**ФреймВорк коллекций Java**

Java Collections Framework - это набор различных классов и интерфейсов, предназначенный для работы с коллекциями объектов в языке программирования Java.

Дерево наследования в Java Collections Framework:

- Collection: базовый интерфейс, который представляет собой группу объектов (List, Set, Queue)

- List: упорядоченная коллекция, которая позволяет дублировать элементы (ArrayList, LinkedList)

- Set: неупорядоченная коллекция, которая не содержит дублирующихся элементов (HashSet, TreeSet)

- Queue: коллекция, в которой элементы добавляются и удаляются в порядке FIFO (LinkedList)

- Map: интерфейс, представляющий отображение ключей на значения (HashMap, TreeMap)

Каждый тип коллекции в Java Collections Framework имеет свои особенности и используется для решения определенных задач. Например, ArrayList эффективно работает при доступе к элементам по индексу, HashSet обеспечивает быструю проверку наличия элемента, а TreeMap автоматически сортирует элементы по ключу.

**Javadoc - что, зачем, какие есть аннотации**

Javadoc - это инструмент в JDK, который автоматически генерирует документацию на основе комментариев в исходном коде Java. Он позволяет разработчикам создавать читаемую и понятную документацию для своего кода.

Зачем используется Javadoc:

1. Для документирования классов, методов, переменных и пакетов.

2. Для создания удобной и понятной документации для других разработчиков, которые будут использовать ваш код.

3. Для автоматической генерации HTML-файлов с документацией.

Некоторые аннотации, используемые в Javadoc:

1. @param - описывает параметр метода.

2. @return - описывает возвращаемое значение метода.

3. @throws - описывает исключение, выбрасываемое методом.

4. @author - указывает автора класса или метода.

5. @version - указывает версию класса или метода.

Использование Javadoc помогает улучшить понимание кода, облегчает его поддержку и сотрудничество между разработчиками.

-Коллекции

\* List – Список Элементов, в порядке добавления, имеют переопределённый ToString

Реализации:

ArrayList – Доступ по индексу (дешёвое), но вставлять элементы тяжело

LinkedList – Каждый следующий элемент имеет ссылку на предыдущий и следующий, обращение по индексу дорогое, но вставлять в любую точку дёшево.

\* Set не допускает количество дублей, не сохраняет порядок дополнения

HashSet – наиболее быстрый тип, хранит внутри себя map, быстро отвечает на вопрос содержится ли что-то внутри него, порядок определяет ключ хэша

LinkedHashSet – сохраняет порядок дополнения, из за этого немного медленнее хээшсета

TreeSet – Содержит внутри себя дерево, легко сортирует элементы благодаря этому добавление нового элемента происходит дороже из за авт. Сортировки

\* Map содержит в себе ключ и значение, быстрое обращение

HashMap – Просто помещает ключ и элемент в коллекцию.

LinkedHashMap – сохраняет элементы в порядке добавления

TreeMap – сортирует по ключу

Put – Добавляет элемент, либо если он существует перезаписывает его

\* Queue: коллекция, в которой элементы добавляются и удаляются в порядке FIFO (LinkedList)

HashMap – почему лежит один