HỌC PHẦN:	**HỆ ĐIỀU HÀNH**
BÀI:	**LAB 03**
THỜI GIAN:	**8 TIẾT**

GV: Dương Văn Hải

Nội dung

Đồng bộ hóa các tiểu trình sử dụng Monitor

<u>Hướng dẫn và yêu cầu</u>

1) Xem lại cấu trúc tổng quát của bài toán đồng bộ sử dụng cơ chế Sleep and Wakeup

Giải thuật:

```
while (1) {
  if (busy) {
    blocked=blocked+1;
    sleep();
  }
  else busy=1;
  critical_section();
  busy=0;
  if (blocked) {
    wakeup(process);
    blocked=blocked-1;
  }
  noncritical_section();
}
```

2) Giới thiệu cơ chế đồng bộ hóa Monitor (trong C#)

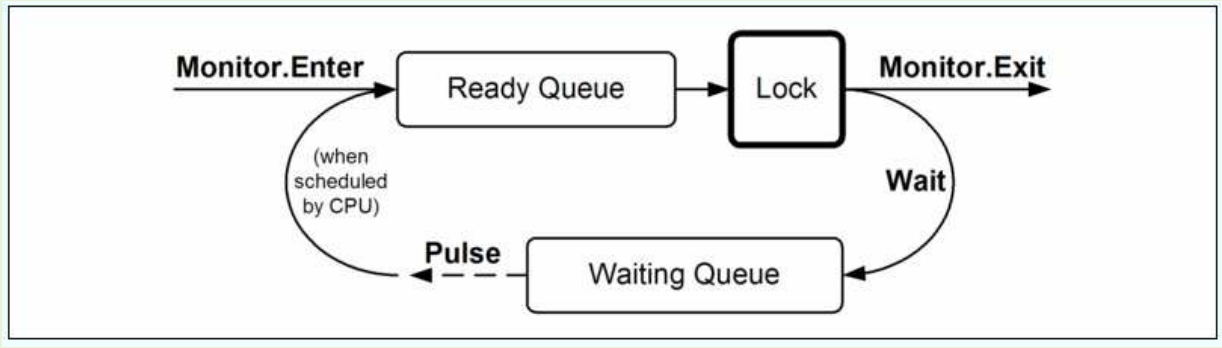
Cơ chế Monitor:

- Là cơ chế đồng bộ hóa thông dụng nhất.
- Lớp này cho phép một tiểu trình đơn thu lấy khóa (lock) trên một đối tượng bằng cách gọi phương thức tĩnh ***Monitor.Enter.*** Bằng cách thu lấy khóa trước khi truy xuất một tài nguyên hay dữ liệu dùng chung, ta chắc chắn rằng chỉ có một tiểu trình có thể truy xuất tài nguyên đó cùng lúc. Một khi đã hoàn tất với tài nguyên, tiểu trình này sẽ giải phóng khóa (unlock) để tiểu trình khác có thể truy xuất nó. Khối mã thực hiện công việc này thường được gọi là vùng CS.
- Có thể sử dụng một đối tượng bất kỳ đóng vai trò làm khóa, và sử dụng từ khóa **this** để thu lấy khóa trên đối tượng hiện tại.
- Khi một tiểu trình đang chiếm giữ khóa, các tiểu trình khác khi muốn lấy khóa trên cùng một đối tượng sẽ bị block và được thêm vào hàng đợi sẵn sàng (ready queue) của khóa này cho đến khi tiểu trình chủ (tiểu trình đang giữ khóa) giải phóng nó bằng phương thức tĩnh ***Monitor.Exit***. Khi tiểu trình chủ gọi Exit, một trong các tiểu trình từ hàng đợi sẵn sàng sẽ thu lấy khóa.

Áp dụng trong C#:

- sử dụng từ khóa ***lock***
- Khối mã được gói trong lệnh ***lock*** tương đương với gọi **Monitor.Enter** khi đi vào khối mã này, và gọi **Monitor.Exit** khi đi ra khối mã này.
- Tiểu trình chủ (sở hữu khóa) có thể gọi ***Monitor.Wait*** để giải phóng khóa và tự đặt nó vào hàng đợi chờ (wait queue). Các tiểu trình trong hàng đợi chờ sẽ bị block cho đến khi tiểu trình chủ gọi phương thức ***Pulse*** hay ***PulseAll*** của lớp Monitor. Phương thức ***Pulse*** di chuyển một trong các tiểu trình từ hàng đợi chờ vào hàng đợi sẵn sàng, còn phương thức ***PulseAll*** thì di chuyển tất cả các tiểu trình.
- Khi một tiểu trình đã được di chuyển từ hàng đợi chờ vào hàng đợi sẵn sàng, nó có thể thu lấy khóa trong lần giải phóng kế tiếp.
- Cần hiểu rằng các tiểu trình thuộc hàng đợi chờ sẽ không thu được khóa, chúng sẽ đợi vô hạn định cho đến khi chúng ta gọi ***Pulse*** hay ***PulseAll*** để di chuyển chúng vào hàng đợi sẵn sàng.

