**## Compte Rendu de Session**

Markdown

---

Projet: Usine à Tests

Date: 2025-07-09

Version: 14.0 (Améliorations Ergonomiques et Centralisation)

---

### 1. Résumé des Échanges du Jour

La session a été consacrée à la finalisation du débogage et, surtout, à l'implémentation de deux chantiers majeurs d'amélioration ergonomique visant à centraliser et simplifier le processus de déploiement des tests.

\* \*\*Finalisation du Débogage :\*\* Nous avons résolu le dernier bug bloquant lié à l'envoi des pièces jointes. L'erreur `getPublishedUrl is not a function` a été identifiée comme une confusion entre un objet `Fichier` et un objet `Formulaire` dans le script `Moteur.js`. La correction a été apportée en modifiant la fonction `lancerCreationSysteme` pour qu'elle retourne le bon objet, et en ajustant l'appel dans `lancerDeploiementComplet`.

\* \*\*Chantier 1 : Centralisation du Pilotage :\*\* Nous avons implémenté avec succès une nouvelle commande `"🚀 Déployer un test de A à Z..."` dans l'interface de `[MOTEUR] Usine à Tests`. Cette commande orchestre désormais l'intégralité du processus :

1. Demande à l'utilisateur la ligne à traiter depuis `[CONFIG]`.

2. Génère le Kit (Formulaire + Feuille de réponses).

3. Met à jour le statut dans `[CONFIG]` à "Actif - Déclencheur à activer".

4. Affiche une fenêtre de guidage pour l'activation finale.

\* \*\*Chantier 2 : Automatisation du Lien de Partage :\*\* Intégré au chantier 1, le processus de déploiement récupère maintenant automatiquement l'URL publique du formulaire généré et l'inscrit dans une nouvelle colonne `Lien\_Formulaire\_Public` dans la feuille `[CONFIG]`. Cette URL est également présentée à l'utilisateur dans la fenêtre de guidage finale.

\* \*\*Discussions Stratégiques :\*\* Nous avons discuté de deux autres améliorations possibles ("Touche Entrée" et mise à jour automatique du statut final) et avons décidé conjointement de ne \*\*pas\*\* les implémenter, jugeant que la complexité ajoutée l'emportait sur le gain ergonomique, afin de préserver la robustesse du système.

### 2. État Actuel du Projet

Le projet a atteint une nouvelle phase de maturité, avec un workflow de déploiement grandement amélioré et simplifié.

\* \*\*Tâches réalisées :\*\*

\* Correction du bug final sur les pièces jointes (`getPublishedUrl`).

\* Développement complet de la fonctionnalité "Déployer un test de A à Z".

\* Automatisation de la récupération et de l'enregistrement du lien public du formulaire.

\* Création d'une interface utilisateur de guidage pour l'activation finale du Kit.

\* \*\*Tâches en cours :\*\* Aucune.

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

1. \*\*Phase de Test par l'Utilisateur :\*\* Validation complète par vos soins des nouvelles fonctionnalités de déploiement.

2. \*\*(Prochaine Étape Majeure) Conception de l'Application Universelle :\*\* Lancer la phase de réflexion et de spécification, en utilisant le `Prompt Maître`, pour un système de tests générique piloté uniquement par la base de données.

### 3. Contexte Historique du Projet

Après une phase critique de stabilisation (restauration du `TraitementReponses.gs`) et de débogage des fonctionnalités clés (e-mails à 3 niveaux, pièces jointes), le projet s'est concentré sur l'amélioration de l'expérience utilisateur. La centralisation du workflow marque une étape majeure, réduisant le nombre de manipulations manuelles de trois à une seule, avec un guidage clair pour la dernière action requise.

### 4. Orientations Stratégiques

La session a confirmé l'orientation stratégique visant à trouver le juste équilibre entre \*\*automatisation et simplicité de maintenance\*\*. En renonçant volontairement à des complexités techniques non essentielles, nous renforçons la robustesse et la clarté de l'architecture, qui restent les piliers du projet. Le focus se porte désormais sur l'ajout de nouvelles capacités à forte valeur ajoutée.

### 5. Derniers Scripts et Ressources Clés

Les deux fichiers modifiés aujourd'hui sont les piliers du nouveau workflow.

