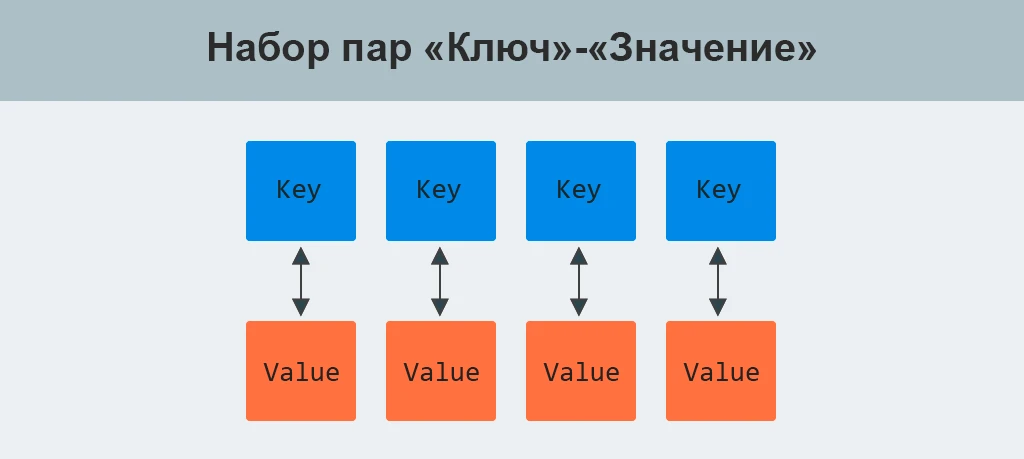
**Список всех коллекций**

[Java Syntax](https://javarush.ru/quests/QUEST_JAVA_SYNTAX)

[Уровень 8](https://javarush.ru/quests/lectures?quest=QUEST_JAVA_SYNTAX&level=8), Лекция 2



— Привет, Амиго.

— Здорова, Риша.

— Элли мне сказала, что ты хотел больше примеров коллекций. Сейчас я дам тебе несколько. Хочу представить тебе список коллекций и интерфейсов:

|  |  |  |
| --- | --- | --- |
| **Интерфейс** | **Класс/Реализация** | **Описание** |
| List | ArrayList | Список |
| LinkedList | Список |
| Vector | Вектор |
| Stack | Стек |
| Set | HashSet | Множество |
| TreeSet | Множество |
| SortedSet | Отсортированное множество |
| Map | HashMap | Карта/Словарь |
| TreeMap | Карта/Словарь |
| SortedMap | Отсортированный словарь |
| Hashtable | Хеш-таблица |

— Хм. Как их много. Четыре List’а, три Set’а и четыре Map’а.

— Да, это все различные реализации интерфейсов List, Set и Map.

— А чем отличаются различные реализации?

— Об этом как раз сегодня мы тебе и расскажем. Просто подожди немного.

— Может, у тебя уже накопились какие-нибудь вопросы?

— Как вывести список на экран я знаю. А как вывести Set и Map?

— У элементов списка (List) есть четко заданный порядок, поэтому их можно вывести просто по номеру. У Set и Map строго заданного порядка элементов нет. Собственно говоря, порядок их элементов может меняться при удалении или добавлении какого-нибудь элемента.

— Надо же.

— Поэтому для работы с элементами коллекций были придуманы специальные объекты – **итераторы**. С их помощью можно пройтись по всем элементам коллекции, даже если у них нет номера, а только имена (Map), или вообще нет имён – Set.

**-Примеры:**

Вывод на экран элементов Set

public static void main(String[] args)

{

**Set**<String> set = new **HashSet**<String>();

set.add("Mama");

set.add("Mila");

set.add("Ramu");

//получение итератора для множества

**Iterator**<String> iterator = set.iterator();

**while** (iterator.hasNext()) //проверка, есть ли ещё элементы

{

//получение текущего элемента и переход на следующий

String text = iterator.next();

System.out.println(text);

}

}

2

Задача

Java Syntax,  8 уровень,  2 лекция

Набираем код Ӏ Java Syntax: 8 уровень, 2 лекция

Java Syntax: 8 уровень, 2 лекция. Иногда думать не надо, строчить надо! Как ни парадоксально звучит, порой пальцы «запоминают» лучше, чем сознание. Вот почему во время обучения в секретном центре JavaRush вы иногда встречаете задания на набор кода. Набирая код, вы привыкаете к синтаксису и зарабатываете немного материи. А ещё — боретесь с ленью.

