Online Account System

..for Unity3D

by

Konstantinos Vasileiadis

• Prerequisites:

Please note that before beginning, you should already have access to:

~ A server with Php support
~ A MySQL Database

Additionally, if you want to use our email activation and / or password recovery features:
    ~ The server must also have the ability to send emails

Here's a couple of lists with companies that provide both for free:
[http://www.ironspider.ca/website/freewebhost2.htm]
[http://www.hostsearch.com/free-web-hosting]

• We Provide:

.. both the *necessities*:

- One-Click database creation and setup
  - Add your own custom fields!

- Single-Line C# commands for
  - Registration
  - Login
  - Email Activation
  - Password Recovery

.. and the *luxuries*:

- Associate instances of your own custom classes with the player's account..!

- Single-Line C# commands to Download, Edit and Re-Upload the user's entire Account entry as a list of easily editable / readable MySQL fields

- C# Examples to get you started
  - Ready to Use GUI Scripts for Login, Registration, Recovery
  - A small 90-line script that demonstrates the workflow process.

All our are codes are well documented and come with:

- Standard level of Security
  - One-Way Password Hashing (using the SHA1 algorithm)
  - All PHP scripts are protected against MySQL injection techniques

- Error handling on 2 different levels:
  - Debug.LogError entries of WWW/MySQL/PHP errors, formatted for visibility
  - User-Friendly error formatting for some of the most common errors

- Debug.Log at critical regions, always tagged "AccountSystem: " for clarity

Reliable customer support via Email (vasileiadi@ceid.upatras.gr)

• How to Get Started:

After Importing the Asset, get started in 3 simple steps:

1) Upload the scripts located in the "/Account System/Server-Side Scripts/" folder to your web server.

2) From the Unity Menu Bar click on Window → Online Account System

    i.   .. fill in your info (Web Server url, Database hostname, ..)

    ii. .. customize your Account System

    iii. .. add extra fields to your database

3) Hit "Initialize Database"

That's it..!

You have created and set-up your own Account System!

To watch it in action, load our Demo Scene (**/Account System/Demo/AS_DemoScene**).

You can also take a look at our 90-lines-long and very well documented example that will get you through the basics.
It is located at **/Account System/C# Scripts/Demo/AS_ShortDemo.cs**.

When you're feeling a bit more confident, you can use
Account System/Demo/AS_LoginGUI.cs and
Account System/Demo/AS_AccountManagement.cs as a basis for your final GUI.

• Things you can Customize:

Here's a few functions / classes you could customize in order to use the provided examples for your own project (for your convenience, they have been thouroughly documented).

CustomInfo Class:

- **Functionality** -> Used for storing custom classes to the database as a single field, associated with the account.
- **Where is it ?** -> "Account System\C# Scripts\AS_CustomInfo.cs"
- **How To Use it** -> Use accountInfo.[Get|Set]CustomInfo() to Retrieve|Store your class.
- **Customize it!** -> Add any [Serializable] subclasses you want to it!

Successful Login Function:

- **Functionality** -> Used to add logic when the user successfully logs in (by default we re-direct him to the account management GUI)
- **Where is it ?** -> "Account System\Demo\AS_LoginGUI.cs" @SuccessfulLogin ()
- **How To Use it** -> Just call it ;)
- **Customize it!** -> Add any custom logic you want after successful logins (load the first level, initialize some preferences, ..)

Offline Field Check Function:

- **Functionality** -> Used for checking registration fields offline.
- **Where is it ?** -> "Account System\C# Scripts\Back-End\AS_Login.cs" @CheckFields ()
- **How To Use it** -> Call it with an AccountInfo instance and it will run checks on its fields.
- **Customize it!** -> Add your specific checks (valid email, age > 18, etc..)

For any Queries / Suggestions,
reach me at mr.kon.vas@gmail.com