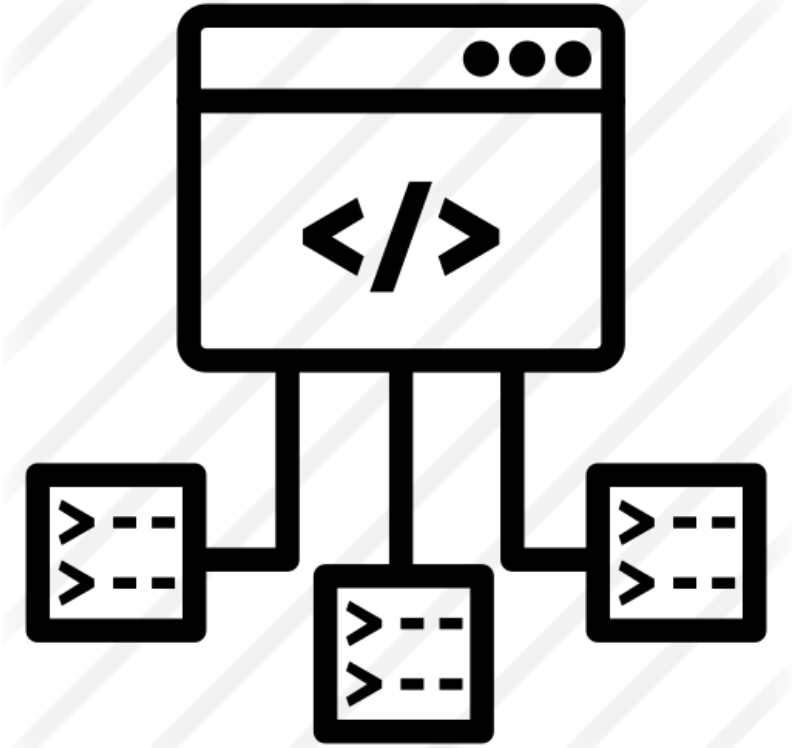
JAVA COLLECTION FRAMEWORK



IN THE NAME OF ALLAH

بسم الله الرحمن الرحيم

THE GRACIOUS, THE MERCIFUL.

LECTURE: 13
Java Collection Framework

Before Starting Collections Framework first we have to Know what is **Data Structured**.

What is Data Structure

It is the way by which we can store the data in efficient way

Time

Space

Data Structure and algorithm like, Searching, sorting, Deleting, Insertion etc

- int num = 10;
- Char ch = ‘A’;

We cannot add multiple values,
We use primitive data type for small projects

Types of Data Structure

Primitive Data Structure

- boolean
- char
- Byte, short, int, long
- Float, double

Non-Primitive Data Structure

Linear Data Structure

- String
- Arrays

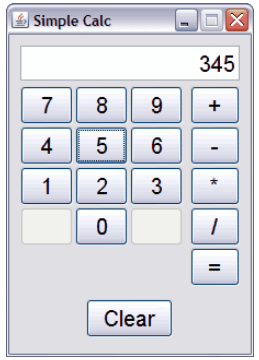
- List, Set, Queue
- ArrayList, LinkedList, HashSet, LinkedHashSet etc

