**II. Краткий обзор сторонних библиотек коллекций**

Итак, Я хотел бы сделать обзор следующих сторонних библиотек: guava, apache, trove и gs-collections. Почему именно эти библиотеки? Guava и Apache Commons Collections очень популярны и встречались мне почти в любом Java проекте, Trove — тоже очень популярная библиотека, когда нужно уменьшить память и улучшить производительность работы с коллекциями. GS-collections — судя по оценкам весьма популярная библиотека на github'e (>1300 звезд), больше неё набрала «звезд» только guava. Так же мельком захвачу несколько других популярных библиотек.   
  
Итак, для начала рассмотрим что предлагают различные библиотеки, их главные фишки (Предупреждение: это очень субъективно, для кого-то главными фишками будут совсем другие возможности библиотек).

**2.1 Фишки разных библиотек коллекций**

Давайте сделаем небольшой обзор главных фишек (на мой взгляд) разных библиотек коллекций:  
1) **Guava** — данная коллекция от гугл практически самая популярная после стандартного фреймворка коллекций, она  
добавляет ряд интересных коллекций, но самая главная «фишка» это скорее богатый набор классов-утилит со статическими  
методами, расширяющими возможности Collections для работы со стандартными коллекциями, чем новые виды коллекций.  
Стандартные коллекции она практически не заменяет.  
  
2) **Apache Commons Collections** — данная коллекция ближайший «конкурент» guava, она так же предоставляет ряд  
интересных коллекций, утилит по работе со стандартными коллекциями Java, а так же большое количество wrapper'ов для  
изменения поведения коллекций. Кроме того она предоставляет свою реализацию map'ы с более простым механизм  
итерирования по ней.  
  
3) **Trove** — фишка данной коллекции в первую очередь в производительности и сокращении памяти, поэтому она  
предлагает более быстрые реализации стандартных коллекций (и требующие меньше памяти), а так же коллекции  
примитивных типов. *(уступает следующим двум - кол-во багов, полнота покрытия интерфейсов, производительность, активность поддержки, и т. д.)*  
  
4) **GS-collections** — фишка данной коллекции в идее объединить методы обработки, такие как сортировка и классы  
коллекций для создания замены использования статических методов классов-утилит. Данная библиотека предлагает замену  
практически всех стандартным коллекциям и добавляет несколько новых.

5) **Fastutil коллекции** - коллекция для работы с примитивными типами.

**III. Альтернативные виды коллекций в разных библиотеках**

Тут я попробую кратко рассмотреть, какие новые интересные виды коллекций можно найти в разных библиотеках:

**3.1 Альтернативные виды коллекций у Guava**

**Официальная информация:** [документация](https://github.com/google/guava/wiki), [исходные коды](https://github.com/google/guava), [javadoc](http://google.github.io/guava/releases/snapshot/api/docs/overview-summary.html).  
Как подключить к проекту:

Maven, Gradle

Maven

<dependency>

<groupId>com.google.guava</groupId>

<artifactId>guava</artifactId>

<version>18.0</version>

</dependency>

Gradle

dependencies {

compile 'com.google.guava:guava:18.0'

}

Гугл разработал ряд интересных дополнений к существующим коллекциям, которые весьма и весьма полезны, если вы можете  
использовать guava библиотеку в своем проекте. Можно сказать, что эти коллекции давно уже стали стандартом де-факто  
для большинства Java проектов, поэтому для любого опытного разработчика важно знать их, даже если он по каким-то  
причинам не может использовать их в своих проектах, зачастую на собеседованиях можно услышать вопросы по guava  
коллекциям.  
  
Давайте рассмотрим их поподробнее. Для начала рассмотрим интерфейсы основных коллекций и группы классов в guava.  
  
