---

Projet: Usine à Tests

Date: 2025-07-14

Version: 3.4

---

### 1. Résumé des Échanges du Jour

La session a été consacrée à la finalisation de la transition vers l'architecture V2 en s'attaquant à la dette technique des anciens formats de questions. L'objectif principal était la création d'un outil de migration pour convertir les questions existantes (tests "MBTI", "Couleurs", "Ancres", etc.) vers la nouvelle structure pilotée par `JSON`.

Nous avons commencé par définir la logique de conversion pour les questions de type `CHOIX\_BINAIRE`, en les mappant vers un `JSON` de mode `QRM\_CAT`. Après avoir écrit une première version du script, un processus de débogage itératif nous a permis de résoudre plusieurs problèmes :

1. \*\*Portée du Script :\*\* Correction d'une erreur où le script cherchait l'onglet à traiter dans le fichier `[MOTEUR]` au lieu de la `[BDD]`.

2. \*\*Formatage du JSON :\*\* Modification du script pour générer un `JSON` sur une seule ligne afin d'optimiser l'affichage dans la feuille de calcul.

3. \*\*Évolution de la Logique :\*\* Le script a été enrichi pour gérer un deuxième type de question, `ECHELLE`, en introduisant une nouvelle structure `JSON` (`ECHELLE\_NOTE`).

4. \*\*Débogage Final :\*\* Identification et résolution d'une "panne silencieuse" due à des cellules cibles non vides et à une erreur dans une version de débogage du script.

La session s'est conclue par la validation de l'outil de migration final, désormais capable de traiter plusieurs formats de questions. L'utilisateur a confirmé avoir migré l'ensemble de ses onglets de questions, rendant la base de données entièrement compatible avec le moteur universel V2.

### 2. État Actuel du Projet

\* \*\*Tâches réalisées :\*\*

\* Finalisation du script de l'outil de migration (`MigrationV1.gs`), capable de convertir les questions de type `CHOIX\_BINAIRE` et `ECHELLE`.

\* Mise à jour de l'interface utilisateur du `[MOTEUR] Usine à Tests V2` pour inclure un menu d'accès à l'outil de migration.

\* Standardisation de la nomenclature du projet sur \*\*`[MOTEUR] Usine à Tests V2`\*\*.

\* Migration complète par l'utilisateur de tous les onglets de questions dans la `[BDD]` vers le format V2.

\* \*\*Tâches en cours :\*\*

\* Génération de ce document de synthèse.

\* \*\*Prochaines étapes planifiées :\*\*

\* \*\*Test de bout en bout :\*\* Lancement de la génération d'un test complet (ex: "Couleurs") pour valider toute la chaîne V2 (configuration, génération, soumission, calcul des résultats).

\* \*\*Développement d'un validateur `JSON` :\*\* Pour fiabiliser la saisie manuelle des questions.

\* \*\*Enrichissement des Blocs Méta :\*\* Pour permettre la gestion de blocs de plusieurs questions.

### 3. Contexte Historique du Projet

L'objectif du projet est de passer d'une série de systèmes de tests spécialisés et rigides à une "Usine Universelle" (V2) capable de générer n'importe quel type de test à partir d'une simple configuration. La logique de calcul est externalisée dans les données (`JSON`) plutôt que d'être codée en dur. La session d'aujourd'hui a été l'étape finale et critique pour rendre cette vision V2 opérationnelle, en fournissant l'outil indispensable pour rendre les données historiques compatibles avec le nouveau moteur universel.

### 4. Orientations Stratégiques

Les actions du jour renforcent directement les principes directeurs de la V2 :

\* \*\*JSON comme Source Unique de Vérité :\*\* La création de l'outil de migration est l'incarnation de ce principe, transformant l'ancienne logique implicite des colonnes `Options`/`Logique` en une structure `JSON` explicite et universelle.