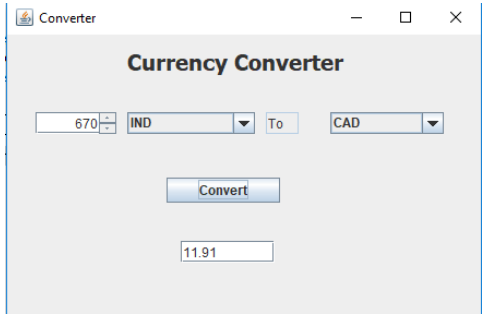
Non-Linear Data Structure

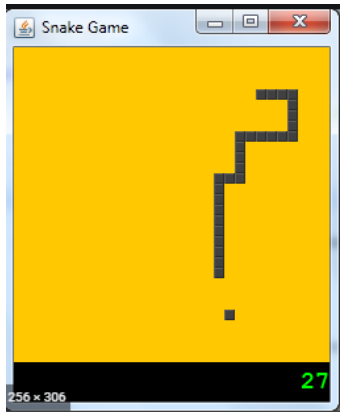
- Graph
- Trees

NOTE: Non-Primitive Data Types we can store multiple Data in Single Entity

Primitive Data Structure

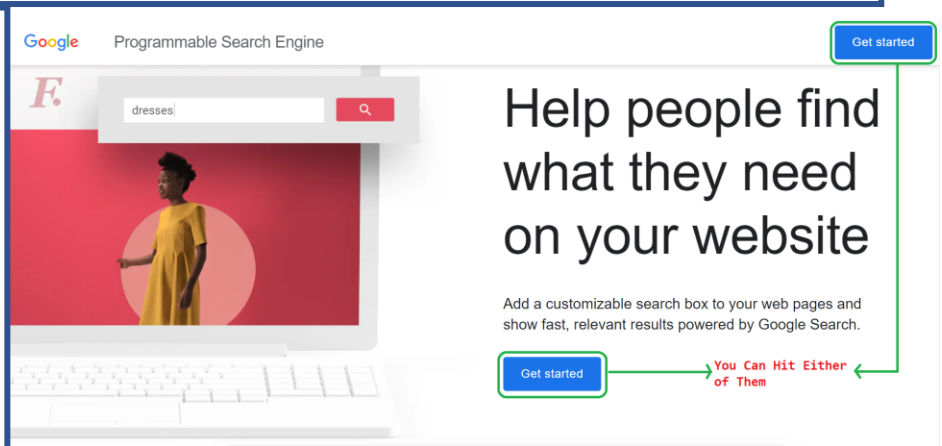




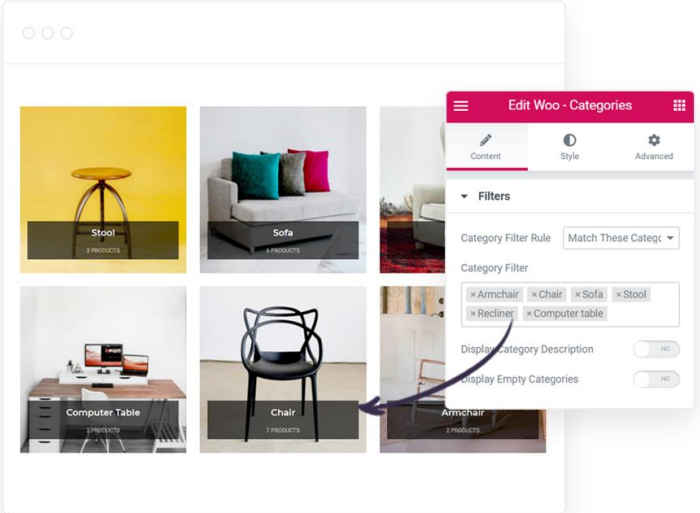


Non-Primitive Data Structure

Searching
Deleting
Sorting,
Insertion etc



=E-Commerce
=Social Networking
=Online Banking



Types of Data Structure

Primitive Data Structure

- boolean
- char
- Byte, short, int, long
- Float, double

Non-Primitive Data Structure

Linear Data Structure

- String
- Arrays
- List, Set, Queue
- ArrayList, LinkedList, HashSet, LinkedHashSet etc

Non-Linear Data Structure

- Graph
- Trees

Part of collection framework

What is Java Collections Framework

- The **Java collections framework** gives the programmer access to prepackaged data structures as well as to algorithms for manipulating them.

- A collection is an object that can hold references to other objects. The collection interfaces declare the operations that can be performed on each type of collection.

- The classes and interfaces of the collections framework are in package java.util.

Arrays

1. Arrays can store **Primitive** and **Non-Primitive** (Objects) Data Types

- Int[] a={10,20,30};

- Class test {
  Test[] arrobj = {obj1, obj2, obj3};
  }

2. Array can store only homogeneous (similar) types of data.

3. Array size is fixed, we cannot increase or decrease the size of an array at runtime.

4. Arrays are in-build features of java & thus we have to developed the algorithm

Collection Framework

1. Collection Framework can contain only non-primitive type of data.

```
ArrayList al= new ArrayList();
al.add(obj1);
al.add(10);
al.add('R');
```

