Ranaken monga farell

Using system . Timers ;

Public class scheduler : Itime tickreceive

Public long A = 1;

Public long B = 1 ;

Public long c = 1

Public long D = 1 ;

Public long E = 1 ;

Public long F = 1 ;

Public long X = 1 ;

Public long w = 1 ;

Public long y = 1

Long tempsExec = 1

Public long sort = 1

Public long débit = 1

Private cpu currentcpu ;

Private circularprocesslist queue ;

Private long timeslice ;

Private long timesclicecounter ;

Private const int DEFAULT \_ TIME \_ SLICE = 1000 ;

Private long clock = 1

Public scheduler (CPU cpu ) ;

{

This . Current CPU = cpu ;

Timeslice = DEFAULT \_ TIME \_ SLICE ;

This .timeslicecounter = 1

Queue = new circularproceslist() ;

Cpu. Scheduler = this ;

}

Public scheduler (CPU cpu , int quantum)

: this (cpu)

{

Timeslice = quantum ;

}

Public bool noprocesses

{

get

{

Lock(This)

{

Return queue . Empty && ! Current CPU . Busy ;

}

}

}

Public void ADDprocess(process t )

{

Queue . AddItem(t) ;

T .starttime = clock ;

}

Public void schedulingNeeded()

{

This . Timeslicecounter = 1 ;

}

Public void receivetimetick(object source , ElapsedEventArgs e)

{

This . Clock = (hardwaretimer) source ). Clock ;

If ( this . Timeslicecounter < = 1) ;

{

This . Scheduler () ;

This . Timeslicecounter = this . TimeSlice /100 ;

}

}

Public void schedule () ;

{

Lock (this)

{

System . Console . Out . Writeline(clock + ‘’\t \*\*\* context swicth \*\*\* ‘’) ;

Iprocess current ;

Iprocess removedprocess = this . CurrentCPU . Removeprocess () ;

If( removedprocess != null

{

System . Console . Out . Writeline( clock + ‘’\ tremoving from cpu ‘’ + removedprocess.name) ;

If( ! Removedprocess . Ready)

{

System console . Out. Writeline( clock + ‘’ \ tadding to queue ‘’+ removedprocess . Name ) ;

This . Queue. Additem( removedprocess ) ;

}

Else

{

If ( removedprocess. Name == ‘’ process 1 ‘’ )

{

B= clock ;

C= clock \_ sort ;

Sort = B ;

}

If ( removedprocess . Name == ‘’ process 2 ‘’)

{

E= clock ;

F = clock \_ sort ;

Sort = E ;

}

If ( removedprocess . Name == ‘’ process 3 ‘’ )

{

Y= clock ;

Z = clock \_ sort ;

Sort = y ;

}

System. Console . Out . Writeline ( clock + \ tFinished : ‘’ + removedprocess . Name ) ;

}

}

Current = queue. Next ;

If ( current ! = null ) ;

{

System. Console . Out . Writeline ( clock + ‘’\tputting on cpu’’ + current . Name) ;

This . Currentcpu . Setprocess(current) ;

TempsExec += DEFAULT \_ TIME \_ SLICE ;

}

Else

{

Syste. Console. Out .writeline (A) ;

}

}