🎭 Patagrain — Tutoriel d'installation

Ce tutoriel installe l'overlay dans **OBS Studio**, de zéro jusqu'au premier live : les écrans, le chat, les alertes (follows, abonnements, bits, raids, dons), puis l'habillage de 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 (réglages, canevas OBS, les 3 gestes de base)
3. Les scènes, fiche par fiche : Starting soon · Pause · Fin · Cam seule · Contenu · Jeu · JDR
4. Les sources à la carte : Alertes · Chat · Bandeau · Objectif · Cadre cam
5. Les transitions : Rideau · Coup d'épée · Jet de dé
6. Brancher le chat et les alertes (follows, abonnements, bits, raids, dons, objectif, sons des alertes, les Grelots)
7. Habiller la chaîne Twitch
8. Tester sans être en live
9. Personnaliser
10. Dépannage

1. Ce qu'il y a dans le dossier

```
patagrain/
├── reglages.html        ← LA page pour changer les textes et réglages (elle écrit mes-reglages.js)
├── config.js            ← les valeurs par défaut (remplacé à chaque mise à jour de l'overlay)
├── mettre-a-jour.cmd    ← met l'overlay à jour (double-clic), en gardant tes réglages : voir 2.1
├── fabriquer-images.cmd ← fabrique SANS OBS le planning de la semaine, les images du kit, les vidéos de transition (double-clic) : voir 7
├── mes-reglages.js      ← TES réglages, écrits par reglages.html (apparaît au 1er enregistrement ; à garder)
├── index.html           ← la vitrine : 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, le bouffon) et actualiser-obs.lua pour OBS : voir 9
├── assets/              ← logo, emblèmes, épée, chapeau, le bouffon, polices/
├── design/              ← moodboard et pistes de logo
└── css/  js/            ← le moteur (pas besoin d'y toucher)
```

Toutes les pages sont dessinées en **1920 × 1080** et **s'adaptent toutes seules** à la taille de la source (1440p compris), en restant nettes.

Le bouffon (la mascotte) joue un petit numéro sur chaque écran : cirque sur Starting soon, rêve au coin du feu sur Pause, il surveille la taverne (le chat) sur Cam seule et Contenu, il descend avec les alertes accroché à sa corde, il se cache sous son chapeau posé sur la cam en scène Jeu, et il dit au revoir sur l'écran de Fin. Pour le cacher sur une page : ajoute `?bouffon=0` à son adresse (geste C). Pour le cacher partout : `reglages.html` › **Le bouffon**.

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/…`) et **ce que tu fais aujourd'hui**.

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 (ou automatiquement avec le script de la section 2.8).

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

En un clic (Windows) : OBS › **Outils › Scripts** › `actualiser-obs.lua` (voir 2.8) › bouton **Mettre à jour l'overlay**. Ou, OBS fermé, double-clique sur `mettre-a-jour.cmd` dans le dossier de l'overlay.

- L'overlay télécharge sa dernière version sur le serveur des overlays, et l'installe à la place de l'ancienne.
- **Ton** `mes-reglages.js` **n'est jamais remplacé** : tes réglages sont gardés. Les nouvelles fonctions arrivent avec leurs valeurs par défaut.
- L'ancienne version est d'abord copiée dans `sauvegardes\<date>` (les 3 dernières sont gardées) : si tu avais modifié un fichier à la main, il est là.
- Avec le bouton d'OBS, les sources s'actualisent et les webcams se replacent toutes seules. Sous la case, OBS affiche la version installée et le résultat.

La page de téléchargement (le zip complet, le tuto, « Ce qui a changé ») : l'adresse de `reglages.html` › **Mises à jour**.

À la main (sans le bouton) : copie les fichiers du nouveau zip par-dessus les anciens (remplacer). `mes-reglages.js` n'est pas dans le zip : il reste en place. Puis, 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 :

Paramètre	Valeur conseillée
Résolution de base (canevas)	`2560 × 1440` (celle de ton écran)
Résolution de sortie	`1920 × 1080`
Filtre de mise à l'échelle	Lanczos

Dans les fiches, les positions sont données pour **les deux canevas** : prends la colonne qui correspond au tien.