2. We can store only heterogeneous (different) types of data.

3. We can increase or decrease the size of collection at runtime.

4. Collections framework is an API which provides the predefined classes, interfaces and methods.

Collection Framework

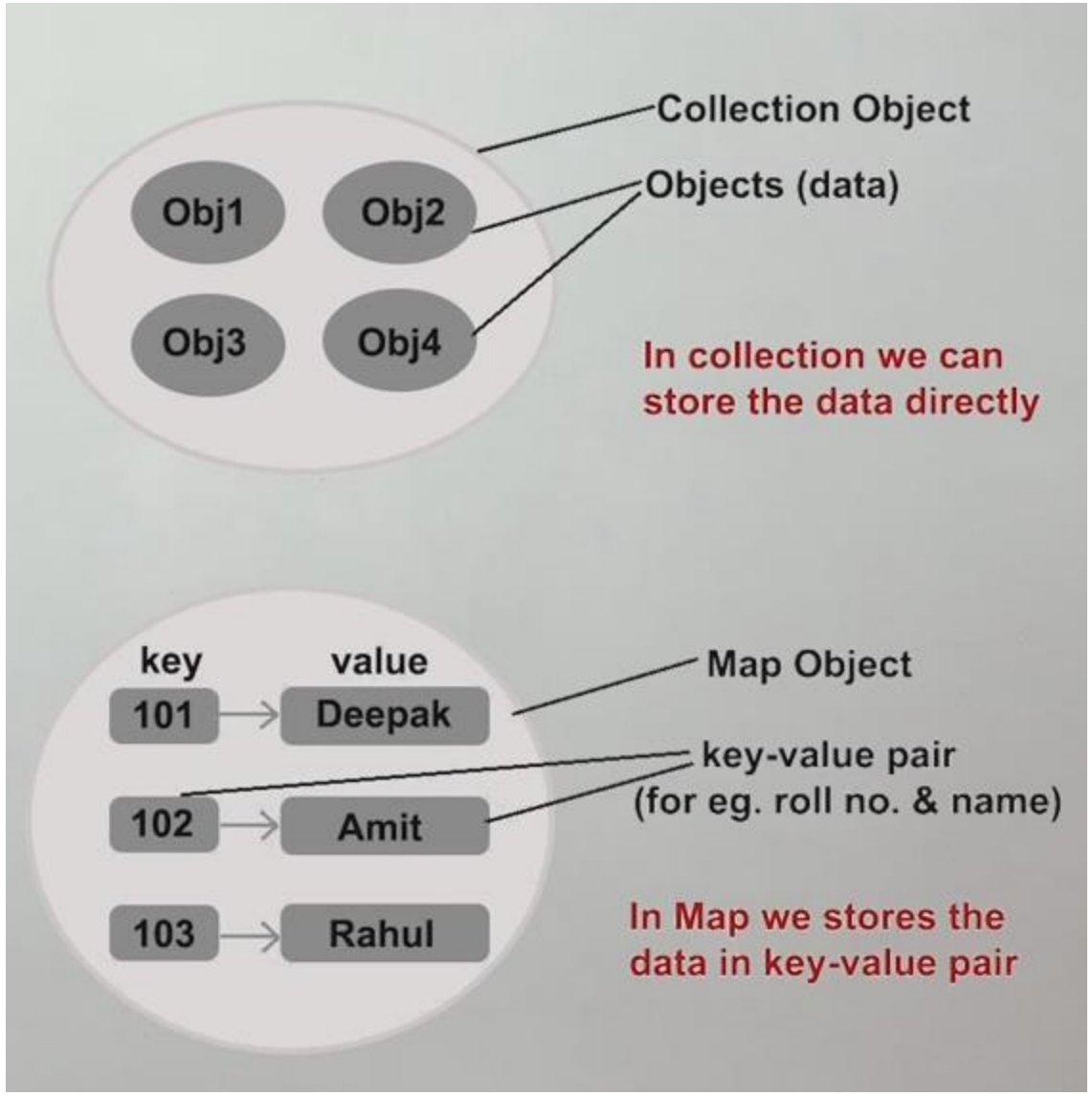
- **Collection:** It is the single entity or object which we can store multiple data.
- **Framework:** Represent the library.
- **It is the set of predefined classes & interfaces which is used to store multiple data.**
- **It contain 2 main parts:**

