UNIVERSITE PROTESTANTE AU CONGO

FACULTE DES SCIENCES INFORMATIQUES

BP : 4745

KINSHASA II/ LINGWALA

TP DE PROGRAMMATION SYSTEMES SUR L' EXPERIENCE AVEC PLUSIEURS THREADS DANS UN PROCESSUS

Par :

- **ESAKI MUNDEKE Christian**
- **KONDOLI NKEBI Jonathan**
- **BAILONGAKA LINZABE Hénock**
- **NZEMBELE KASHALA Samuel**
- **BEEMBE KONDEMO Grace**
- **TSHINIOKA KABASELE Jérémie**
- **ASSOKWAMA WEMBO Emmanuel**
- **KOMBA BAKULU Daniel**
- **KAKUDJI KINENGO Alain**
- **NGOY MANGO Ken**
- **KOY Floris**
- **PALUKU MBUSA Grace**
- **NLANDU MASUEKAMA Chadrack**
- **TSHIYOYO KAPIA Henock**
- **NGANGA KABASUBASU SAMUEL**

PROMOTION : L1 FASI

PROFESSEUR : Patrick MUKALA

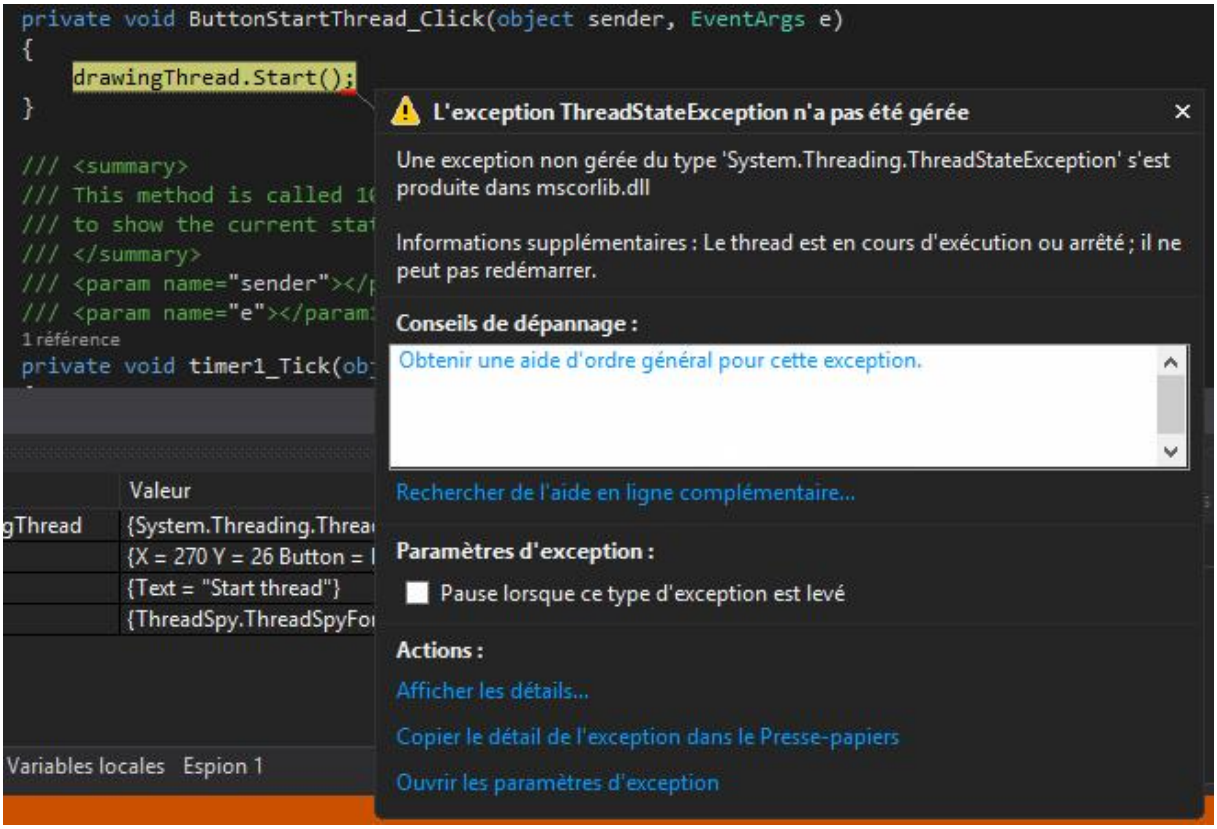
Année Académique 2020-2021

RESOLUTION

a. Au démarrage de l'application
   - Nous avons vérifiés et constaté que l'état du thread est **UnStarted** (non démarré)
   - En cliquant sur le bouton « Start thread », nous avons vu que le nouveau thread est en cours d'exécution car il a ajouté 20 caractères de « a » à la zone de texte de sortie et en attendant nous avons vu l'état du nouveau thread. En vrai on a deux états pendant que les caractères sont ajoutés notamment **Running** et **WaitSleepJoin** :
   - Lorsque le thread est terminé, nous avons vu qu'aucun autre caractère n'est ajouté et l'état du thread change pour devenir **Stopped**