**Внимание**: если таблица не помещается целиком, попробуйте уменьшить масштаб страницы или открыть в другом браузере.

|  |  |  |  |
| --- | --- | --- | --- |
| **Название** | **Описание** | **Примеры реализаций** | **Примеры использования** |
| [**ImmutableCollection**](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ImmutableCollection.html) [**ImmutableList**](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ImmutableList.html) [**ImmutableSet**](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ImmutableSet.html) … и т.д. | Хотя в стандартном фреймворке Java коллекций есть возможность сделать коллекцию неизменяемой вызвав Collections.unmodifiableCollection (unmodifiableList или unmodifiableMap), но этот подход не самый оптимальный, так как отдельный тип для неизменяемых коллекций позволяет быть уверенным, что это коллекция действительно неизменяемая, вместо ошибок времени исполнения, при попытке изменить коллекцию, будут ошибки во время компиляции проекта, к тому же в стандартном фреймворке коллекций Java неизменяемые коллекции по-прежнему тратят ресурсы на поддержку синхронизации при многопоточном чтении и т. п. операциях, в то время как ImmutableCollection guava «знают» что они неизменяемые и оптимизированы с учетом этого. | JDK: [ImmutableCollection](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ImmutableCollection.html), [ImmutableList](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ImmutableList.html), [ImmutableSet](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ImmutableSet.html), [ImmutableSortedSet](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ImmutableSortedSet.html), [ImmutableMap](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ImmutableMap.html), [ImmutableSortedMap](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ImmutableSortedMap.html) Guava: [ImmutableMultiset](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ImmutableMultiset.html), [ImmutableSortedMultiset](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ImmutableSortedMultiset.html), [ImmutableMultimap](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ImmutableMultimap.html), [ImmutableListMultimap](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ImmutableListMultimap.html), [ImmutableSetMultimap](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ImmutableSetMultimap.html), [ImmutableBiMap](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ImmutableBiMap.html), [ImmutableClassToInstanceMap](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ImmutableClassToInstanceMap.html), [ImmutableTable](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ImmutableTable.html) | — если публичный метод возвращает коллекцию, которую гарантировано не должны менять другие классы, — если известно что значения коллекции больше никогда не должны меняться |
| [**Multiset**](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/Multiset.html) | Коллекция аналогичная Set, но позволяющая дополнительно считать количество добавлений элемента. Очень полезна для тех задач, когда нужно не только знать есть ли данный элемент в данном множестве, но и посчитать их количество (самый простой пример подсчет количества упоминаний тех или иных слов в каком-либо тексте). То есть данная коллекция более удобный вариант коллекции Map<T, Integer>, с методами специально предназначенными для подобных коллекций, позволяет очень сильно сократить количество лишнего кода в таких случаях. | [HashMultiset](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/HashMultiset.html), [TreeMultiset](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/TreeMultiset.html), [LinkedHashMultiset](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/LinkedHashMultiset.html), [ConcurrentHashMultiset](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ConcurrentHashMultiset.html), [ImmutableMultiset](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ImmutableMultiset.html) [SortedMultiset](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/SortedMultiset.html) | — подсчет кол-ва вхождений слов в тексте — подсчет кол-ва букв в тексте — подсчет кол-ва любых объектов |
