GSF Backend

An Extended Backend Framework based on GameSpark.

Version 0.81

GSF Backend is NOT officially associated with GameSpark. But is dedicated to make backend development easier.

Before Importing the package:

- Sign in or register: https://www.gamesparks.com/
- Download the SDK: https://www.assetstore.unity3d.com/en/#!/content/78351
- To use the development game on the portal:
  - In GameSparks/Resources/GameSparkSettings.asset, set Api key: w313478WG59f
  - Set Api Secret: m3YNThDypcAxz2bbaCBpVCpSLM0xluLt
  - Press Get My Custom SDK to download latest cloud code responses on your client.
  - You can send me an email at delmarle.damien@gmail.com to be added to the development game.

ROADMAP: https://trello.com/b/4O5TF0Bn/gsframework

Introduction

This framework have been designed to be used for as many types of game as possible.
But its the perfect match for mobile social games like clash of clan.

This framework can help you:
- To give an ONLINE feeling to your player
- Persist data between all devices
- Let player communicate with them, send email, chat.
- Create groups with common objectives like clan or guilds
- Match player between them so they can visit other player base, house.. or attack it.

Project Structure:

The project have been split in **Modules,** each module will be a folder which describe a feature like “Mail” each of the module will contain script and prefabs related to itself. They can de removed without breaking the project and having any dependencies.
They will have in common few scripts located in **Core, Events & UI.** So in short all the modules are **optional**. You can pickup the pickup you are interested and remove the other one.

Code Structure:

The logic is separated in 2 pieces, RequestResponse & UI they will communicate between them using events without noticing they exist, so its easy to replace UI code with your own.
The only exception is CharacterCreation because it require more interaction between UI and game logic.
Most of the scripts derive from **Module** class which help with built in events and debug related to Gamespark

So if I want to create a new module and follow the project structure?

- Create MyModule: Module
- Override subscribe and unsubscribe event related to sending and receiving messages in here
- CreateUiMyModule:Module
- Override subscribe and unsubscribe event related to calling function from MyModule and receive UI updates from it

How Can I play the demo?
Locate **GameSparkSettings.asset** and click the button **[Get My Custom SDK]** so your version of the client will be updating with most recent server responses. You will notice that

API key and API secret need to be setup: check the first page of docs to access the development server.

Send me an email at delmarle.damien@gmail.com so I can add you to the group and let you see all the backend setup and logic on game spark portal.

Then Open **Assets/Demo/Scenes/Core.** You can start the scene immediately and test all features.

Each time an account is registered it will be cached on playerPrefs, Select **DataManager** object in the scene and set **ClearPlayerPrefsOnAwake = true** to remove cached accounts.

Modules

- Authentication
- User Details
- Mails
- Clan
- Character: creation, load, delete, action, xp, level up
- Leaderboard: rank top player by xp daily total, reward player when reset
- Achievement
- Multiple UI element to display notifications

Keep your backend optimized :

- Typically the areas that produce problems within a configuration are in relation to DataBase Usage.
- Ensure that your database queries, should the data set have the potential to grow substantially, are efficient in that they are searching over indexed fields.
- In the event that your data is time relevant and may become obsolete, it may be prudent to include TTL (time to live) indexing so that obsolete data entries expire.

- When using Schedulers, ensure they are staggered. The server can handle scheduler frequency well, but is not as efficient in handling sudden high loads.

- When utilizing modules maintain manageable module sizes. Breaking them up by purpose and behavior. massive modules or deeply nested modules greatly increase the execution time of scripts and can lead to timeouts. Especially if accompanied with inefficient queries. When the player base ramps up you will run into issues.

