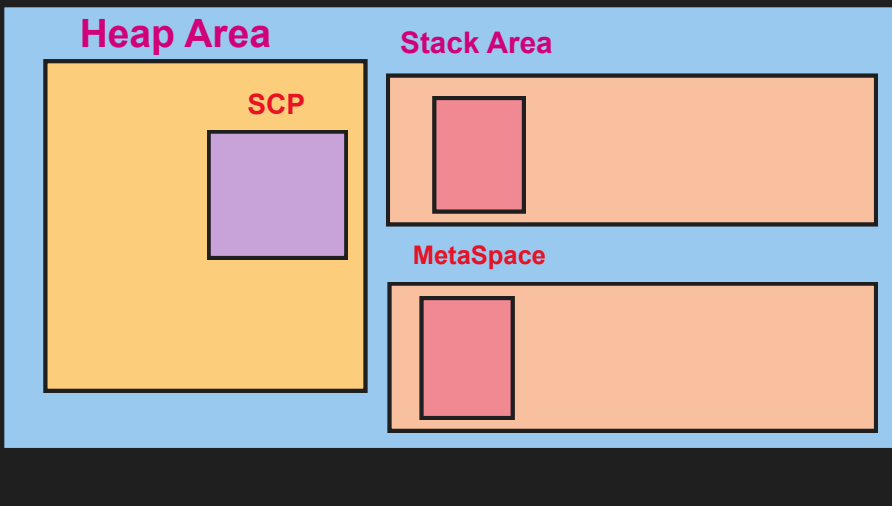


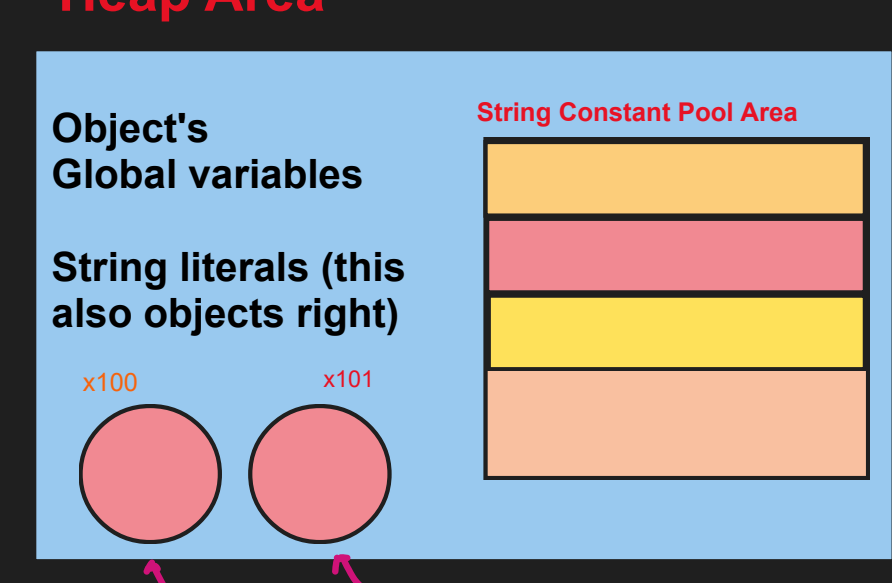
Java memory management



Native area:- Native area is allocated to Java through system "RAM".

The Java developer team splitted into this native area to 3 parts from Java 8 onwards

- 1) Heap Area
- 2) Stack Area
- 3) meta space



Heap Area:- Here, Java will store only global variables & Objects, and one more Area is there inside heap that is "String constant pool Area" (SCP). SCP not allow to duplicate value's, if it is duplicate it will pointing to old value reference.

```
public class MyMemoryClass {  
    // Global variables  
    int a = 10;  
    int b = 20;  
}  
  
// non-static method  
public void add(int a, int b){  
    int sum = a+b;  
}  
  
public static void main(String[] args) {  
  
    // String literals  
    String myName = "yogesh joga";  
  
    String newName = "yogesh joga";  
  
    String mySis = "teju";  
  
    // String object  
    String st = new String("yogesh joga");  
  
    // object creating  
  
    MyMemoryClass myMemoryClass = new MyMemoryClass();  
  
    // local variables  
    int hy = 100;  
    int hyy = 200;  
  
    // calling instance method by using object reference  
    myMemoryClass.add(hy, hyy);  
}  
}
```

Here take this code as reference , this code contains both static and non-static's, Global variables & local variables String objects and String literals objects.

Connect the flow of this code :-

Above the code and chart is indicated which references are storing into heap memory and different places and different object References.