| [**Multimap**](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/Multimap.html) | Практически любой опытный Java разработчик сталкивался с необходимостью использовать структуры вроде Map<K, List<V>> или Map<K,Set<V>>, при этом приходилось писать много лишнего кода, для упрощения работы в библиотеку guava были введены Multimap, то есть коллекции, позволяющие просто работать со случаями когда один ключ и много значений у этого ключа. В отличии от конструкций вроде Map<K,Set<V>>, Multimap предоставляет ряд удобных функций для сокращения кода и упрощения алгоритмов. | [ArrayListMultimap](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ArrayListMultimap.html), [HashMultimap](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/HashMultimap.html), [LinkedListMultimap](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/LinkedListMultimap.html), [LinkedHashMultimap](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/LinkedHashMultimap.html), [TreeMultimap](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/TreeMultimap.html), [ImmutableListMultimap](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ImmutableListMultimap.html), [ImmutableSetMultimap](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ImmutableSetMultimap.html) | — реализация отношений один ко многим, таких как: учитель — ученики отдел — работники начальник — подчиненные |
| [**BiMap**](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/BiMap.html) | Достаточно часто встречаются ситуации, когда требуется создать Map'у работающую в обе стороны, то есть когда ключ и значение могут меняться местами (например, русско-английский словарь, когда в одном случае требуется получить по русскому слову — английское, в другом наоброт по английскому-русское). Обычно, это решается созданием двух Map, где в одной будет ключ1-ключ2, в другой ключ2-ключ1). BiMap позволяет решить эту задачу с помощью лишь одной коллекции. К тому же это исключает проблемы и ошибки синхронизации при использовании двух коллекций. | [HashBiMap](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/HashBiMap.html), [ImmutableBiMap](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ImmutableBiMap.html), [EnumBiMap](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/EnumBiMap.html), [EnumHashBiMap](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/EnumHashBiMap.html) | — словарь для перевода с одного языка в другой и обратно, — любая конвертация данных в обе стороны, |
| [**Table**](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/Table.html) | Эта коллекция служит для замены коллекций вида Map<FirstName, Map<LastName, Person>>, которые неудобны в использовании. | [HashBasedTable](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/HashBasedTable.html), [TreeBasedTable](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/TreeBasedTable.html), [ImmutableTable](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ImmutableTable.html), [ArrayTable](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ArrayTable.html) | — таблица, например, как в Excel — любые сложные структуры данных с большим количеством столбцов, |
| [**ClassToInstanceMap**](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ClassToInstanceMap.html) | Иногда нужно хранить в Map'e не ключ-значение, а тип-значение этого типа, для этого служит данная коллекция. То есть технически это более удобный и безопасный аналог Map <Class <? extends B>, B> | [MutableClassToInstanceMap](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/MutableClassToInstanceMap.html), [ImmutableClassToInstanceMap](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ImmutableClassToInstanceMap.html). |  |
| [**RangeSet**](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/RangeSet.html) | Коллекция для хранения разных открытых и закрытых отрезков числовых значений, при этом отрезки могут объединятся с друг другом. | [ImmutableRangeSet](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ImmutableRangeSet.html), [TreeRangeSet](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/TreeRangeSet.html) | Геометрические отрезки Временные отрезки |
| [**RangeMap**](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/RangeMap.html) | Коллекция, похожая на RangeSet, но при этом отрезки никогда не объединяются друг с другом. | [ImmutableRangeMap](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/ImmutableRangeMap.html), [TreeRangeMap](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/collect/TreeRangeMap.html) | Геометрические отрезки Временные отрезки |
| [**LoadingCache**](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/cache/LoadingCache.html) | Коллекция, похожая на ConcurrentMap, но при этом можно указать время какое будет хранится каждый элемент. Очень удобная коллекция для организации кэшей, подсчета количества ввода ошибочных паролей за какой-то промежуток времени и т.п. задач | [ForwardingLoadingCache](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/cache/ForwardingLoadingCache.html), [ForwardingLoadingCache.SimpleForwardingLoadingCache](http://docs.guava-libraries.googlecode.com/git/javadoc/com/google/common/cache/ForwardingLoadingCache.SimpleForwardingLoadingCache.html) | кэши, хранение ошибочных попыток ввода пароля и т.п. |