\* \*\*Zéro Convention Implicite :\*\* La définition de modes de traitement clairs (`QRM\_CAT`, `ECHELLE\_NOTE`) dans le `JSON` élimine toute ambiguïté sur la manière dont une question doit être interprétée par les scripts, que ce soit pour la génération du formulaire ou le calcul des scores.

### 5. Architecture Technique & Ressources Clés

#### 5.1. État des Scripts et Fichiers

Voici les versions finales et complètes des fichiers ajoutés ou modifiés aujourd'hui dans le projet `[MOTEUR] Usine à Tests V2`.

\*\*Fichier `MigrationV1.gs` (NOUVEAU)\*\*

```javascript

// =================================================================================

// FONCTION DE MIGRATION V1 -> V2 (JSON)

// RÔLE : Convertit les questions d'un ancien format (Options/Logique)

// vers le nouveau format V2 (Paramètres (JSON)).

// VERSION : 1.4 - Version finale et corrigée

// =================================================================================

/\*\*

\* Fonction principale appelée depuis le menu de l'interface utilisateur.

\*/

function lancerMigrationV1versV2() {

try {

const ID\_BDD = '1m2MGBd0nyiAl3qw032B6Nfj7zQL27bRSBexiOPaRZd8';

const ui = SpreadsheetApp.getUi();

const response = ui.prompt(

'Outil de Migration V1 -> V2',

'Veuillez entrer le nom exact de l\'onglet dans la BDD à migrer :',

ui.ButtonSet.OK\_CANCEL);

if (response.getSelectedButton() == ui.Button.OK && response.getResponseText() != '') {

const sheetName = response.getResponseText().trim();

const bdd = SpreadsheetApp.openById(ID\_BDD);

if (!bdd) { throw new Error(`Impossible d'ouvrir la BDD avec l'ID fourni.`); }

const sheet = bdd.getSheetByName(sheetName);

if (!sheet) { throw new Error(`L'onglet "${sheetName}" est introuvable dans la BDD.`); }

const resultat = convertirQuestionsEnJSON(sheet);

ui.alert(

'Migration Terminée',

`Rapport pour l'onglet "${sheetName}":\n\n` +

`- Lignes traitées : ${resultat.lignesTraitees}\n` +

`- Questions converties : ${resultat.questionsConverties}\n` +

`- Lignes ignorées : ${resultat.lignesIgnorees}\n` +

`- Erreurs rencontrées : ${resultat.erreurs.length}` +

(resultat.erreurs.length > 0 ? `\n\nConsultez les logs ("Affichage > Journaux") pour le détail des erreurs.` : ''),

ui.ButtonSet.OK);

}

} catch (e) {

SpreadsheetApp.getUi().alert(`Une erreur est survenue : ${e.message}`);

console.error(`Erreur lors du lancement de la migration : ${e.stack}`);

}

}

/\*\*

\* Cœur de la logique de conversion. Lit une feuille et met à jour les lignes.

\* @param {GoogleAppsScript.Spreadsheet.Sheet} sheet La feuille de calcul à traiter.

\* @returns {object} Un objet contenant les statistiques de la migration.

\*/

function convertirQuestionsEnJSON(sheet) {

const dataRange = sheet.getDataRange();

const values = dataRange.getValues();

const headers = values.shift();

const colIndex = {

type: headers.indexOf('TypeQuestion'),

options: headers.indexOf('Options'),

logique: headers.indexOf('Logique'),

description: headers.indexOf('Description'),

json: headers.indexOf('Paramètres (JSON)')

};

if (colIndex.type === -1 || colIndex.options === -1 || colIndex.logique === -1 || colIndex.json === -1) {

throw new Error("Colonnes requises ('TypeQuestion', 'Options', 'Logique', 'Paramètres (JSON)') manquantes.");

}

let questionsConverties = 0;

let lignesIgnorees = 0;

const erreurs = [];

values.forEach((row, index) => {

const jsonCell = row[colIndex.json];

if (jsonCell) {

lignesIgnorees++;

return;

}

const typeQuestion = row[colIndex.type];

const optionsStr = row[colIndex.options];

const logiqueStr = row[colIndex.logique];

const descriptionStr = colIndex.description !== -1 ? row[colIndex.description] : "";

let jsonPayload = null;

try {

switch (typeQuestion) {

case 'CHOIX\_BINAIRE':

if (optionsStr && logiqueStr) {

const optionsArray = optionsStr.toString().split(';').map(s => s.trim());

const logiqueArray = logiqueStr.toString().split(';').map(s => s.trim());

if (optionsArray.length !== logiqueArray.length) {

throw new Error(`CHOIX\_BINAIRE: Le nombre d'options (${optionsArray.length}) et de logiques (${logiqueArray.length}) ne correspond pas.`);

}

jsonPayload = {

mode: 'QRM\_CAT',

options: optionsArray.map((libelle, i) => ({ libelle: libelle, profil: logiqueArray[i], valeur: 1 }))

};

}

break;

case 'ECHELLE':

if (optionsStr && logiqueStr) { // La description n'est pas bloquante

const echelle = optionsStr.toString().split(';').map(s => parseInt(s.trim(), 10));

const labels = descriptionStr ? descriptionStr.toString().split(';').map(s => s.trim()) : ["", ""];

jsonPayload = {

mode: 'ECHELLE\_NOTE',

profil: logiqueStr.toString().trim(),

echelle\_min: Math.min(...echelle),

echelle\_max: Math.max(...echelle),

label\_min: labels[0] || "",

label\_max: labels[1] || ""

};

}

break;

default:

lignesIgnorees++;

break;

}

if (jsonPayload) {

sheet.getRange(index + 2, colIndex.json + 1).setValue(JSON.stringify(jsonPayload));

questionsConverties++;

} else {

lignesIgnorees++;

}

} catch (e) {

const errorMessage = `Erreur à la ligne ${index + 2}: ${e.message}`;

console.error(errorMessage);

erreurs.push(errorMessage);

}

});

return {

lignesTraitees: values.length,

questionsConverties: questionsConverties,

lignesIgnorees: lignesIgnorees,

erreurs: erreurs

};

}

