

Forum: [Hack Forums](#)

Thread: [Simple Dork Scanner](#)

Decim - May 5, 2016, 6:54 am

Dork Scanner

(No offence if you're the owner of that website, just picked a random one from the Website sub-forum.)

Wrote this in my free time.

Nothing special, just a simple program that scans a URL against a given list of dorks.

It's pretty fast and hasn't given me any issues.

Put the dorks in the executable directory with the name 'dorks.txt' or set your own path in the program.

Source: <https://github.com/arcanecfg/Dork-Scanner/>

//Someone should probably move this to the C# section, posted it here by accident.

Enjoy.

2344lkn - May 5, 2016, 12:18 pm

Nice work Arcane. Love how simple this is and gives anyone the ability to add there own dorks or paths. Not only that you are even presenting a nice report at the end gj!

BahNahNah - May 6, 2016, 1:58 am

Some small suggestions:

-Length is faster than .Count() and should be used when possable
Though the performance difference dosent realy matter here, its good practice.

-Use the Console.WriteLine formatting

This:

Code Console.WriteLine("Successfully loaded " + dorkList.Count() + " dorks.");Can be

Code Console.WriteLine("Successfully loaded {0} dorks.", dorkList.Length);

//or

Console.WriteLine(\$"Successfully loaded {dorkList.Length} dorks.");

-Initial variable values

Spoiler [\(Click to View\)](#)

Why not

Code string dorkPath = "dorks.txt";

Personal prefrence, this deceleration is a bit odd

Code string[] dorkList = { string.Empty };If you want to initialize an empty array

Code `string[] dorkList = new string[];`
//or even just null
`string[] dorkList = null;`
-No checking before string modifications
Spoiler [\(Click to View\)](#)

There are no checks to verify that the string is not empty. If there is a blank line in the file, this will throw an error.

-Not disposing of objects
Good practice, dispose your objects when you are finished with them.
in the startScan function, this
Spoiler [\(Click to View\)](#)

Can be
Code using `(StreamReader sReader = new StreamReader(newRes.GetResponseStream()))`
{
 `myResp = sReader.ReadToEnd();`
}Or, if you want to dispose of them manually
Code `StreamReader sReader = new StreamReader(newRes.GetResponseStream());`
`myResp = sReader.ReadToEnd();`
`sReader.Dispose();`They do the same thing.

-Personal preference, don't check against true/false in IF statements
Code `if (myResp.Contains("404") == false)`Can be
Code `if (!myResp.Contains("404"))`

Aveling - May 6, 2016, 1:59 am

Thank you for sharing the source code, it will help people like me understand this kind of programs better.

Decim - May 6, 2016, 4:46 am

(May 5, 2016 12:18 PM)2344lkn Wrote: [\[CITE\]](#) Nice work Arcane. Love how simple this is and gives anyone the ability to add their own dorks or paths. Not only that you are even presenting a nice report at the end gj!Glad you liked it mate.

(May 6, 2016 01:58 AM)BahNahNah Wrote: [\[CITE\]](#) Some small suggestions:

-.Length is faster than .Count() and should be used when possible
Though the performance difference doesn't really matter here, it's good practice.

-Use the Console.WriteLine formatting
This:
Code `Console.WriteLine("Successfully loaded " + dorkList.Count() + " dorks.");`Can be
Code `Console.WriteLine("Successfully loaded {0} dorks.", dorkList.Length);`
//or
`Console.WriteLine($"Successfully loaded {dorkList.Length} dorks.");`
-Initial variable values
Spoiler [\(Click to View\)](#)

Why not

```
Code string dorkPath = "dorks.txt";
```

Personal preference, this deceleration is a bit odd

```
Code string[] dorkList = { string.Empty };If you want to initialize an empty array
```

```
Code string[] dorkList = new string[];
```

```
//or even just null
```

```
string[] dorkList = null;
```

-No checking before string modifications

Spoiler ([Click to View](#))

There are no checks to verify that the string is not empty. If there is a blank line in the file, this will throw an error.

-Not disposing of objects

Good practice, dispose your objects when you are finished with them.

in the startScan function, this

Spoiler ([Click to View](#))

Can be

```
Code using (StreamReader sReader = new StreamReader(newRes.GetResponseStream()))
{
```

```
    myResp = sReader.ReadToEnd();
```

```
}Or, if you want to dispose of them manually
```

```
Code StreamReader sReader = new StreamReader(newRes.GetResponseStream());
```

```
myResp = sReader.ReadToEnd();
```

```
sReader.Dispose();They do the same thing.
```

-Personal preference, don't check against true/false in IF statements

```
Code if (myResp.Contains("404") == false)Can be
```

```
Code if (!myResp.Contains("404"))I really appreciate you taking the time to go through the code thoroughly and giving me suggestions.
```

Some of the stuff, the pointless initialisation and forgetting to close the StreamReader for example, was the result of my careless debugging.

The console line formatting is just a bad habit that stuck with me, I'll try my best to fight it in the future.

Never knew .Length was faster than Count(), I'll keep that in mind as well.

Thanks again.

(May 6, 2016 01:59 AM)iNeo19 Wrote: ([CITE](#)) Thank you for sharing the source code, it will help people like me understand this kind of programs better.Happy to hear that.