\*\*`Moteur.js` (Version 3.2)\*\*

```javascript

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

// FICHIER : Moteur.js

// RÔLE : Fonctions principales de création et d'orchestration des tests.

// VERSION : 3.2 - Correctif getPublishedUrl

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

/\*\*

 \* Fonction de compatibilité pour une ancienne méthode (ne fait plus rien).

 \*/

function lancerCreationFormulaire() {

  SpreadsheetApp.getUi().alert("Cette fonction est l'ancienne méthode et n'est plus maintenue activement.");

}

/\*\*

 \* Orchestre la création d'un système de formulaire complet à partir d'une ligne de configuration.

 \*/

function lancerCreationSysteme(rowIndex) {

  try {

    console.log("Lancement de la création pour la ligne " + rowIndex + "...");

    const config = getConfigurationFromRow(rowIndex);

    Logger.log("CONFIG REÇUE DANS LE MOTEUR : " + JSON.stringify(config, null, 2));

    if (config['Statut'].toLowerCase() !== 'en construction') {

      console.log("Le statut pour la ligne " + rowIndex + " est '" + config['Statut'] + "'. Création ignorée.");

      return null;

    }

    const nomFichierComplet = "[" + config['Type\_Test'] + "] " + config['Titre\_Formulaire\_Utilisateur'];

    const systemIds = getSystemIds();

    if (!systemIds.ID\_TEMPLATE\_TRAITEMENT) {

      throw new Error("La clé 'ID\_TEMPLATE\_TRAITEMENT' est introuvable dans sys\_ID\_Fichiers.");

    }

    let dossierCible;

    if (config['ID\_Dossier\_Cible']) {

      dossierCible = DriveApp.getFolderById(config['ID\_Dossier\_Cible']);

    } else {

      if (!systemIds.ID\_DOSSIER\_CIBLE\_GEN) {

        throw new Error("La clé 'ID\_DOSSIER\_CIBLE\_GEN' est introuvable dans sys\_ID\_Fichiers.");

      }

      dossierCible = DriveApp.getFolderById(systemIds.ID\_DOSSIER\_CIBLE\_GEN);

    }

    console.log("Utilisation du dossier cible : " + dossierCible.getName());

    const templateFile = DriveApp.getFileById(systemIds.ID\_TEMPLATE\_TRAITEMENT);

    const sheetFile = templateFile.makeCopy(nomFichierComplet, dossierCible);

    const reponsesSheetId = sheetFile.getId();

    console.log("Kit de traitement copié : " + sheetFile.getName());

    const form = FormApp.create(nomFichierComplet);

    form.setDestination(FormApp.DestinationType.SPREADSHEET, reponsesSheetId);

    form.setProgressBar(true);

    form.setDescription(config['Titre\_Formulaire\_Utilisateur']);

    console.log("Formulaire créé : " + form.getTitle());

    const formFile = DriveApp.getFileById(form.getId());

    formFile.moveTo(dossierCible);

    try {

      console.log("Génération des questions pour le test : " + config['Type\_Test']);

      if (!systemIds.ID\_BDD) {

        throw new Error("La clé 'ID\_BDD' est introuvable dans sys\_ID\_Fichiers.");

      }

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

      const toutesLesFeuillesBDD = bdd.getSheets();

      const regexLangues = new RegExp('^Questions\_' + config['Type\_Test'] + '\_([A-Z]{2})$', 'i');

      const languesAInclure = [];

      toutesLesFeuillesBDD.forEach(feuille => {

        const match = feuille.getName().match(regexLangues);

        if (match && match[1]) {

          languesAInclure.push({ code: match[1].toUpperCase(), nomComplet: getLangueFullName(match[1]), feuille: feuille });

        }

      });

      if (languesAInclure.length === 0) {

        throw new Error("Aucune feuille de questions trouvée pour le type '" + config['Type\_Test'] + "'.");

      }

      form.addTextItem().setTitle("Nom Complet / Full Name").setRequired(true);

      form.addTextItem().setTitle("Adresse Email").setValidation(FormApp.createTextValidation().requireTextIsEmail().build()).setRequired(true);

      const itemLangue = form.addMultipleChoiceItem().setTitle("Langue / Language").setRequired(true);

      const choices = [];

      languesAInclure.forEach(langue => {

        const page = form.addPageBreakItem().setTitle("Questions (" + langue.nomComplet + ")");

        choices.push(itemLangue.createChoice(langue.nomComplet, page));

        const nbQuestionsDisponibles = langue.feuille.getLastRow() - 1;

        let nbQuestionsAUtiliser = (config['nbQuestions'] && config['nbQuestions'] > 0) ? Math.min(config['nbQuestions'], nbQuestionsDisponibles) : nbQuestionsDisponibles;

        if (nbQuestionsAUtiliser <= 0) return;

        const questionsData = langue.feuille.getRange(2, 1, nbQuestionsAUtiliser, 6).getValues();