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 / Hauteur** : la taille de ton canevas (`2560` × `1440` , ou `1920` × `1080`).
5. Coche les cases indiquées dans la fiche (par exemple *Actualiser le navigateur quand la scène devient active*).
6. **OK**.

2.4 Geste B — placer une source au pixel près

Pour la webcam, le jeu ou une capture, sous un overlay :

1. Clique sur la source dans la liste, puis **Ctrl + E** (ou clic droit › *Transformer* › *Modifier la transformation*).
2. **Position** : les valeurs 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** : la largeur et la 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.)

La source remplit alors tout le cadre, sans bande noire : ce qui dépasse est coupé.

2.5 Geste C — ajouter une option dans l'adresse

Certaines pages acceptent des options (durée, coin de la cam…). Un fichier local n'en accepte pas :

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

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

Plusieurs options se séparent par `&` : `...jeu.html?cam=bas-gauche&bouffon=0`

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, le bouffon 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 »**.
2. Ajoutes-y `sources/alertes.html` (fiche 4.1).
3. 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**. Les alertes tout en haut, l'overlay au-dessus de la webcam et du jeu.

2.7 Le chat et le bandeau : déjà dans les scènes

Les scènes **Cam seule**, **Contenu**, **Jeu** et **Pause** contiennent **déjà** leur chat et leur bandeau, placés pile à côté des trous de la cam et du jeu (et le bouffon est accoudé à la carte du chat). **Tu n'as rien à ajouter.**

Les pages `sources/chat.html` et `sources/bandeau.html` servent seulement pour **une scène à toi**, ou si tu veux placer le chat autrement : dans ce cas, ajoute `?chat=0` (ou `?bandeau=0`) à l'adresse de la scène (geste C) pour retirer celui qui est intégré, puis ajoute la source séparée.

Et 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**.
2. Onglet **Scripts** › **+** › choisis `outils/actualiser-obs.lua` (dans le dossier `patagrain`).
3. À droite apparaissent :
   - le bouton **Actualiser toutes les sources Navigateur** ;
   - la case **Actualiser tout seul les sources de l'overlay quand les réglages changent** (cochée) : dès que tu cliques **Enregistrer** dans `reglages.html`, les sources se mettent à jour en 2 secondes, sans

rien toucher ;
    - le **Dossier de l'overlay** (trouvé tout seul).
4. **Fermer**. Le script reste installé (OBS le recharge à chaque démarrage).

Tu peux aussi lui donner un **raccourci clavier** : **Paramètres › Raccourcis clavier** › « Actualiser toutes les sources Navigateur ».

> ⚠️ Une page actualisée repart de zéro : évite d'enregistrer des réglages pendant le compte à rebours de Starting soon.

Mettre à jour l'overlay (le même script)

Le bouton **Mettre à jour l'overlay (tes réglages sont gardés)** installe la dernière version (voir 2.1, « Quand tu reçois une nouvelle version »). Sous le bouton : la version installée, et le résultat de la dernière mise à jour.

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).

3. Les scènes, fiche par fiche

3.1 🎭 Starting soon — `scenes/demarrage.html`

À quoi ça sert : l'écran d'avant-live, « Le spectacle va commencer », avec le compte à rebours « Jet d'initiative ». Le bouffon fait son numéro de cirque (il jongle avec trois dés, fait le poirier, salue) ; à zéro, **il lance un d20 qui retombe sur 20** et le compteur annonce « Les dés sont jetés ! ».

Dans OBS :

1. **Scènes › +** : « Starting soon ».
2. **Sources › + › Scène** › « Global — Alertes » (voir 2.6).
3. Geste A avec `scenes/demarrage.html`, case **Actualiser le navigateur quand la scène devient active** cochée (le compte à rebours repart à chaque fois).
4. Ordre final : `Global — Alertes` puis `Écran`.

