Exception Handling
- Runtime problems
- Error, Exception

Throwable
- All the errors and exceptions

1. Checked Exception
2. UnChecked Exception

try
catch
throw
throws
finally

```
try(){

}
```

Add on

interface Closeable extends AutoClosable -> 1.5

interface Autocloseable -> 1.7


```
protected Object clone() throws CloneNotSupportedException{

reuturn
}
```

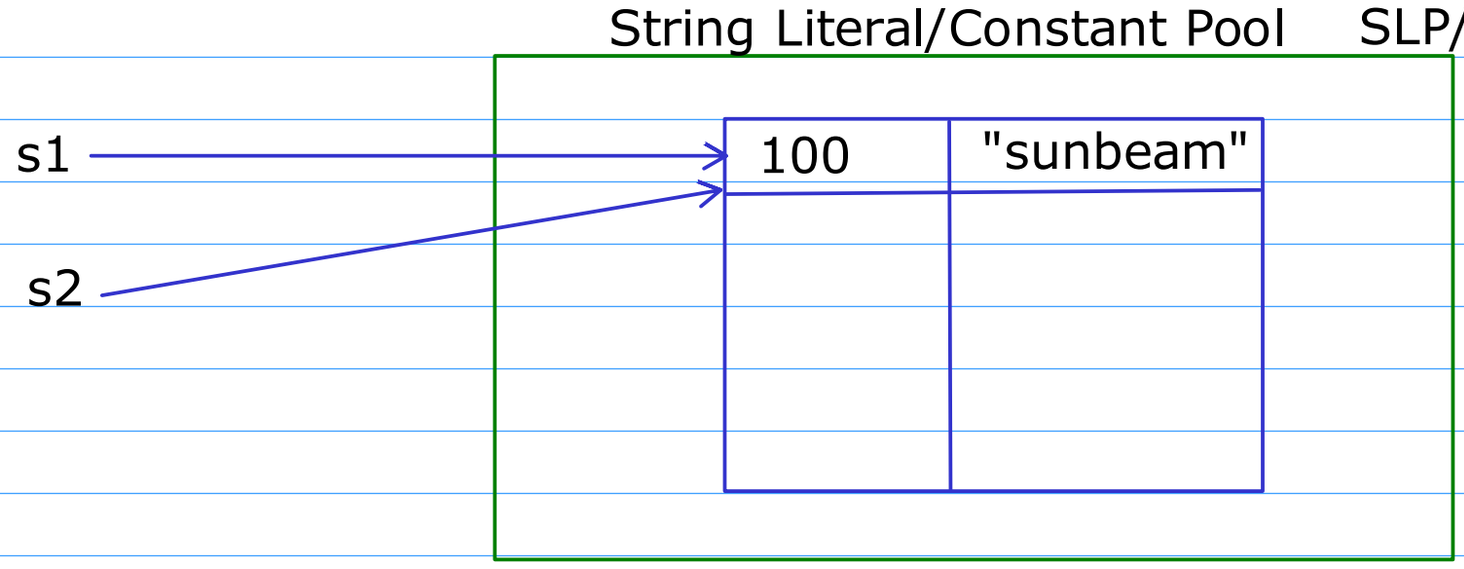
Person
Employee
Date
Time

```
Person p = new Person();
Employee e = new Employee();
```