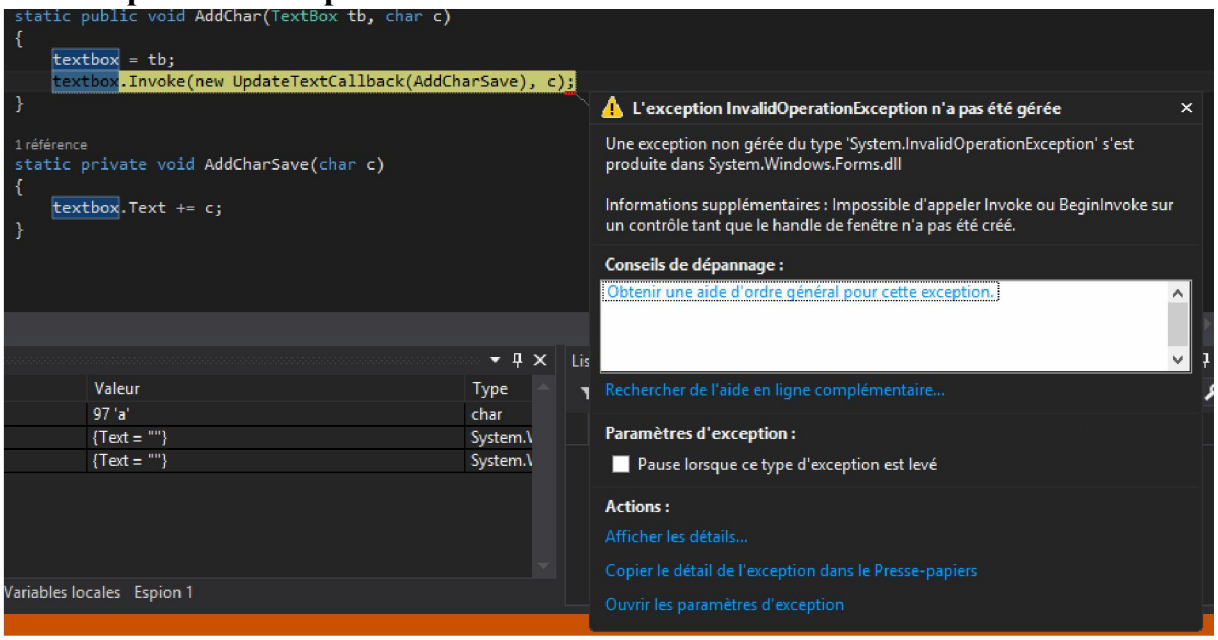
b. Nous avons utilisés le gestionnaire des taches pour afficher le nombre de threads du noyau dans cette application et on a constaté c'est qui a été dit.

c. Etant donné qu'un thread ne peut pas être démarré deux fois, le type d'exception levée est **ThreadStateException** lorsqu'on clique deux fois sur le bouton Start thread.



d. Une fois l'application fermée et redémarrée, nous avons cliqués sur le bouton Start thread et arrêtés immédiatement comme demandé et nous avons constaté

que l'application n'a pas pu se fermer correctement et l'exception **InvalidoperationException** est obtenue.



e. Nous avons changé le programme comme demandé et le **ThreadStart** n'est plus utilisé explicitement

f. Nous avons changé le programme comme demandé de telle sorte que nous puissions plus avoir besoin de la classe **DrawingRunnable.cs** et nous avons copiés la méthode Run de la classe DrawingRunnable à la classe **ThreadSpyForm.cs**

```
public void Run()
{


    for (int i = 0; i < this.cptRun; i++)
    {
        Thread.Sleep(300);
        TextBoxHelper.AddChar(this.TextBoxOutput, this.c);
    }


}
```

Et on passe cette méthode en paramètre pour permettre l'exécution de notre programme.

```
this.ts = new ThreadStart(this.Run);
drawingThread = new Thread(ts);
```

g. Nous avons changé le programme pour qu'un nouveau thread démarre à chaque fois lorsqu'on clique sur le bouton Start thread et comme on doit afficher progressivement chaque lettre de l'alphabet, nous avons commencés par déclarer un tableau de lettre de la manière suivante :

