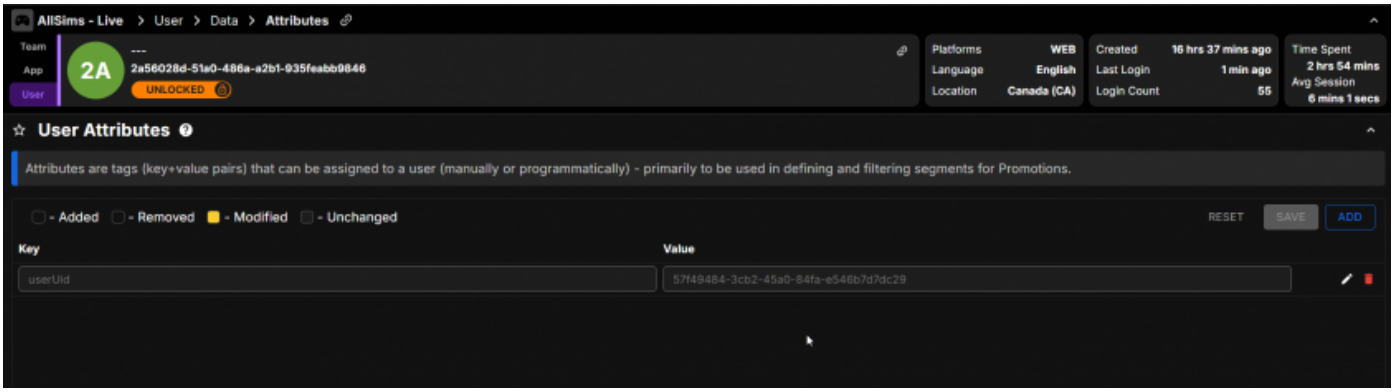
BrainCloud

Scripts

- On organise le nom des scripts de la manière suivantes : {folderName}_{subFolderName}_{ScriptName}
- Dans le cas des scripts appelable par le client on ajoute "Cmd" comme prefix du "ScriptName"
- Exemples:
  - acc1_CmdCreateTeam (pour les commandes de create team)
  - ent1_Utils (pour les scripts utilaire spécifique à un jeu)
  - acc1_Bills
  - CmdSaveSegment (Pour une commande générique)

Faire qu'un user se log comme un autre user

Si un étudiant se fait changer de date par Cheneiliere en cours de session et qu'il veut continuer sur son ancien user, nous pouvons editer l'attribute userUid du User braincloud.



C'est le même système que pour les user copiés.

Message/Events

- Code clients : Assez simple, On ajoute les binding pour get les events dans le

backendBrainCloud qui ajouterai une entrée dans le messageDispatcher.
- Code Serveur : Il faudrait faire un systeme plus centralisé sur le serveur afin de ne pas avoir plusieurs moyen de send des events
- Les events semblent mieux que les messages pour gerer notre systeme de live update

Documents Save/Load

- On peut modifier le système déjà mis en place, documents_utils, et cela devrait être fonctionnel pour tout le monde.
- Create : Dans gamesparks lorsqu'on crée un object, il n'est pas directement créer, il faut le save par la suite sinon il ne sera pas persisté. Cependant, dans braincloud lorsqu'on crée un object il est directement créer avec les informations spécifiuer.

Document Queries

- Possible d'ajouter des single et des compound indexes en json (MongoDB). Il n'y a pas beaucoup de documentation sur BrainCloud sur les indexes, mais il y en a dans celle de MongoDB.
- La boîte Options dans le popup de création d'indexes sert pour ajouter les propriétés aux indexes (comme sparse, TTL, case insensitive).
- Portal: Plus de capacités de recherche que GameSparks, mais sans l'interface user-friendly. Les Where Queries sont faites avec du json. Pour faire une query avec un champ spécifique du document, il faut utiliser le prefix "data." La documentation en lien avec les queries est assez simple et compréhensible.
- Code: Pour un élément spécifique, il est possible d'utiliser getRandomEntitiesMatching. Pour plusieurs éléments (comme pour obtenir toutes les simulations d'un prof), il serait plus favorable d'utiliser getEntityPage.
- Semble avoir la même limite de 100 objets retournés pour une requête particulière. La documentation suggère d'utiliser getEntityPageOffset pour obtenir le reste des objets (ex: si 150 objets sont demandés). ***À tester***

Scheduled Scripts

- En utilisant la Job Queue dans l'onglet Monitoring, puis Global Monitoring, il est très facile d'exécuter une job planifiée et il est possible d'exécuter le même script à nouveau en appelant ~~scheduleRunScriptMillisUTC~~ (ne fonctionne pas pour l'instant) scheduleRunScriptUTC (depricated depuis la version 4.6) du script service.
- En se construisant un script Template pour les Scheduled Jobs, il sera possible compartimenté les scripts appelés dans des jobs individuelles.
- Malgré un script de ScheduledJob qui permettrait d'ignorer les délais d'exécution, certains délais supplémentaires sont ajouter en recréant une nouvelle job.
- param pour schedule la job: /_job/_scheduled_job

```
{
    "timerDelay": 1,
    "scriptName": "/_job/generate_results_job",
    "scriptData": {},
    "lastExecutedTime" : -1}
```

REST API / nodeJS Client

- We can use https://www.npmjs.com/package/braincloud?activeTab=readme in combination with the S2S request to call differents scripts
- The nodejs is relatively simple, the hard thing is to find the right objectName or methodName but they can be found here

  http://getbraincloud.com/apidocs/apiref/?java#capi

  ex.:

  ```
  var scriptData = {
  "profil1": "ec11299f-ce5c-4485-9725-072205223783"
  };
  brainCloud.script.runScript("SendMessage", scriptData, OnSendMessageResponse);
  ```

- Pour la création d'utlisateur on n'a pas accès aux bonne méthodes à partir de nodeJS mais, on peut run un script qui, lui, peut créer un user. Sinon notre autre option serait de s'authentifier directement comme le user que nous voulont créer à partir de nodeJS

Deployment to Live

- Les Global Entities doivent être manuellement exporter/importer lors d'un déploiement. Étant donné qu'il est possible de migrer le data des custom entities avec le déploiement, il serait plus avantageux de les utiliser pour le static data.

Revision #26
Created Wed, Dec 2, 2020 4:33 PM by Frédéric Messier
Updated Wed, Mar 18, 2026 7:41 PM by Nicolas Babin