A collection that contains ****no duplicate**** elements.

Sets contain no pair of elements e1 and e2 such that ****e1.equals(e2)****, and at most one null element.

Same elements in this context refers to two objects are logically identical.

Example

a1 = new Apple("red");

a2 = new Apple("red");

a1.equals(a2); // if true, then we must override equals() method in Apple class and indicate the color comparison. // also by contract, we have to override hashCode() method to ensure the same elements return same hashCode.

HashSet - Internally using HashMap to save the elements as the keys, value is a dummy Object o = new Object();

See Map session for more detail.

class Apple {

private String color;

Apple(String color) {

this.color = color;

}}//...public void doSomething(){

Set<Apple> aSet = new HashSet<>();

aSet.add(new Apple("RED"));

aSet.add(new Apple("RED"));

aSet.size(); // 2, because Apple doesn't override equals method. by default ==.

// so hashSet treat two new Apple as two objects.}//...

// if override equals using color field. aSet.size() will be 1. Try it.