**3.2 Новые виды коллекций из Apache Commons Collections**

**Официальная информация**: [документация](http://commons.apache.org/proper/commons-collections/), [исходные коды](http://commons.apache.org/proper/commons-collections/source-repository.html), [документация пользователя](http://commons.apache.org/proper/commons-collections/userguide.html), [javadoc](http://commons.apache.org/proper/commons-collections/javadocs/api-release/overview-summary.html).  
Как подключить к проекту:

Мaven,Gradle,Ivy

Maven

<dependency>

<groupId>org.apache.commons</groupId>

<artifactId>commons-collections4</artifactId>

<version>4.0</version>

</dependency>

Gradle

'org.apache.commons:commons-collections4:4.0'

Ivy

<dependency org="org.apache.commons" name="commons-collections4" rev="4.0"/>

|  |  |  |  |
| --- | --- | --- | --- |
| **Название** | **Описание** | **Примеры реализаций** | **Примеры использования** |
| [**Unmodifiable**](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/Unmodifiable.html) | Интерфейс аналогичный Immutable классам guava | [UnmodifiableBag](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/bag/UnmodifiableBag.html), [UnmodifiableBidiMap](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/bidimap/UnmodifiableBidiMap.html), [UnmodifiableCollection](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/collection/UnmodifiableCollection.html), [UnmodifiableList](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/list/UnmodifiableList.html), [UnmodifiableMap](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/map/UnmodifiableMap.html) и т.п. | во всех случаях когда нужно создать не модифицированную коллекцию |
| [**IterableMap**](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/IterableMap.html) | Аналог интерфейса Map, но позволяющий итерироваться по Map напрямую без создания entry set. Используется почти во всех реализациях Map в данной библиотеке. | [HashedMap](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/map/HashedMap.html), [LinkedMap](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/map/LinkedMap.html), [ListOrderedMap](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/map/ListOrderedMap.html) и ряд других | Такие же как у обычной map'ы |
| [**OrderedMap**](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/OrderedMap.html) | Позволяет создавать Map'ы, упорядоченные по порядку добавления, но не использующие сортировку | [LinkedMap](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/map/LinkedMap.html), [ListOrderedMap](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/map/ListOrderedMap.html) | В случаях, когда обычно используется отдельно List и отдельно Map'а |
| [**BidiMap**](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/BidiMap.html) | Аналог BiMap из Guava, то есть возможность получать значение по ключу, так и ключ по значению | [TreeBidiMap](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/bidimap/TreeBidiMap.html), [DualHashBidiMap](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/bidimap/DualHashBidiMap.html), [DualLinkedHashBidiMap](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/bidimap/DualLinkedHashBidiMap.html), [DualTreeBidiMap](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/bidimap/DualTreeBidiMap.html) и т.п. | Любые конвертации один к одному, которые требуется выполнять в обе стороны |
| [**Bags**](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/Bag.html) | Аналог Multiset из Guava, то есть возможность сохранять количество элементов каждого типа | [CollectionBag](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/bag/CollectionBag.html), [HashBag](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/bag/HashBag.html), [SynchronizedBag](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/bag/SynchronizedBag.html), [TreeBag](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/bag/TreeBag.html) и другие | подсчет кол-ва любых объектов |
| [**BoundedCollection**](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/BoundedCollection.html), [BoundedMap](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/BoundedMap.html) | Позволяет создавать динамические коллекции, ограниченные каким-то размером сверху | [CircularFifoQueue](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/queue/CircularFifoQueue.html), [FixedSizeList](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/list/FixedSizeList.html), [FixedSizeMap](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/map/FixedSizeMap.html), [LRUMap](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/map/LRUMap.html) | в случае, когда вы точно знаете что в коллекции не может быть больше определенного количества элементов |
| [**MultiMap**](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/MultiMap.html) | Аналог Multimap из Guava, то есть возможность сохранять множество элементов для одного ключа | MultiValueMap | для коллекций со связями один ключ – много значений |
| [**Trie**](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/Trie.html) | Коллекция для создания и хранения упорядоченных деревьев | PatriciaTrie | создание деревьев |
| [**TreeList**](http://commons.apache.org/proper/commons-collections/javadocs/api-release/org/apache/commons/collections4/list/TreeList.html) | Замена ArrayList и LinkedList, если требуется вставить элемент в середину списка, так как в данном списке данные хранятся в виде дерева, что позволяет с одной стороны относительно быстро получать данные по индексу, с другой стороны быстро вставлять данные в середину списка. | TreeList | замена LinkedList при частых добавлениях/ударениях в середине списка |

**3.3 Trove коллекции**

В отличии от остальных библиотек альтернативных коллекций, Trove не предлагает никакие новые уникальные виды  
коллекций, зато предлагает оптимизацию существующих:  
Во-первых, как известно, примитивные типы Java нельзя добавить в стандартные коллекции, только их обертки, что резко  
увеличивает занимаемую память и несколько ухудшает производительность коллекций. Trove предлагает набор коллекций,  
ключи и значения которых могут содержать примитивные типы.  
  
Во-вторых, стандартные коллекции часто реализованы не самым оптимальным способом по потреблению памяти, например, каждый элемент HashMap храниться в отдельном объекте, а HashSet это HashMap хранящая фейковые объекты вместо ключей. Trove предлагает свои реализации таких коллекций на основе массивов и открытой адресации, что позволяет значительно сократить требуемую память и в некоторых случаях улучшить производительность.  
  
**Update:** В комментариях, к статье было высказано мнение что Trove плохо использовать в новых проектах, так как он по всех параметрам уступает fastutil или GS (кол-во багов, полнота покрытия интерфейсов, производительность, активность поддержки, и т. д.). К сожалению, у меня нет возможности сейчас провести полноценный анализ/сравнение Trove с fastutil и GS, поэтому не могу проверить данное мнение, просто учитывайте его при выборе библиотеки альтернативных коллекций.   
  
