Objets de synchronisation entre threads en C#

ManualResetEvent

ManualResetEvent permet aux threads de communiquer entre eux par la signalisation. En général, cette communication concerne une tâche qu'un thread donné doit réaliser avant que d'autres threads puissent poursuivre.

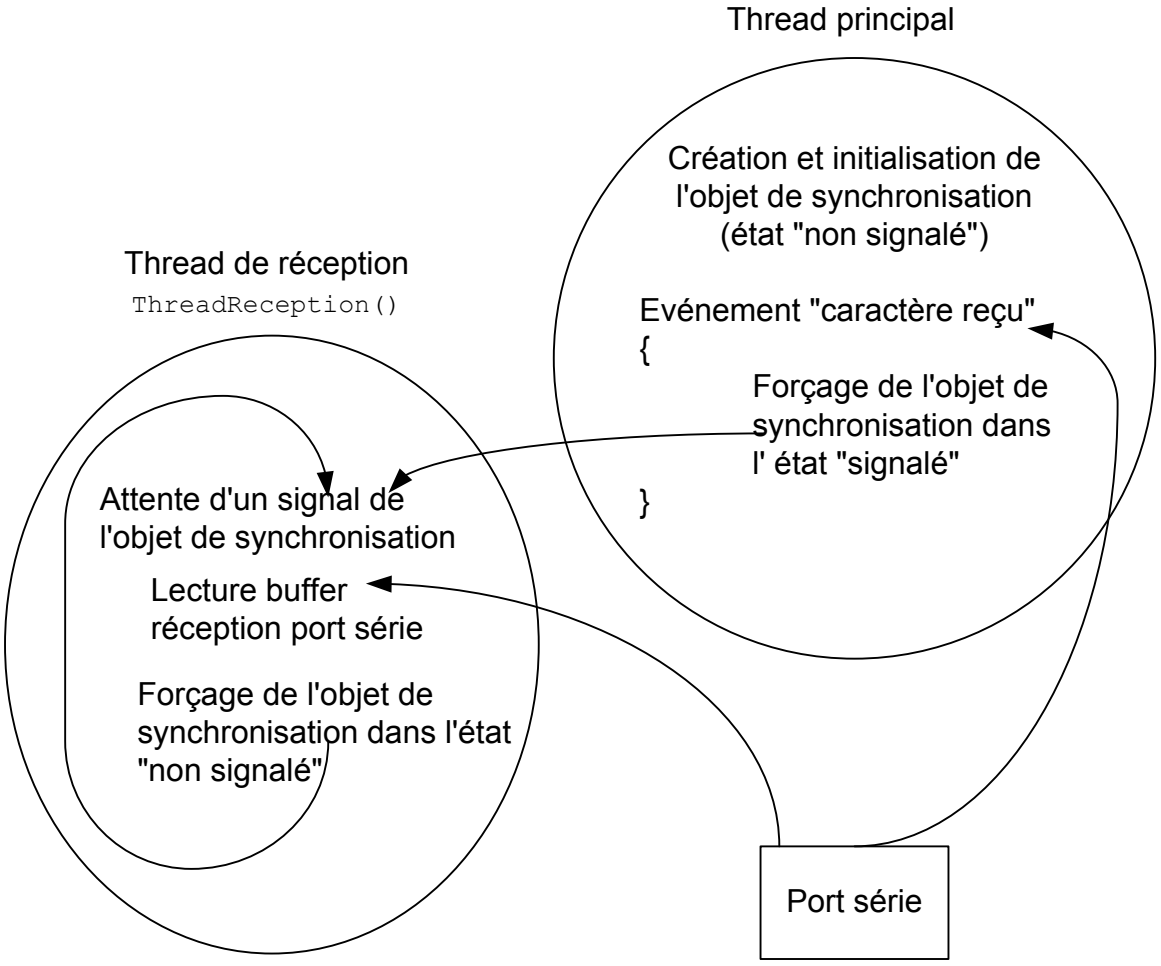
Les threads qui appellent WaitOne() sur ManualResetEvent se bloquent en attendant le signal. Le thread qui "contrôle" ManualResetEvent appelle Set() pour positionner ManuelResetEvent à l'état "signalé" et ainsi signaler aux threads en attente qu'ils peuvent poursuivre. Tous les threads en attente sont libérés.

ManualResetEvent conserve cet état jusqu'à sa réinitialisation manuelle

Cette réinitialisation manuelle est réalisée par l'appel de Reset() de ManuelResetEvent.

Par exemple lorsqu'un thread commence une activité qui doit être terminée avant que d'autres threads puissent s'exécuter, il appelle Reset() pour affecter l'état "non signalé" à ManualResetEvent. Il est possible de contrôler l'état initial d'un objet ManualResetEvent en passant une valeur booléenne au constructeur, true si l'état initial est signalé, sinon false.

ManualResetEvent peut également s'utiliser avec les méthodes staticWaitAll et WaitAny.



```
namespace EssaiSerie
{
    public partial class Form1 : Form
    {
        string MaChaineAEmettre;
        string MaChaineCompleteRecue;
        Thread MonThread;
        ManualResetEvent manualEvent;

        public Form1()
        {
            InitializeComponent();
            MaChaineCompleteRecue = "";
            MonThread = new Thread(new ThreadStart(ThreadReception));
            manualEvent = new ManualResetEvent(false);
        }

        public void ThreadReception()
        {
            // Tant que le thread n'est pas tué, on travaille
            while (Thread.CurrentThread.IsAlive)
            {

                manualEvent.WaitOne();
                string MaChaineRecue = "";
                MaChaineRecue = serialPort1.ReadExisting();
                MaChaineCompleteRecue = MaChaineCompleteRecue + MaChaineRecue;
                manualEvent.Reset();

            }
        }

         private void serialPort1_DataReceived(object sender,
            System.IO.Ports.SerialDataReceivedEventArgs e)
         {
            manualEvent.Set();

         }
```

AutoResetEvent

AutoResetEvent permet aux threads de communiquer entre eux par la signalisation. En général, cette communication concerne une ressource à laquelle les threads doivent accéder de façon exclusive.

Un thread attend un signal en appelant WaitOne sur AutoResetEvent. Si l'objet AutoResetEvent a l'état non signalé, le thread se bloque et attend que le thread qui contrôle actuellement la ressource signale la disponibilité de celle-ci en appelant Set.

L'appel de Set indique à AutoResetEvent de libérer un thread en attente. AutoResetEvent conserve l'état signalé jusqu'à la libération d'un seul thread en attente **puis reprend automatiquement l'état non signalé**. En l'absence de thread en attente, l'état reste indéfiniment signalé.

Remarque importante :

Il n'y a aucune garantie que chaque appel à la méthode Set libère un thread. Si deux appels sont trop proches, et si le second appel se produit avant la libération d'un thread, un seul thread est libéré. Tout se passe comme si le second appel ne s'était pas produit. De même, si Set est appelé lorsqu'il n'y a pas de threads en attente et si AutoResetEvent est déjà signalé, l'appel n'a aucun effet.

Vous pouvez contrôler l'état initial d'un AutoResetEvent en passant une valeur booléenne au constructeur, true si l'état initial est signalé, sinon false.

AutoResetEvent peut également s'utiliser avec les méthodes staticWaitAll et WaitAny.