🚀 John Von Gurt — Tutoriel d'installation

Ce tutoriel installe l'overlay dans **OBS Studio**, de zéro jusqu'au premier live, puis habille ta **chaîne Twitch**. Chaque page a sa **fiche** : tu peux suivre une fiche seule, sans lire le reste.

> 💡 Pour tout voir en direct : ouvre `index.html` dans ton navigateur. La version à lire confortablement de ce tutoriel est `TUTO.pdf`.

Sommaire

1. Ce qu'il y a dans le dossier
2. Avant de commencer (config, réglages OBS, les 3 gestes de base)
3. Les scènes, fiche par fiche : Démarrage · Pause · Fin · Cam seule · Contenu · Jeu
4. Les sources à la carte : Alertes · Chat · Bandeau · Objectif · Cadre cam (puis « 4bis » : les sons des scènes Démarrage et Fin)
5. Les transitions : Sas · Passage
6. Brancher le chat et les alertes (StreamElements, journal, objectif, sons des alertes)
7. Habiller la chaîne Twitch
8. Tester sans être en live
9. Personnaliser
10. Dépannage

1. Ce qu'il y a dans le dossier

```
johnvongurt/
├── reglages.html       ← LA page pour changer les textes et réglages (elle écrit mes-reglages.js)
├── fabriquer-images.cmd ← fabrique SANS OBS le planning de la semaine, les images du kit, les vidéos de transition (double-clic) : voir 7
├── config.js            ← les valeurs par défaut (remplacé à chaque mise à jour de l'overlay)
├── mes-reglages.js      ← TES réglages, écrits par reglages.html (apparaît au 1er enregistrement ; à garder)
├── index.html           ← aperçu de tout
├── TUTO.md / TUTO.pdf   ← ce tutoriel
├── TUTO-JDR.md / .pdf  ← le tuto de la scène JDR (parties de jeu de rôle)
├── pupitre-jdr.html    ← le pupitre JDR : les PV pendant la partie, dans un dock d'OBS (voir TUTO-JDR)
├── CONCEPT.md / .pdf    ← le résumé du projet (DA, choix, ce qu'il reste à faire)
├── scenes/              ← les écrans et overlays de scène (une page = une scène OBS)
├── sources/             ← les éléments à poser où tu veux (alertes, chat…)
├── sons/                ← tes propres sons d'alerte, si tu veux (facultatif, voir 6.8)
├── transitions/         ← les transitions (+ videos/ : prêtes pour OBS)
├── chaine/              ← kit de chaîne Twitch (kit.html + export/ : les PNG)
├── outils/              ← les scripts (vidéos, images du kit, PDF) et actualiser-obs.lua pour OBS : voir 9
├── design/moodboard.html
└── css/  js/            ← le moteur (pas besoin d'y toucher)
```

Toutes les pages font **1920 × 1080**, avec un fond transparent là où il faut.

2. Avant de commencer

2.1 Remplir les réglages avec `reglages.html`

1. Dans le dossier de l'overlay, double-clique sur `reglages.html` : la page s'ouvre dans ton navigateur (**Edge** ou **Chrome**).
2. Vérifie au minimum, dans **La chaîne** : le **nom affiché** et ton **identifiant Twitch** (celui de l'adresse `twitch.tv/…`).
3. Dans **Objectif** : choisis ce que la barre affiche (**followers** ou **abonnés**) et ton objectif. Les vrais nombres de la chaîne viennent de StreamElements (section 6.6).
4. Clique 💾 **Enregistrer mes réglages** (en bas, ou **Ctrl + S**).
5. **La première fois**, une fenêtre s'ouvre : choisis le **dossier de l'overlay** (celui qui contient `reglages.html`), puis **Sélectionner le dossier**. Le navigateur demande s'il peut modifier les fichiers : clique **Modifier les fichiers** (ou **Autoriser**). La page écrit alors tes réglages dans le fichier `mes-reglages.js`, à côté de `config.js` : rien à copier à la main. Les fois suivantes, elle s'en souvient et enregistre directement (au plus, le navigateur redemande l'autorisation).
6. Dans OBS : **clic droit sur la source › Actualiser** pour voir le changement.

Un **point** • à côté d'un réglage veut dire qu'il a changé et n'est pas encore enregistré. La section **Alertes** montre un aperçu de chaque alerte avec tes textes, et la section **Tester** ouvre les pages en mode test.
Où vont tes réglages ? `config.js` contient les **valeurs par défaut** de l'overlay ; `mes-reglages.js` contient **seulement ce que tu as changé**, appliqué par-dessus. En haut de la page, un encadré dit combien de réglages perso tu as. Remettre un réglage à sa valeur d'origine le retire de `mes-reglages.js`.

> ℹ️ Avec **Firefox**, la page ne peut pas écrire dans le dossier : elle **télécharge** `mes-reglages.js`, à mettre dans le dossier de l'overlay (à la place de l'ancien). Préfère Edge ou Chrome.

À la main (sans la page) : ouvre `config.js` avec le **Bloc-notes** (clic droit › *Ouvrir avec* › *Bloc-notes*). Garde les guillemets `"…"` autour des textes et la virgule `,` en fin de ligne, enregistre, puis **Actualiser** dans OBS. Attention : un changement fait à la main dans `config.js` sera perdu à la prochaine mise à jour ; dans `reglages.html`, il est gardé.

Quand tu reçois une nouvelle version de l'overlay

1. Copie les nouveaux fichiers par-dessus les anciens (remplacer).
2. **Ton** `mes-reglages.js` **n'est pas dans la nouvelle version** : il reste en place, tes réglages sont gardés. Les nouvelles fonctions arrivent avec leurs valeurs par défaut.
3. Dans OBS : actualise les sources.

> 🔁 **La toute première fois seulement** (si tes réglages étaient encore dans l'ancien `config.js`) : **avant** de remplacer les fichiers, fais une copie de ton `config.js` (ex. sur le bureau). Après la mise à jour, ouvre `reglages.html` › 📥 **Reprendre les réglages d'un ancien config.js** › choisis cette copie : tes réglages reviennent dans le formulaire (marqués •). Clique **Enregistrer**, c'est fini : ils sont maintenant dans `mes-reglages.js`.

2.2 Régler le canevas d'OBS (une seule fois)

Paramètres › Vidéo : résolution de base `1920 × 1080`, résolution de sortie `1920 × 1080`.
Les positions des fiches sont données pour ce canevas.

2.3 Geste A — ajouter une source Navigateur

C'est la même manipulation pour **tous** les fichiers `.html` :

1. Sélectionne la scène, puis dans **Sources** : **+** › **Navigateur**.
2. Donne un nom à la source, puis **OK**.
3. Coche **Fichier local**, **Parcourir**, choisis le fichier `.html` indiqué dans la fiche.
4. **Largeur :** `1920`, **Hauteur :** `1080`.
5. Coche les cases indiquées dans la fiche.
6. **OK**.