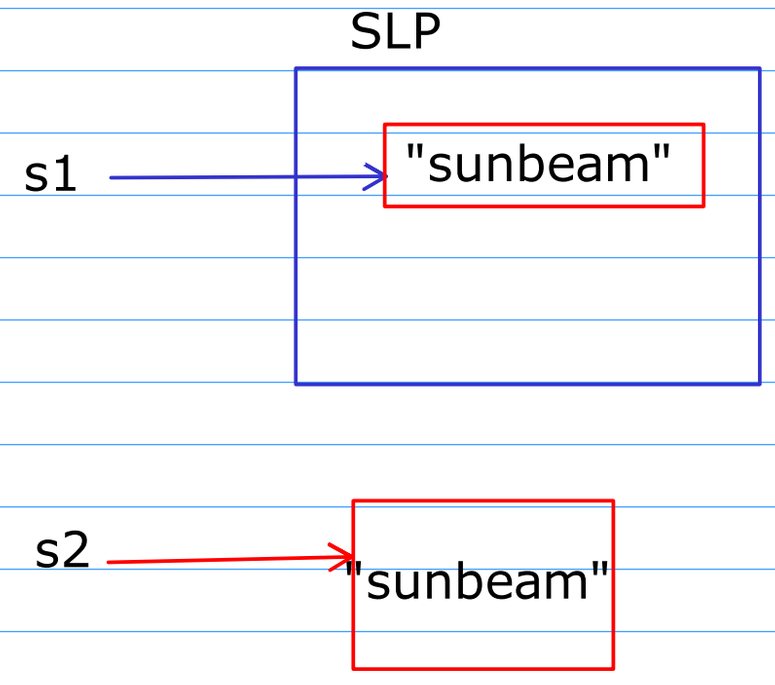
```
String name = "sunbeam";
```

Heap

String Literal/Constant Pool SLP/ SCP



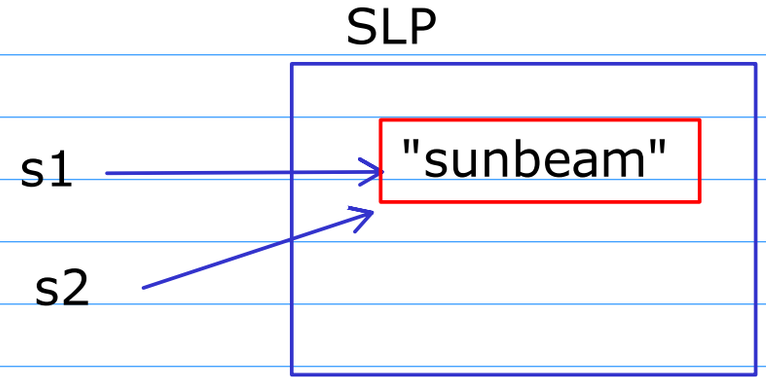

```java
public static void main(String[] args) {
    String s1 = "sunbeam";
    String s2 = new String("sunbeam");

    System.out.println("s1 - " + s1);
    System.out.println("s2 - " + s2);
    System.out.println("s1==s2 - " + (s1 == s2));    // false
    System.out.println("s1.equals(s2) - " + s1.equals(s2));
}                                                    // true
```



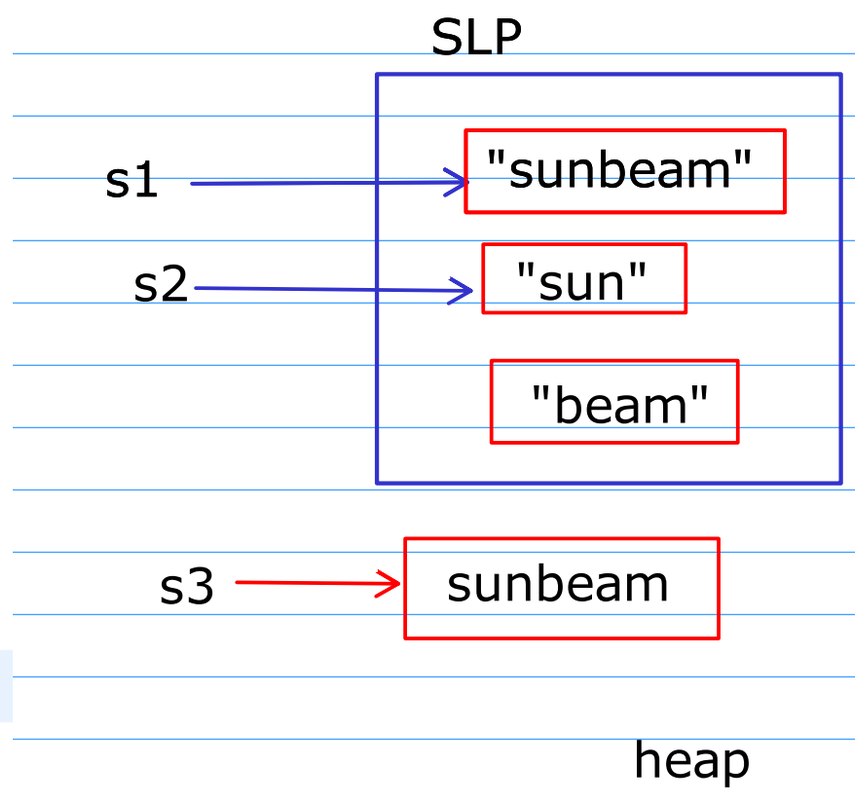

```java
public static void main(String[] args) {
    String s1 = "sunbeam";
    String s2 = "sun" + "beam";

    System.out.println("s1 - " + s1);
    System.out.println("s2 - " + s2);
    System.out.println("s1==s2 - " + (s1 == s2));
    System.out.println("s1.equals(s2) - " + s1.equals(s2));
}
```

