Absolument. J'ai bien mémorisé les points essentiels de notre session, notamment la résolution du problème des e-mails et la nouvelle architecture de débogage. Le document ci-dessous est une synthèse complète, conçue pour être notre nouvelle référence.

Markdown

---

Projet: "Usine à Tests"

Date: "2025-07-15"

Version: "5.0"

---

### 1. Résumé des Échanges du Jour

La session d'aujourd'hui a été entièrement consacrée à la \*\*résolution d'un bug critique et persistant concernant l'envoi des e-mails de restitution\*\*, et à l'amélioration de notre méthodologie de débogage pour l'avenir.

Le processus de résolution a suivi plusieurs étapes clés :

\* \*\*Confirmation du Problème :\*\* Malgré les correctifs précédents, les e-mails n'étaient pas envoyés comme attendu (le répondant ne recevait rien, ou le formateur recevait des copies incorrectes).

\* \*\*Itérations de Diagnostic :\*\* Nous avons exploré plusieurs pistes, ce qui a mené à la correction de plusieurs scripts (`Utilities.gs`, `Logique\_Universel.gs`, `TraitementReponses.gs`). Chaque correction a résolu un problème latent (typo, logique d'envoi, etc.) mais n'a pas résolu le problème de fond.

\* \*\*Pivot Architectural (Décision Clé) :\*\* Face à la difficulté de déboguer en modifiant directement les scripts de production, nous avons validé votre proposition de créer une \*\*architecture de débogage dédiée\*\*.

\* \*\*Implémentation de la Nouvelle Architecture :\*\* Nous avons créé une librairie externe (`LoggerLib`) et un interrupteur de débogage local (`MODE\_DEBUG\_ACTIF`) dans les scripts du `[TEMPLATE]` pour obtenir des logs détaillés sans altérer le code principal.

\* \*\*Diagnostic Final :\*\* Grâce à cette nouvelle architecture, nous avons pu analyser des logs fiables qui ont révélé que le problème n'était pas un bug de code, mais une \*\*colonne de configuration manquante\*\* (`Repondant\_Email\_Actif`) dans la feuille `[CONFIG] Usine à Tests`.

\* \*\*Résolution :\*\* L'ajout de cette colonne a définitivement résolu le problème, validant ainsi l'ensemble de la chaîne de traitement des e-mails.

### 2. État Actuel du Projet

\* \*\*Tâches réalisées :\*\*

\* Correction et stabilisation de l'ensemble de la logique d'envoi d'e-mails.

\* Mise à jour des scripts `TraitementReponses.gs` (v3.5) et `Utilities.gs` (v3.5) pour inclure une logique de débogage robuste et synchronisée.

\* Création d'une librairie partagée `LoggerLib` pour la centralisation des logs.

\* Correction de la configuration de la feuille `[CONFIG] Usine à Tests` par l'ajout de la colonne `Repondant\_Email\_Actif`.

\* Validation complète de la chaîne de traitement, y compris l'envoi d'e-mails au répondant et les copies au formateur avec un sujet différencié.

\* \*\*Tâches en cours :\*\*

\* Génération de ce document de synthèse.

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

\* Mettre à jour le "Manuel d'Architecture" pour y intégrer la nouvelle architecture de débogage et les dernières corrections.

\* Passer la variable `MODE\_DEBUG\_ACTIF` à `false` dans le `[TEMPLATE]` pour que les futurs kits générés soient en mode production.

### 3. Contexte Historique du Projet

La session d'aujourd'hui fait suite à la validation fonctionnelle de la chaîne V2. Alors que nous pensions le système stable, un bug subtil dans la logique d'envoi d'e-mails a persisté. Cette session a donc été une validation en profondeur qui nous a forcés non seulement à corriger le bug, mais aussi à \*\*améliorer notre propre outillage de développement\*\* en créant une architecture de débogage réutilisable, une étape significative vers la maturité et la maintenabilité du projet.

### 4. Orientations Stratégiques

La session a renforcé nos principes directeurs de manière inattendue :

\* \*\*Zéro Convention Implicite :\*\* Le bug était causé par une convention implicite (la supposition que la colonne `Repondant\_Email\_Actif` existait). Le fait de devoir l'ajouter explicitement pour que le système fonctionne est l'incarnation même de ce principe.

\* \*\*Le Code plus Intelligent que la Configuration :\*\* La fonction `\_findReponseByKeyHint` qui cherche maintenant une liste de mots-clés (`['email', 'e-mail']`) est une amélioration directe de ce principe.

\* \*\*Nouvelle Décision d'Architecture :\*\* Nous avons acté qu'une séparation claire entre le code de production et le code de débogage est nécessaire. L'utilisation d'une \*\*librairie de log externe contrôlée par un interrupteur local\*\* est désormais une nouvelle norme pour ce projet.

### 5. Architecture Technique & Ressources Clés

#### 5.1. État des Scripts et Fichiers

Voici les versions finales et complètes des scripts, incluant toutes les corrections et la nouvelle architecture de débogage.