2.4 Geste B — placer une source au pixel près

1. Clique sur la source (webcam, jeu, capture), puis **Ctrl + E**.
2. **Position** : x et y de la fiche.

3. **Type de zone de délimitation** : *Mettre à l'échelle à l'extérieur de la zone*.
4. **Taille de la zone de délimitation** : largeur et hauteur de la fiche.
5. Coche **Rogner à la zone de délimitation**, puis **Fermer**.

> 💡 **Voir les chiffres directement dans OBS** : `reglages.html` › **Options des scènes** › coche **Afficher la taille et la position des zones**, puis **Enregistrer** et actualise. Chaque zone (webcam, contenu, jeu) affiche sa taille et sa position : tu les recopies dans Ctrl + E. Décoche ensuite. (Ou, pour une seule source : `?zones=1` dans l'adresse, geste C.)

2.5 Geste C — ajouter une option dans l'adresse

1. Dans la source Navigateur, **décoche** *Fichier local*.
2. Colle l'adresse complète dans **URL**, avec les options après un `?` :

```
file:///G:/Projets/Overlay/johnvongurt/scenes/demarrage.html?minutes=10
```

Plusieurs options se séparent par `&` : `...jeu.html?cam=bas-gauche&test=1`

Plus simple, et pour de bon : `reglages.html` **› Options des scènes.** Le coin de la webcam de la scène Jeu (ou « Pas de webcam »), le chat, le bandeau de chaque scène s'y règlent une fois pour toutes, sans toucher aux adresses dans OBS. Une option écrite dans l'adresse d'une source passe **avant** ce réglage (pratique pour une source particulière).

2.6 Astuce — une seule source d'alertes pour toutes les scènes

1. **Scènes › +** : crée une scène **« Global — Alertes »** et ajoutes-y `sources/alertes.html` (fiche 4.1).
2. Dans chaque autre scène : **Sources › + › Scène** › « Global — Alertes », tout **en haut** de la liste.

> 🔑 **Règle d'or** : dans OBS, ce qui est **en haut** de la liste s'affiche **par-dessus**.

2.7 Le chat et le bandeau : déjà dans les scènes

Les scènes contiennent **déjà** leur chat et leur bandeau, placés pile à côté des zones de la cam et du jeu : **tu n'as rien à ajouter.**
La source séparée `sources/chat.html` (ou `sources/bandeau.html`) sert seulement pour une scène à toi, ou pour placer le chat autrement : dans ce cas, ajoute `?chat=0` (ou `?bandeau=0`) à l'adresse de la scène pour retirer celui qui est intégré, puis ajoute la source séparée.

Quand tu changes de scène, le chat **réaffiche les derniers messages** (ceux des 10 dernières minutes, réglable dans `reglages.html` **› Le chat**) : il ne repart pas à vide.

2.8 Actualiser toutes les sources d'un coup (après un changement de réglages)

OBS n'a pas de bouton pour actualiser toutes les sources Navigateur : l'overlay en fournit un, sous forme de petit script OBS.

1. OBS › **Outils › Scripts** › onglet **Scripts** › **+** › choisis `outils/actualiser-obs.lua` (dans le dossier `johnvongurt`).

2. À droite : le bouton **Actualiser toutes les sources Navigateur**, et la case **Actualiser tout seul les sources de l'overlay quand les réglages changent** (cochée) : après **Enregistrer** dans `reglages.html`, les sources se mettent à jour en 2 secondes.
3. **Fermer** : le script reste installé. Raccourci clavier possible : **Paramètres › Raccourcis clavier** › « Actualiser toutes les sources Navigateur ».

> ⚠️ Une page actualisée repart de zéro (un compte à rebours recommence).

Refaire les vidéos de transition et les images du kit (le même script)

Les **vidéos de transition** et les **images du kit Twitch** sont des fichiers déjà fabriqués : après un changement de **couleurs**, du **nom** ou des **textes du kit** dans `reglages.html`, elles gardent l'ancien rendu (la page des réglages te le rappelle quand tu enregistres). Deux boutons les refont avec tes réglages :

- **Refaire les vidéos de transition (avec tes couleurs)** : refait `transitions/videos/*.webm`, puis OBS recharge tout seul tes transitions Stinger de l'overlay. Compte 1 à 2 minutes par vidéo ; pendant ce temps, ces transitions sont coupées (un changement de scène se fait sans animation).
- **Refaire les images du kit Twitch** : refait `chaine/export/*.png` (profil, bannière, hors-ligne, panneaux, emotes, badges) ; le dossier s'ouvre à la fin, il reste à les envoyer sur Twitch (section 7).

OBS reste utilisable pendant ce temps. Le résultat s'affiche sous les boutons (rouvre la fenêtre des scripts pour le voir) et dans **Journal des scripts**.

> **La première fois**, il faut **Node.js** (et **ffmpeg** pour les vidéos). S'ils manquent, le message le dit : **reclique sur le bouton**, ils s'installent tout seuls (2 à 5 minutes, avec l'installeur de Windows), puis ça continue.

Placer les webcams tout seul (le même script)

Dans la même fenêtre **Outils › Scripts**, le script a aussi :

- le bouton **Placer les webcams sur toutes les scènes (et les ajouter là où elles manquent)** : dans chaque scène qui affiche une page de l'overlay (`scenes/jeu.html`, `contenu.html`, `cam-seule.html`…), ta webcam est mise **pile dans sa zone**, à la bonne taille, et rognée. Si une scène n'a pas encore de webcam, elle y est ajoutée, juste sous l'overlay ;
- la case **Replacer tout seul les webcams quand les réglages changent** (cochée) : tu changes la webcam de la scène Jeu dans `reglages.html` › **Options des scènes**, tu enregistres… et dans OBS la webcam se déplace en même temps que son cadre.

Pour que le script reconnaisse ta webcam, **son nom doit contenir « cam »** (ex. « Webcam »). Une carte d'acquisition de console n'est jamais déplacée (sauf si son nom contient « cam »). Une scène réglée **Pas de webcam** : la webcam y est cachée, puis réaffichée quand tu la remets. Les positions viennent de `js/zones.js` (les mêmes que dans les fiches ci-dessous) ; si ton canevas est en 1440p, le script fait la conversion tout seul.

> Le rognage automatique demande **OBS 30.1 ou plus récent**. Avec un OBS plus ancien, si la webcam déborde de son cadre : clic droit › **Transformer** › **Rogner** à la main (geste B).

> ✅ **Plus besoin de cocher « Actualiser le navigateur quand la scène devient active »** : le démarrage, la pause, la fin et les transitions **repartent tout seuls de zéro à chaque fois que leur scène passe à l'antenne**, et se mettent en pause quand elle n'est plus à l'écran. Cocher la case ne gêne pas.