```

\*\*Fichier `Interface V2.js` (MODIFIÉ)\*\*

```javascript

// =================================================================================

// FICHIER : Interface V2.js

// RÔLE : Création du menu utilisateur et fonctions appelées par ce menu.

// VERSION : 4.2 - Ajout de l'outil de migration V1 -> V2 dans le menu

// =================================================================================

/\*\*

 \* Crée le menu personnalisé dans l'interface utilisateur de Google Sheets à l'ouverture.

 \*/

function onOpen() {

  SpreadsheetApp.getUi()

    .createMenu('🏭 Usine à Tests')

    .addItem("🚀 Déployer un test de A à Z...", "orchestrateurDeploiementComplet\_UI")

    .addSeparator()

    .addItem("Générer un test (choisir la ligne)", "lancerCreationDepuisPilote\_UI")

    .addItem("Traiter TOUTES les nouvelles demandes", "orchestrateurCreationAutomatique\_UI")

// --- AJOUT ---

    .addSeparator()

    .addItem("🔧 Migrer les Questions (V1 -> V2)", "lancerMigrationV1versV2")

    .addToUi();

}

// ... (Le reste du fichier reste inchangé) ...

function orchestrateurDeploiementComplet\_UI() {

// ...

}

function lancerCreationDepuisPilote\_UI() {

// ...

}

function orchestrateurCreationAutomatique\_UI() {

// ...

}

```

#### 5.2. Architecture Détaillée et Traçabilité

\* \*\*Projet : `[MOTEUR] Usine à Tests V2`\*\*

\* \*\*Rôle :\*\* Usine centrale contenant les scripts pour générer les tests et les outils de maintenance.

\* \*\*Fichiers notables :\*\*

\* `Moteur V2.js` : (Inchangé) Cœur de la logique de génération des tests.

\* `Interface V2.js` : \*\*MODIFIÉ\*\*. La fonction `onOpen()` a été mise à jour pour ajouter un menu "🔧 Migrer les Questions (V1 -> V2)" qui appelle `lancerMigrationV1versV2`.