1. java.util.Collection — **In collection we can store data directly**

2. java.util.Map — **In Map we can store data through key value pair form**

1. java.util.Collection

2. java.util.Map



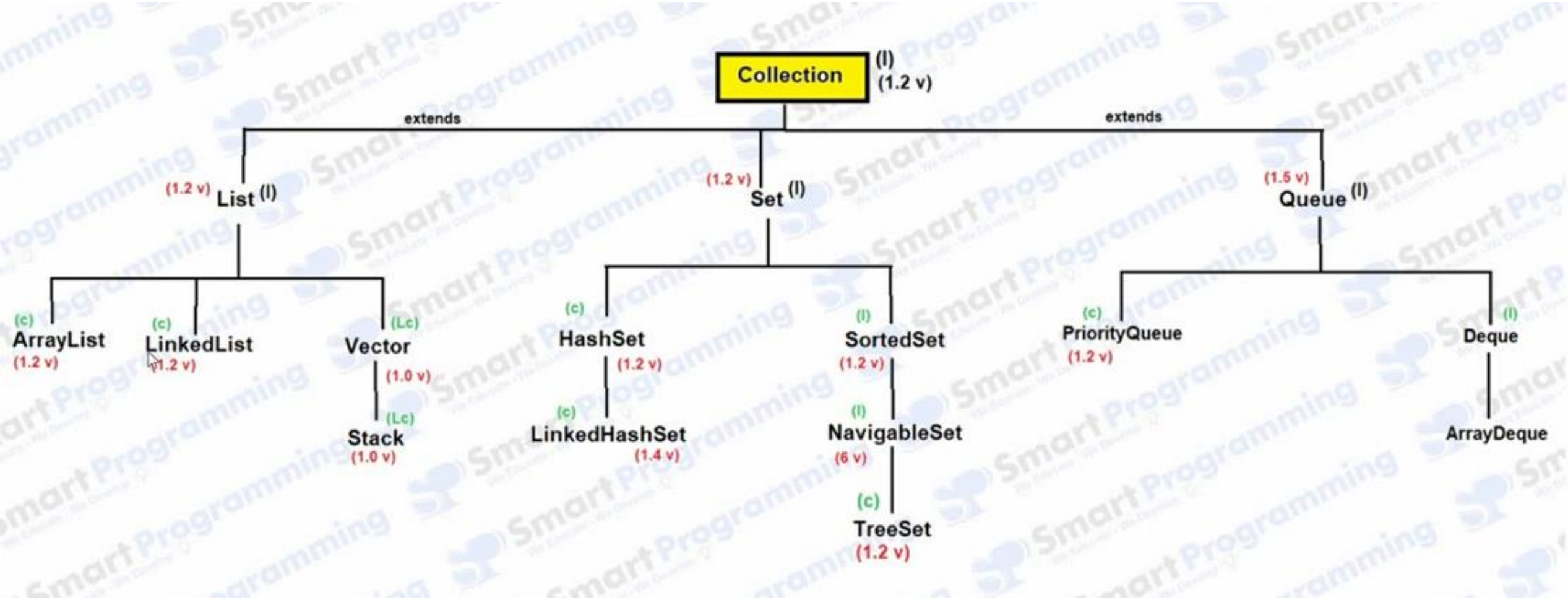
WHAT IS COLLECTION FRAMEWORK, COLLECTION & COLLECTIONS?

COLLECTION FRAMEWORK (API):
It is an API which contains predefined classes & interfaces.