true

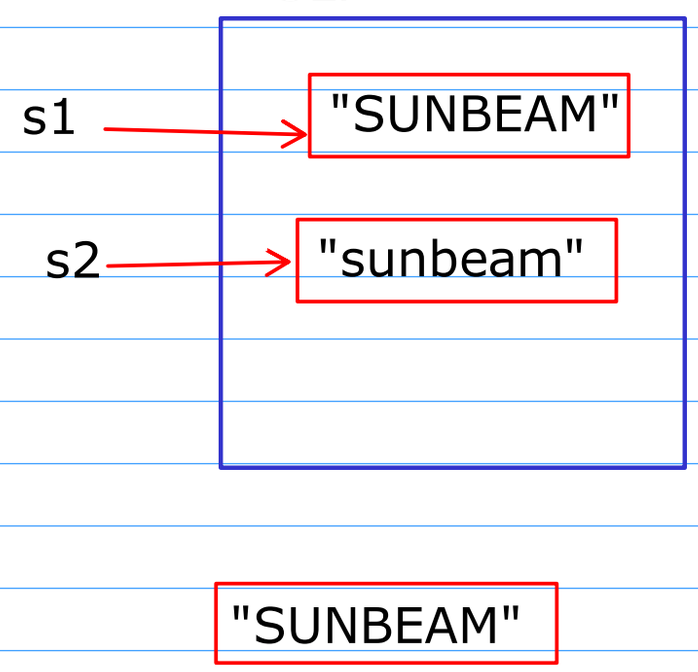
true



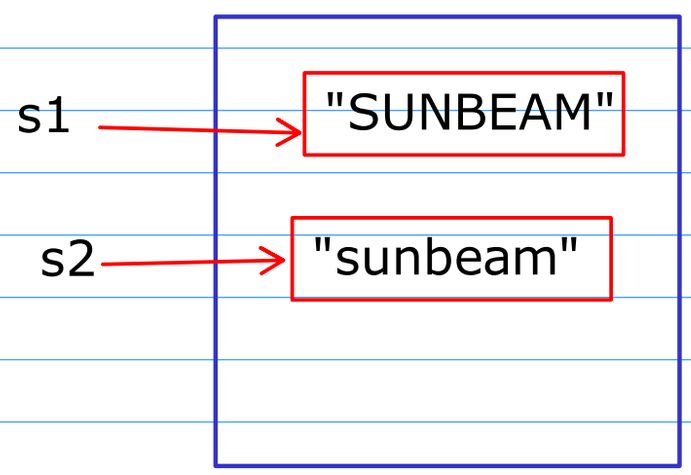
```java
public static void main(String[] args) {
    String s1 = "sunbeam";
    String s2 = "sun";
    String s3 = s2 + "beam"; // runtime

    System.out.println("s1 - " + s1);
    System.out.println("s3 - " + s3);
    System.out.println("s1==s3 - " + (s1 == s3));      // false
    System.out.println("s1.equals(s3) - " + s1.equals(s3));
}                                                      //true
```




```java
public static void main(String[] args) {
    String s1 = "SUNBEAM";
    String s2 = "sunbeam";
    s2.toUpperCase();

    System.out.println("s1 - " + s1);
    System.out.println("s2 - " + s2);
    System.out.println("s1==s2 - " + (s1 == s2));
    System.out.println("s1.equals(s2) - " + s1.equals(s2));

    // System.out.println("s3 - " + s3);
    // System.out.println("s1==s3 - " + (s1 == s3));
    // System.out.println("s1.equals(s3) - " + s1.equals(s3));
}
```