**Официальная информация:** [документация](http://trove.starlight-systems.com/), [исходные коды](https://bitbucket.org/trove4j/trove), [javadoc](http://trove4j.sourceforge.net/javadocs/).  
Как подключить к проекту:

Maven, Gradle, Ivy

Maven

<dependency>

<groupId>net.sf.trove4j</groupId>

<artifactId>trove4j</artifactId>

<version>3.0.3</version>

</dependency>

Gradle

'net.sf.trove4j:trove4j:3.0.3'

Ivy

<dependency org="net.sf.trove4j" name="trove4j" rev="3.0.3"/>

|  |  |  |
| --- | --- | --- |
| **Название** | **Аналог JDK** | **Описание** |
| [THashMap](http://trove4j.sourceforge.net/javadocs/gnu/trove/map/hash/THashMap.html) | [HashMap](https://docs.oracle.com/javase/8/docs/api/java/util/HashMap.html) | Реализация Map интерфейса, которая использует хеш-таблицу с алгоритмом "[открытой адресации](https://ru.wikipedia.org/wiki/%D0%A5%D0%B5%D1%88-%D1%82%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D0%B0#.D0.9E.D1.82.D0.BA.D1.80.D1.8B.D1.82.D0.B0.D1.8F_.D0.B0.D0.B4.D1.80.D0.B5.D1.81.D0.B0.D1.86.D0.B8.D1.8F)" для разрешения коллизий (в отличии от HashMap где используется [метод цепочек](https://ru.wikipedia.org/wiki/%D0%A5%D0%B5%D1%88-%D1%82%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D0%B0#.D0.9C.D0.B5.D1.82.D0.BE.D0.B4_.D1.86.D0.B5.D0.BF.D0.BE.D1.87.D0.B5.D0.BA)). Это позволяет не хранить и не создавать объекты класса Node, при этом сильно экономится память и, в некоторых случаях, улучшается производительность. |
| [THashSet](http://trove4j.sourceforge.net/javadocs/gnu/trove/set/hash/THashSet.html) | [HashSet](https://www.google.pl/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=0CBwQFjAAahUKEwiVzK_m49vIAhWh33IKHefACK4&url=https%3A%2F%2Fdocs.oracle.com%2Fjavase%2F8%2Fdocs%2Fapi%2Fjava%2Futil%2FHashSet.html&usg=AFQjCNGmyLjywjCtkqKIp64UrriPBWfrrg) | Реализация Set интерфейса, которая использует хеш-таблицу с алгоритмом "[открытой адресации](https://ru.wikipedia.org/wiki/%D0%A5%D0%B5%D1%88-%D1%82%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D0%B0#.D0.9E.D1.82.D0.BA.D1.80.D1.8B.D1.82.D0.B0.D1.8F_.D0.B0.D0.B4.D1.80.D0.B5.D1.81.D0.B0.D1.86.D0.B8.D1.8F)" для разрешения коллизий |
| [TLinkedHashSet](http://trove4j.sourceforge.net/javadocs/gnu/trove/set/hash/TLinkedHashSet.html) | [LinkedHashSet](http://docs.oracle.com/javase/7/docs/api/java/util/LinkedHashSet.html) | Аналог LinkedHashSet, но используя хеш-таблицы с алгоритмом "[открытой адресации](https://ru.wikipedia.org/wiki/%D0%A5%D0%B5%D1%88-%D1%82%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D0%B0#.D0.9E.D1.82.D0.BA.D1.80.D1.8B.D1.82.D0.B0.D1.8F_.D0.B0.D0.B4.D1.80.D0.B5.D1.81.D0.B0.D1.86.D0.B8.D1.8F)" |
| [TLinkedList](http://trove4j.sourceforge.net/javadocs/gnu/trove/list/linked/TLinkedList.html) | [LinkedList](http://docs.oracle.com/javase/8/docs/api/java/util/LinkedList.html) | Более производительный аналог связного списка, однако накладывающий ряд ограничений на данные. |
| [TByteArrayList](http://trove4j.sourceforge.net/javadocs/gnu/trove/list/array/TByteArrayList.html), [TIntArrayList](http://trove4j.sourceforge.net/javadocs/gnu/trove/list/array/TIntArrayList.html) и т.п. | [ArrayList](https://docs.oracle.com/javase/8/docs/api/java/util/ArrayList.html) | Аналог ArrayList, который непосредственно хранит примитивные числовые значения, что резко сокращает затраты памяти и ускоряет обработку. Есть коллекции для всех семи примитивных числовых типов, шаблон наименования T[Тип]ArrayList |