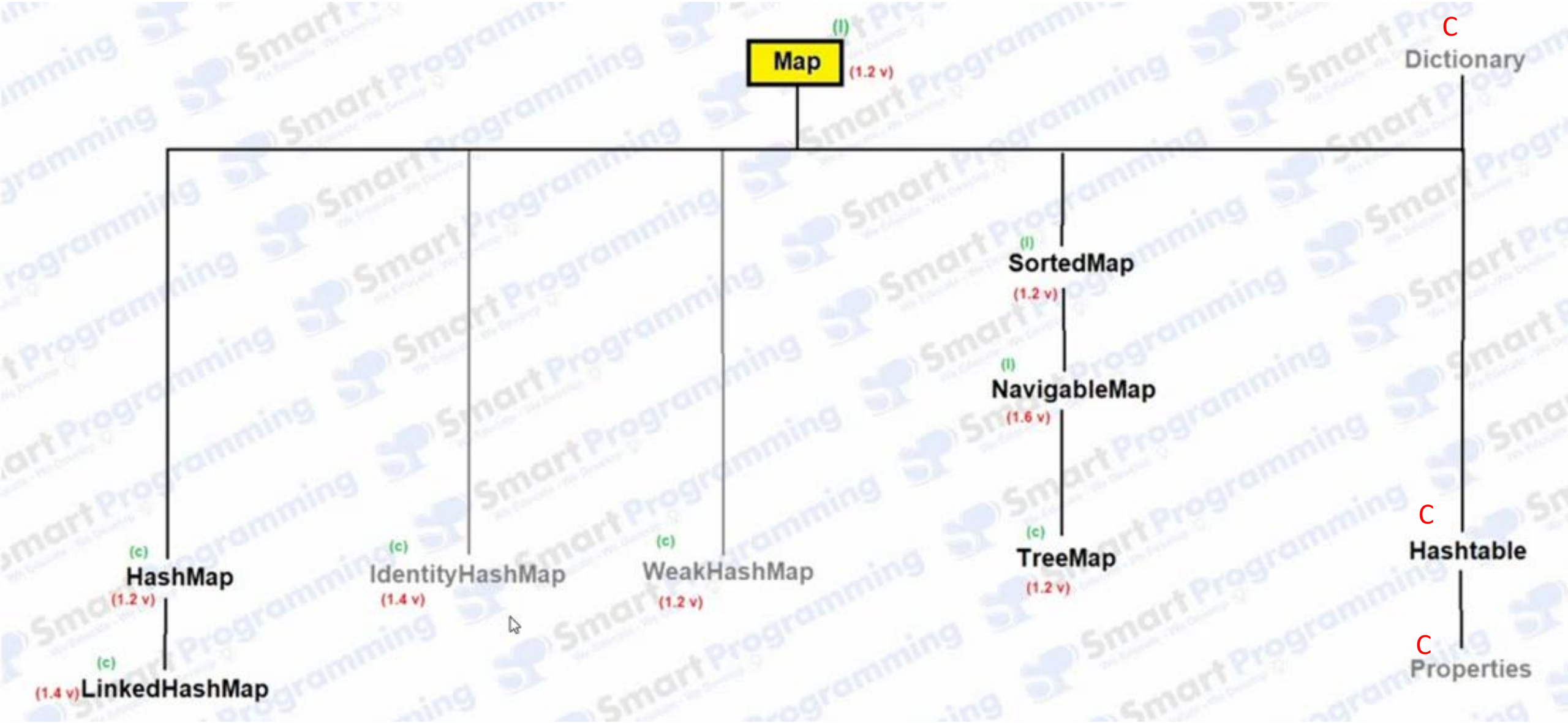
COLLECTION (INTERFACE):
It is the root interface (present in java.util package) of all the collection objects

COLLECTIONS (UTILITY CLASS):
It is the utility class which contains only static methods

