

Forum:[Hack Forums](#)

Thread: [Parallelize HttpWebRequest \[Really Hard C#\]](#)

FryPan - July 4, 2018, 11:39 am

Hi there,

What I'm trying to do:

I log into a bunch of accounts (instances of Account.cs) on the program.

Each account is set a TXT of people (realIDs) to message.

They all do it at the same time.

[Account.cs](#):

myId is my own ID (not needed in send message method).

authToken is used to login (not needed in send message method).

[HttpMethods.cs](#):

realId above is the ID of a person to message.

If anyone can solve this one for me, I'm even willing to reward. This shit is frying my brain. ?

Washi - July 4, 2018, 11:43 am

(Jul 4, 2018 11:39 AM)FryPan Wrote: [\[CITE\]](#) If anyone can solve this one for me, I'm even willing to reward. This shit is frying my brain. ?

Solve what exactly? You did not specify the problem that you are having?

Charger - July 4, 2018, 11:45 am

Read up on Multi Threading ([example](#))

FryPan - July 4, 2018, 11:46 am

(Jul 4, 2018 11:45 AM)Charger Wrote: [\[CITE\]](#) Read up on Multi Threading ([example](#))

Yes, I know about multi threading, but I'm asking on help to set it up:

IE: Parallel.For the account instances, and then normal loop through the TXT realIDs?

(Jul 4, 2018 11:43 AM)Washi Wrote: [\[CITE\]](#) Solve what exactly? You did not specify the problem that you are having?

Sorry for not making it clear. I'm trying to:

Import a bunch of accounts (Account acc = new Account("user", "pass"))

Assign a txt of realIDs/Accounts to message (contains 1000 users)

They all do it at the same time.

Charger - July 4, 2018, 11:49 am

(Jul 4, 2018 11:46 AM)FryPan Wrote: [\[CITE\]](#) Yes, I know about multi threading, but I'm asking on help to set it up:

IE: Parallel.For the account instances, and then normal loop through the TXT realIDs?

Sorry for not making it clear. I'm trying to:

Import a bunch of accounts (Account acc = new Account("user", "pass"))

Assign a txt of realIDs/Accounts to message (contains 1000 users)

They all do it at the same time.

You can't fire 1000 requests at the same time. Your computer can't handle it nor can the server. What you are trying to do is what multi threading does.

Zettabit - July 4, 2018, 12:06 pm

Also, directly supplying the 'message' into your preset JSON will break if the message is not in proper format.

I would also add some sort of response checking to make sure the server returned that the message was sent successfully.

Tommy Chong - July 4, 2018, 1:42 pm

Multithreading is exactly what you need.

I used the threadpool for a similar purpose.

<https://www.dotnetperls.com/threadpool>

```
Code for (int i = 0; i < _maxThreads; i++)
{
    ThreadPool.QueueUserWorkItem(new WaitCallback(Worker), i);
}
```

```
private void Worker(object state)
{
    // call your method that does the HTTP requests
}
```

FryPan - July 4, 2018, 2:01 pm

(Jul 4, 2018 01:42 PM)Tommy Chong- Wrote: [\[CITE\]](#) Multithreading is exactly what you need.

I used the threadpool for a similar purpose.

<https://www.dotnetperls.com/threadpool>

```
Code for (int i = 0; i < _maxThreads; i++)
{
    ThreadPool.QueueUserWorkItem(new WaitCallback(Worker), i);
}
```

```
private void Worker(object state)
{
    // call your method that does the HTTP requests
}
```

Thanks for the only message on topic. Any chance you know of an example that is similar but uses tasks?

(Jul 4, 2018 11:49 AM)Charger Wrote: [\[CITE\]](#) You can't fire 1000 requests at the same time. Your computer can't handle it nor can the server.

What you are trying to do is what multi threading does.

You read the thread wrong

Tommy Chong - - July 4, 2018, 4:09 pm

(Jul 4, 2018 02:01 PM)FryPan Wrote: [\[CITE\]](#) Thanks for the only message on topic. Any chance you know of an example that is similar but uses tasks?

You read the thread wrong

Is there any reason why you want to use tasks over threadpool?

alexconnolly - July 5, 2018, 3:44 am

More control over a task than a thread pool.

Thread pool is like a magic invisible wall kinda like a glory hole where you just shove shit in and you see what happens.

Tasks allows more control over this shit and is non-glory hole version.

A good read is here for OP wanting to use tasks:

docs.microsoft.com/en-us/dotnet/standard/parallel-programming/task-based-asynchronous-programming

FryPan - July 5, 2018, 6:06 am

(Jul 5, 2018 03:44 AM)alexconnolly Wrote: [\[CITE\]](#) More control over a task than a thread pool.

Thread pool is like a magic invisible wall kinda like a glory hole where you just shove shit in and you see what happens.

Tasks allows more control over this shit and is non-glory hole version.

A good read is here for OP wanting to use tasks:

docs.microsoft.com/en-us/dotnet/standard/parallel-programming/task-based-asynchronous-programming

Thanks bud. Nice analogy lmao.

Tommy Chong - July 5, 2018, 10:10 am

(Jul 5, 2018 03:44 AM)alexconnolly Wrote: [\[CITE\]](#) More control over a task than a thread pool.

Thread pool is like a magic invisible wall kinda like a glory hole where you just shove shit in and you see what happens.

Tasks allows more control over this shit and is non-glory hole version.

A good read is here for OP wanting to use tasks:

docs.microsoft.com/en-us/dotnet/standard/parallel-programming/task-based-asynchronous-programming

You are right about this, but for the simple tasks OP need to perform, the threadpool should be sufficient enough

FryPan - July 5, 2018, 1:30 pm

(Jul 5, 2018 10:10 AM)Tommy Chong- Wrote: [\[CITE\]](#) You are right about this, but for the simple tasks OP need to perform, the threadpool should be sufficient enough

I'm going to be using Parallel.For & then a normal loop in that one.

Closed User - July 5, 2018, 2:52 pm

Definitely tasks, people mentioning threadpool should get lynched.

Tasks provides more control, way easier debugging and cleaner code

EDIT: If you use async that is, if you don't use async might aswell use backgroundworker lol

flAmingw0rm - July 5, 2018, 2:57 pm

You can also look up [SemaphoreSlim](#). It allows you to control the maximum amount of tasks you can run in parallel.

Closed User - July 5, 2018, 2:58 pm

(Jul 5, 2018 02:57 PM)flAmingw0rm Wrote: [\[CITE\]](#) You can also look up [SemaphoreSlim](#). It allows you to control the maximum amount of tasks you can run in parallel.

Nevermind, missread.

You can still control that very easily without Semaphore though