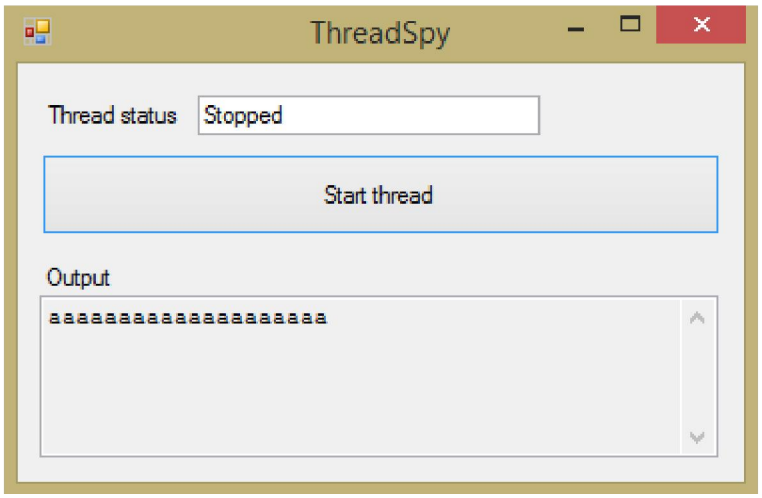
```
private char[] lettres = { 'a', 'b', 'c','d','e','f','g','h','i','j','k','l','m','n','o','p','r','s','t' };
```

Chaque fois qu'on clique sur le bouton Start thread on imprimera 20 caractères d'une lettre et grâce à un compteur le tableau va s'incrémenter et passer à une autre lettre de sorte que nous puissions voir la différence entre le thread.

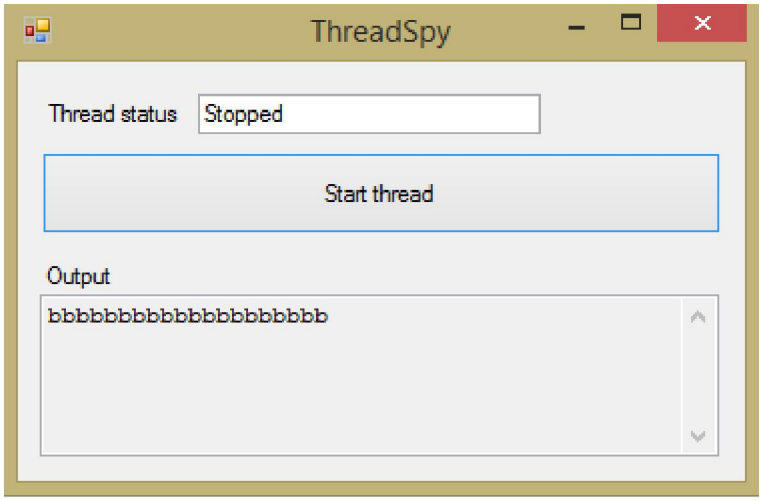
Cette instruction nous permet que lorsque ça fini avec l'impression de 20 caractères pour une lettre, avant de passer à la lettre suivante que ça puisse nettoyer tout dans le champ Output

```
this.TextBoxOutput.Clear();
```

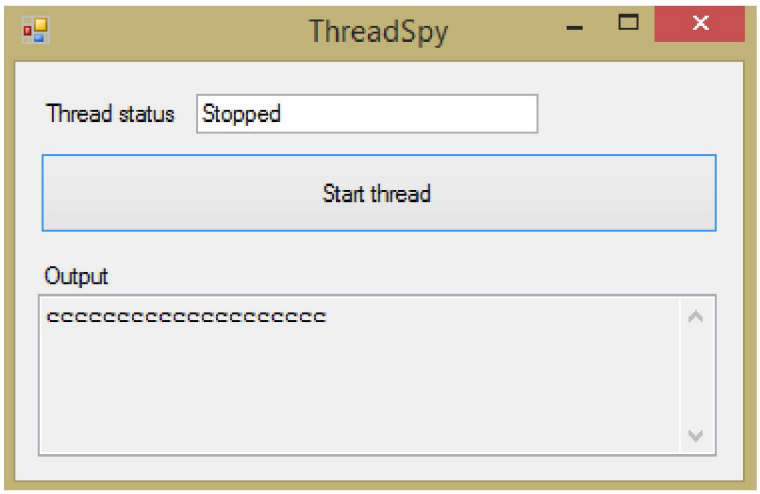
- Le 1er thread en cliquant sur le bouton « Start thread », on obtient l'impression de la lettre « **a** » 20 fois



- Le 2iem thread en cliquant sur le bouton « Start thread », on obtient l'impression de la lettre « **b** » 20 fois



- Le 3iem thread en cliquant sur le bouton « Start thread », on obtient l'impression de la lettre « **c** » 20 fois



- Etc.