3. Les scènes, fiche par fiche

3.1 🚀 Démarrage — `scenes/demarrage.html`

À quoi ça sert : la fusée se ravitaille sur le pas de tir au rythme du compte à rebours, pendant que les vérifications se cochent (la ligne du bas du panneau dit ce qui est en cours). À T-30, un message « Ravitaillement terminé » apparaît quelques secondes sous la fusée ; de T-10 à T-1, le décompte dans le panneau ; à T-0, **décollage** ! « Décollage ! » puis « Lancement réussi » s'affichent dans le panneau, qui se ferme ensuite : la caméra **suit la fusée** dans l'espace.

Dans OBS :

1. **Scènes › +** : « Démarrage ».
2. **+ › Scène** › « Global — Alertes ».
3. Geste A avec `scenes/demarrage.html`.
4. Pour les sons : clic droit sur la source › **Propriétés** › coche **Contrôler l'audio via OBS** (le volume se règle ensuite dans le mélangeur audio).

Changer la durée sans rien ouvrir : dans OBS, clic droit sur la source › **Interagir**, bouge la souris : un bouton ⚙ **DURÉE** apparaît en haut à gauche (invisible à l'antenne). Choisis la durée, elle est gardée pour les fois suivantes.
Options : `?minutes=10` (passe avant le bouton ⚙ DURÉE ; 5 min par défaut, réglable dans `reglages.html` › Démarrage, avec tous les textes) · `?heure=20:30` (heure fixe : le compteur arrive à zéro à 20 h 30 ; aussi dans `reglages.html` › **… ou heure fixe**, prioritaire sur les minutes)
Vérifier : ouvre `scenes/demarrage.html?minutes=0.5` dans ton navigateur, le décollage arrive en 30 s.

3.2 ⏸ Pause — `scenes/pause.html`

À quoi ça sert : l'écran clair « Transmission en pause », la fusée en vol dans l'anneau de chargement.

Dans OBS :

1. **Scènes › +** : « Pause ».
2. **+ › Scène** › « Global — Alertes ».
3. Geste A avec `scenes/pause.html`.

Le chrono « En pause depuis » repart de zéro à chaque passage à l'antenne.

3.3 🌙 Fin — `scenes/fin.html`

À quoi ça sert : l'alunissage (jambes déployées, poussière, drapeau), puis « Mission accomplie ».

Dans OBS :

1. **Scènes › +** : « Fin ».

2. Geste A avec `scenes/fin.html` (l'alunissage rejoue à chaque passage à l'antenne).
3. Pour les sons : coche **Contrôler l'audio via OBS** dans les propriétés de la source.

3.4 🎙️ Cam seule — `scenes/cam-seule.html`

À quoi ça sert : la discussion : ta cam en grand, le chat à côté, le bandeau en bas.

Dans OBS :

1. **Scènes › +** : « Cam seule ».
2. **+ › Scène** › « Global — Alertes ».
3. Geste A avec `scenes/cam-seule.html`.
4. **+ › Périphérique de capture vidéo** › « Webcam », choisis ta caméra.
5. Geste B sur la webcam : position `80`, `210` · taille `1260` × `709`.
6. Ordre final : `Global — Alertes` · `Overlay` · `Webcam`.

Vérifier : adresse `cam-seule.html?apercu=1` (geste C), la zone de la cam est grisée. Retire `?apercu=1` ensuite.

3.5 📱 Contenu — `scenes/contenu.html`

À quoi ça sert : la cam et le chat en colonne à gauche, le contenu à droite.

Dans OBS :

1. **Scènes › +** : « Contenu ».
2. **+ › Scène** › « Global — Alertes ».
3. Geste A avec `scenes/contenu.html`.
4. **+ › Périphérique de capture vidéo** › « Webcam ».
5. **+ › Capture de fenêtre** (ou *Capture d'écran*, *Navigateur*…) › « Contenu ».
6. Geste B :

Source	Position (x, y)	Taille (l × h)
Webcam	`60`, `250`	`480` × `270`
Contenu	`580`, `190`	`1280` × `720`

7. Ordre final : `Global — Alertes` · `Overlay` · `Webcam` · `Contenu`.

Options : `?cam=0` (pas de webcam : le chat prend toute la hauteur ; n'ajoute pas la source Webcam).
Le voyant REC n'est que sur la cam : le cadre du contenu n'en a pas.

3.6 🎮 Jeu — `scenes/jeu.html`

> Un cadre fin (carré, coins à peine arrondis) fait le tour de l'écran, collé aux bords : le jeu ne dépasse jamais. `?coins=0` le retire (réglable dans `reglages.html` › Options des scènes). Même cadre dans la scène Contenu : `?cadre=0` pour le retirer.

À quoi ça sert : le jeu en plein écran, une petite cam dans un coin, le chat en transparence.

Dans OBS :

1. **Scènes › +** : « Jeu ».
2. **+ › Scène** › « Global — Alertes ».
3. Geste A avec `scenes/jeu.html`.
4. **+ › Périphérique de capture vidéo** › « Webcam ».
5. **+ › Capture de jeu** › « Jeu ».
6. Geste B : le **Jeu** en `0`, `0`, `1920` × `1080` ; la **Webcam** en `400` × `225`, selon le coin :

Coin (option `cam`)	Position de la webcam (x, y)
`bas-droite` *(défaut)*	`1456`, `735`
`bas-gauche`	`64`, `735`
`haut-droite`	`1456`, `180`
`haut-gauche`	`64`, `180`

7. Ordre final : `Global — Alertes` · `Overlay` · `Webcam` · `Jeu`.

Options : `?cam=bas-gauche` (le chat passe automatiquement de l'autre côté) · `?cam=0` (pas de webcam : pas de cadre, n'ajoute pas la source Webcam ; le chat reste à sa place) · `?cam=1200,700,420,236` (position perso : x, y, largeur, hauteur).

Position perso (au pixel près) : `reglages.html` › **Options des scènes** › **La webcam de la scène Jeu** › **Position perso** (X, Y, largeur, hauteur, ou fais glisser la cam sur le plan) ; ou dans l'adresse : `?cam=1200,700,420,236`. **Le plus simple pour la poser dans OBS** : le bouton **Placer les webcams** du script `actualiser-obs.lua` (voir 2.8), qui suit aussi la position perso.

3.7 🎲 JDR — `scenes/jdr.html` (les parties de jeu de rôle)

Tout est dans le tuto à part : `TUTO-JDR.pdf` (les personnages, la scène dans OBS, le pupitre, le dépannage). En bref :

À quoi ça sert : tes parties de jeu de rôle. La table de jeu en grand (CodexVTT ou un autre site), un **cadre par personnage**
dessous (son image, son nom, ses PV, une ressource, « à terre », « son tour »), le nom de la campagne et le chat à droite.
Les personnages se règlent dans `reglages.html` › **Scène JDR** ; leurs PV se changent pendant la partie dans le **pupitre**
(`pupitre-jdr.html`, un dock d'OBS : TUTO-JDR, chapitre 4).