Sơ đồ chuyển đổi trạng thái của tiểu trình khi sử dụng Monitor:



3) Ví dụ

Xây dựng lớp thứ nhất như sau:

```
class ThreadSyncExample
    {
        private static object DoiTuongDungChung = new Object();
        private static void DisplayMessage()
        {
             Console.WriteLine("{0} : Tiểu trình đã bắt đầu, Đang dành Khóa (lock)...",
                DateTime.Now.ToString("HH:mm:ss.ffff"));
            // Thu lấy khóa trên đối tượng consoleGate.
            try
            {
                Monitor.Enter(DoiTuongDungChung);
                Console.WriteLine("{0} :
                {1}",DateTime.Now.ToString("HH:mm:ss.ffff"),
                    "Đã dành được khóa trên đối tượng DoiTuongDungChung, Đang đợi...");
                // Đợi cho đến khi Pulse được gọi trên đối tượng consoleGate.
                Monitor.Wait(DoiTuongDungChung);
                Console.WriteLine("{0}:Tiểu trình tiếp tục chạy, đang kết thúc.",
                    DateTime.Now.ToString("HH:mm:ss.ffff"));
            }
            finally
            {
                Monitor.Exit(DoiTuongDungChung);
```

```
            }
        }

        public void ThreadSync()
        {
            // Thu lấy khóa trên đối tượng DoiTuongDungChung.
            lock (DoiTuongDungChung)
            {
             //Tạo và khởi chạy ba tiểu trình mới (chạy phương thức DisplayMesssage).
                for (int count = 0; count < 3; count++)
                {
                    (new Thread(new ThreadStart(DisplayMessage))).Start();
                }
            }
            Thread.Sleep(1000);
            // Đánh thức một tiểu trình đang chờ.
            Console.WriteLine("{0} : {1}", DateTime.Now.ToString("HH:mm:ss.ffff"),
                              "Nhấn Enter để lấy một tiểu trình đang chờ.");
            Console.ReadLine();
            // Thu lấy khóa trên đối tượng DoiTuongDungChung.
            lock (DoiTuongDungChung)
            {
                // Pulse một tiểu trình đang chờ.
                Monitor.Pulse(DoiTuongDungChung);
            }
            // Đánh thức tất cả các tiểu trình đang chờ.
            Console.WriteLine("{0} : {1}", DateTime.Now.ToString("HH:mm:ss.ffff"),
                                "Nhấn Enter để lấy tất cả các tiểu trình đang chờ.");
            Console.ReadLine();
            // Thu lấy chốt trên đối tượng consoleGate.
            lock (DoiTuongDungChung)
            {
                // Pulse tất cả các tiểu trình đang chờ.
                Monitor.PulseAll(DoiTuongDungChung);
            }
            // Nhấn Enter để kết thúc.
            Console.WriteLine("Hàm Main đã hoàn thành. Nhấn Enter.");
            Console.ReadLine();
        }

    }
```

Xây dựng lớp chính để gọi lớp trên như sau:

```
class Program
    {   static void Main(string[] args)
        {
            ThreadSyncExample t = new ThreadSyncExample();
            t.ThreadSync();
        }
     }
```

4) Giải thích ý nghĩa của từng dòng lệnh và ý nghĩa của cả chương trình trên.

5) Sử dụng cơ chế Monitor và ví dụ đã cho ở trên để cài đặt bài toán sản xuất và tiểu thụ với gợi ý như sau:

   - Xây dựng một lớp KhoHang có 2 phương thức *Ghi dữ liệu vào kho* và *Đọc dữ liệu từ kho*. Cả hai phương thức này có sử dụng cơ chế đồng bộ Monitor để loại trừ lẫn nhau.
   - Xây dựng lớp SanXuat để ghi N mục dữ liệu vào kho.
   - Xây dựng lớp TieuThu để lấy N mục dữ liệu từ kho.
   - Xây dựng lớp chính (có hàm Main) để khởi tạo 2 tiểu trình (một tiểu trình của lớp sản xuất và một tiểu trình của lớp tiêu thụ).

6) Xây dựng chương trình có sử dụng cơ chế Monitor để giải bài toán sản xuất phân tử nước.

WISH YOU ACHIEVE GREAT SUCCESS IN OUR LAB03