Options (geste C) : `?minutes=10` (durée, 5 min par défaut) · `?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) · `?titre=…` · `?titreDuJour=…` · `?bouffon=0`
Vérifier : ouvre `scenes/demarrage.html?minutes=0.2` dans ton navigateur : 12 s plus tard, le bouffon lance le dé.

3.2 🔥 Pause — `scenes/pause.html`

À quoi ça sert : le « Repos court » : le bouffon rêve au coin du feu (un d20 roule dans sa bulle et tombe **au hasard** : il dort tant que le dé ne fait pas 1 ; sur un 1, il se réveille en sursaut, grille un chamallow, le croque et se rendort — environ toutes les 2 à 3 minutes. En test, `?test=1` : le 2e lancer fait 1), et le chat « La taverne » à droite.

Dans OBS :

1. **Scènes › +** : « Pause ».
2. **+ › Scène** › « Global — Alertes ».
3. Geste A avec `scenes/pause.html`, case **Actualiser le navigateur quand la scène devient active** cochée.

Options : `?minutes=10` (affiche « Retour dans 10:00 ») · `?titre=…` · `?sousTitre=…` · `?chat=0` · `?bouffon=0`

3.3 👋 Fin — `scenes/fin.html`

À quoi ça sert : « Fin de la session », remerciements, la carte « Prochaine quête » ; le bouffon fait coucou, une courbette et un saut de joie, sa bulle alterne « Merci d'être venus ! » et « À bientôt, aventuriers ! ».

Dans OBS :

1. **Scènes › +** : « Fin ».
2. Geste A avec `scenes/fin.html`, case **Actualiser le navigateur quand la scène devient active** cochée.

Options : `?prochainStream=Jeudi%2020h30` (ou remplis « Prochain stream » dans `reglages.html`) · `?titre=…` · `?sousTitre=…` · `?bouffon=0`
Les phrases de la bulle : `reglages.html` › **Le bouffon**.

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

À quoi ça sert : l'écran « blabla » : accueil et discussion, ta cam en grand avec la plaque dorée au logo de la chaîne dessous, le chat à côté (le bouffon dépasse du haut de la carte, ses grelots tintent à chaque message), 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 :

Canevas	Position (x, y)	Taille (l × h)
2560 × 1440	`93` , `133`	`1653` × `931`
1920 × 1080	`70` , `100`	`1240` × `698`

6. Ordre final : `Global — Alertes` · `Overlay` · `Webcam`.

Options : `?chat=0` · `?bandeau=0` · `?bouffon=0` (voir 2.7)
Vérifier : mets temporairement l'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 : le contenu (navigateur, vidéo, fenêtre…) à gauche, la cam (avec la plaque dorée au logo dessous) et le chat empilés à droite (avec le bouffon, en plus petit).

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 sur chacune :

Source	Position 1440p	Taille 1440p	Position 1080p	Taille 1080p
Contenu	`80` , `133`	`1707` × `960`	`60` , `100`	`1280` × `720`
Webcam	`1867` , `133`	`613` × `345`	`1400` , `100`	`460` × `259`

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

Options : `?titre=Mon%20titre` (étiquette au-dessus du contenu, `?titre=` pour la masquer) · `?cam=0` (pas de webcam : le chat prend toute la hauteur ; n'ajoute pas la source Webcam) · `?chat=0` · `?bandeau=0` · `?bouffon=0`

3.6 ⚔️ Jeu — `scenes/jeu.html`

À quoi ça sert : le jeu en plein écran, une petite cam dans un coin, le chat en transparence et une barre fine. **Le chapeau posé sur la cam est celui du bouffon** : environ toutes les 3 minutes (le délai varie un peu) et à chaque follow, il se lève dessous, passe la tête, regarde à gauche, à droite, fait un clin d'œil, puis replonge sous son chapeau, qui retombe en tanguant.