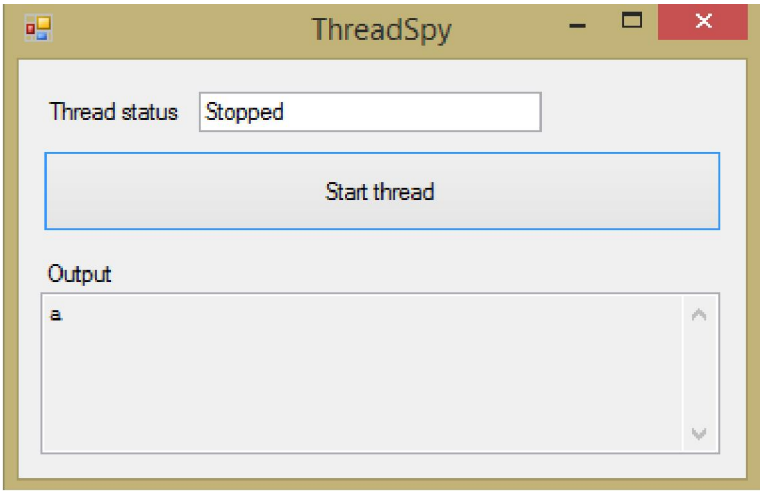
h. Nous avons changes le programme pour que chaque nouveau thread en imprime un de plus caractère que le thread précèdent.

Pour imprimer les caractères de suite, il nous a fallu concaténer la chaine **c** pour avoir le nombre de caractère à afficher et cela avec cette instruction :

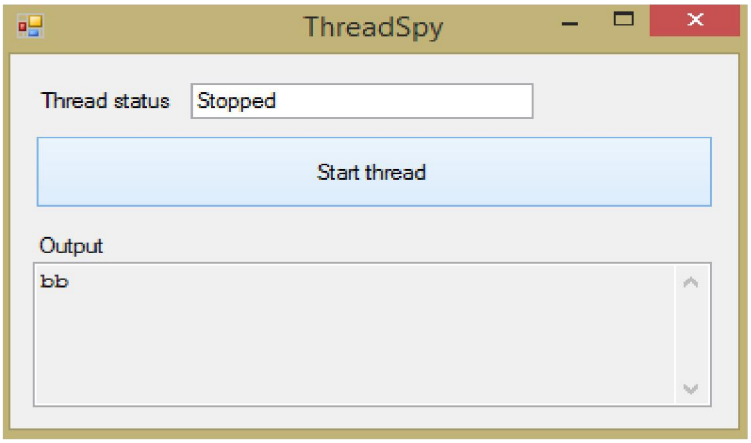
```
do
{
    car += c;
    i = i + 1;
} while (i < cpt);
```

Une variable NrOfchars va nous servir de point de départ et point d'arriver pour savoir le nombre de fois qu'on doit concaténer la chaine.

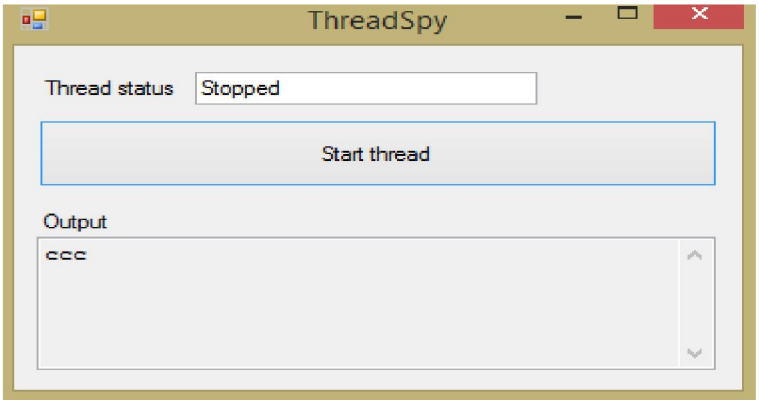
- Le 1er thread imprime une fois la lettre « **a** »



- Le 2ieme thread imprime deux fois la lettre « **b** »



- Le 3ieme thread imprime trois fois la lettre « **c** »



- Etc.