Dans OBS (détails : TUTO-JDR, chapitre 3) :

1. **Scènes › +** : « JDR ». Tes alertes en haut.
2. **Geste A** avec `scenes/jdr.html` › « Overlay JDR ».
3. **+ › Capture de fenêtre** › « Table » : la fenêtre de ton navigateur ouvert sur la table de jeu, **sous** l'overlay.
4. **Geste B** sur la table : position `60`, `60`, taille `1440` × `810` (1440p : `80`, `80`, `1920` × `1080`).

5. + › **Image** pour chaque personnage, **sous** l'overlay, dans la case de son portrait (positions : TUTO-JDR, 3.4).
6. + › **Capture audio d'application** › Discord (pour que le stream entende les joueurs).

Options : `?chat=0` · `?titre=…` · `?apercu=1` (la zone de la table et les cases des portraits) · `?test=1` (une table de démonstration)

4. Les sources à la carte

À poser **en plus**, dans n'importe quelle scène. Position et taille avec `?x=&y=&l=&h=` (pixels 1920 × 1080).

4.1 🔔 Alertes — `sources/alertes.html`

À quoi ça sert : follows, abonnements, bits, raids et dons, avec un petit son radio.

Dans OBS :

1. Dans la scène « Global — Alertes » (voir 2.6) : geste A avec `sources/alertes.html`.
2. Coche **Contrôler l'audio via OBS** et règle le volume dans le mélangeur audio.

Options : `?position=haut` (défaut), `centre` ou `bas` · `?test=1`
Nécessite StreamElements (section 6).

4.2 💬 Chat — `sources/chat.html`

À quoi ça sert : le « Canal de communication » seul, pour une scène à toi.
Dans OBS : geste A, puis geste C pour le placer.
Options : `?x=1400&y=160&l=460&h=760` · `?flottant=1` (sans panneau) · `?disparition=45` · `?test=1`

4.3 📰 Bandeau — `sources/bandeau.html`

À quoi ça sert : dernière recrue, dernier abonné, dernier soutien et objectif.
Options : `?x=80&y=980&l=1760&h=76` · `?compact=1` · `?test=1`
Choisir les cases : `reglages.html` › **Bandeau d'infos** (par exemple, décoche le dernier soutien si tu ne reçois ni dons ni bits, et le dernier abonné si ta chaîne n'est pas encore affiliée).

4.4 🌍 Objectif — `sources/objectif.html`

À quoi ça sert : la jauge où la fusée va de la Terre à la Lune à chaque follow (ou abonnement).
Options : `?x=560&y=60&l=800&h=150` · `?test=1`

4.5 🎥 Cadre cam — `sources/cam.html`

À quoi ça sert : le cadre « Flux caméra » seul (voyant REC, plaque de nom), pour une cam placée où tu veux.

Le chrono REC est **le même dans toutes les scènes** : changer de scène ne le remet pas à zéro. Il repart de 0 quand OBS lance le stream ou l'enregistrement (ou après 10 minutes sans aucune page de l'overlay ouverte).

Dans OBS :

1. Place ta webcam et note sa position et sa taille (Ctrl + E).
2. Geste A avec `sources/cam.html` **au-dessus** de la webcam, puis geste C avec les mêmes chiffres : `?x=1456&y=735&l=400&h=225`.

Options : `?nom=1` (plaque « Nom — Grade ») · `?titre=Flux%20caméra` (`?titre=` pour masquer) · `?apercu=1`

4bis. Les sons (scènes Démarrage et Fin)

Tout est déjà branché, rien à installer. Les sons sont des fichiers dans `assets/audio/` :

Fichier	Quand
`preparation.ogg`	Démarrage : ambiance du pas de tir (ventilation, purges de vapeur, radio lointaine), en boucle
`chauffe.ogg`	À **T-15 s** : les moteurs chauffent (le grondement monte en volume et en hauteur jusqu'au décollage), en boucle
`decollage.ogg`	À **T-3 s** : allumage, puis le grand « boum » pile au décollage
`propulseur.ogg`	Après le décollage : boucle du propulseur tant que la fusée vole (s'adoucit peu à peu)
`atterrissage.ogg`	Fin : descente moteurs allumés, contact avec le sol au moment où la fusée se pose
`musique.ogg`	Fin : musique d'ambiance en boucle, après l'atterrissage

La voix du compte à rebours (T-15 s : « T moins quinze secondes », puis « dix, neuf… un », puis « Décollage ») :

- ce sont **tes propres enregistrements** : `assets/audio/voix/15.mp3`, `10.mp3` … `1.mp3` et `0.mp3`, utilisés individuellement ;
- si un fichier manque, **rien n'est joué pour ce nombre** : il n'y a aucune voix synthétique.

Tes propres sons AVANT les 15 dernières secondes (annonce à T-60, musique à T-30…) : dans `reglages.html` › Sons › Réglages avancés › **Annonces avant T-15 s** (ou `config.js` › `audio` › `reperes`), écris **une ligne par son** : `secondes restantes | fichier`.

- `60` → joue `assets/audio/voix/60.mp3` (ou `.ogg` / `.wav`) quand il reste 60 secondes ;
- `30 | sons/ouverture.mp3` → joue ce fichier quand il reste 30 secondes (chemin depuis le dossier de l'overlay, ou un chemin complet `C:\\…\\son.mp3`) ;
- un repère n'est joué **que si le compte à rebours est assez long** pour l'atteindre (un compte à rebours de 45 s ignore « 60 »), et **une seule fois** par lancement ;
- ils suivent le volume général et le volume de la voix. Pour tester sans attendre : `demarrage.html?minutes=1.5`.

Réglages : `reglages.html` › Sons (ou `config.js` › `audio`) : on/off, volume général, volume de ta voix ; en « Réglages avancés » : annonces avant T-15 s et volume de chaque son. Dans l'adresse d'une scène : `?`

`audio=0` coupe tout, `?volume=0.4` règle le volume.

Changer un son : remplace le fichier en gardant **le même nom** (`.ogg`). Les sons fournis sont fabriqués par ordinateur (`outils/generer-sons.py` , pour les refaire ou les retoucher) : pour un rendu plus réaliste, remplace-les par de vrais enregistrements dont tu as les droits.

Tester : ouvre `scenes/demarrage.html?minutes=0.5` dans ton navigateur et clique une fois sur la page (le navigateur n'autorise le son qu'après un clic ; pas OBS). Le décollage arrive en 30 s : on entend le pas de tir, la montée à T-15 s, la voix, le décollage puis le propulseur.

5. Les transitions

Les vidéos sont **déjà prêtes** dans `transitions/videos/` (fond transparent). Pour chacune :

1. Panneau **Transitions de scène** › **+** › **Stinger**, donne-lui un nom.
2. **Fichier vidéo** : la vidéo du tableau.
3. **Type de point de transition** : *Temps (millisecondes)* ; **Point de transition** : la valeur du tableau.
4. **OK**.