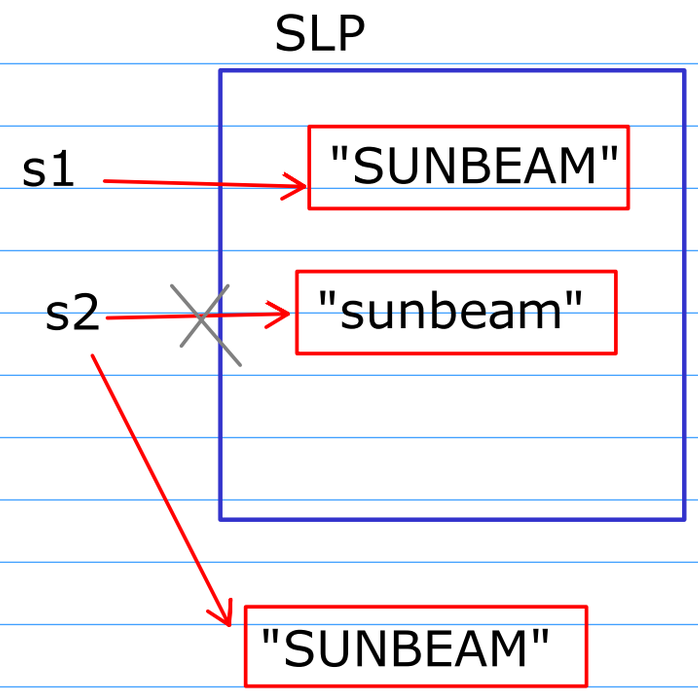

```java
public static void main(String[] args) {
    String s1 = "SUNBEAM";
    String s2 = "sunbeam";
    String s3 = s2.toUpperCase();

    System.out.println("s1 - " + s1);
    System.out.println("s2 - " + s2);      //false
    System.out.println("s1==s2 - " + (s1 == s2));
    System.out.println("s1.equals(s2) - " + s1.equals(s2));
                                                  //false
    System.out.println("s3 - " + s3);     //false
    System.out.println("s1==s3 - " + (s1 == s3));
    System.out.println("s1.equals(s3) - " + s1.equals(s3));
}                                                 //true
```



```java
public static void main(String[] args) {
    String s1 = "SUNBEAM";
    String s2 = "sunbeam";
    s2 = s2.toUpperCase();

    System.out.println("s1 - " + s1);
    System.out.println("s2 - " + s2);
    System.out.println("s1==s2 - " + (s1 == s2));
    System.out.println("s1.equals(s2) - " + s1.equals(s2));

}
```

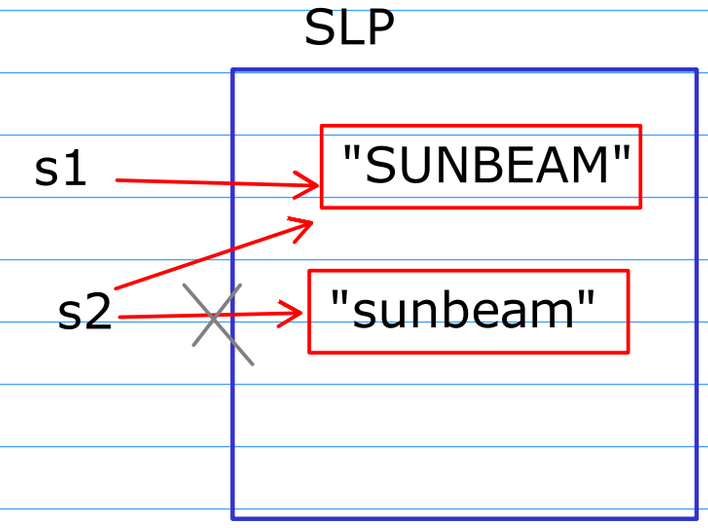
//false

// true




```java
public static void main(String[] args) {
    String s1 = "SUNBEAM";
    String s2 = "sunbeam";
    s2 = s2.toUpperCase().intern(); // SUNBEAM

    System.out.println("s1 - " + s1);
    System.out.println("s2 - " + s2);
    System.out.println("s1==s2 - " + (s1 == s2));
    System.out.println("s1.equals(s2) - " + s1.equals(s2));

}
```

run time operation that creates
a new String object on heap is now forced to be created on SLP




```
class Voter{
String name;
String city;
String pincode;
String area;

}
```

```
Voter v1 = new  Voter("Anil","Pune","411057","Hinijewadi");
```

1 lakh Voters

1 lakh objects of Voter -> Their could be 4 lakh total objects of String

1 lakh objects of Voter -> Their could be < 1 lakh total objects of String

Mutable Strings

1. StringBuffer (Thread safe)
2. StringBuilder (Not Thread safe)