

Outils BrainCloud et import/export des scripts

But

Cette page documente les outils qui synchronisent les Cloud Scripts entre le dépôt et les applications BrainCloud. Ils sont séparés du debugger local : ici on parle de **déploiement/synchronisation des scripts**, pas d'exécution métier.

Fichiers principaux

- `backend/braincloud/api_builder_utils.js` ;
- `backend/braincloud/import_server_scripts.bat` ;
- `backend/braincloud/export_server_scripts.bat` ;
- `backend/braincloud/api_builder/` : CLI BrainCloud Builder + configuration.

api_builder_utils.js

Ce script enveloppe le CLI `bc-builder`. Il peut récupérer la liste des scripts, lire un script distant, comparer les versions et importer/exporter selon l'application sélectionnée.

Les environnements gérés explicitement incluent Dev, Dev2 et Demo. Le script permet aussi des options de remplacement forcé selon le workflow choisi.

Flux d'import

flowchart LR

LOCAL[server_scripts/] --> UTIL[api_builder_utils.js]

```
UTIL --> CLI[bc-builder]
CLI --> APP[BrainCloud Dev / Dev2 / Demo]
```

Flux d'export

```
flowchart LR
APP[BrainCloud] --> CLI[bc-builder]
CLI --> UTIL[api_builder_utils.js]
UTIL --> LOCAL[server_scripts/]
```

Avant un import

1. Vérifier l'environnement ciblé.
2. Vérifier le diff Git des Server Scripts.
3. Vérifier si des fichiers ont aussi changé directement dans BrainCloud.
4. Éviter un *force override* global sans comprendre les différences.
5. Après import, exécuter une commande représentative ou le debugger local avec la même logique.

Relation avec les autres outils

Besoin	Outil
Valider une fonction métier isolée	Tests automatisés
Exécuter un Cloud Script avec de vraies données	Debug local des Server Scripts
Envoyer/récupérer les scripts BrainCloud	api_builder / import-export scripts
Importer les static data Google Sheets	<code>node.js/exports/</code>

Commandes exactes

Importer les Cloud Scripts depuis BrainCloud vers le dépôt

```
cd node.js/backend/braincloud
import_server_scripts.bat
# équivalent :
node api_builder_utils.js import_scripts
```

Exporter/publier les Cloud Scripts vers BrainCloud

```
cd node.js/backend/braincloud
export_server_scripts.bat
# équivalent :
node api_builder_utils.js export_scripts
```

`api_builder_utils.js` reconnaît aussi les cibles Demo/Dev2 selon les arguments du workflow. Vérifier le prompt/log de destination avant toute publication.

Copier une simulation entre environnements

`exports/data_importer_braincloud.js` est un outil distinct de la static data. Il copie une **simulation runtime** et les documents associés.

```
cd node.js
node exports/data_importer_braincloud.js copySim <simUid> <source> <destination>
```

Sources autorisées : `live`, `dev`, `dev2`, `demo`. Destinations : `dev`, `dev2`, `demo`. **Le script refuse explicitement une destination live.**

Documents copiés

- `simulation;`
- `activity;`
- `results;`
- `actDataGameSpec;`
- `teams;`
- `teamsLogbooks;`
- `evaluationTeam;`
- `teamsModules;`
- `userDetails;`
- `evaluation.`

Le script crée/authentifie aussi les users nécessaires dans la destination et maintient les mappings de profils copiés.

Sync vs async

Le script a deux chemins : `copyActs()` pour le flow classique et `copyAllActsAsync()` / `copyAsyncTeam()` pour les simulations async, où certains UUIDs d'activity sont construits par équipe.

Copier une activité vers MongoDB pour debug

`api_builder_utils.js` expose aussi `copyAct [actUid] [backEndType]`. C'est utile pour préparer des données réelles dans le MongoDB utilisé par le debugger local.

Checklist opérationnelle

1. Confirmer source et destination.
2. Ne jamais recopier de credentials du script dans la documentation/log public.
3. Copier.
4. Vérifier simulation + teams + activity + results dans la destination.
5. Tester login avec un utilisateur copié.
6. Pour By Module, vérifier également les `teamsModules`.
7. Si un nouveau entity type est ajouté au produit, vérifier si cet outil doit le copier.

Revision #2

Created 2026-09-18 19:12:56 UTC by Mathis

Updated 2026-09-18 20:09:30 UTC by Mathis