

Forum:[Hack Forums](#)

Thread: [Splitting text into strings](#)

K???er - July 13, 2017, 6:07 pm

If I have a .txt file with this for example:

```
username1:password1
username2:password2
username3:password3
```

How am I able to load first username1 into the "username" string and password2 into the "password" string? And then once that's done by logging in, going back to the login screen and then loading username2 in the "username" string and password2 in the "password" string?

```
string Accounts = File.ReadAllText("C:\\file.txt");
```

I tried this but then I end up getting all the accounts in that string and that's not how I want it.

I know this sounds confusing, but I can't explain it better. I hope someone can help me out.

Thanks!

0xDEAD10CC - July 13, 2017, 6:48 pm

File.ReadLines/ReadAllLines() and string.Split() function. It's not confusing at all and pretty simple to do.

K???er - July 14, 2017, 3:27 am

(Jul 13, 2017 06:48 PM)0xDEAD10CC Wrote: [\[CITE\]](#) File.ReadLines/ReadAllLines() and string.Split() function. It's not confusing at all and pretty simple to do.

Yeah, but how can I make from the part before the ":" a string and from the part after it another string?

Abducted - July 14, 2017, 3:32 am

```
Code List<string[2]> accounts = new List<string[2]>();
List<string> file = File.ReadAllLines("C:\\file.txt").ToList();
foreach (int line in file) : accounts.Append(line.Split(":", StringSplitOptions.None));
```

Might work, might not. But you get the idea.

0xDEAD10CC - July 14, 2017, 2:25 pm

(Jul 14, 2017 03:27 AM)K???er Wrote: [\[CITE\]](#) Yeah, but how can I make from the part before the ":" a string and from the part after it another string?

If you really "knew" you would have been able to comprehend that I gave you the solution. I already answered this question...

Siy? - July 14, 2017, 6:00 pm

You find your way to get a string, line for line, then you use this:

```
string username = line.Split(':')[0];
string password = line.Split(':')[1];
```

Oliver Sykes - July 14, 2017, 10:16 pm

protip: You won't just call `.Split()` once. It would be easy to spoonfeed code, but trust me, it makes it a pain in the ass to learn later on.

temp19 - July 15, 2017, 12:20 am

Go in Google.com and search how you can read a TXT File.

Then search how does `String.Split` function works.

And then figure out by your self .

idb. - July 15, 2017, 2:46 am

(Jul 14, 2017 06:00 PM)Siy? Wrote: [\[CITE\]](#) You find your way to get a string, line for line, then you use this:

```
string username = line.Split(':')[0];
string password = line.Split(':')[1];
What if the password contains a :?
```

OP, use `SubString` and `IndexOf`.

BahNahNah - July 15, 2017, 3:59 pm

(Jul 15, 2017 02:46 AM)idb. Wrote: [\[CITE\]](#) What if the password contains a :?

OP, use `SubString` and `IndexOf`.

This should work too.

```
Code string[] split = line.Split(new char[] { ':' }, 2);
```

0xDEAD10CC - July 15, 2017, 5:52 pm

(Jul 15, 2017 02:46 AM)idb. Wrote: [\[CITE\]](#) What if the password contains a :?

OP, use `SubString` and `IndexOf`.

`Split` still works as shown..

idb. - July 15, 2017, 10:17 pm

(Jul 15, 2017 03:59 PM)BahNahNah Wrote: [\[CITE\]](#) This should work too.

```
Code string[] split = line.Split(new char[] { ':' }, 2);
```

Mmmmmmmmm. Forgot about that overload of `String.Split`. Good one bud!

(Jul 15, 2017 05:52 PM)0xDEAD10CC Wrote: [\[CITE\]](#) `Split` still works as shown..

And in the instance I quoted, under the circumstances I stated, would return an inaccurate password. Bah's solution is accurate. Other dudes was not.

0xDEAD10CC - July 17, 2017, 8:08 am

(Jul 15, 2017 10:17 PM)idb. Wrote: [\[CITE\]](#) Mmmmmmmmm. Forgot about that overload of `String.Split`. Good one bud!

And in the instance I quoted, under the circumstances I stated, would return an inaccurate password. Bah's solution is accurate. Other dudes was not.

In that context it seemed like you were suggesting to use the methods `Substring()` and `IndexOf()` over `Split()` entirely for the one reason you mentioned, and didn't suggest the alternative overload of `IndexOf()` at all. In that context it seemed like you were blaming the `Split()` function. People who

mentioned the function could have been correct, it was just that one snippet that was not.

And my analysis of your post appears to be correct too:

Quote:Forgot about that overload of `String.Split`. Good one bud!

I wasn't saying that his snippet with `Split()` was right, I was saying that `Split()` still works if used properly, which is still a true statement regardless of the context of your post.