Вывод на экран элементов List

public static void **main**(String[] args)

{

**List**<String> list = new **ArrayList**<String>();

list.add("Mama");

list.add("Mila");

list.add("Ramu");

**Iterator**<String> iterator = list.iterator();//получение итератора для списка

while (iterator.hasNext()) //проверка, есть ли ещё элементы

{

//получение текущего элемента и переход на следующий

String text = iterator.next();

System.out.println(text);

}

}

Вывод на экран элементов Map

public static void main(String[] args)

{

//все элементы хранятся в парах

**Map**<String, String> map = new **HashMap**<String, String>();

map.put("first", "Mama");

map.put("second", "Mila");

map.put("third", "Ramu");

**Iterator**<Map.Entry<String, String>> iterator = map.entrySet().iterator();

**while** (iterator.hasNext())

{

//получение «пары» элементов

Map.Entry<String, String> pair = iterator.next();

String **key** = pair.getKey(); //ключ

String **value** = pair.getValue(); //значение

System.out.println(**key** + ":" + **value**);

}

}

— Ничего себе! И что все это значит?

— На самом деле все довольно просто. Сначала мы получаем у коллекции специальный объект-iterator. У него есть всего два метода

1. Метод next() возвращает очередной элемент коллекции.

2. Метод hasNext() проверяет, есть ли еще элементы, которые не возвращал next().

— Ага. Похоже ситуация начинает проясняться. Давай я расскажу, как я все это понял.

— Так. Сначала надо вызвать у коллекции метод iterator(), чтобы получить этот волшебный объект-итератор.

— Затем мы в цикле, пока есть еще неполученные элементы, получаем их по одному. Получаем мы очередной элемент коллекции с помощью вызова next(), а проверяем, есть ли еще элементы в итераторе с помощью hasNext(). Все верно?

— Да, примерно все так и есть. Но самое интересное будет сейчас.

— В Java есть сокращённая запись работы с итераторами. По аналогии с **while**, в **for** был добавлен еще один специальный оператор «**for each**» — «**для каждого**». Обозначается тоже ключевым словом **for**.

— Оператор for each используется только при работе с коллекциями и контейнерами. В нем неявно используется итератор, но мы видим уже полученный элемент.

— Давай я покажу тебе длинную и сокращенную работу с итератором:

Длинная запись

public static void main(String[] args)

{

**Set**<String> set = new **HashSet**<String>();

set.add("Mama");

set.add("Mila");

set.add("Ramu");

**Iterator**<String> iterator = set.iterator();

**while** (iterator.hasNext())

{

String text = iterator.next();

System.out.println(text);

}

}

Сокращенная запись

public static void main(String[] args)

{

**Set**<String> set = new **HashSet**<String>();

set.add("Mama");

set.add("Mila");

set.add("Ramu");

**for** (String text : set)

{

System.out.println(text);

}

}

— Обрати внимание: в правой нижней таблице нет ни зелёных, ни красных слов. Фактически 3 строки заменяются на одну:

Длинная запись

**Iterator**<String> iterator = set.iterator();

**while** (iterator.hasNext())

{

String text = iterator.next();

Сокращенная запись

**for** (String text : set)

— Выглядит шикарно. Так мне нравится гораздо больше!

— Давай посмотрим на те же примеры, что и выше, только записанные короче:

Вывод на экран элементов Set

public static void main(String[] args)

{

**Set**<String> set = new **HashSet**<String>();

set.add("Mama");

set.add("Mila");

set.add("Ramu");

**for** (String text : set)

{

System.out.println(text);

}

}

Вывод на экран элементов List

public static void main(String[] args)

{

**List**<String> list = new **ArrayList**<String>();

list.add("Mama");

list.add("Mila");

list.add("Ramu");

**for** (String text : list)

{

System.out.println(text);

}

}

Вывод на экран элементов Map

public static void main(String[] args)

{

**Map**<String, String> map = new **HashMap**<String, String>();

map.put("first", "Mama");

map.put("second", "Mila");

map.put("third", "Ramu");

**for** (Map.Entry<String, String> pair : map.entrySet())

{

String **key** = pair.getKey(); //ключ

String **value** = pair.getValue(); //значение

System.out.println(**key** + ":" + **value**);

}

}

— Так это же другое дело!

— Рад, что тебе понравилось.