DAY-61

--------

yield() method:

---------------

\* yield() --> static method present inside the Thread class.

\* yield() is used to bring the current executing Thread from run state to runnable state other threads of the same priority.

\* In case if thier are no other threads of same priority in runnable state then the current thread would keep executing even when yield() called.

refer thread life cycle diagram:-

EXAMPLE:

--------

// EXAMPLE FOR yield() method:-

class Child extends Thread

{

public void run()

{

for (int i=1;i<=5 ;i++ )

{

try

{

System.out.println("child thread");

//System.out.println();

Thread.sleep(5000);

Thread.yield();

}

catch (InterruptedException e)

{

}

}

}

}

class Demo1

{

public static void main(String[] args)

{

Child c = new Child();

c.start();

for (int i=1;i<=5 ;i++ )

{

try

{

System.out.println("Main thread");

//System.out.println();

Thread.sleep(5000);

}

catch (InterruptedException e)

{

}

}

}

}

OUTPUT

-------

Main thread

child thread

Main thread

child thread

Main thread

child thread

Main thread

child thread

Main thread

child thread

join() method:

---------------

\* join() allows the current executing thread to go to waiting state until the other thread finised the execution.

\* The Thread executing join statement will go to waiting state untill the another thread completes its execution (main thread goes to waiting state).

\* Once the respective Thread completes its execution the main thread will come out of waiting state.

ie:-(comes from waiting state to runnable state).

\* join method is overloaded method in Thread class :

thread.join()

thread.join(long ms)

thread.join(long ms,int ns)

\* join method throws interrupted exception(it is checked exception).It is complusory for the caller method to handle the exception or duck the exception

using throws keyword.

refer thread life cycle diagram:-

EXAMPLE:

--------

//EXAMPLE FOR join() method.

import java.util.\*;

class Download extends Thread

{

public void run()

{

for (int i=1;i<=100 ;i++ )

{

try

{

System.out.println(i+" downloading...!");

Thread.sleep(150);

}

catch (InterruptedException e)

{

}

}

}

}

class Demo2

{

public static void main(String[] args)

{

Download d = new Download();

d.start();

try

{

d.join();

}

catch (InterruptedException e)

{

}

System.out.println("download completes.");

System.out.println("Enter y to watch the video");

Scanner sc = new Scanner(System.in);

if (sc.next().equalsIgnoreCase("y"))

{

for (int i=1;i<=100 ;i++ )

{

try

{

System.out.println(i+" played...!");

Thread.sleep(150);

}

catch (InterruptedException e)

{

}

}

}

}

}

OUTPUT:

--------

1 downloading...!

2 downloading...!

3 downloading...!

4 downloading...!

5 downloading...!

6 downloading...!

7 downloading...!

8 downloading...!

9 downloading...!

10 downloading...!

11 downloading...!

12 downloading...!

13 downloading...!

14 downloading...!

15 downloading...!

16 downloading...!

17 downloading...!

18 downloading...!

19 downloading...!

20 downloading...!

21 downloading...!

22 downloading...!

23 downloading...!

24 downloading...!

25 downloading...!

26 downloading...!

27 downloading...!

28 downloading...!

29 downloading...!

30 downloading...!

31 downloading...!

32 downloading...!

33 downloading...!

34 downloading...!

35 downloading...!

36 downloading...!

37 downloading...!

38 downloading...!

39 downloading...!

40 downloading...!

41 downloading...!

42 downloading...!

43 downloading...!

44 downloading...!

45 downloading...!

46 downloading...!

47 downloading...!

48 downloading...!

49 downloading...!

50 downloading...!

51 downloading...!

52 downloading...!

53 downloading...!

54 downloading...!

55 downloading...!

56 downloading...!

57 downloading...!

58 downloading...!

59 downloading...!

60 downloading...!

61 downloading...!

62 downloading...!

63 downloading...!

64 downloading...!

65 downloading...!

66 downloading...!

67 downloading...!

68 downloading...!

69 downloading...!

70 downloading...!

71 downloading...!

72 downloading...!

73 downloading...!

74 downloading...!

75 downloading...!

76 downloading...!

77 downloading...!

78 downloading...!

79 downloading...!

80 downloading...!

81 downloading...!

82 downloading...!

83 downloading...!

84 downloading...!

85 downloading...!

86 downloading...!

87 downloading...!

88 downloading...!

89 downloading...!

90 downloading...!

91 downloading...!

92 downloading...!

93 downloading...!

94 downloading...!

95 downloading...!

96 downloading...!

97 downloading...!

98 downloading...!

99 downloading...!

100 downloading...!

download completes.

Enter y to watch the video

y

1 played...!

2 played...!

3 played...!

4 played...!

5 played...!

6 played...!

7 played...!

8 played...!

9 played...!

10 played...!

11 played...!

12 played...!

13 played...!

14 played...!

15 played...!

16 played...!

17 played...!

18 played...!

19 played...!

20 played...!

21 played...!

22 played...!