New Stack Area

Stack (first in last out)

this is the main concept of stack

Also next we will jump into Java stack memory area. first in last out

Last in first out

Let's take above code as a example :-

```
public class MyMemoryClass {  
    // Global variables  
    int a = 10;  
    int b = 20;  
}  
  
// non-static method  
public void add(int a, int b){  
    int sum = a+b;  
}  
  
public static void main(String[] args) {  
  
    // String literals  
    String myName = "yogesh joga";  
  
    String newName = "yogesh joga";  
  
    String mySis = "teju";  
  
    // String object  
    String st = new String("yogesh joga");  
  
    // object creating  
  
    MyMemoryClass myMemoryClass = new MyMemoryClass();  
  
    // local variables  
    int hy = 100;  
    int hyy = 200;  
  
    // calling instance method by using object reference  
    myMemoryClass.add(hy, hyy);  
}  
}
```

Stack Area

add	x101
hyy	200
hy	100
myMemoryClass	x101
st	x100
mySis	xs101
newName	xs100
my name	xs100

Stack is created based on running threads.

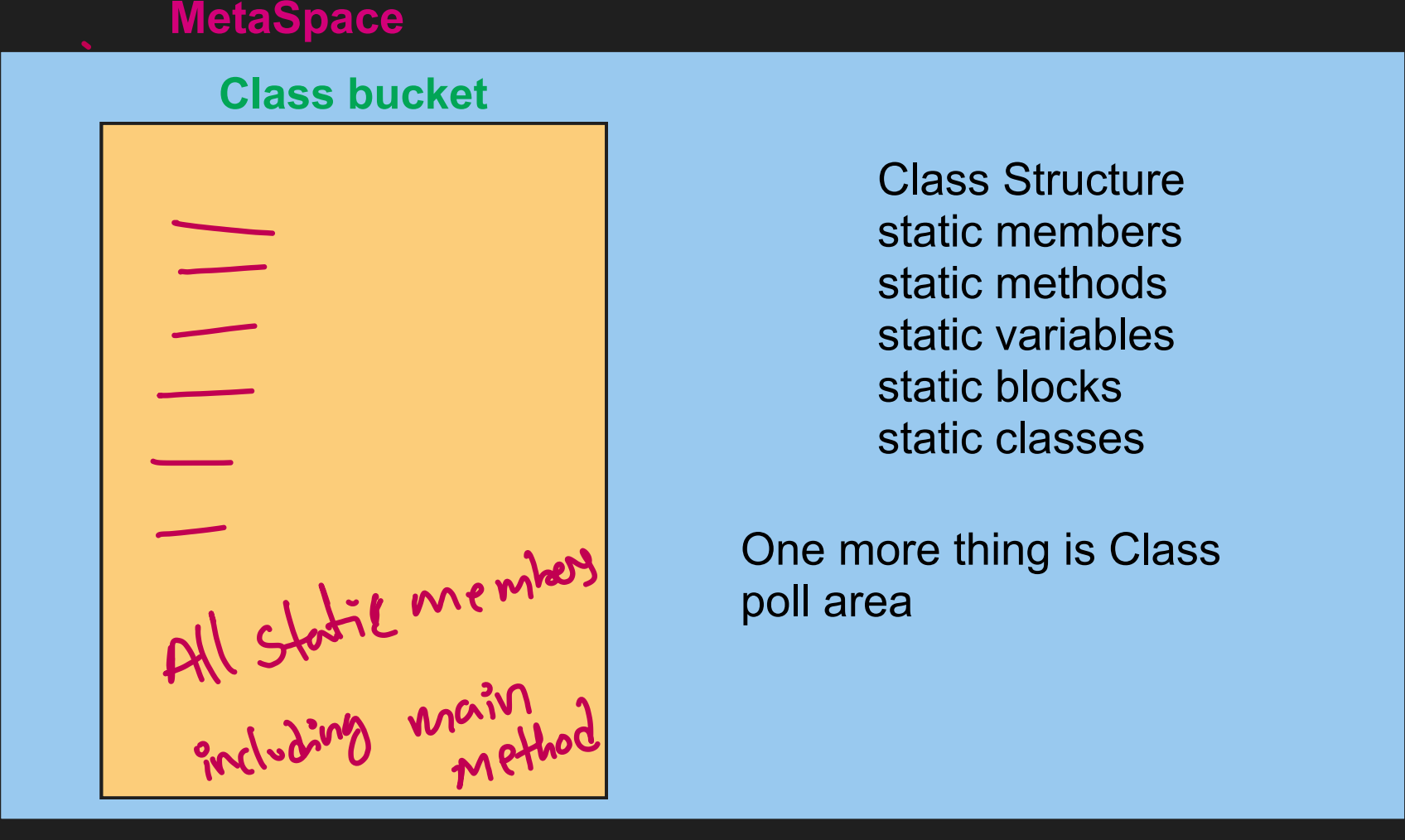
Stack frame

myName is Reference the actual value is String literal object object address is xs100

Let's understand the Concept wise

Stack Area is only calling which of variables and method are participating the execution flow ,

the Global variables are not used any more in the generation that is the reason these variables are not stored in stack frames



that's enough

yogesh joga

10/6/2023