Waiting for a Set of Tasks to Complete

The framework provides a **Task.WhenAll** method for this purpose. This method takes

several tasks and returns a task that completes when all of those tasks have completed:

Waiting for Any Task to Complete

You have several tasks and need to respond to just one of them completing. The most

common situation for this is when you have multiple independent attempts at an operation, with a first-one-takes-all kind of structure. For example, you could request stock quotes from multiple web services simultaneously, but you only care about the first one that responds.

using System;

using System.Collections.Generic;

using System.Diagnostics;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication1

{

class Program

{

static void Main()

{

Task taskA =ProcessTasksAsync();

taskA.Wait();

Console.ReadLine();

}

static async Task ProcessTasksAsync()

{

// Create a sequence of tasks.

Task<int> taskA = DelayAndReturnAsync(2);

Task<int> taskB = DelayAndReturnAsync(3);

Task<int> taskC = DelayAndReturnAsync(1);

// Await all processing to complete

int [] t= await Task.WhenAll(taskA, taskB,taskC);

foreach(var x in t)

Console.WriteLine(x);

}

static async Task<int> DelayAndReturnAsync(int val)

{

await Task.Delay(TimeSpan.FromSeconds(val));

return val;

}

}

}

Use the **Task.WhenAny** method. This method takes a sequence of tasks and returns a

task that completes when any of the tasks complete. The result of the returned task isthe task that completed.

using System;

using System.Collections.Generic;

using System.Diagnostics;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication1

{

class Program

{

static void Main()

{

Task taskA =ProcessTasksAsync();

taskA.Wait();

Console.ReadLine();

}

static async Task ProcessTasksAsync()

{

// Create a sequence of tasks.

Task<int> taskA = DelayAndReturnAsync(2);

Task<int> taskB = DelayAndReturnAsync(3);

Task<int> taskC = DelayAndReturnAsync(1);

// Await all processing to complete

Task<int> t= await Task.WhenAny(taskA, taskB);

Console.WriteLine(t.Result);

}

static async Task<int> DelayAndReturnAsync(int val)

{

await Task.Delay(TimeSpan.FromSeconds(val));

return val;

}

}

}