

Simple PlayFab Auth

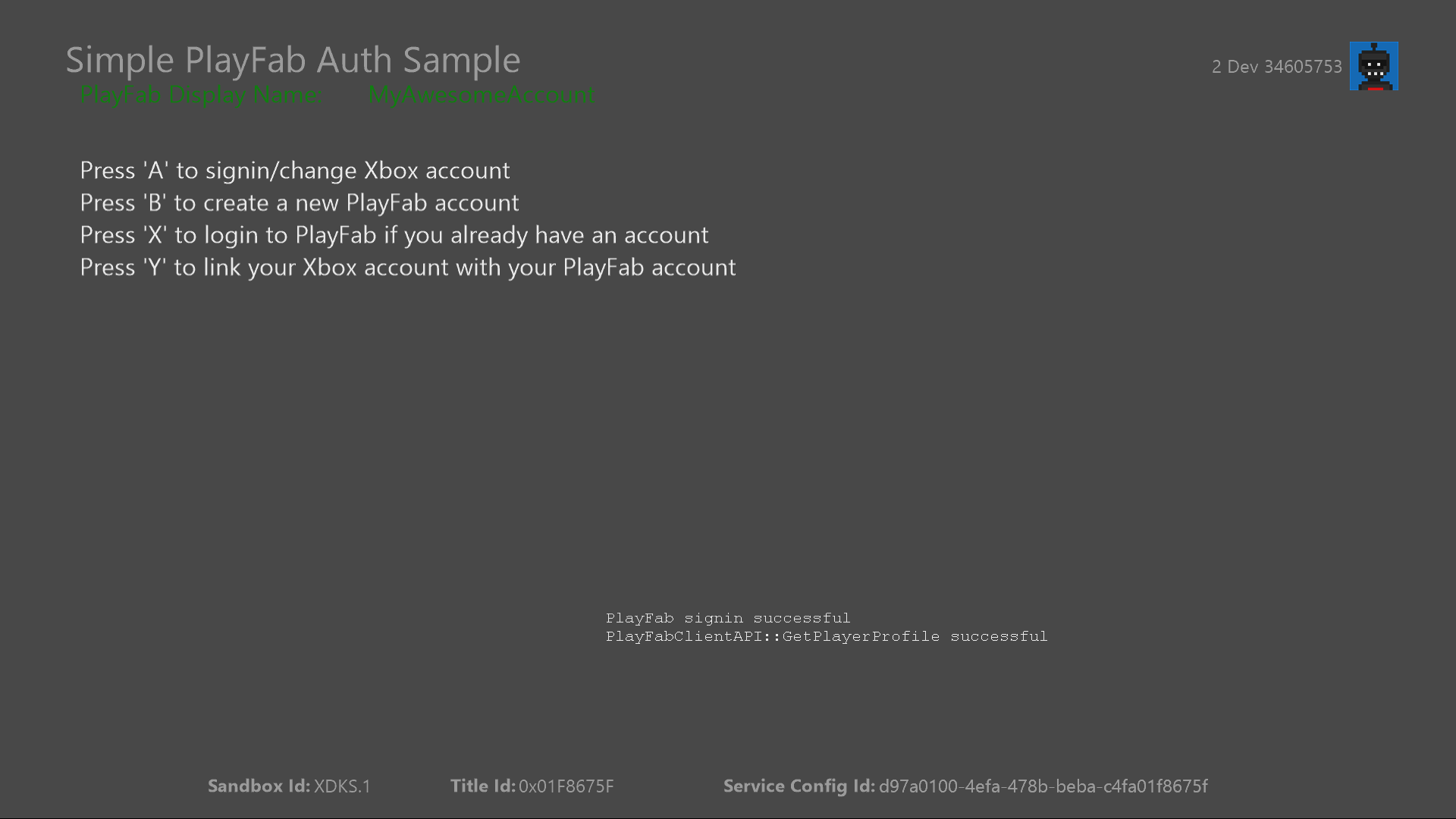
*\* This sample is compatible with the April 2018 XDK*

# Description

This sample demonstrates signing an Xbox user in with PlayFab to use PlayFab services.

# Using the sample

This sample is configured to work in the XDKS.1 sandbox.



|  |  |
| --- | --- |
| Action | Gamepad |
| Open account picker to choose an Xbox profile | A button |
| Register for a new PlayFab account | B button |
| Seamless PlayFab login | X Button |
| Link Xbox profile with existing PlayFab account | Y Button |
| Exit | View button |

# Implementation notes

In this version of the XDK, the sample also requires your Xbox One console to use the XDKS.1 **SandboxID**. To switch to this **SandboxID**, in the Xbox One XDK command prompt, use:

xbconfig sandboxid=XDKS.1

The PlayFab title ID must be set in the sample initialization code. This sample uses the PlayFab title ID “DC0” which is assigned to the **PlayFabSettings::titleId** static field.