Transition	Vidéo	Point de transition
🚪 Sas (portes blindées)	`sas.webm`	`1100 ms`
🚀 Passage (la fusée traverse l'écran)	`passage.webm`	`1000 ms`

Une transition par scène : clic droit sur la scène › **Remplacer la transition**.
Sans vidéo : ajoute `transitions/sas.html` tout en haut de la scène, transition d'OBS sur **Coupure** : le sas s'ouvre à chaque arrivée sur la scène.

Refaire les vidéos (après un changement de couleurs) : le plus simple, OBS › **Outils** › **Scripts** › **Refaire les vidéos de transition** (2.8). Ou dans PowerShell :

```
cd G:\Projets\Overlay\johnvongurt
node outils/generer-transitions.mjs
```

6. Brancher le chat et les alertes

Deux choses différentes :

Quoi	Comment ça arrive	À installer
Le chat	l'overlay lit ton chat Twitch directement	**rien** : ton identifiant Twitch dans les réglages suffit
Les alertes, le bandeau et l'objectif (follows, abonnements, bits, raids, dons)	par **StreamElements**, un service gratuit (par internet)	**rien** : un compte StreamElements et son jeton, une fois (6.2)

6.1 Le chat

1. `reglages.html` › **La chaîne** › **Identifiant Twitch** : celui de ton adresse `twitch.tv/…`. Enregistre.
2. C'est tout : dans les scènes, le chat s'affiche dès qu'un message arrive.

Les **emotes Twitch** s'affichent en image ; les **bots** (liste dans `reglages.html` › **Le chat**) et les **commandes** qui commencent par `!` sont cachés ; un message **supprimé par un modo**, ou ceux d'un spectateur **banni**, disparaissent aussi de l'overlay ; en changeant de scène, les messages des 10 dernières minutes sont réaffichés.
Il ne montre pas les messages envoyés **avant** l'ouverture d'OBS, ni les emotes des extensions 7TV, BTTV ou FFZ.

6.2 Brancher StreamElements (une seule fois, 3 minutes)

StreamElements est un service **gratuit**, par internet : **rien à installer** sur le PC, rien à lancer avant le live.

1. Va sur **streamelements.com** › **Login** › **Twitch**, connecte-toi avec **ton compte de streamer** et autorise. (Déjà un compte StreamElements ? Connecte-toi simplement.)
2. Ouvre la page de ton compte : **https://streamelements.com/dashboard/account/channels** (ou : ton avatar en haut à droite › **Account** › onglet **Channels**).
3. Clique **Show secrets** (« Afficher les secrets »), puis le bouton **copier** à côté de **JWT Token** : une très longue suite de lettres et de chiffres.
4. Ouvre `reglages.html` › **StreamElements** › colle-le dans **Ton jeton StreamElements**, puis 💾 **Enregistrer**.
5. Dans OBS : **clic droit sur la source d'alertes › Actualiser** (et sur les scènes, pour le bandeau et l'objectif).

> 🔒 **Ce jeton est un secret** (c'est la clé de ton compte StreamElements) : ne le montre pas en live, ne l'envoie à personne. Il reste dans `mes-reglages.js`, sur ton PC : les mises à jour de l'overlay ne l'emportent jamais. S'il a fuité : même page › **Show secrets** › bouton pour en refaire un, puis recolle le nouveau.

✅ **À chaque live** : rien à faire. L'overlay se branche tout seul à l'ouverture d'OBS, et se rebranche tout seul si internet coupe un moment.

6.3 Vérifier que l'overlay est bien branché

OBS n'a pas de console (F12) : l'overlay a donc son propre **journal**, affiché directement dans la source.

1. Dans OBS, double-clic sur la source d'alertes.
2. Décoche **Fichier local** et colle dans **URL** l'adresse de la page suivie de `?journal=1` (geste C) : `file:///G:/Projets/Overlay/johnvongurt/sources/alertes.html?journal=1`
3. **OK** : un panneau sombre apparaît en haut à gauche de l'écran. Tu dois y lire ✅ **Connecté à StreamElements**, puis 📊 **Compteurs de la chaîne** avec ton nombre de followers et d'abonnés.
   - « ⚠️ Pas de jeton StreamElements » : colle ton jeton dans `reglages.html` (6.2).
   - « ❌ StreamElements refuse le jeton » : il est mal copié (il manque un bout ?) ou il a été refait : recopie-le (6.2).

- « … connexion à StreamElements » qui reste seul : pas d'internet sur le PC, ou un pare-feu bloque `astro.streamelements.com`.
- « ⚠️ StreamElements désactivé » : coche « Se connecter à StreamElements » dans `reglages.html`.

4. Chaque événement reçu s'y ajoute (ex. `follow → follow · Pseudo`), avec **les données brutes** reçues de StreamElements en dessous.
5. Une fois vérifié, **retire** `?journal=1` de l'adresse (sinon le panneau reste à l'écran pendant le live).

Le journal marche sur toutes les pages (`?journal=1`), et aussi avec `?test=1` pour voir passer les fausses alertes.

6.4 Événement par événement

Une fois StreamElements branché, **rien à régler par événement** : l'overlay les écoute tous. Les textes se changent dans `reglages.html` › **Alertes** (avec un aperçu).

Événement Twitch	Alerte
Follow	**Nouvelle recrue** — rejoint l'équipage
Abonnement	**Astronaute certifié** — signe pour la mission
Réabonnement	**Astronaute vétéran** — rempile pour X mois
Abonnement offert	**Billet offert** — offre un abonnement à…
Pluie d'abonnements	**Pluie de billets !** *(grande alerte)*
Bits	**Carburant reçu** — ajoute X bits au réservoir
Raid	**Flotte en approche !** *(grande alerte)*
Don	**Soutien de mission** — finance la mission
Objectif atteint	**Objectif atteint !** *(grande alerte)*

Abonnements, abonnements offerts et bits demandent une chaîne **affiliée** ou **partenaire**. Les alertes passent **une par une** (file d'attente) : rien n'est perdu pendant un raid.

6.5 Les dons (facultatif)

Twitch ne gère pas les dons en argent. Avec StreamElements, ils sont **déjà branchés** : ta page de dons est `streamelements.com/<ton pseudo>/tip` (à mettre dans un panneau de ta chaîne, « Soutenir »). Pour recevoir l'argent : streamelements.com › **Revenue** › **Tipping settings**, puis relie ton compte PayPal (ou un autre moyen proposé).

Chaque don arrive tout seul dans l'overlay, avec son montant. *(Les dons faits par un autre service, Ko-fi, Streamlabs ou Tipeee, n'arrivent pas : passe par la page de dons StreamElements.)* Pour l'habiller aux couleurs de l'overlay (bannière et fond) : section 7.

6.6 L'objectif (followers OU abonnés, avec les vrais nombres)

Une seule barre, qui affiche au choix **les followers** ou **les abonnés**. Chacun a son nom, son objectif et son compteur : on bascule quand on veut, la barre est tout de suite juste.

