Bài 12.8: Hủy luồng

- ✓ Tổng quan
- ✓ Các kiểu hủy luồng
- ✓ Lắng nghe và phản hồi yêu cầu hủy luồng

Tổng quan

- ✓Từ phiên bản .NET framework 4, .NET sử dụng một mô hình hợp nhất gọi là sự hủy hợp tác để hủy các tác vụ chạy bất đồng bộ.
- ✓Mô hình này dựa trên một đối tượng gọi là mã thông báo hủy(cancellation token).
- ✓Các bước thực hiện của mô hình hủy hợp tác:
  - ✓ B1: tạo một đối tượng CancellationTokenSource dùng để quản lý và gửi các thông báo hủy tới từng token hủy.
  - ✓ B2: truyền token trả về bởi CancellationTokenSource.Token vào mỗi tác vụ hoặc luồng đang lắng nghe sự kiện hủy.
  - ✓ B3: triển khai cách thức phản hồi của mỗi task hoặc thread khi nhận được mã hủy.
  - ✓ B4: gọi Cancel method từ đối tượng tạo ở B1 để tạo và gửi thông báo hủy luồng.

Tổng quan

- ✓Mô hình này hỗ trợ các tính năng sau:
  - ✓ Việc hủy bỏ dựa trên tinh thần hợp tác và không bắt buộc với nơi lắng nghe. Nơi lắng nghe sự kiện hủy sẽ quyết định cách thức triển khai việc hủy tốt nhất theo mong muốn.
  - ✓ Yêu cầu hủy độc lập với việc lắng nghe. Một đối tượng gọi hành động hủy bỏ có thể kiểm soát khi nào(nếu có) việc hủy được kích hoạt.
  - ✓ Đối tượng yêu cầu hủy đưa ra thông báo hủy với tất cả các bản sao của mã thông báo chỉ bằng 1 lời gọi tới phương thức.
  - ✓ Nơi lắng nghe có thể đăng ký nhiều mã thông báo đồng thời bằng cách kết hợp chúng vào một mã thông báo được liên kết.
  - ✓ Code của người dùng và code trong thư viện có thể gửi thông báo và phản hồi tới yêu cầu hủy của nhau.
  - ✓ Nơi lắng nghe có thể được thông báo về các yêu cầu hủy bỏ bằng cách thăm dò, đăng ký callback, hoặc chờ đợi.

Các kiểu hủy luồng

- ✓CancellationTokenSource: đối tượng tạo một mã hủy và đưa ra yêu cầu hủy tới tất cả các bản sao của mã hủy đó.
- ✓CancellationToken: kiểu gọn nhẹ value type được truyền vào 1 hoặc nhiều nơi lắng nghe, thường dưới dạng tham số của phương thức. Nơi lắng nghe sẽ giám sát giá trị của thuộc tính IsCancellationRequested bằng cách thăm dò, callback hoặc chờ xử lý.
- ✓OperationCanceledException: nạp chồng constructor của exception này cho phép nơi lắng nghe tự văng ngoại lệ để xác minh nguồn gốc của sự hủy và thông báo tới bên khác rằng đã phản hồi yêu cầu hủy.
- ✓Các kiểu được triển khai của mô hình hủy hợp tác trong .NET: System.Threading.Tasks.Parallel, Task, Task<Tresult>, Linq.ParallelEnumerable.

Hủy hành động vs hủy đối tượng

- ✓Mô hình hủy hợp tác trong thư viện ngầm định tới việc hủy các hành động, không phải hủy các đối tượng.
- ✓Yêu cầu hủy có nghĩa là hành động đó phải kết thúc càng sớm càn tốt ngay sau khi bất kỳ yêu cầu dọn dẹp nào được thực hiện.
- ✓Sau khi thuộc tính IsCancellationRequested được thiết lập giá trị true, nó không thể reset về false. Do đó mã hủy không thể tái sử dụng sau khi nó đã được hủy.

Lắng nghe và phản hồi yêu cầu hủy

- ✓Trong delegate của người dùng, nơi triển khai của hành động có thể hủy bỏ sẽ xác định cách kết thúc hành động đó khi nhận được yêu cầu hủy.
- ✓Trong nhiều trường hợp, các delegate này chỉ thực hiện các hành động dọn dẹp được yêu cầu và sau đó kết thúc luôn.
- ✓Với các trường hợp phức tạp hơn, có thể cần phải thông báo tới code trong thư viện rằng hành động hủy bỏ đã xảy ra. Trong trường hợp này, để kết thúc hành động đang thực hiện ta gọi ThrowIfCancellationRequested().
- ✓Hành động trên kích hoạt ngoại lệ OperationCanceledException.
- ✓Lắng nghe bằng thăm dò: thường áp dụng với các tính toán lặp đi lặp lại hoặc đệ quy.
- ✓Lắng nghe bằng cách đăng ký callback: áp dụng với các hành động có thể bị phong tỏa theo cách mà nó không thể kiểm tra được giá trị của mã hủy một cách kịp thời.

Ví dụ lắng nghe thăm dò

```
0 references | 0 changes | 0 authors, 0 changes
static void Main()
{
    Console.OutputEncoding = Encoding.UTF8;
    CancellationTokenSource cts = new CancellationTokenSource();
    ThreadPool.QueueUserWorkItem(new WaitCallback(DoSomeWork), cts.Token);
    Thread.Sleep(4000);

    // tiến hành hủy
    cts.Cancel();
    Console.WriteLine("==> Tiến hành hủy luồng...");
    Thread.Sleep(3000);
    cts.Dispose(); // giải phóng đối tượng
}

1 reference | 0 changes | 0 authors, 0 changes
static void DoSomeWork(object obj)
{
    CancellationToken token = (CancellationToken)obj;
    for (int i = 0; i < 100000; i++)
    {
        if(token.IsCancellationRequested)
        {
            Console.WriteLine("==> Yêu cầu hủy được thực thi ở vòng lặp thứ " + (i + 1));
            Console.WriteLine("...");
            break;
        }
        Thread.SpinWait(50000);
    }
}
```

Ví dụ lắng nghe sử dụng callback

```
0 references | 0 changes | 0 authors, 0 changes
static void Main()
{
    CancellationTokenSource cts = new CancellationTokenSource();
    Console.OutputEncoding = Encoding.UTF8;
    StartWebRequest(cts.Token);
    Console.WriteLine("==> Nhấn phím bất kỳ để kết thúc: ");
    Console.ReadKey();
    Console.WriteLine();
    cts.Cancel();
}

1 reference | 0 changes | 0 authors, 0 changes
static void StartWebRequest(CancellationToken token)
{
    var target = "https://braniumacademy.net";
    var edgePath = @"C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe";
    Console.WriteLine("==> Bắt đầu mở yêu cầu truy cập trang web...");
    StartProcess(edgePath, target);
    token.Register(() =>
    {
        StopProcess("msedge");
        Console.WriteLine("==> Đã hủy yêu cầu mở trang web.");
    });
}
```

Lắng nghe và phản hồi yêu cầu hủy

- ✓Lắng nghe bằng cách sử dụng một trình chờ đợi sự kiện: áp dụng cho các hành động có thể hủy bỏ đang chờ đợi một sự đồng bộ nguyên thủy và có thể bị phong tỏa.
- ✓Lắng nghe nhiều mã hủy cùng lúc: áp dụng với các hành động cần phải lắng nghe cùng lúc nhiều mã hủy. Lúc này ta tạo một token liên kết để liên kết các token lại.

Nội dung tiếp theo

Đồng bộ hóa dữ liệu trong đa luồng