23 played...!

24 played...!

25 played...!

26 played...!

27 played...!

28 played...!

29 played...!

30 played...!

31 played...!

32 played...!

33 played...!

34 played...!

35 played...!

36 played...!

37 played...!

38 played...!

39 played...!

40 played...!

41 played...!

42 played...!

43 played...!

44 played...!

45 played...!

46 played...!

47 played...!

48 played...!

49 played...!

50 played...!

51 played...!

52 played...!

53 played...!

54 played...!

55 played...!

56 played...!

57 played...!

58 played...!

59 played...!

60 played...!

61 played...!

62 played...!

63 played...!

64 played...!

65 played...!

66 played...!

67 played...!

68 played...!

69 played...!

70 played...!

71 played...!

72 played...!

73 played...!

74 played...!

75 played...!

76 played...!

77 played...!

78 played...!

79 played...!

80 played...!

81 played...!

82 played...!

83 played...!

84 played...!

85 played...!

86 played...!

87 played...!

88 played...!

89 played...!

90 played...!

91 played...!

92 played...!

93 played...!

94 played...!

95 played...!

96 played...!

97 played...!

98 played...!

99 played...!

100 played...!

sleep() method:-

--------------

\* when we call the sleep method the current executing thread will go to sleeping state(alt the execution) till time specified.

\* sleep method throws interrupted exception(it is checked exception).It is complusory for the caller method to handle the exception or duck the exception

using throws keyword.

interrupt method:

-----------------

\* when interrupt() is called the thread in the waiting state or sleeping state will be interrupted.

\* when interrupt() is called a new InterruptException is been created.

refer thread life cycle diagram:-

EXAMPLE:

--------

// Example for sleep() method and interrupt() method.

class MyChild extends Thread

{

public void run()

{

for (int i=1;i<=5 ;i++ )

{

try

{

System.out.println("iam lazy thread..!");

Thread.sleep(10000);

}

catch (InterruptedException e)

{

System.out.println("i got interrupted..!");

}

}

}

}

class Demo3

{

public static void main(String[] args)

{

MyChild c = new MyChild();

c.start();

c.interrupt();

System.out.println("End of the main thread");

}

}

OUTPUT:

--------

End of the main thread

iam lazy thread..!

i got interrupted..!

iam lazy thread..!

iam lazy thread..!

iam lazy thread..!

iam lazy thread..!

SYNCHRONIZATION

--------------------------------------

synchronization method:

-----------------------

\* synchronization is modifier applicable only for method and blocks.

\* If multiple thread try to access or operate the same java object then it leads to problem of "data inconsistency".

\* usage of synchronized modifier will make only one thread to access the resource at a time which will resolve the data inconsistency problem.

\* internally synchronization concept works on the logic of lock and release. This lock and release mechanism is handled by JVM.

refer dia:-

EXAMPLE-1:

--------

// Example for non-synchronization

class Display

{

public void player(String name)

{

try

{

for (int i=0;i<=5;i++)

{

System.out.println("batsman in the order is :"+ name);

//System.out.println();

Thread.sleep(2000);

}

}

catch (InterruptedException e)

{

}

}

}

class MyThread extends Thread

{

Display d;

String name;

public MyThread(Display d, String name)

{

this.d=d;

this.name=name;

}

public void run()

{

d.player(name);

}

}

class Demo4

{

public static void main(String[] args)

{

Display d = new Display();

MyThread t1= new MyThread(d,"Sachin Tendulkar");

MyThread t2= new MyThread(d,"Rahul Dravid");

MyThread t3= new MyThread(d,"Sourav Ganguly");

t1.start();

t2.start();

t3.start();

}

}

OUTPUT:

--------

batsman in the order is :Rahul Dravid

batsman in the order is :Sourav Ganguly

batsman in the order is :Sachin Tendulkar

batsman in the order is :Sachin Tendulkar

batsman in the order is :Rahul Dravid

batsman in the order is :Sourav Ganguly

batsman in the order is :Sachin Tendulkar

batsman in the order is :Sourav Ganguly

batsman in the order is :Rahul Dravid

batsman in the order is :Sourav Ganguly

batsman in the order is :Rahul Dravid

batsman in the order is :Sachin Tendulkar

batsman in the order is :Sourav Ganguly

batsman in the order is :Sachin Tendulkar

batsman in the order is :Rahul Dravid

batsman in the order is :Sourav Ganguly

batsman in the order is :Rahul Dravid

batsman in the order is :Sachin Tendulkar

Press any key to continue . . .