A. Choisir ce que la barre affiche

1. `reglages.html` › **Objectif** › **Ce que la barre affiche** : **Les followers** ou **Les abonnés**.
2. Pour chacun : le **nom de l'objectif** et l'**objectif à atteindre** (ex. `50`).
3. 💾 **Enregistrer**, puis actualise les sources dans OBS.

Une seule source peut aussi afficher l'autre compteur sans toucher aux réglages : ajoute `?objectif=sub` (ou `?objectif=follow`) à son adresse.

B. Les vrais nombres de la chaîne (automatique avec StreamElements)

Rien à faire de plus : dès que StreamElements est branché (6.2), l'overlay lit le **vrai total** de followers et d'abonnés de ta chaîne, à l'ouverture de chaque page puis à chaque changement. Rien n'est modifié sur Twitch : il ne fait que lire.

Vérification : ajoute `?journal=1` à la source d'objectif (6.3). Une ligne **« 📊 Compteurs de la chaîne »** doit apparaître avec tes nombres, juste après « ✅ Connecté à StreamElements ».

> Les abonnés : seulement si la chaîne est **affiliée** ou **partenaire** (sinon il n'y en a pas). La case **Les vrais nombres de la chaîne, depuis StreamElements** (`reglages.html` › **Objectif**) doit rester cochée (c'est le cas par défaut).

Sans StreamElements : décoche cette case et mets ton vrai nombre dans **Followers : nombre de départ** (ou **Abonnés : nombre de départ**) ; le changer remet ce compteur à cette valeur. L'overlay compte alors seulement ce qui arrive pendant qu'OBS est ouvert.

6.7 Tester les vraies alertes

- **Sans Twitch** : `?test=1` sur une page (fausses alertes toutes les 9 secondes, sans toucher au vrai compteur), ou `reglages.html` › **Tester**.
- **Pour de vrai** : demande à un ami (ou à un deuxième compte) de suivre la chaîne, OBS ouvert.

> ⚠️ **Pas encore vérifié sur un vrai live** : si une alerte montre « Quelqu'un » ou « ? », fais une capture d'écran du journal (6.3) pour faire corriger l'overlay.

6.8 Les sons des alertes

Chaque alerte a **son propre son**, pour savoir ce qui se passe à l'oreille, même en pleine partie :

Alerte	Le son
Follow	grésillement radio et deux bips
Abonnement	trois bips qui montent et un long « biiip » de confirmation
Réabonnement	un petit message en morse (· · —)
Abonnement offert	une double tonalité, comme un téléphone
Pluie d'abonnements	une rafale de bips qui montent
Bits	une pièce de borne d'arcade
Raid	sirène d'alarme puis quatre bips
Don	un accord doux
Objectif atteint	compte à rebours 3, 2, 1… et décollage

Les écouter : `reglages.html` › **Sons des alertes**, bouton ▶ à côté de chaque alerte.

Mettre ton propre son (un mp3, wav ou ogg, court de préférence) :

1. Copie ton fichier dans le dossier `sons/` de l'overlay, par exemple `sons/follow.mp3`.
2. `reglages.html` › **Sons des alertes** : dans la case de l'alerte, écris `sons/follow.mp3`. Clique ▶ pour vérifier.
3. **Enregistrer**, puis dans OBS : clic droit sur la source des alertes › **Actualiser**.

Écris `aucun` dans une case pour que cette alerte reste silencieuse ; vide la case pour revenir au son de l'overlay.
Le volume général et le bouton « son » sont dans `reglages.html` › **Alertes** ; dans OBS, le volume se règle aussi dans le mélangeur audio (case **Contrôler l'audio via OBS** de la source des alertes).

7. Habiller la chaîne Twitch

Tous les visuels sont dans `chaine/` : ouvre `chaine/kit.html` pour les voir. Les images prêtes à envoyer sont dans `chaine/export/`.

Visuel	Fichier	Où l'envoyer sur Twitch
Photo de profil	`profil.png` (800 × 800)	Tableau de bord des créateurs › Paramètres › Chaîne › **Marque** › Photo de profil
Bannière de profil	`banniere.png` (1200 × 480)	… › **Marque** › Bannière de profil
Écran hors-ligne	`hors-ligne.png` (1920 × 1080)	… › **Marque** › Bannière du lecteur vidéo
Panneaux de bio	`panneau-a-propos.png`, `panneau-planning.png`, `panneau-regles.png`, `panneau-materiel.png`, `panneau-commandes.png`, `panneau-soutenir.png`, `panneau-mes-creations.png`, et les réseaux : `panneau-discord.png`, `panneau-youtube.png`, `panneau-tiktok.png`, `panneau-instagram.png`, `panneau-x-twitter.png` (320 × 160)	Ta chaîne › onglet **À propos** › **Modifier les panneaux** › +
Page de dons StreamElements	`dons-banniere.png` (640 × 200) et `dons-fond.png` (1920 × 1080)	streamelements.com › **Revenue** › **Tipping settings** (pas à pas ci-dessous)
Emotes	`emote-decollage`, `casque`, `houston`, `o7`, `gg`, `lune` (-112, -56, -28)	Tableau de bord › **Récompenses des spectateurs** › Emotes *(affilié ou partenaire)*
Badges d'abonné	`badge-mois-1` (1 galon), `-3` (2 galons), `-6` (3 galons), `-9` (planète), `-12` (fusée) (-72, -36, -18)	Tableau de bord › **Récompenses des spectateurs** › Badges d'abonné

Les panneaux, pas à pas :

1. Va sur ta chaîne, onglet **À propos**, active **Modifier les panneaux**.
2. **+** › **Ajouter un panneau texte ou image**.
3. **Image** : choisis le PNG du panneau. **Description** : écris le texte (qui tu es, le planning, les règles du chat…).
4. **Envoyer**, puis recommence pour les autres panneaux.

Les panneaux des réseaux (Discord, YouTube, TikTok, Instagram, X) : à l'étape 3, remplis aussi **Lien de l'image** avec l'adresse du réseau (ton invitation Discord `https://discord.gg/…`, ta chaîne YouTube…) : un clic sur l'image y mène. Tu n'as pas un de ces réseaux ? Retire son nom dans `reglages.html` › **Kit de chaîne Twitch** › Panneaux de bio. Un autre réseau (Twitch, Kick, Bluesky) : ajoute son nom, il prend son logo.

