Dt : 30/10/2023

faq:

define UnSafe state of an Application?

=>If more than user using same programming resource(Class or Object or method),then the application will be executing under UnSafe State known as UnSafe State Application.

=>UnSafe state applications will generate Wrong results.

Note:

=>These UnSafe state applications can be converted into Safe State applications using Thread-Synchronization-process.

==

faq:

define Thread Synchronization process?

=>The process of ordering threads for execution is known as Thread Synchronization process.

=>Thread Synchronization process can be done in two ways:

1.Mutual Exclusion process

2.Thread Communication process

1.Mutual Exclusion process:

=>The process of locking programming resources and ordering the threads

for execution is known as Mutual Exclusion process.

=>Mutual Exclusion process can be done in three ways:

(a)synchronized block - Object locking process

(b)synchronized method - Instance method locking process

(c)static synchronization - Class Locking process

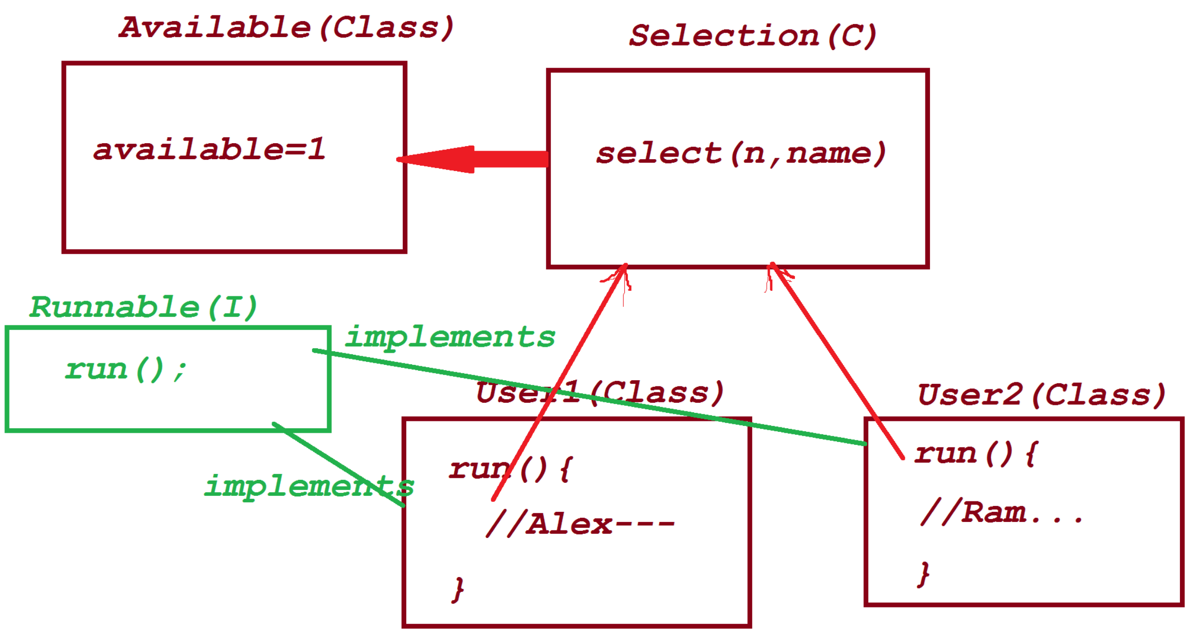
(a)synchronized block - Object locking process

=>The process of declaring statements with 'synchronized' keyword is known as synchronized block.