\*\*Nouveau Projet : `LoggerLib` (Apps Script)\*\*

```javascript

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

// == LIBRAIRIE : LoggerLib

// == VERSION : 1.0

// == RÔLE : Outil de débogage centralisé pour l'Usine à Tests.

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

function logStart(donneesRepondant) {

Logger.log("--- DÉBUT PISTAGE ---");

if (donneesRepondant && donneesRepondant.emailRepondant) {

Logger.log("Répondant identifié : " + donneesRepondant.nomRepondant + " (" + donneesRepondant.emailRepondant + ")");

} else {

Logger.log("AVERTISSEMENT : Données du répondant non fournies ou e-mail manquant.");

}

}

function logResults(resultatsCalcules) {

if (resultatsCalcules && Object.keys(resultatsCalcules).length > 0) {

Logger.log("Profil final calculé : " + resultatsCalcules.profilFinal);

Logger.log("Données de score : " + JSON.stringify(resultatsCalcules.scoresData));

} else {

Logger.log("AVERTISSEMENT : Le calcul n'a retourné aucun résultat.");

}

}

function logEmailQueue(listeAdresses) {

if (listeAdresses && listeAdresses.size > 0) {

Logger.log("Liste finale des destinataires déterminée : [" + Array.from(listeAdresses).join(', ') + "]");

} else {

Logger.log("AVERTISSEMENT : Aucune adresse dans la file d'attente d'envoi.");

}

}

function logEmailAttempt(adresse, sujet) {

Logger.log(`TENTATIVE D'ENVOI -> Destinataire: ${adresse} | Sujet: ${sujet}`);

}

function logSuccess(message) {

Logger.log(`SUCCÈS : ${message}`);

}

function logError(message, erreur) {

Logger.log(`ERREUR : ${message} | Détails : ${erreur.message}`);

}

function logEnd() {

Logger.log("--- FIN PISTAGE ---");

}

**Projet [TEMPLATE] Kit de Traitement V2**

**Fichier TraitementReponses.gs (Version 3.5)**

JavaScript

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

// == FICHIER : TraitementReponses.gs (Le Chef d'Orchestre)

// == VERSION : 3.5 (Passe le switch de debug aux utilitaires)

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

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

// == INTERRUPTEUR DE DÉBOGAGE

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

const MODE\_DEBUG\_ACTIF = true; // Mettre sur 'false' pour la production

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

function onOpen() { /\* ... (inchangé) ... \*/ }

function creerDeclencheur() { /\* ... (inchangé) ... \*/ }

function retraiterReponse\_UI() { /\* ... (inchangé) ... \*/ }

function onFormSubmitTrigger(e) { processSubmissionData(e.namedValues, null); }

function retraiterReponse(rowIndex, langueOverride) { /\* ... (inchangé) ... \*/ }

function processSubmissionData(namedValues, langueOverride) {

try {

const config = getTestConfiguration();

const langueOrigine = getOriginalLanguage(namedValues);

const langueCible = langueOverride ? langueOverride.toUpperCase() : langueOrigine;

const reponse = {

emailRepondant: \_findReponseByKeyHint(namedValues, ['email', 'e-mail', 'mail']),

nomRepondant: \_findReponseByKeyHint(namedValues, ['nom', 'name', 'prénom']),

toutesLesReponses: namedValues

};

if (MODE\_DEBUG\_ACTIF) { LoggerLib.logStart(reponse); }

const resultats = calculerResultats(reponse.toutesLesReponses, langueCible, config);

if (MODE\_DEBUG\_ACTIF) { LoggerLib.logResults(resultats); }

if (resultats && Object.keys(resultats).length > 0) {

buildAndSendEmails(config, reponse, resultats, langueCible, MODE\_DEBUG\_ACTIF);

Logger.log("Traitement V2 réussi pour " + (reponse.emailRepondant || "un utilisateur inconnu"));

} else {

Logger.log("Le calcul des résultats V2 n'a retourné aucune donnée.");

}

} catch (err) {

if (typeof LoggerLib !== 'undefined' && MODE\_DEBUG\_ACTIF) { LoggerLib.logError("Erreur critique dans processSubmissionData", err); }

Logger.log("ERREUR CRITIQUE V2 : " + err.toString() + "\n" + err.stack);

}

}

function \_findReponseByKeyHint(namedValues, hints) {

for (const key in namedValues) {

const lowerCaseKey = key.toLowerCase();

const aUnMatch = hints.some(hint => lowerCaseKey.includes(hint.toLowerCase()));

if (aUnMatch) {

if (namedValues[key] && namedValues[key][0]) {

return namedValues[key][0];

}

}

}

return null;

}

**Fichier Utilities.gs (Version 3.5)**

JavaScript

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

// == FICHIER : Utilities.gs - VERSION 3.5 (Reçoit le switch de debug)