Le panneau Planning de la semaine : dans `reglages.html` › **Kit de chaîne Twitch** › **Planning**, écris une ligne par jour : `Mercredi | 20h30`, ou avec le jeu : `Mercredi | 20h30 | Minecraft`. Juste en dessous,

choisis la **semaine** dans le calendrier, puis clique 🖼️ **Fabriquer l'image du planning** : la page enregistre le planning, OBS fabrique l'image en quelques secondes et la page te la montre. Elle est rangée dans `chaine/export/plannings/` (ex. `planning-2027-semaine-08.png`, avec « Semaine 8 · du 22 au 28 février » sous le titre) : **un fichier par semaine, les anciens restent**. Sur Twitch : ta chaîne › **À propos** › **Modifier les panneaux** › panneau Planning › change l'image.
Il faut **Edge ou Chrome**. L'image est fabriquée par **OBS** s'il est ouvert avec le script `actualiser-obs.lua` à jour (après une mise à jour de l'overlay, recharge-le : Outils › Scripts › ↻) ; **sans OBS** (pour préparer tes streams à l'avance), double-clique sur `fabriquer-images.cmd` dans le dossier de l'overlay quand la page te le propose : il prend la demande et la page affiche l'image. Tu peux aussi le lancer seul : un petit menu propose le planning d'une semaine (tape son numéro, ou Entrée pour cette semaine), les images du kit ou les vidéos de transition. Planning vide = le panneau titre « Planning ».

Le panneau Mes créations : dans sa description, mets tes liens (site, boutique, portfolio…), et dans **Lien de l'image** le principal. Tu préfères « Mes sites » (avec un globe) ? Renomme-le dans la liste des panneaux de `reglages.html`.

Le panneau Commandes : dans sa description, liste les commandes du chat, une par ligne avec ce qu'elle fait (ex. `!discord` : le lien du Discord · `!planning` : les jours de live). Elles se créent dans streamelements.com › **Chatbot** › **Chat commands** (le bot StreamElements doit être sur ta chaîne : **Chatbot** › **Join channel**). L'overlay cache déjà ces messages et les réponses du bot dans le chat de l'écran.

La page de dons StreamElements, pas à pas :

1. streamelements.com › **Revenue** › **Tipping settings**.
2. Dans l'apparence de la page : **Banner** (bannière) → `chaine/export/dons-banniere.png` ; **Background** (fond) → `chaine/export/dons-fond.png`.
3. **Save**, puis ouvre `streamelements.com/<ton pseudo>/tip` pour voir le résultat. Le formulaire de don se pose au milieu du fond : le décor est sur les côtés exprès.
4. Mets ce lien dans la description du panneau **Soutenir**.

Changer les textes (slogan, planning, titres des panneaux, titre et phrase de la page de dons) : `reglages.html` › **Kit de chaîne Twitch**, puis refais les images : OBS › **Outils › Scripts › Refaire les images du kit Twitch** (2.8). Ou dans PowerShell :

```
cd G:\Projets\Overlay\johnvongurt
node outils/exporter-chaine.mjs
```

> Les noms des menus Twitch changent parfois un peu : si tu ne trouves pas un intitulé, cherche « Marque » ou « Récompenses des spectateurs » dans le tableau de bord.

8. Tester sans être en live

Option	Effet
`?test=1`	faux messages de chat et fausses alertes qui défilent
`?apercu=1`	affiche les zones de la cam et du jeu
`?journal=1`	affiche le journal : connexion à StreamElements et derniers événements reçus (6.3)
`?objectif=sub` / `?objectif=follow`	la barre d'objectif affiche les abonnés / les followers, quel que soit le réglage (6.6)
`?chat=0` · `?bandeau=0`	retire le chat ou le bandeau intégré à la scène
`?cam=0`	*(Jeu, Contenu)* pas de webcam : le cadre disparaît et le chat s'agrandit
`?cam=1200,700,420,236`	*(Jeu)* webcam à une position perso : x, y, largeur, hauteur (pixels 1920 × 1080)
`?minutes=0.5`	*(démarrage)* compte à rebours de 30 s, pour voir le décollage vite
`?theme=clair` / `?theme=sombre`	force le thème

Le mode test **ne modifie pas** le vrai compteur de l'objectif. Le plus simple : ouvre `index.html` (tout y tourne en mode test), ou `reglages.html` › **Tester**.

9. Personnaliser

Presque tout se règle dans `reglages.html` (2.1), sans toucher au code.

Je veux changer…	Où
Les textes, titres, messages, l'objectif	`reglages.html` (ou `config.js` à la main)
Le vocabulaire des alertes (avec aperçu)	`reglages.html` › Alertes
Le son de chaque alerte, ou ton propre fichier	`reglages.html` › Sons des alertes (voir 6.8)
La durée, le volume ou le son des alertes	`reglages.html` › Alertes
Les bots masqués, la mémoire du chat	`reglages.html` › Le chat
Les cases du bandeau (dons, abonnés…)	`reglages.html` › Bandeau d'infos
Les visuels de la chaîne	`reglages.html` › Kit de chaîne Twitch (puis `node outils/exporter-chaine.mjs`, voir « Les scripts »)
Le coin ou la position exacte de la webcam, pas de webcam, le chat ou le bandeau d'une scène	`reglages.html` › Options des scènes
Les couleurs, une ambiance (Bleu glace, Rouge Mars, Halloween…)	`reglages.html` › Couleurs (voir ci-dessous)
Les couleurs d'origine, les polices	début de `css/theme.css`
Changer l'objectif affiché, ou le compter à la main	`reglages.html` › Objectif (6.6)

Dans les textes des alertes, `{nom}`, `{montant}`, `{mois}`, `{nombre}` et `{destinataire}` sont remplacés automatiquement.

Changer d'ambiance en un clic

1. Ouvre `reglages.html` › section **Couleurs**.
2. Clique une ambiance toute prête — ❄️ **Bleu glace**, 💚 **Vert terminal**, 🌌 **Violet nébuleuse**, 🔴 **Rouge Mars**, 💗 **Rose néon**, 🌼 **Clair**, 🎃 **Halloween**, 🎄 **Noël** — ou change une couleur à la main (le nuancier, ou un code comme `#FF7A1A`). ↺ remet la couleur d'origine d'une seule couleur ; ↺ **Couleurs d'origine** les remet toutes.
3. **Enregistrer**, puis actualise les sources dans OBS (ou laisse faire le script de la section 2) : toutes les scènes, sources et alertes prennent ces couleurs.

Créer ta propre ambiance (ex. Batman) et la garder

1. Dans `reglages.html` › **Couleurs**, règle les couleurs comme tu veux (nuancier ou code).
2. Sous **Mes ambiances**, tape un nom (ex. `Batman`) puis clique 💾 **Sauvegarder ces couleurs** : l'ambiance est écrite tout de suite dans `mes-reglages.js`, avec son nom et ses couleurs. Même nom qu'une ambiance existante = elle est remplacée (la page demande confirmation).
3. Elle apparaît ensuite en bouton : **un clic** remet toutes ses couleurs, puis **Enregistrer** pour que l'overlay les prenne. × la supprime.

Sauvegarder une ambiance ne change pas les couleurs de l'overlay : seul **Enregistrer** le fait.