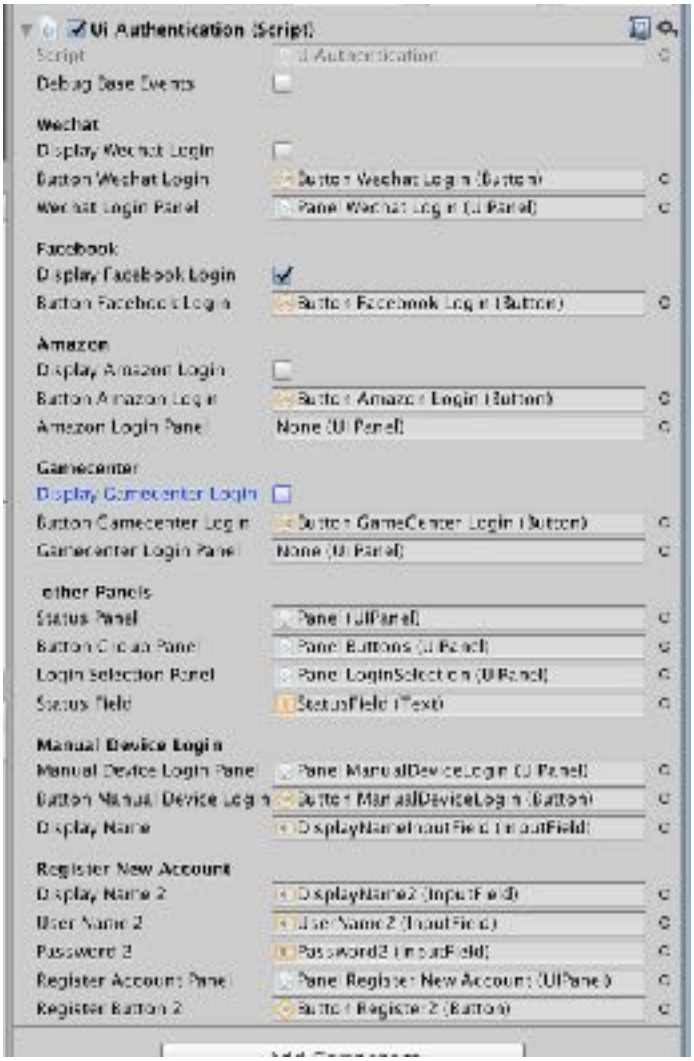
In Depth

Auth Module:

You can select what option to display on the UI. Using Facebook will require you to first import Facebook SDK and add “FACEBOOK” to scripting define symbols.
In the current version

Gsf_Authentication:
LoginIfSavedLocally allow you to check if the player have cached credentials in player prefs and reuse them.
ForceDeviceLogin will use devicelogin if no credential were cached instead of letting player select login mode

You can clear player prefs on awake on the DataManager object.



Profile Module:

Gsf_UserDetails, UiProfile, UIUpdateUserDetails

It use base request like AccountDetailsRequest but also add other function to update the player profile. Note that it is dissociated with character and gameplay as much as possible.

Character Module:

Gsf_Character, UiProfile, UIUpdateUserDetails

Load, Create, Delete Character, do actions, that server will interpret as experience, experience can be changed in the server settings. The xp will also trigger leaderboards

Clan Module:

Gsf_Character, UiProfile, UiUpdateUserDetails

Allow to get a list of all clans, join a clan, leave a clan, destroy the clan & get own clan data.

Email Module:

Gsf_CMail, UiMail,

Allow to get list of own email, delete & read email entries, send new email using display name. When a player receive an email he will receive a new message or a notification if his phone is closed.

Leaderboard Module:

Gsf_Leaderboard, UiLeaderboard,

Currently used to load and display Xp leaderboard, receive notification when the ranking on the leaderboard is changed by sending a notification, List players around yourself or look at top players

Other components:

GsfMaster: Master component that track state with Gamespark, only one active at a time
Dispatcher: Generic event framework
UiNotice: can queue and display message very easily
UiNotification: block player input until he confirm
UiDialogBox: block player input until he make a choice by clicking one of the options

UiCurrency: simply displaying currency by their ID, will be updated if the currency is changed on server
SceneLoader: will display an animated loading screen when going from one scene to an other
SoundManager: used to play pooled sounds
UiPanel: control state to show or hide an UI panel
LocalCache: save and load keys into playerprefs
DataManager: Hold data in a generic way so its easy to implement other plugin like AntiCheatToolKit later.