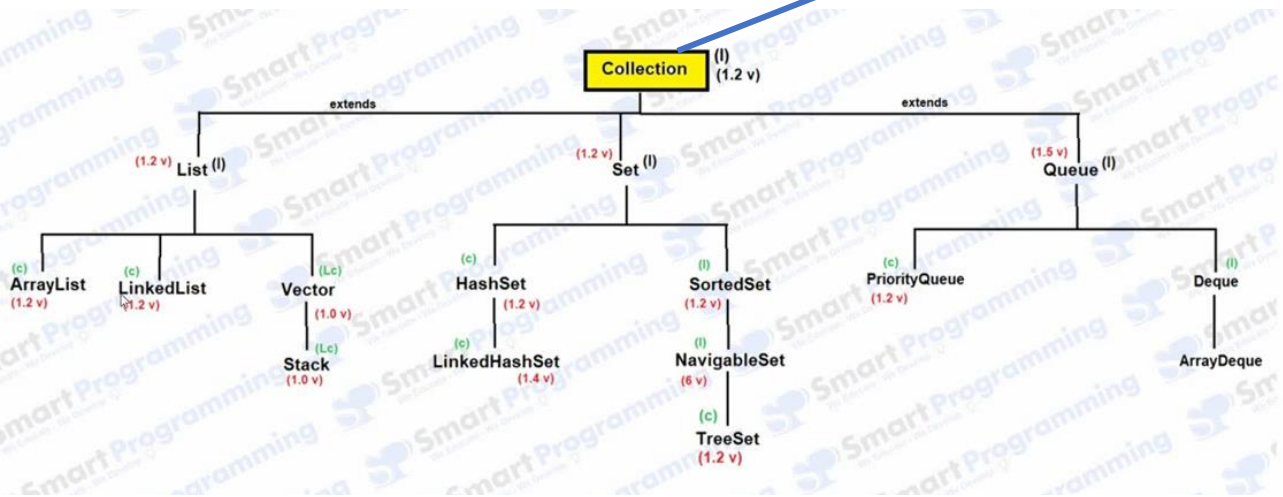
Collection Hierarchy

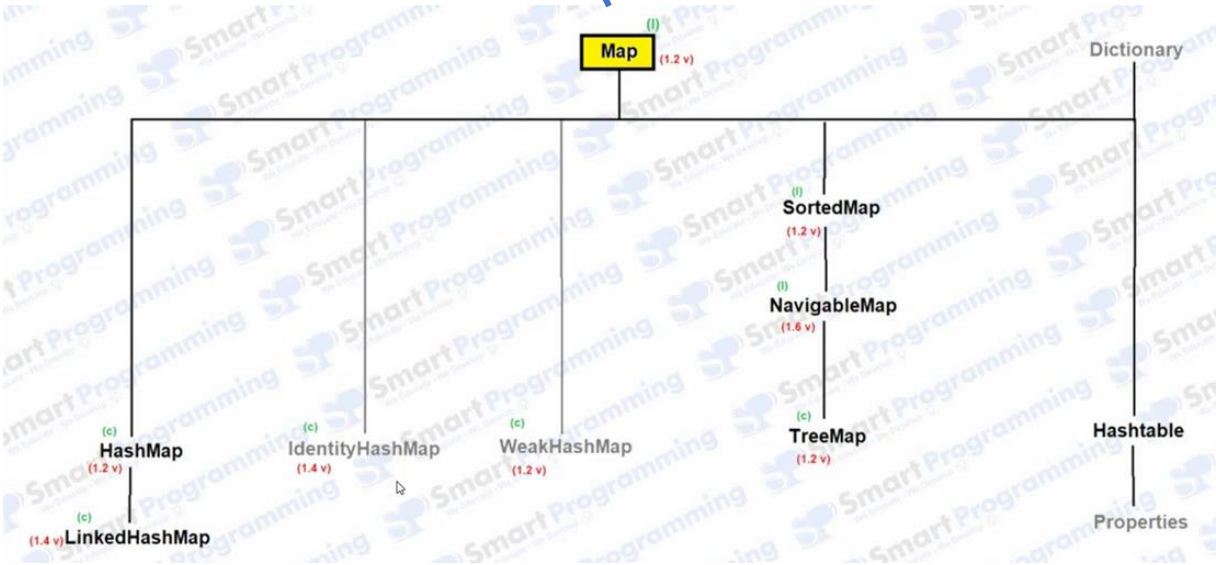


Map Hierarchy



If we combine both Collection and Map it is called **Collection Framework**





Create a collection object in ArrayList

```java
import java.util.ArrayList;

public class Main {
    public static void main(String[] args) {
        //arraylist
        ArrayList al = new ArrayList();
        al.add(10);
        al.add("Rehman");
        al.add('R');

        System.out.println(al);
```

```
"C:\Users\Rehman Ali\.jdks\openjdk-17.0.2\bin\java.exe" "-ja
[10, Rehman, R]

Process finished with exit code 0
```

Create a collection object in HashSet

```java
import java.util.ArrayList;
import java.util.HashSet;

public class Main {
    public static void main(String[] args) {
        //arraylist
        HashSet hs = new HashSet();
        hs.add("Adnan");
        hs.add(10);
        hs.add('C');

        System.out.println(hs);
```

```
"C:\Users\Rehman Ali\.jdks\openjdk-17.0.2\bin\java.exe" "-javaa
[C, 10, Adnan]

Process finished with exit code 0
```

Create a collection object in HashMap

```java
import java.util.ArrayList;
import java.util.HashMap;
import java.util.HashSet;

public class Main {
    public static void main(String[] args) {
        //arraylist
        HashMap hm = new HashMap();
        hm.put(01,"Adnan");
        hm.put(02,"Hameed");
        hm.put(03, "Rehan");

        System.out.println(hm);
```

```
"C:\Users\Rehman Ali\.jdks\openjdk-17.0.2\bin\java.exe" "-javaa
{1=Adnan, 2=Hameed, 3=Rehan}

Process finished with exit code 0
```

We will cover in future:

- ❑ Java Array
- ❑ Java ArrayList
- ❑ Java LinkedList
- ❑ ArrayList vs LinkedList
- ❑ Java List Interface
- ❑ Java HashSet
- ❑ Java LinkedHashSet
- ❑ Java TreeSet
- ❑ Queue & PriorityQueue
- ❑ Deque & ArrayDeque
- ❑ Java Map Interface
- ❑ Java HashMap
- ❑ Working of HashMap
- ❑ Java LinkedHashMap
- ❑ Java TreeMap
- ❑ Java Hashtable
- ❑ HashMap vs Hashtable
- ❑ Java EnumSet
- ❑ Java EnumMap
- ❑ Collections class
- ❑ Sorting Collections
- ❑ Java Vector
- ❑ Java Stack