        questionsData.forEach((q\_data, index) => {

          const [id, type, titre, options, logique, description] = q\_data;

          creerItemFormulaire(form, type, id + ': ' + titre, options, description);

          if (index === questionsData.length - 1) {

            page.setGoToPage(FormApp.PageNavigationType.SUBMIT);

          }

        });

      });

      itemLangue.setChoices(choices);

    } catch (e) {

      throw new Error("Impossible de générer les questions. Erreur : " + e.message);

    }

    console.log("--- Création terminée avec succès ! N'oubliez pas d'activer le traitement dans la feuille de calcul générée. ---");

    return { formObject: form, formFile: formFile, sheetFile: sheetFile, nomFichierComplet: nomFichierComplet };

  } catch(e) {

    console.error("ERREUR (ligne " + rowIndex + ") : " + e.toString() + "\n" + e.stack);

    SpreadsheetApp.getUi().alert("Une erreur est survenue lors de la création pour la ligne " + rowIndex + ": " + e.message);

    throw e;

  }

}

/\*\*

 \* Orchestre le traitement de toutes les lignes "En construction".

 \*/

function orchestrateurCreationAutomatique() {

  const ssConfig = SpreadsheetApp.openById(ID\_FEUILLE\_CONFIGURATION);

  const configSheet = ssConfig.getSheetByName("Paramètres Généraux");

  if (!configSheet) {

    throw new Error("L'onglet 'Paramètres Généraux' est introuvable.");

  }

  const data = configSheet.getDataRange().getValues();

  const headers = data.shift();

  const colIndex = {};

  headers.forEach((header, i) => { if (header) colIndex[header] = i; });

  const STATUT\_COL = colIndex['Statut'];

  let lignesTraitees = 0;

  data.forEach((row, index) => {

    const rowIndex = index + 2;

    if (row[STATUT\_COL] === 'En construction') {

      try {

        Logger.log("Début du traitement pour la ligne " + rowIndex + "...");

        const resultats = lancerCreationSysteme(rowIndex);

        if (resultats) {

            const idUnique = resultats.sheetFile.getId().slice(0, 8) + '-' + resultats.formFile.getId().slice(0, 8);

            configSheet.getRange(rowIndex, STATUT\_COL + 1).setValue('Actif');

// ... autres mises à jour de cellules

            Logger.log("SUCCÈS ligne " + rowIndex + ". Statut mis à 'Actif'.");

            lignesTraitees++;

        }

      } catch (e) {

        configSheet.getRange(rowIndex, STATUT\_COL + 1).setValue('ERREUR');

        Logger.log("ERREUR ligne " + rowIndex + ": " + e.toString());

      }

    }

  });

  SpreadsheetApp.flush();

  return lignesTraitees;

}

/\*\*

 \* Gère le déploiement complet (création + mise à jour du statut + lien public).

 \*/

function lancerDeploiementComplet(rowIndex) {

Logger.log(`Lancement du déploiement complet pour la ligne ${rowIndex}...`);

const resultatsCreation = lancerCreationSysteme(rowIndex);

if (!resultatsCreation) {

Logger.log(`La création pour la ligne ${rowIndex} a été ignorée (statut non valide).`);

return null;

}

Logger.log(`Kit créé pour la ligne ${rowIndex}. ID Sheet: ${resultatsCreation.sheetFile.getId()}`);

const formUrl = resultatsCreation.formObject.getPublishedUrl();

const configSheet = SpreadsheetApp.openById(ID\_FEUILLE\_CONFIGURATION).getSheetByName("Paramètres Généraux");

const headers = configSheet.getRange(1, 1, 1, configSheet.getLastColumn()).getValues()[0];

const colIndex = {};

headers.forEach((header, i) => { if (header) colIndex[header] = i; });

const STATUT\_COL = colIndex['Statut'];

const ID\_UNIQUE\_COL = colIndex['Id\_Unique'];

const NOM\_FICHIER\_COL = colIndex['Nom\_Fichier\_Complet'];

const ID\_FORM\_COL = colIndex['ID\_Formulaire\_Cible'];

const ID\_SHEET\_COL = colIndex['ID\_Sheet\_Cible'];

const LIEN\_FORM\_COL = colIndex['Lien\_Formulaire\_Public'];

const idUnique = resultatsCreation.sheetFile.getId().slice(0, 8) + '-' + resultatsCreation.formFile.getId().slice(0, 8);

configSheet.getRange(rowIndex, STATUT\_COL + 1).setValue('Actif - Déclencheur à activer');

configSheet.getRange(rowIndex, ID\_UNIQUE\_COL + 1).setValue(idUnique);

configSheet.getRange(rowIndex, NOM\_FICHIER\_COL + 1).setValue(resultatsCreation.nomFichierComplet);

if (ID\_FORM\_COL !== undefined) configSheet.getRange(rowIndex, ID\_FORM\_COL + 1).setValue(resultatsCreation.formFile.getId());

if (ID\_SHEET\_COL !== undefined) configSheet.getRange(rowIndex, ID\_SHEET\_COL + 1).setValue(resultatsCreation.sheetFile.getId());