EXAMPLE:

--------

// Example for Synchronization

class Display

{

public synchronized void player(String name)

{

try

{

for (int i=0;i<=5;i++)

{

System.out.println("batsman in the order is :"+ name);

//System.out.println();

Thread.sleep(2000);

}

}

catch (InterruptedException e)

{

}

}

}

class MyThread extends Thread

{

Display d;

String name;

public MyThread(Display d, String name)

{

this.d=d;

this.name=name;

}

public void run()

{

d.player(name);

}

}

class Demo4

{

public static void main(String[] args)

{

Display d = new Display();

MyThread t1= new MyThread(d,"Sachin Tendulkar");

MyThread t2= new MyThread(d,"Rahul Dravid");

MyThread t3= new MyThread(d,"Sourav Ganguly");

t1.start();

t2.start();

t3.start();

}

}

OUTPUT:

-------

batsman in the order is :Rahul Dravid

batsman in the order is :Rahul Dravid

batsman in the order is :Rahul Dravid

batsman in the order is :Rahul Dravid

batsman in the order is :Rahul Dravid

batsman in the order is :Rahul Dravid

batsman in the order is :Sachin Tendulkar

batsman in the order is :Sachin Tendulkar

batsman in the order is :Sachin Tendulkar

batsman in the order is :Sachin Tendulkar

batsman in the order is :Sachin Tendulkar

batsman in the order is :Sachin Tendulkar

batsman in the order is :Sourav Ganguly

batsman in the order is :Sourav Ganguly

batsman in the order is :Sourav Ganguly

batsman in the order is :Sourav Ganguly

batsman in the order is :Sourav Ganguly

batsman in the order is :Sourav Ganguly

EXAMPLE-2

----------

// Example-2 w.r.t synchronization

class Customer

{

public synchronized void withdraw(String mode)

{

for (int i=1;i<=3 ;i++ )

{

System.out.println("withdraw money through"+ mode);

try

{

Thread.sleep(1000);

}

catch (InterruptedException e)

{

}

}

}

}

class BankThread extends Thread

{

Customer c;

String mode;

public BankThread(Customer c, String mode)

{

this.c=c;

this.mode=mode;

}

public void run()

{

c.withdraw(mode);

}

}

class Demo5

{

public static void main(String[] args)

{

Customer c = new Customer();

BankThread t1 = new BankThread(c,"phone");

BankThread t2 = new BankThread(c,"ATM");

BankThread t3 = new BankThread(c,"google pay");

t1.start();

t2.start();

t3.start();

}

}

OUTPUT:

--------

withdraw money throughATM

withdraw money throughATM

withdraw money throughATM

withdraw money throughphone

withdraw money throughphone

withdraw money throughphone

withdraw money throughgoogle pay

withdraw money throughgoogle pay

withdraw money throughgoogle pay

NOTE: If multiple synchronized method are present then only one thread can access the method at a time becuase class level lock is released only

once for a thread.

--> but, instance method can be accessed by any other different thread simultaneoulsy.

synchronized blocks:

--------------------

\* when we make use of synchronized method the efficiency of the program would reduce.

\* If the thread is executing synchronized method the other threads who wants to be executed the method which is synchronized as to wait.

\* Incase if we dont want the whole method to be synchronized but only few lines of method is synchronized then we must use synchronized blocks.

\* synchronized block is having 3 types of parameter:

1: synchronized(this)

{

}

--> if a thread got lock of current object lock then only it is allowed to execute this area.

2: synchronized(object ref)

{

}

--> if a thread got lock of perticular object lock then only it is allowed to execute this area.

3: synchronized(class name)

{

}

--> if a thread got lock of perticular class level lock then only it is allowed to execute this area.

EXAMPLE:

---------

// Example for Synchronization block

class Display

{

public /\*synchronized\*/ void player(String name)

{

synchronized(this)

{

try

{

for (int i=0;i<=5;i++)

{

System.out.println("batsman in the order is :"+ name);

//System.out.println();

Thread.sleep(2000);

}

}

catch (InterruptedException e)

{

}

}

}

}

class MyThread extends Thread

{

Display d;

String name;

public MyThread(Display d, String name)

{

this.d=d;

this.name=name;

}

public void run()

{

d.player(name);

}

}

class Demo6

{

public static void main(String[] args)

{

Display d = new Display();

MyThread t1= new MyThread(d,"Sachin Tendulkar");

MyThread t2= new MyThread(d,"Rahul Dravid");

MyThread t3= new MyThread(d,"Sourav Ganguly");

t1.start();

t2.start();

t3.start();

}

}

OUTPUT:

---------

batsman in the order is :Rahul Dravid

batsman in the order is :Rahul Dravid

batsman in the order is :Rahul Dravid

batsman in the order is :Rahul Dravid

batsman in the order is :Rahul Dravid

batsman in the order is :Rahul Dravid

batsman in the order is :Sachin Tendulkar

batsman in the order is :Sachin Tendulkar

batsman in the order is :Sachin Tendulkar

batsman in the order is :Sachin Tendulkar

batsman in the order is :Sachin Tendulkar

batsman in the order is :Sachin Tendulkar

batsman in the order is :Sourav Ganguly

batsman in the order is :Sourav Ganguly

batsman in the order is :Sourav Ganguly

batsman in the order is :Sourav Ganguly

batsman in the order is :Sourav Ganguly

batsman in the order is :Sourav Ganguly