| [TCharLinkedList](http://trove4j.sourceforge.net/javadocs/gnu/trove/list/linked/TCharLinkedList.html), [TFloatLinkedList](http://trove4j.sourceforge.net/javadocs/gnu/trove/list/linked/TFloatLinkedList.html) и т.п. | [LinkedList](https://docs.oracle.com/javase/8/docs/api/java/util/LinkedList.html) | Аналог LinkedList для хранения семи примитивных числовых типов, шаблон наименования T[Тип]LinkedList |
| [TByteArrayStack](http://trove4j.sourceforge.net/javadocs/gnu/trove/stack/array/TByteArrayStack.html), [TLongArrayStack](http://trove4j.sourceforge.net/javadocs/gnu/trove/stack/array/TLongArrayStack.html) | [ArrayDequery](https://docs.oracle.com/javase/8/docs/api/java/util/ArrayDeque.html) | Реализация стека для хранения примитивных числовых типов, шаблон наименования T[Тип]LinkedList |
| [TIntQueue](http://trove4j.sourceforge.net/javadocs/gnu/trove/queue/TIntQueue.html), [TCharQueue](http://trove4j.sourceforge.net/javadocs/gnu/trove/queue/TCharQueue.html) | [ArrayDequery](https://docs.oracle.com/javase/8/docs/api/java/util/ArrayDeque.html) | Реализация очереди для хранения примитивных числовых типов, шаблон наименования T[Тип]Queue |
| [TShortHashSet](http://trove4j.sourceforge.net/javadocs/gnu/trove/set/hash/TShortHashSet.html), [TDoubleHashSet](http://trove4j.sourceforge.net/javadocs/gnu/trove/set/hash/TDoubleHashSet.html) | [HashSet](https://docs.oracle.com/javase/8/docs/api/java/util/HashSet.html) | Реализация Set интерфейса для хранения примитивных типов, с алгоритмом [открытой адресации](https://ru.wikipedia.org/wiki/%D0%A5%D0%B5%D1%88-%D1%82%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D0%B0#.D0.9E.D1.82.D0.BA.D1.80.D1.8B.D1.82.D0.B0.D1.8F_.D0.B0.D0.B4.D1.80.D0.B5.D1.81.D0.B0.D1.86.D0.B8.D1.8F), шаблон наименования T[Тип]HashSet |
| [TLongLongHashMap](http://trove4j.sourceforge.net/javadocs/gnu/trove/map/hash/TLongLongHashMap.html), [TFloatObjectHashMap](http://trove4j.sourceforge.net/javadocs/gnu/trove/map/hash/TFloatObjectHashMap.html), [TShortObjectHashMap](http://trove4j.sourceforge.net/javadocs/gnu/trove/map/hash/TShortObjectHashMap.html)и т.п. | [HashMap](https://docs.oracle.com/javase/8/docs/api/java/util/HashMap.html) | Реализация Map интерфейса для хранения примитивных типов, с алгоритмом [открытой адресации](https://ru.wikipedia.org/wiki/%D0%A5%D0%B5%D1%88-%D1%82%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D0%B0#.D0.9E.D1.82.D0.BA.D1.80.D1.8B.D1.82.D0.B0.D1.8F_.D0.B0.D0.B4.D1.80.D0.B5.D1.81.D0.B0.D1.86.D0.B8.D1.8F), шаблон наименования T[Тип][Тип]HashMap, где тип может быть Object |

**3.4 GS-collections коллекции**

Основная фишка данной библиотеке в том что нелогично и некрасиво то что методы обработки коллекций (сортировки, поиска) не добавлены в сами классы коллекций, а используется Collections.sort и т.п. методы, поэтому GS-collections предложили идею «богатых» коллекций (rich collections), которые хранят в себе все методы обработки, поиска, сортировки, то есть вместо Collections.sort(list) вызывается просто list.sort. Поэтому библиотека предлагает свои аналоги стандартных коллекций и дополнительно ряд новых коллекций.  
**Официальная информация**: [документация](https://github.com/goldmansachs/gs-collections/wiki), [исходные коды](https://github.com/goldmansachs/gs-collections), [документация пользователя](http://www.goldmansachs.com/gs-collections/documents/GS%20Collections%20Reference%20Guide%205.0.0.pdf), [javadoc](http://www.goldmansachs.com/gs-collections/javadoc/6.1.0/index.html).  
Как подключить к проекту:

Мaven,Gradle,Ivy

**Maven**

<dependency>

<groupId>com.goldmansachs</groupId>

<artifactId>gs-collections-api</artifactId>

<version>6.2.0</version>

</dependency>

<dependency>

<groupId>com.goldmansachs</groupId>

<artifactId>gs-collections</artifactId>

<version>6.2.0</version>

</dependency>

<dependency>

<groupId>com.goldmansachs</groupId>

<artifactId>gs-collections-testutils</artifactId>

<version>6.2.0</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>com.goldmansachs</groupId>

<artifactId>gs-collections-forkjoin</artifactId>

<version>6.2.0</version>

</dependency>

**Gradle**

compile 'com.goldmansachs:gs-collections-api:6.2.0'

compile 'com.goldmansachs:gs-collections:6.2.0'

testCompile 'com.goldmansachs:gs-collections-testutils:6.2.0'

compile 'com.goldmansachs:gs-collections-forkjoin:6.2.0'

**Ivy**

<dependency org="com.goldmansachs" name="gs-collections-api" rev="6.2.0" />

<dependency org="com.goldmansachs" name="gs-collections" rev="6.2.0" />

<dependency org="com.goldmansachs" name="gs-collections-testutils" rev="6.2.0" />

<dependency org="com.goldmansachs" name="gs-collections-forkjoin" rev="6.2.0"/>

|  |  |  |
| --- | --- | --- |
| **Название** | **Аналог JDK** | **Описание** |
| [FastList](http://www.goldmansachs.com/gs-collections/javadoc/6.1.0/com/gs/collections/impl/list/mutable/FastList.html) | [ArrayList](https://docs.oracle.com/javase/8/docs/api/java/util/ArrayList.html) | Аналог ArrayList с возможностью использовать функции вроде sort, select и т.п. прямо у объекта коллекции |
| [UnifiedSet](http://www.goldmansachs.com/gs-collections/javadoc/6.1.0/com/gs/collections/impl/set/mutable/UnifiedSet.html) | [HashSet](https://docs.oracle.com/javase/8/docs/api/java/util/HashSet.html) | Аналог HashSet. См FastList |
| [TreeSortedSet](http://www.goldmansachs.com/gs-collections/javadoc/6.1.0/com/gs/collections/impl/set/sorted/mutable/TreeSortedSet.html) | [TreeSet](https://docs.oracle.com/javase/8/docs/api/java/util/TreeSet.html) | Аналог TreeSet. См FastList |
| [UnifiedMap](http://www.goldmansachs.com/gs-collections/javadoc/6.1.0/com/gs/collections/impl/map/mutable/UnifiedMap.html) | [HashMap](https://docs.oracle.com/javase/8/docs/api/java/util/HashMap.html) | Аналог HashMap. См FastList |
| [TreeSortedMap](http://www.goldmansachs.com/gs-collections/javadoc/6.1.0/com/gs/collections/impl/map/sorted/mutable/TreeSortedMap.html) | [TreeMap](https://docs.oracle.com/javase/8/docs/api/java/util/TreeMap.html) | Аналог TreeMap. См FastList |
| [HashBiMap](http://www.goldmansachs.com/gs-collections/javadoc/6.1.0/com/gs/collections/impl/bimap/mutable/HashBiMap.html) | - | Реализация BiMap, см. Guava |
| [HashBag](http://www.goldmansachs.com/gs-collections/javadoc/6.1.0/com/gs/collections/impl/bag/mutable/HashBag.html) | - | Реализация Multiset, см. Guava |
| [TreeBag](http://www.goldmansachs.com/gs-collections/javadoc/6.1.0/com/gs/collections/impl/bag/sorted/mutable/TreeBag.html) | - | Реализация отсортированного BiMap, см. Guava |
| [ArrayStack](http://www.goldmansachs.com/gs-collections/javadoc/6.1.0/com/gs/collections/impl/stack/mutable/ArrayStack.html) | [ArrayDeque](https://docs.oracle.com/javase/8/docs/api/java/util/ArrayDeque.html) | Реализация стека с порядком «last-in, first-out», похожего на класс Stack JDK |
| [FastListMultimap](http://www.goldmansachs.com/gs-collections/javadoc/6.1.0/com/gs/collections/impl/multimap/list/FastListMultimap.html) | - | Реализация Multimap, см. Guava |
| [IntArrayList](http://www.goldmansachs.com/gs-collections/javadoc/6.1.0/com/gs/collections/impl/list/mutable/primitive/IntArrayList.html), [FloatHashSet](http://www.goldmansachs.com/gs-collections/javadoc/6.1.0/com/gs/collections/impl/set/mutable/primitive/FloatHashSet.html), [ArrayStack](http://www.goldmansachs.com/gs-collections/javadoc/6.1.0/com/gs/collections/impl/stack/mutable/ArrayStack.html), [HashBag](http://www.goldmansachs.com/gs-collections/javadoc/6.1.0/com/gs/collections/impl/bag/mutable/HashBag.html), [ByteIntHashMap](http://www.goldmansachs.com/gs-collections/javadoc/6.1.0/com/gs/collections/impl/map/mutable/primitive/ByteIntHashMap.html) | - | Коллекции примитивных различных типов, принцип наименования такой же как у trove, но кроме аналогов JDK, так же существуют аналоги коллекций Stack, Bag |