if (LIEN\_FORM\_COL !== undefined) configSheet.getRange(rowIndex, LIEN\_FORM\_COL + 1).setValue(formUrl);

SpreadsheetApp.flush();

Logger.log(`Ligne ${rowIndex} mise à jour avec le statut 'Actif - Déclencheur à activer'.`);

return {

nomFichier: resultatsCreation.nomFichierComplet,

urlSheet: resultatsCreation.sheetFile.getUrl(),

urlForm: formUrl

};

}

**Interface.js (Version 4.1)**

JavaScript

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

// FICHIER : Interface.js

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

// VERSION : 4.1 - Ajout du lien public du Form dans la fenêtre de guidage

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

/\*\*

 \* 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")

    .addSeparator()

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

    .addToUi();

}

/\*\*

 \* Orchestre le déploiement complet d'un test depuis l'UI.

 \* Gère la génération du kit ET guide l'utilisateur pour l'activation et le partage.

 \*/

function orchestrateurDeploiementComplet\_UI() {

  const ui = SpreadsheetApp.getUi();

  const response = ui.prompt(

    '🚀 Déploiement de A à Z',

    'Entrez le numéro de la ligne à déployer entièrement :',

    ui.ButtonSet.OK\_CANCEL

  );

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

    return;

  }

  const rowIndex = parseInt(response.getResponseText(), 10);

  if (isNaN(rowIndex) || rowIndex <= 1) {

    ui.alert('Numéro de ligne invalide. Veuillez entrer un nombre supérieur à 1.');

    return;

  }

  ui.alert('Lancement du déploiement complet... Cette opération peut prendre un moment.');

  try {

    const resultats = lancerDeploiementComplet(rowIndex);

    if (resultats && resultats.urlSheet && resultats.urlForm) {

      const htmlOutput = HtmlService.createHtmlOutput(

        `<h4>✅ Déploiement Réussi !</h4>` +

`<p>Le kit "<b>${resultats.nomFichier}</b>" a été généré.</p><hr>` +

`<p><b>1. Voici le lien public du formulaire à partager :</b></p>` +

`<p style="margin-top:10px;"><a href="${resultats.urlForm}" target="\_blank" style="background-color:#34A853; color:white; padding:8px 12px; text-decoration:none; border-radius:4px;">Copier ou ouvrir le lien du Formulaire</a></p><br>` +

        `<p><b>2. ACTION FINALE REQUISE (pour que le test fonctionne) :</b></p>` +

`<p>Cliquez sur le lien ci-dessous, puis dans le menu :<br>` +

        `<b>&nbsp;&nbsp;&nbsp;⚙️ Actions du Kit -> Activer le traitement des réponses</b>.</p>` +

        `<p style="margin-top:10px;"><a href="${resultats.urlSheet}" target="\_blank" style="background-color:#4285F4; color:white; padding:8px 12px; text-decoration:none; border-radius:4px;">Ouvrir le Kit pour l'activer</a></p>`

      )

      .setWidth(500)

      .setHeight(320);

      ui.showModalDialog(htmlOutput, "Déploiement Terminé");

    } else {

      ui.alert(`ℹ️ Le déploiement pour la ligne ${rowIndex} a été ignoré (le statut n'était probablement pas 'En construction').`);

    }

  } catch (e) {

    Logger.log(`ERREUR Critique lors du déploiement complet (ligne ${rowIndex}) : ${e.toString()}`);

    ui.alert(`❌ ERREUR : Le déploiement a échoué pour la ligne ${rowIndex}. Consultez les logs pour les détails. Message : ${e.message}`);

  }

}

// ... autres fonctions ...

**6. Actions Recommandées / Prochaines Étapes**

* **Action Immédiate pour l'Utilisateur :** Tester de bout en bout le nouveau workflow "Déployer un test de A à Z..." pour s'assurer que le comportement est conforme aux attentes.
* **Prochaine session :** Lorsque vous serez prêt, nous pourrons commencer à utiliser le Prompt Maître pour définir les spécifications de votre idée d'"application universelle".

**Consigne pour Gemini (à reporter dans la prochaine synthèse)**

Lors de la rédaction du prochain résumé de travail, réinscrire systématiquement l'architecture complète des fichiers et des scripts pour maintenir une connaissance contextuelle parfaite du projet. **Architecture Fichiers et Scripts de Référence :**

* **[BDD] Tests & Profils** (Google Sheet)
* **[TEMPLATE] Kit de Traitement** (Projet Apps Script)
  + TraitementReponses.gs (V2.5)
  + Utilities.gs (V3.2)
  + Logique\_Couleurs.gs (V2.1)
  + Logique\_MBTI.gs
  + Logique\_Ancres.gs
* **[MOTEUR] Usine à Tests** (Projet Apps Script)
  + Moteur.js (V3.2)
  + Interface.js (V4.1)
  + Utils.gs
* **[CONFIG] Usine à Tests** (Google Sheet)
  + menu.gs
  + FormulaireUI.html