=>In synchronized block the lock is applied on the Object and known as Object Locking process.

syntax:

```
synchronized(ref_var)
{
  //statements
}
```



Ex-program:

p1 : Available.java

```
package p1;
public class Available {
   public static int available=1;
}
```

p1 : Selection.java

```
package p1;
import java.util.*;
public class Selection
{
    public void select(int n,String name)
    {
     if(n<=Available.available)  {
          System.out.println(n+" Tickets booked for 
"+name+" Time: "+new Date());
          try {
              Thread.sleep(2000);
          }catch(Exception e) {e.printStackTrace();}
          Available.available=Available.available-n;

      }else {
          System.out.println("Tickets not available for 
"+name);
      }
     }
}
```

p1 : User1.java

```
package p1;
public class User1 implements Runnable
{
   public Selection ob=null;
   public User1(Selection ob)
   {
       this.ob=ob;
   }
    @Override
   public void run()
   {
          synchronized(ob)
          {
       ob.select(1, "Alex");
          }
   }
}
```

p1 : User2.java

```
package p1;
public class User2 implements Runnable
{
       public Selection ob=null;
       public User2(Selection ob)
       {
            this.ob=ob;
       }
         @Override
       public void run()
       {
             synchronized(ob)
             {
           ob.select(1, "Ram");
             }
       }
}
```

p2 : DemoThread5.java(MainClass)

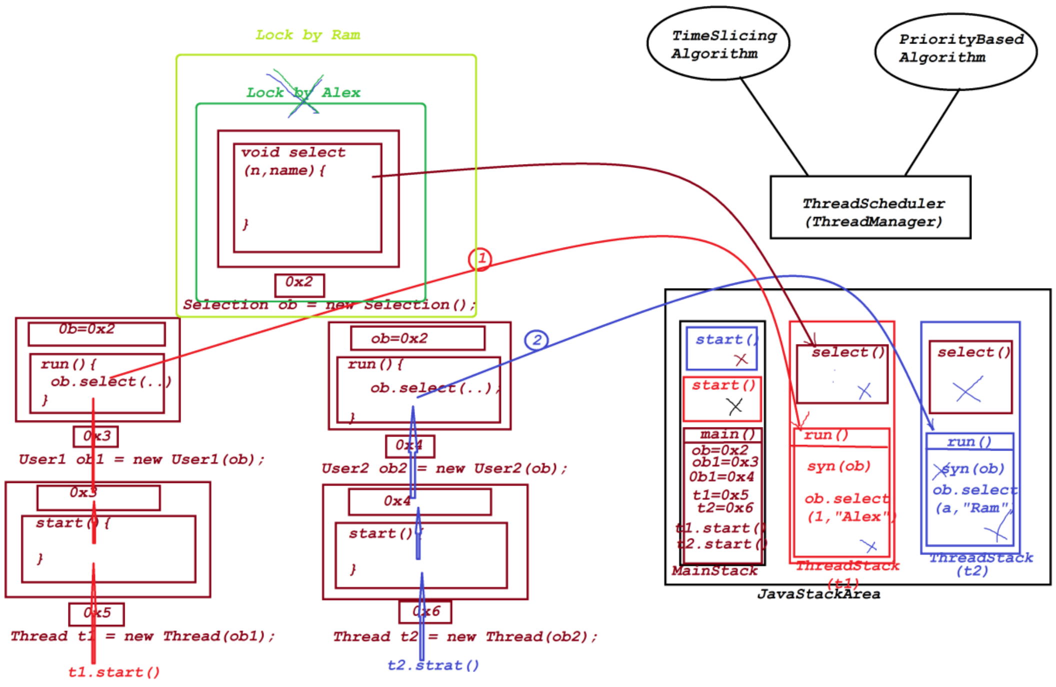
```
package p2;
import p1.*;
public class DemoThread5
{
     public static void main(String[] args)
     {
         Selection ob = new Selection();
         User1 ob1 = new User1(ob);//Con_call
         User2 ob2 = new User2(ob);//Con_call

         Thread t1 = new Thread(ob1);
         Thread t2 = new Thread(ob2);

         t1.start();
         t2.start();
     }
}
```

o/P:

```
1 Tickets booked for Alex Time: Mon Oct 30 19:10:00 IST 2023
```

Tickets not available for Ram

Execution flow of above Application:



===
=