# Logging in with PlayFab

Windows::Xbox::System::User::GetTokenAndSignatureAsync 🡪 PlayFab::PlayFabClientAPI::LoginWithXbox

The simplest method of signing a player in with PlayFab uses the **LoginWithXbox** API in the Windows Xbox C++ PlayFab SDK. As long as an Xbox profile is logged in, this method can be used to programmatically login to a PlayFab account without requiring user input. Like all PlayFab APIs, LoginWithXbox takes a “request” object and a callback that will be executed upon successful completion of the HTTP request. It can optionally take a second callback to handle failures, and a void pointer to arbitrary data that will be passed to the callbacks. This API takes a **LoginWithXboxRequest** object which manages login options.

|  |  |
| --- | --- |
| Field | Description |
| XboxToken | A token generated from the **GetTokenAndSignatureAsync** method using the endpoint <https://playfabapi.com>. |
| CreateAccount | If set to “true”, then **LoginWithXbox** will create a new account for the Xbox profile if one does not already exist.  If set to “false”, then the login will only succeed if the Xbox profile is already linked to a PlayFab account. Otherwise, the login attempt will fail and the error callback will be executed.  This sample does not force the creation of new accounts to demonstrate how to determine if a profile is already associated with a PlayFab account. |

Once a PlayFab account has successfully been logged in a PlayFab token will be stored in the **PlayFabSettings::clientSessionTicket** static object. It will be used for subsequent PlayFab API calls. Tokens will expire and must be refreshed every four hours. The stored token will be overwritten if LoginWithXbox is called again with another user. Tokens can also be managed in client code; by replacing PlayFabSettings::clientSessionTicket on the fly a game can have multiple PlayFab users playing at once.

# Linking an Xbox profile with an existing PlayFab account

PlayFab::PlayFabClientAPI::LoginWithPlayFab 🡪 Windows::Xbox::System::User::GetTokenAndSignatureAsync 🡪 PlayFab::PlayFabClientAPI::LinkXboxAccount

PlayFab is a multi-platform solution so it is possible players already have a PlayFab account, but have not yet associated it with their Xbox profile. Instead of creating a new profile, the **LinkXboxAccount** enables this scenario. The **LinkXboxAccountRequest** contains the following properties.

|  |  |
| --- | --- |
| Field | Description |
| XboxToken | A token generated from the **GetTokenAndSignatureAsync** method using the endpoint <https://playfabapi.com>. |
| ForceLink | If the PlayFab account is already associated with another Xbox profile, setting this to **true** will unlink the existing profile and link the new profile. If this is set to **false** and another profile is already associated with the account, then the call to LinkXboxAccount will return an error. |

Notably, the request object passed to this API does not contain any fields indicating which PlayFab account to link with. This is because every PlayFab API call requires a PlayFab profile returned from one of the login methods. The **LoginWithPlayFab** API allows a player to sign in using a username and password, but does not require an Xbox token. Once a user logs in, then the PlayFab profile can be linked using LinkXboxAccount.

*It is important to note that all usage of PlayFab services must be associated with an Xbox profile. A game should never allow a player to log in to a PlayFab account without linking it to an Xbox profile.*

# Creating a new PlayFab account

PlayFab::PlayFabClientAPI::RegisterPlayFabUser 🡪 Windows::Xbox::System::User::GetTokenAndSignatureAsync 🡪 PlayFab::PlayFabClientAPI::LinkXboxAccount

If a player’s Xbox profile is not linked and they do not have an existing PlayFab account, then the PlayFab API provides a few ways to create a new account. The simplest way to create a new account is with **LoginWithXbox** with **CreateAccount** set to true. This allows you to programmatically generate a new account for the player with minimal input required, but players may object to an account being created on their behalf without their explicit input.

Instead, this sample demonstrates the use of **RegisterPlayFabUser** to create a new account with player input, then link that account using the **LinkXboxAccount** API detailed in the previous section. The **RegisterPlayFabUserRequest** object has the following properties.

|  |  |
| --- | --- |
| Field | Description |
| DisplayName | The name associated with the PlayFab account that is displayed in game. This must be between 3 and 25 characters long. |
| Password | The password the player would like to use to login with their PlayFab account. This must be between 6 and 100 characters long. |
| UserName | This is the user name that is used to log in with a PlayFab account. It is not meant to be displayed within a game. It must be between 3 and 20 characters long. |
| RequireBothUserNameAndEmail | If this is set to **true**, then the user is required to provide both a user name and an email address to create a new account. If it is set to **false** an account can be created with only one of the two. |

This is not an exhaustive list of the properties associated with a RegisterPlayFabUserRequest; it is only the properties that are used in this sample. Check the [PlayFab online documentation](https://api.playfab.com/documentation/client/method/LinkXboxAccount) for a complete list of all available fields.

This sample demonstrates creating a new PlayFab account using only the **UserName** property, and provides a default value, the player’s gamertag, for both the UserName and the **DisplayName**. This is to minimize the input required from the player and get signed up with as little friction as possible.

Once a PlayFab account is created a token will be cached in the PlayFabSettings automatically and you can start making calls to PlayFab services. The first call should be LinkWithXbox to associate the player’s Xbox profile with the new PlayFab account immediately.

*It is important to note that all usage of PlayFab services must be associated with an Xbox profile. A game should never allow a player to log in to a PlayFab account without linking it to an Xbox profile.*

# Update history

**Initial Release:** May 2018

# Privacy Statement

When compiling and running a sample, the file name of the sample executable will be sent to Microsoft to help track sample usage. To opt-out of this data collection, you can remove the block of code in Main.cpp labeled “Sample Usage Telemetry”.

For more information about Microsoft’s privacy policies in general, see the [Microsoft Privacy Statement](https://privacy.microsoft.com/en-us/privacystatement/).