2344lkn - May 6, 2016, 8:35 am

Thanks BahNahNah for the optimizations and better practices. I've been messing around with C# for around a year I'm no expert but I picked up on a few things thanks again.

SL x TnT - May 6, 2016, 2:02 pm

(May 6, 2016 04:46 AM)Arcanecfg Wrote: ([CITE](#)) Glad you liked it mate.

I really appreciate you taking the time to go through the code thoroughly and giving me suggestions.

Some of the stuff, the pointless initialisation and forgetting to close the StreamReader for example, was the result of my careless debugging.

The consoleline formatting is just a bad habit that stuck with me, I'll try my best to fight it in the future.

Never knew .Length was faster than Count(), I'll keep that in mind as well.

Thanks again.

Happy to hear that.

Adding on to his post a little.

You're doing a lot of string manipulation when you don't have to.

```
Code targetURL = Console.ReadLine();
```

```
//... firewall didn't like this line
```

You never check to see if the address is valid until later when it'll throw an exception.

If you use Uri.TryCreate here, you can make sure their input is valid. From there, use the Host property.

Read through the methods (GetLeftPart, specifically) and properties. You'll find some stuff that'll make your life easier.

Using a WebClient will make your code easier to read.

```
Code if (myResp.Contains("404") == false)
```

You're more likely to get a 404 exception than a site giving a 200 and displaying a 404 page. If any of the pages you request contain "404," it'll put it in the "bad" list when it's valid.

Catch as a WebException. From there, Google how to get the status code and determine which errors you want to determine as "bad." For example, a "timeout" might not be something you want to determine as bad.

Decim - May 6, 2016, 7:55 pm

(May 6, 2016 02:02 PM)SL x TnT Wrote: [\[CITE\]](#) Adding on to his post a little.

You're doing a lot of string manipulation when you don't have to.

```
Code targetURL = Console.ReadLine();
```

```
//... firewall didn't like this line
```

You never check to see if the address is valid until later when it'll throw an exception.

If you use Uri.TryCreate here, you can make sure their input is valid. From there, use the Host property.

Read through the methods (GetLeftPart, specifically) and properties. You'll find some stuff that'll make your life easier.

Using a WebClient will make your code easier to read.

```
Code if (myResp.Contains("404") == false)
```

You're more likely to get a 404 exception than a site giving a 200 and displaying a 404 page. If any

of the pages you request contain "404," it'll put it in the "bad" list when it's valid.

Catch as a `WebException`. From there, Google how to get the status code and determine which errors you want to determine as "bad." For example, a "timeout" might not be something you want to determine as bad.

I probably should be checking the URL first, thanks.

As for the 404, some websites have custom error pages set up and they won't return an exception when the URL is invalid.

SL x TnT - May 6, 2016, 9:39 pm

(May 6, 2016 07:55 PM)Arcanecfg Wrote: [\[CITE\]](#) I probably should be checking the URL first, thanks.

As for the 404, some websites have custom error pages set up and they won't return an exception when the URL is invalid.

Then they should fix their site.

If they have a custom 404 page that doesn't have a 404 status, it's impossible to reliably check it.

Ooze - May 16, 2016, 6:45 am

posting here to remind myself later

mich3k3r - June 16, 2019, 7:38 pm

nice these post can be usefull thanks for sharing

deiski - June 17, 2019, 9:16 pm

(May 6, 2016 01:58 AM)BahNahNah Wrote: [\[CITE\]](#) Some small suggestions:

-`.Length` is faster than `.Count()` and should be used when possible

Though the performance difference doesn't really matter here, it's good practice.

-Use the `Console.WriteLine` formatting

This:

`Code Console.WriteLine("Successfully loaded " + dorkList.Count() + " dorks.");`Can be

`Code Console.WriteLine("Successfully loaded {0} dorks.", dorkList.Length);`

`//or`

`Console.WriteLine($"Successfully loaded {dorkList.Length} dorks.");`

-Initial variable values

Spoiler [\(Click to View\)](#)

Why not

`Code string dorkPath = "dorks.txt";`

Personal preference, this declaration is a bit odd

`Code string[] dorkList = { string.Empty };`If you want to initialize an empty array

`Code string[] dorkList = new string[];`

`//or even just null`

`string[] dorkList = null;`

-No checking before string modifications

Spoiler [\(Click to View\)](#)

There are no checks to verify that the string is not empty. If there is a blank line in the file, this will

throw an error.

-Not disposing of objects

Good practice, dispose your objects when you are finished with them.
in the startScan function, this

Spoiler ([Click to View](#))

Can be

```
Code using (StreamReader sReader = new StreamReader(newRes.GetResponseStream()))  
{
```

```
    myResp = sReader.ReadToEnd();
```

```
}Or, if you want to dispose of them manually
```

```
Code StreamReader sReader = new StreamReader(newRes.GetResponseStream());
```

```
myResp = sReader.ReadToEnd();
```

```
sReader.Dispose();They do the same thing.
```

-Personal preference, don't check against true/false in IF statements

Code if (myResp.Contains("404") == false)Can be

Code if (!myResp.Contains("404"))Very informative post, thanks heaps.

gigajew - June 17, 2019, 9:19 pm

I believe this is called a admin panel scanner and you may be able to achieve faster results with queue and socket combination

Thanks for the contribution keep it up champ