**3.5 Fastutil коллекции**

Давайте очень кратко рассмотрим эту библиотеку для работы с коллекциями примитивных типов.  
Подробнее можно найти информацию: [документация](http://fastutil.di.unimi.it/), [исходные коды](https://github.com/vigna/fastutil), [javadoc](http://fastutil.di.unimi.it/docs/)

|  |  |
| --- | --- |
| **Название** | **Описание** |
| Byte2DoubleOpenHashMap, IntArrayList, IntArrayPriorityQueue и т.п. | Коллекции различных примитивных типов, принцип наименования [Тип]ArrayList, [Тип]ArrayPriorityQueue и т.п. для списков или множеств, и [ТипКлюча]2[ТипЗначения]OpenHashMap и т.п. для Map. |
| IntBigList, DoubleOpenHashBigSet и т.п. | Коллекции различных примитивных типов очень Большого размера, эти коллекции позволяют использовать long элементов, вместо int. Внутри данные, как правило, хранятся как массивы массивов. Не рекомендуется использовать подобные коллекции там где хватит обычных, так как потери производительности могут достигать примерно 30%, однако такие коллекции позволяют работать с действительно большим количеством данных |

**3.6 Прочие библиотеки коллекций и немного о производительности примитивных коллекций**

Кроме Trove и Fastutil есть ещё несколько известных библиотек, реализующих коллекции примитивных типов и более быстрые аналоги стандартных коллекций:  
1) [**HPPC**](http://labs.carrotsearch.com/hppc.html) — High Performance Primitive Collections for Java, так же предоставляет примитивные коллекции аналогичные коллекциям из JDK,  
2) **[Koloboke](https://github.com/OpenHFT/Koloboke)** (другое имя HFTC) — библиотека так же служит для реализации высокопроизводительных примитивных коллекций.  
  
Если интересно сравнение производительности разных библиотек советую посмотреть эту [статью](http://java-performance.info/hashmap-overview-jdk-fastutil-goldman-sachs-hppc-koloboke-trove-january-2015/), только нужно учитывать, что тестировали только коллекции HashMap и в определенных условиях. К тому же, замеряли только скорость работы, не учитывая занимаемую память (например, HashMap jdk могут занимать намного больше памяти чем аналоги от trove), а иногда память может быть даже более важной чем производительность.  
  
**Update:** В комментариях, к статье было высказано мнение что Trove плохо использовать в новых проектах, так как он по всех параметрам уступает fastutil или GS (кол-во багов, полнота покрытия интерфейсов, производительность, активность поддержки, и т. д.). К сожалению, у меня нет возможности сейчас провести полноценный анализ/сравнение Trove с fastutil и GS, поэтому не могу проверить данное мнение, просто учитывайте его при выборе библиотеки альтернативных коллекций.