

M09-UF2-PR01

# PROCESS SCHEDULER

## Enunciat de la pràctica:

En aquest exercici treballarem la gestió de processos.

L'objectiu del següent exercici és crear un planificador de processos, és a dir un programa que sigui capaç de generar nous processos, trackejar-los i controlar quins quants se n'estan executant en cada moment.

Aquesta pràctica es farà per parelles que pot designar el professor. Segueix els següents passos per la realització de la pràctica:

### Desenvolupament del procés "client":

Genera el programa client. Aquest s'encarregarà de generar els butlletins d'una escola (com fa Sallenet).

Haurem de mostrar per consola, una simulació de les notes que rebeu, per exemple:

```
Alumne: Benito Camelas  
Curs: DAM1  
M01: 6  
M02: 8  
M03: 10  
... : ...
```

Afegeix prints per tal de generar un butlletí simulat per consola i genera cada nota amb un número aleatori del 0 al 10.

El programa captarà 3 variables que entraran com a paràmetres d'entrada:

- Nom
- Cognom1
- Número de curs (1 o 2)

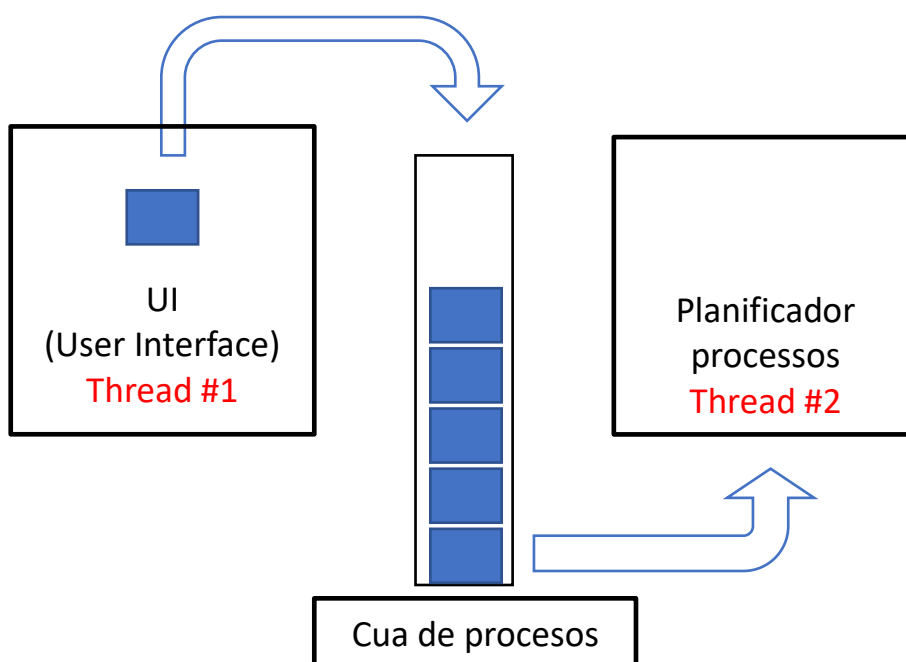
I s'utilitzaran per generar el butlletins (veure els valors subratllats en l'exemple)

Per acabar, afegeix-li un retard de 20 seg. Anem a "suposar" que la infraestructura de l'escola no té gaires recursos i que l'accés a la bbdd és molt lent (més fàcil així per testejar).

## Desenvolupament del procés “Scheduler”:

Un cop fet el programa “client”, anem a fer el planificador de tasques, que haurà de permetre fer crear nous processos, controlar l’execució dels processos simultàniament (hi haurà un màxim de processos executant-se al mateix temps que establirem en 3) i visualitzar l’estat dels processos.

Hi haurà una estructura de dades bàsica per a treballar, que és una cua de processos. Quan l’usuari creï un nou procés aquest passarà a la cua de processos. El planificador de processos anirà executant els processos d’aquesta cua a mesura que els processos anteriors vagin finalitzant, sempre sense superar el màxim de processos permesos.



Per fer el planificador de processos crearem un nou projecte de C# anomenat “Scheduler”.

El programa hauria de tenir un fil d’execució per cada una de les tasques que faci. Seguidament s’explicarà en cadascun dels apartats:

### User Interface

Un fil d’execució serà la UI i permetrà a l’usuari introduir totes les dades referents a la creació de nous processos. És a dir, hauria d’aparèixer un menú per sortir o bé generar un nou procés, i en el cas de voler-lo crear es demanaran els paràmetres d’entrada requerits.

