Keep your eye on the code page: Is this string CP_ACP or UTF-8?

devblogs.microsoft.com/oldnewthing/20150611-00

June 11, 2015

Raymond Chen

A customer had a problem with strings and code pages.

> The customer has a password like `"Müllwagen"` for a particular user. Note the umlaut over the *u*. That character is encoded as the two bytes `C3 BC` according to UTF-8. When the customer passes this password to the `LogonUser` function in order to authenticate the user, the call fails, claiming that the password is invalid.
>
> If we encode the *ü* as the single byte `FC`, then the call to `LogonUser` succeeds.
>
> Therefore, if the string is in UTF-8 form, it needs to be converted, and to do this we use the `MultiByteToWideChar` function. Once converted, the logon is successful.
>
> The problem is that we are not sure if the password being given to the application will encode the *ü* as `C3 BC` or as `FC`. If it arrives as `FC`, and we try to convert it with the `MultiByteToWideChar` function, the *ü* is converted to `U+FFFD`.
>
> If I take the `FC`-encoded string and convert it with the `MultiByteToWideChar` function, passing `CP_ACP` as the first parameter, then it converts successfully (no `U+FFFD`), and the call to `LogonUser` is successful.
>
> For the application, the customer does not want to distinguish the two cases or implement any retry logic or anything like that. Can you help us understand the issue, what we are doing wrong, and how we can fix it?

As the problem is stated, you are screwed.

You have a bunch of bytes, and you don't know what encoding they are in. The byte sequence `C3 BC` might be a UTF-8 encoding of *ü*, or it could be a `CP_ACP` encoding of *Ã¼*. You are stuck with guessing. But for something as important as passwords, you shouldn't guess. You need to know for sure, because an incorrect guess will generate audit entries, and may cause the user to become locked out of the account due to too many incorrect passwords.

This means that you need to make sure that whoever is passing you the string also tells you what encoding it is using.

The customer liaison replied,

> Thanks. I went back and talked to the customer, and it turns out that the password is always in UTF-8 form, so the problem is solved. We will always pass `CP_UTF8` when converting the string.

Raymond Chen

Follow