Les **vidéos de transition** et les **images du kit de chaîne** sont déjà fabriquées : pour qu'elles prennent les nouvelles couleurs, refais-les : OBS › **Outils** › **Scripts** › **Refaire les vidéos de transition** puis **Refaire les**

images du kit Twitch (2.8) ; `reglages.html` te le rappelle quand tu enregistres. Et quand tu reviens aux couleurs d'origine, pareil. (Sans OBS : les scripts ci-dessous.)

Les scripts du dossier `outils/`

Ces scripts **refont les fichiers « fabriqués »** : vidéos de transition, images du kit, PDF… On ne s'en sert qu'après un changement (couleurs, textes du kit, tuto modifié). Tout le reste (réglages, textes, alertes) se fait dans `reglages.html`, sans script.

Une seule fois : installer Node.js (le programme qui lance les scripts `.mjs`)

1. Va sur le site officiel **nodejs.org**, télécharge la version **LTS** et installe-la (tout laisser par défaut, *Suivant* jusqu'au bout). Ou, dans PowerShell : `winget install OpenJS.NodeJS.LTS`
2. Pour **refaire les vidéos de transition**, il faut aussi **ffmpeg** : dans PowerShell, `winget install Gyan.FFmpeg`.
3. Ferme puis rouvre PowerShell.

Lancer un script

1. Ouvre le dossier `johnvongurt` dans l'Explorateur Windows.
2. Clic droit dans un espace vide du dossier › **Ouvrir dans le Terminal** (ou tape `powershell` dans la barre d'adresse du dossier, puis **Entrée**).
3. Tape la commande du tableau, puis **Entrée**. Le script dit ce qu'il fait, puis rend la main.

Script	À quoi il sert	Quand	Commande
`actualiser-obs.lua`	bouton « Actualiser toutes les sources Navigateur » dans OBS, bouton « Placer les webcams » (et replacement automatique), et actualisation automatique quand tes réglages changent	une fois, à installer dans OBS (section 2)	*pas de commande :* OBS › Outils › Scripts › +
`fabriquer-images.cmd`	fabrique sans OBS : le planning d'une semaine (et répond au bouton des réglages), les images du kit, les vidéos de transition	quand OBS est fermé	double-clic (dossier de l'overlay)
`generer-transitions.mjs`	refait les vidéos `transitions/videos/*.webm` (Stinger)	après un changement de couleurs	`node outils/generer-transitions.mjs` *(ffmpeg nécessaire)*
`exporter-chaine.mjs`	refait les images `chaine/export/*.png` (profil, bannière, panneaux, emotes, badges)	après un changement de couleurs ou des textes du kit	`node outils/exporter-chaine.mjs`
`generer-pdf.mjs`	refait `TUTO.pdf` et `CONCEPT.pdf` depuis les `.md`	après une modification de `TUTO.md` ou `CONCEPT.md`	`node outils/generer-pdf.mjs`
`capturer.mjs`	fait une capture PNG d'une page (pour vérifier une animation, ou l'envoyer)	pour vérifier	`node outils/capturer.mjs "scenes/jeu.html?test=1"`
`polices-locales.mjs`	copie dans `assets/polices/` les polices du thème (pour ne plus dépendre d'internet)	seulement si on change de police (déjà fait)	`node outils/polices-locales.mjs` *(internet nécessaire)*

Les scripts se servent de **Microsoft Edge** en coulisses (déjà installé avec Windows) : rien d'autre à installer.

10. Dépannage

Le bouton « Fabriquer l'image du planning » ne fait rien
→ La page dit ce qui bloque, sous le bouton. « Personne n'a pris la demande » : double-clique sur `fabriquer-images.cmd` (sans OBS) ; ou, dans OBS, recharge le script `actualiser-obs.lua` (Outils › Scripts › ↻) et vérifie que son **Dossier de l'overlay** est bien celui que tu as choisi dans `reglages.html`. Il faut Edge ou Chrome (pas Firefox).

Mes réglages ont disparu après une mise à jour de l'overlay
→ Ton `mes-reglages.js` a peut-être été remplacé ou supprimé : il ne doit **pas** faire partie des fichiers que tu copies. S'il te reste une copie, remets-la dans le dossier de l'overlay.
→ Si tes réglages étaient encore dans l'ancien `config.js` (avant `mes-reglages.js`) : `reglages.html` › 📥 **Reprendre les réglages d'un ancien config.js** › choisis une copie de cet ancien fichier, puis **Enregistrer** (section 2.1).

Rien ne s'affiche / page blanche
→ Vérifie `1920` × `1080` sur la source, puis clic droit › **Actualiser**. Si tu viens de modifier `config.js`, cherche un guillemet ou une virgule manquant.

Les polices ne sont pas les bonnes
→ Elles sont dans `assets/polices/` (pas besoin d'internet) : vérifie que le dossier est bien là, à côté de `css/`, puis actualise la source.

Le chat n'affiche rien
→ Vérifie l'identifiant Twitch dans `reglages.html` › **La chaîne** (l'identifiant exact, en minuscules). Le chat n'affiche que les messages envoyés **après** l'ouverture de la page, et il a besoin d'internet.

Les alertes ne s'affichent pas
→ Le jeton StreamElements est-il collé dans `reglages.html` (6.2) ? Regarde le journal (6.3, `?journal=1`) : il dit ce qui coince (pas de jeton, jeton refusé, pas d'internet). Puis actualise la source d'alertes.

Les réglages ne changent rien dans OBS
→ Après **Enregistrer** dans `reglages.html`, il faut actualiser les sources : clic droit › **Actualiser**, ou le script `actualiser-obs.lua` (section 2) qui le fait tout seul.

On n'entend pas le son des alertes
→ Coche **Contrôler l'audio via OBS** sur la source d'alertes et vérifie son volume dans le mélangeur.

Une alerte s'affiche mal (pseudo manquant, « ? »…)
→ OBS n'a pas de console (F12) : ajoute `?journal=1` à l'adresse de la source d'alertes (décoche *Fichier local*, colle l'adresse de la page suivie de `?journal=1`). Un panneau affiche la connexion à StreamElements (« ✅ Connecté ») et chaque événement reçu avec ses données brutes : fais-en une capture d'écran pour faire corriger l'overlay, puis retire `?journal=1`.

Une ancienne image s'affiche encore après une modification
→ Dans le navigateur : **Ctrl + F5** (recharge en ignorant la mémoire du navigateur).
→ Dans OBS : clic droit sur la source › **Propriétés** › **Actualiser le cache de la page actuelle**.

La cam ne tombe pas pile dans le cadre
→ Le plus simple : OBS › **Outils** › **Scripts** › `actualiser-obs.lua` › **Placer les webcams sur toutes les scènes** (le nom de ta webcam doit contenir « cam »). Sinon :
→ Refais le geste B, et vérifie que la webcam est bien **sous** l'overlay.

Le compte à rebours ou l'alunissage ne repart pas de zéro
→ Ils repartent tout seuls quand leur scène passe à l'antenne. Si ce n'est pas le cas, coche **Actualiser le navigateur quand la scène devient active** sur la source.

Bon live, commandant ! 🚀