Un cop obtingudes les dades, les passarem com a arguments al procés i tot seguit el programa haurà d'encuar aquest procés a la cua de processos. (Ojete! No arrenquem el procés encara!)

Un cop fet això, la UI haurà de tornar a demanar dades per a un nou procés, per tant la UI és un bucle que va demanant dades a l'usuari i encuant processos.

### **Cua de processos:**

És una estructura de dades de tipus Queue a nivell de classe i que contindrà processos (Process).

### **Planificador de processos**

El segon fil d'execució serà el planificador de tasques en si. Aquest planificador s'encarregarà de mirar si hi ha elements a la cua de processos i executar-los. A no ser que s'hagi al límit de processos (3), en aquest cas s'haurà d'esperar a que vagin acabant.

El procés d'execució serà en base a la cua creada en la UI. El que es farà serà treure el primer element de la cua de processos i s'executarà amb (Process.Start()).

En cas que no hi hagi processos a la cua o que ja s'hagi arribat al límit el planificador de processos resta a l'espera, o bé perquè hi hagi processos a la cua o bé perquè algun dels processos en execució ha acabat i en pot executar un de nou.

Per tant, el planificador de tasques serà un bucle infinit, que sempre estarà pendent de si pot executar nous processos.

### **Com controlar el nombre de processos que s'estan executant?**

Utilitzarem un semàfor per a controlar que no hi hagi més de 3 processos executant-se al mateix temps.

Per això caldrà utilitzar l'event process\_Exited, que es dispara cada vegada que un procés acaba. És gràcies a aquest event que podrem alliberar una sol·licitud del semàfor (release) i deixar entrar un nou procés.

A més a més, haurem de trackejar l'estat de cada procés de forma que sapiguem en tot moment, quin percentatge de processos estem utilitzant (respecte del màxim de 3). Aquesta informació fes que aparegui en la UI.

Qualsevol índex de copia, suposarà un 0 pel que copia i un 0 pel que es deixa copiar.

## Pistes i informació

Cal recordar que la diferència entre un fil (Thread) i un procés, és que els processos són independents entre ells i cada procés té el seu espai de memòria. En canvi els fils, tots formen part del mateix procés, per tant comparteixen el mateix espai de memòria.

### Sobre el client:

La manera de passar dades a un programa és mitjançant els argument de línies de comandes, que el programa recupera mitjançant l'array *args* del mètode *main*.

```
static void Main(string[] args)
```

Els arguments que ens entren per línia de comanda els recuperem amb el paràmetre *args*: *args[0]*, *args[1]*, etc... En el nostre cas tindrem *args[0]* serà la lletra, *args[1]* serà el nombre de vegades i *args[2]* serà la velocitat.

Per simular els paràmetres per línia de comandes a Visual Studio ho heu de fer a través de les propietats del projecte i la pestanya "Depurar".

### Sobre l'scheduler:

Aquí trobareu informació sobre les propietats, mètodes i events de la classe *Process*:

<https://learn.microsoft.com/en-us/dotnet/api/system.diagnostics.process?view=net-7.0>

I aquí podeu veure informació sobre l'ús de les cues:

<https://learn.microsoft.com/es-es/dotnet/api/system.collections.queue?view=net-8.0>

Busca informació sobre la implementació FIFO (First-In-First-Out).

## Rúbrica d'avaluació i entregabls(10 punts)

Documentació: Entregueu aquest document amb la següent informació:

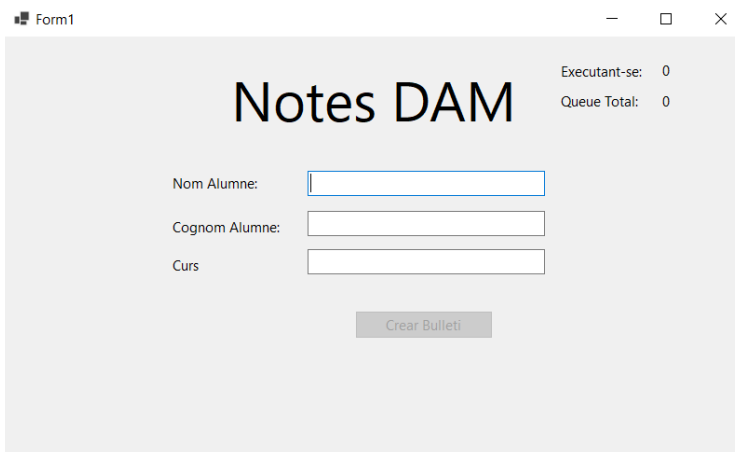
Control d'arguments:

```
Regex onlytext = new Regex(@"^[a-zA-Z0-9\sñ]+$"); //comprova que nomes hi hagi lletres siguin majuscles o minuscles  
Regex intonly = new Regex(@"^[1-2]$"); //comprova que només hi hagi un numero que a la vegada pot ser nomes 1 o 2
```

\*en el cas de que no es compleixin les condicions el boto per a crear el bulletí no s'activara

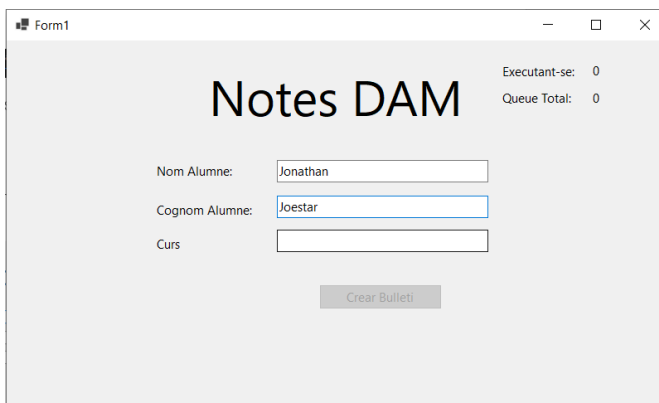
Prova1-

\*al principi del programa el boto estarà deshabilitat



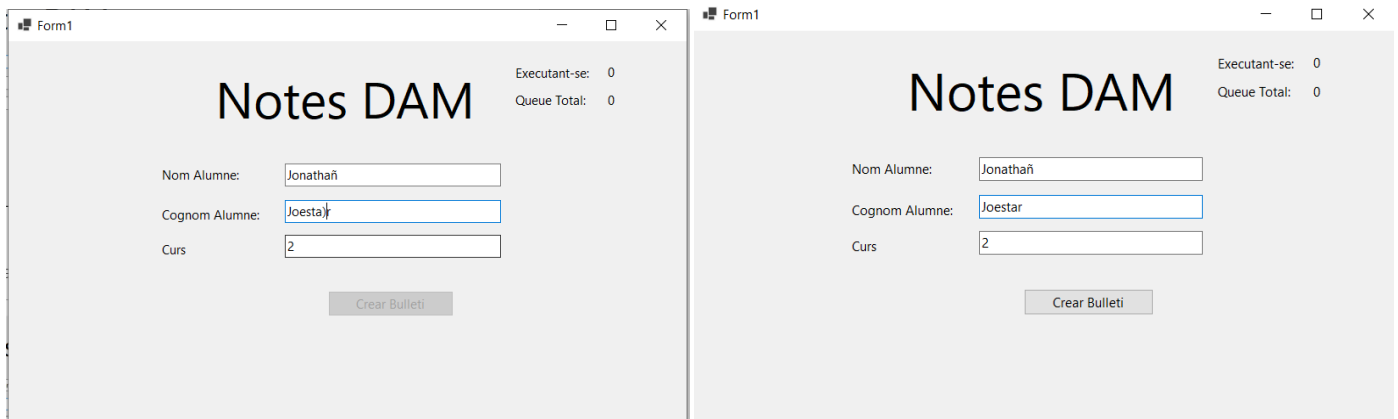
Prova 2-

\*si els 3 camps no estan omplerts amb les dades requerides el botó no s'activara



Prova 3-

\*no s'admeten caràcters especials exepete la "ñ"



Form1

## Notes DAM

Executant-se: 0  
Queue Total: 0

Nom Alumne:

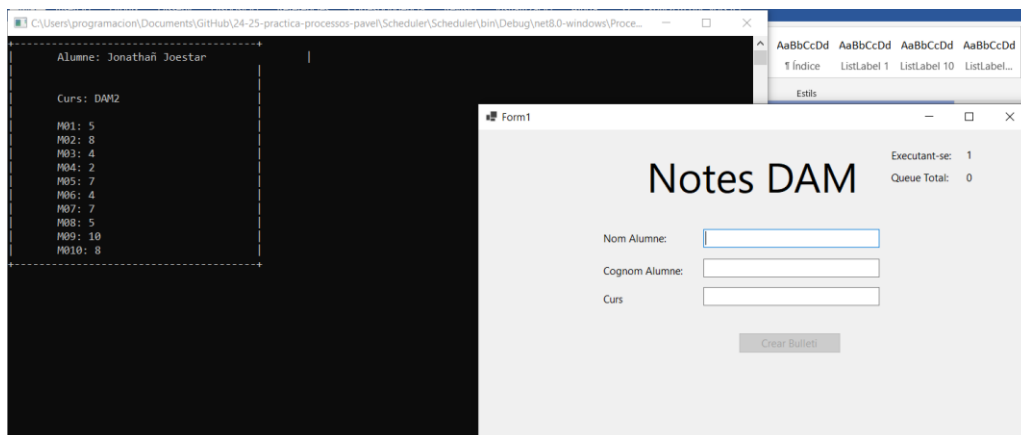
Cognom Alumne:

Curs:

Creació de Processos:

Prova 1-

\*creació de procés normal



Form1

## Notes DAM

Executant-se: 1  
Queue Total: 0

Nom Alumne:

Cognom Alumne:

Curs:

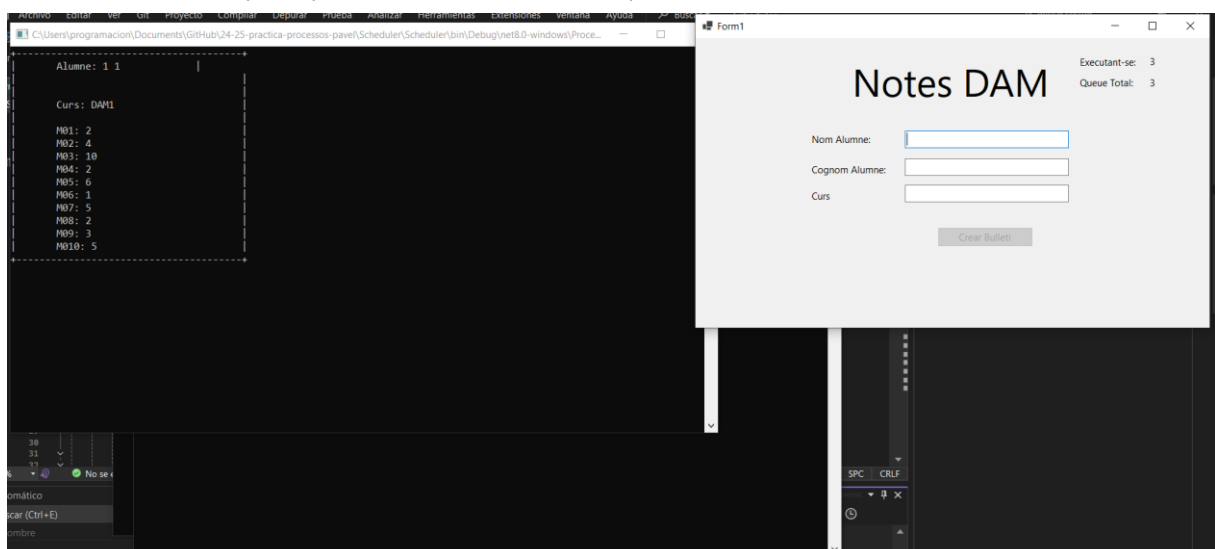
Alumne: Jonathañ Joestar

Curs: DAM2

M01: 5  
M02: 8  
M03: 4  
M04: 2  
M05: 7  
M06: 4  
M07: 7  
M08: 5  
M09: 10  
M10: 8

Prova 2-

\*creació de múltiples processos i control de processos creats



Form1

## Notes DAM

Executant-se: 3  
Queue Total: 3

Nom Alumne:

Cognom Alumne:

Curs:

Alumne: 1 1

Curs: DAM1

M01: 2  
M02: 4  
M03: 10  
M04: 2  
M05: 6  
M06: 1  
M07: 5  
M08: 2  
M09: 3  
M10: 5

\*diferencia de aprox. 60seg