// == RÔLE : Boîte à outils du Kit de Traitement.

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

const ID\_FEUILLE\_PILOTE = "1kLBqIHZWbHrb4SsoSQcyVsLOmqKHkhSA4FttM5hZtDQ";

function getTestConfiguration() { /\* ... (inchangé) ... \*/ }

function getSystemIds() { /\* ... (inchangé) ... \*/ }

function loadTraductions(langueCode) { /\* ... (inchangé) ... \*/ }

function buildAndSendEmails(config, reponse, resultats, langueCode, isDebugMode) {

const niveau = config.Repondant\_Contenu || "Niveau 1 : Simple";

const T = loadTraductions(langueCode);

let corpsEmail = "";

let sujet = T[`SUJET\_EMAIL\_RESULTATS\_${config.Type\_Test.toUpperCase()}`] || `${T.SUJET\_EMAIL\_RESULTATS || "Résultats de votre test"} : ${config.Type\_Test}`;

corpsEmail += (T.SALUTATION || "Bonjour {nom\_repondant},").replace('{nom\_repondant}', reponse.nomRepondant) + "\n\n";

// ... (construction du corps de l'e-mail inchangée) ...

corpsEmail += (T.FORMULE\_POLITESSE || "Bien à vous,") + "\n";

const piecesJointes = findAttachments(config.Type\_Test, resultats.profilFinal, niveau, langueCode);

const adressesUniques = new Set();

if (config.Repondant\_Email\_Actif === 'Oui' && reponse.emailRepondant) { adressesUniques.add(reponse.emailRepondant); }

if (config.Patron\_Email\_Mode === 'Oui' && config.Patron\_Email) { adressesUniques.add(config.Patron\_Email); }

if (config.Formateur\_Email\_Actif === 'Oui' && config.Formateur\_Email) { adressesUniques.add(config.Formateur\_Email); }

if (config.Developpeur\_Email) { adressesUniques.add(config.Developpeur\_Email); }

if (isDebugMode) { LoggerLib.logEmailQueue(adressesUniques); }

adressesUniques.forEach(adresse => {

try {

let sujetFinal = sujet;

if (adresse.toLowerCase() !== reponse.emailRepondant.toLowerCase()) {

sujetFinal = (T.PREFIXE\_COPIE\_EMAIL || "Copie : ") + sujet;

}

if (isDebugMode) { LoggerLib.logEmailAttempt(adresse, sujetFinal); }

MailApp.sendEmail({ to: adresse, subject: sujetFinal, body: corpsEmail, attachments: piecesJointes });

if (isDebugMode) { LoggerLib.logSuccess(`Envoi réussi à ${adresse}`); }

} catch (e) {

if (isDebugMode) { LoggerLib.logError(`Échec de l'envoi à ${adresse}`, e); }

}

});

if (isDebugMode) { LoggerLib.logEnd(); }

}

function findAttachments(typeTest, profilCode, niveauEmail, langueCode) { /\* ... (inchangé) ... \*/ }

function mapQuestionsById(bdd, nomFeuille) { /\* ... (inchangé avec typo corrigée) ... \*/ }

function getOriginalLanguage(reponses) { /\* ... (inchangé) ... \*/ }

**5.2. Architecture Détaillée et Traçabilité**

* **LoggerLib (Apps Script)** (AJOUTÉ): Nouvelle librairie partagée.
  + **Rôle :** Centralise la logique de logging pour le débogage.
  + **Fonctions Principales :** logStart, logResults, logEmailQueue, logEmailAttempt, logSuccess, logError, logEnd.
* **[TEMPLATE] Kit de Traitement V2 (Google Sheet)**
  + **TraitementReponses.gs** (MODIFIÉ):
    - Ajout d'un interrupteur global MODE\_DEBUG\_ACTIF pour contrôler les logs.
    - Mise à jour de processSubmissionData pour appeler LoggerLib et pour passer l'état du switch à buildAndSendEmails.
    - Mise à jour de \_findReponseByKeyHint pour accepter une liste de mots-clés.
  + **Utilities.gs** (MODIFIÉ):
    - Mise à jour de buildAndSendEmails pour accepter le paramètre isDebugMode et appeler LoggerLib en conséquence.
  + **Logique\_Universel.gs** (MODIFIÉ):
    - Correction d'une typo dans la fonction mapQuestionsById (nomFeufile -> nomFeuille).

**5.3. Nomenclature des Variables**

* **MODE\_DEBUG\_ACTIF** (AJOUTÉ)
  + **Type :** Boolean
  + **Origine/Portée :** Constante globale dans TraitementReponses.gs.
  + **Contenu :** true ou false.
  + **Exemple :** const MODE\_DEBUG\_ACTIF = true;

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

* **Fichier : [CONFIG] Usine à Tests**
  + **Onglet Paramètres Généraux :** ... (colonnes existantes) , Repondant\_Email\_Actif (AJOUTÉ).

**6. Actions Recommandées / Prochaines Étapes**

* **Action Immédiate :** Mettre la constante MODE\_DEBUG\_ACTIF à false dans le script TraitementReponses.gs du [TEMPLATE] maître, afin que tous les futurs kits générés soient en mode production par défaut.
* **Prochaine Session :** Comme convenu, nous allons nous baser sur ce document pour mettre à jour le **"Manuel d'Architecture"** général du projet, afin de documenter officiellement la nouvelle architecture de débogage.