\* `MigrationV1.gs` : \*\*AJOUTÉ\*\*. Nouveau fichier contenant toute la logique de l'outil de migration. Sa fonction principale `lancerMigrationV1versV2` est appelée par le menu. La fonction `convertirQuestionsEnJSON` contient la logique de conversion pour les types `CHOIX\_BINAIRE` et `ECHELLE`.

\* \*\*Projet : `[BDD] Tests & Profils`\*\*

\* \*\*Rôle :\*\* Base de données passive contenant toutes les données (questions, profils, traductions).

\* \*\*MODIFIÉ :\*\* Toutes les feuilles de questions (`Questions\_\*`) ont été migrées par l'utilisateur pour que leur colonne `Paramètres (JSON)` soit remplie, les rendant compatibles V2.

#### 5.3. Nomenclature des Variables

La modification principale de la session réside dans l'introduction de nouvelles structures `JSON` pour le champ `Paramètres (JSON)`.

\* \*\*Objet : `JSON` pour question `ECHELLE`\*\*

\* \*\*Nom Précis :\*\* `jsonPayload` (dans le script)

\* \*\*Type :\*\* `Object` sérialisé en `String`

\* \*\*Origine :\*\* Généré par la fonction `convertirQuestionsEnJSON` du script `MigrationV1.gs`.

\* \*\*Contenu Attendu & Structure :\*\*

\* `mode` (String) : "ECHELLE\_NOTE"

\* `profil` (String) : Le code du profil à impacter (ex: "EXPERTISE")

\* `echelle\_min` (Number) : La valeur minimale de l'échelle (ex: 1)

\* `echelle\_max` (Number) : La valeur maximale de l'échelle (ex: 6)

\* `label\_min` (String) : Le libellé pour la valeur minimale (ex: "Pas du tout d'accord")

\* `label\_max` (String) : Le libellé pour la valeur maximale (ex: "Tout à fait d'accord")

\* \*\*Exemple Concret :\*\* `{"mode":"ECHELLE\_NOTE","profil":"EXPERTISE","echelle\_min":1,"echelle\_max":6,"label\_min":"Pas du tout d'accord","label\_max":"Tout à fait d'accord"}`

#### 5.4. Structure des Données (Feuilles et Colonnes)

\* \*\*Fichier : `[BDD] Tests & Profils`\*\*

\* \*\*Onglets : `Questions\_[TypeTest]\_[Langue]`\*\*

\* La migration a rendu ces onglets pleinement compatibles avec le moteur V2. Les colonnes suivantes sont maintenant utilisées conjointement :

\* `TypeQuestion` : Clé d'entrée pour la logique de migration (`CHOIX\_BINAIRE`, `ECHELLE`).

\* `Options`, `Logique`, `Description` : Données sources pour la migration.

\* `Paramètres (JSON)` : Colonne cible contenant le résultat de la migration, qui sera lue par le moteur V2.

### 6. Actions Recommandées / Prochaines Étapes

\* \*\*Action Immédiate :\*\* Procéder au test complet de la chaîne de génération pour un test (ex: "Couleurs") afin de valider l'intégration de bout en bout, de la configuration à la réception des résultats calculés.

\* \*\*Tâches de Développement Suivantes :\*\*

1. \*\*Validateur JSON :\*\* Concevoir et développer un outil pour valider la structure du `JSON` dans la `BDD`, afin de prévenir les erreurs avant la génération.

2. \*\*Blocs Méta Améliorés :\*\* Étudier la faisabilité d'un système de blocs méta pouvant regrouper plusieurs questions.

\* \*\*Pistes de Réflexion :\*\*

\* Comment l'outil de validation `JSON` devrait-il fonctionner ? (Un menu dans la `BDD` ? Une fonction dans le `MOTEUR` qui scanne un onglet ?)

\* Faut-il ajouter d'autres modes de calcul (`JSON`) pour gérer d'autres types de questions existants ?