняется за константное время O(1), но время удаления, получения зависит от распределения хэш-функции.
- LinkedHashMap — это упорядоченная реализация хэш-таблицы. Здесь, в отличии от HashMap, порядок итерирования равен порядку добавления элементов. Данная особенность достигается благодаря двунаправленным связям между элементами (аналогично LinkedList).
- TreeMap — реализация Map основанная на красно-чёрных деревьях. Как и LinkedHashMap является упорядоченной. По-умолчанию, коллекция сортируется по ключам с использованием принципа "natural ordering", но это поведение может быть настроено под конкретную задачу при помощи объекта Comparator, которые указывается в качестве параметра при создании объекта TreeMap.
- WeakHashMap — реализация хэш-таблицы, которая организована с использованием weak references. Другими словами, Garbage Collector автоматически удалит элемент из коллекции при следующей сборке мусора, если на ключ этого элеметна нет жёстких ссылок.

- Реализации интерфейса List – представляют собой упорядоченные коллекции. Кроме того, разработчику предоставляется возможность доступа к элементам коллекции по индексу и по значению.
- Vector — реализация динамического массива объектов. Позволяет хранить любые данные, включая null в качестве элемента. В Vector все операции с данными являются синхронизированными. В качестве альтернативы часто применяется аналог — ArrayList.
- Stack — является расширением коллекции Vector. Добавлена в Java 1.0 как реализация стека LIFO (last-in-first-out). Является частично синхронизированной коллекцией (кроме метода добавления push()). После добавления в Java 1.6 интерфейса Deque, рекомендуется использовать именно реализации этого интерфейса, например ArrayDeque.
- ArrayList — как и Vector является реализацией динамического массива объектов. Позволяет хранить любые данные, включая null в качестве элемента. Реализация основана на обычном массиве. Данную реализацию следует применять, если в процессе работы с коллекцией предплагается частое обращение к элементам по индексу. Из-за особенностей реализации поиндексное обращение к элементам выполняется за константное время O(1).
- LinkedList — позволяет хранить любые данные, включая null. Особенностью реализации данной коллекции является то, что в её основе лежит двунаправленный связный список (каждый элемент имеет ссылку на предыдущий и следующий). Благодаря этому, добавление и удаление из середины, доступ по индексу, значению происходит за линейное время O(n), а из начала и конца за константное O(1). Так же, ввиду реализации, данную коллекцию можно использовать как стек или очередь.

- Реализации интерфейса Set представляют собой неупорядоченную коллекцию, которая не может содержать дублирующиеся данные. Является программной моделью математического понятия «множество».

Webcamp 2015
10 Darvina Str., 13 office
Kyiv, Ukraine

t. +380 (44) 235-6117
E-mail: info@webcamp.com.ua
Site: webcamp.com.ua

- HashSet — реализация интерфейса Set, базирующаяся на HashMap. Внутри использует объект HashMap для хранения данных. В качестве ключа используется добавляемый элемент, а в качестве значения — объект-пустышка (new Object()). Из-за особенностей реализации порядок элементов не гарантируется при добавлении.
- LinkedHashSet — отличается от HashSet только тем, что в основе лежит LinkedHashMap вместо HashSet. Благодаря этому отличию порядок элементов при обходе коллекции является идентичным порядку добавления элементов.
- TreeSet — содержит в себе объект NavigableMap, что и обуславливает его поведение. Предоставляет возможность управлять порядком элементов в коллекции при помощи объекта Comparator, либо сохраняет элементы с использованием "natural ordering".

- Реализации интерфейса Queue описывает коллекции с предопределённым способом вставки и извлечения элементов, а именно — очереди FIFO (first-in-first-out). Помимо методов, определённых в интерфейсе Collection, определяет дополнительные методы для извлечения и добавления элементов в очередь.
- PriorityQueue — является единственной прямой реализацией интерфейса Queue (была добавлена, как и интерфейс Queue, в Java 1.5), не считая класса LinkedList, который так же реализует этот интерфейс, но был реализован намного раньше. Особенностью данной очереди является возможность управления порядком элементов. По-умолчанию, элементы сортируются с использованием «natural ordering», но это поведение может быть переопределено при помощи объекта Comparator, который задаётся при создании очереди. Данная коллекция не поддерживает null в качестве элементов.
- ArrayDeque — реализация интерфейса Deque, который расширяет интерфейс Queue методами, позволяющими реализовать конструкцию вида LIFO (last-in-first-out). Интерфейс Deque и реализация ArrayDeque были добавлены в Java 1.6. Эта коллекция представляет собой реализацию с использованием массивов, подобно ArrayList, но не позволяет обращаться к элементам по индексу и хранение null. Как заявлено в документации, коллекция работает быстрее чем Stack, если используется как LIFO коллекция, а также быстрее чем LinkedList, если используется как FIFO.

Закрепление материала

- Какой интерфейс не является простым последовательным набором элементов, но при этом входит в пакет java.util.Collections ?
- Какой интерфейс представляет коллекцию, которая исключает дублирование элементов?
- Какой интерфейс стоит во главе иерархии коллекций?
- Какой интерфейс представляет коллекцию которая может содержать дублирующие элементы?
- Какой интерфейс представляет двунаправленную очередь?

Дополнительное задание

Используя класс HashMap, создайте небольшую коллекцию и выведите на экран значения пар «ключ-значение» сначала в алфавитном порядке, а затем в обратном. ***

Самостоятельная деятельность учащегося

Задание 1
Выучите основные конструкции и понятия, рассмотренные на уроке.

Задание 2
Создайте коллекцию, в которую можно добавлять покупателей и категорию приобретенной ими продукции. Из коллекции можно получать категории товаров, которые купил покупатель или по категории определить покупателей.

WebCamp
Webcamp 2015
10 Darvina Str., 13 office
Kyiv, Ukraine
t. +380 (44) 235-6117
E-mail: info@webcamp.com.ua
Site: webcamp.com.ua

Задание 3
Несколькими способами создайте коллекцию, в которой можно хранить только целочисленные и вещественные значения, по типу «счет предприятия – доступная сумма» соответственно.

Рекомендуемые ресурсы

Java.util package:
http://docs.oracle.com/javase/7/docs/api/index.html?java/util/Comparator.html

Java Collections Tutorial:
https://docs.oracle.com/javase/tutorial/collections/

WebCamp

Webcamp 2015
10 Darvina Str., 13 office
Kyiv, Ukraine

t. +380 (44) 235-6117
E-mail: info@webcamp.com.ua
Site: webcamp.com.ua