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` , taille du canevas (plein écran) ; la **Webcam** selon le coin choisi :

Coin (option `cam`)	Position 1440p	Position 1080p
`bas-droite` *(défaut)*	`1933` , `987`	`1450` , `740`
`bas-gauche`	`67` , `987`	`50` , `740`
`haut-droite`	`1933` , `173`	`1450` , `130`
`haut-gauche`	`67` , `173`	`50` , `130`

Taille de la webcam : `560` × `315` en 1440p, `420` × `236` en 1080p.

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

Options : `?cam=bas-gauche` (le chat passe automatiquement de l'autre côté) · `?chat=0` · `?bandeau=0` · `?bouffon=0` · `?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.

Le rythme du bouffon : `reglages.html` › **Le bouffon** (« environ toutes les … secondes » : 180 par défaut, et « à chaque follow »).

3.7 🎲 JDR — `scenes/jdr.html`

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

Ces pages se posent **en plus**, dans n'importe quelle scène, pour composer tes propres mises en page. Chacune se règle avec `?x=&y=&l=&h=` (position et taille en pixels 1920 × 1080).

4.1 🔔 Alertes — `sources/alertes.html`

À quoi ça sert : follows, abonnements, bits, raids et dons. La carte descend du plafond sur deux cordes, le bouffon accroché d'une main à une troisième salue de l'autre ; tout se balance, puis remonte. Les grelots tintent (petite fanfare et guirlande de fanions pour les grosses alertes : raid, pluie d'abonnements, objectif).

Dans OBS :

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

C'est aussi là que le bouffon joue **les spectacles des Grelots**, payés par les spectateurs (section 6.9), à la suite des alertes.

Options : `?position=haut` (défaut), `centre` ou `bas` · `?bouffon=0` (la carte seule) · `?test=1` · `?grelots=catapulte` (joue ce spectacle tout de suite, pour essayer) · `?grelots=0` (pas de spectacles sur cette source)
Nécessite StreamElements (section 6).

4.2 💬 Chat — `sources/chat.html`

À quoi ça sert : le chat seul. Il est **déjà** dans les scènes (voir 2.7) : cette source sert pour une scène à toi.

Dans OBS : geste A avec `sources/chat.html`, puis geste C pour le placer.
Options : `?x=1400&y=90&l=480&h=800` · `?flottant=1` (sans carte, messages posés sur le jeu) · `?disparition=45` (messages effacés après 45 s) · `?test=1`

4.3 📰 Bandeau — `sources/bandeau.html`

À quoi ça sert : titre du jour, dernier aventurier (follow), dernier chevalier (abonné), dernier tribut (bits ou don) et objectif. Déjà dans les scènes Cam seule, Contenu et Jeu.
Choisir les cases : `reglages.html` › **Bandeau d'infos** (par exemple, décoche « Tribut » si tu ne reçois ni dons ni bits, et « Chevalier » si ta chaîne n'est pas encore affiliée).
Dans OBS : geste A avec `sources/bandeau.html`.
Options : `?x=60&y=950&l=1800&h=100` · `?compact=1` (dernier aventurier + objectif) · `?test=1`

4.4 ⚔️ Objectif — `sources/objectif.html`

À quoi ça sert : la jauge « Guilde des aventuriers » où l'épée avance à chaque follow (ou abonnement : `reglages.html` › Objectif, ou `?objectif=sub`).
Dans OBS : geste A avec `sources/objectif.html`.
Options : `?x=560&y=40&l=800&h=150` · `?test=1`

4.5 🎩 Cadre cam — `sources/cam.html`

À quoi ça sert : le cadre de webcam seul (chapeau au-dessus, dés en bas), pour une cam placée où tu veux.

Dans OBS :

1. Place ta webcam où tu veux 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=1450&y=740&l=420&h=236`.

Options : `?nom=1` (la plaque dorée avec le logo de la chaîne, sous la cam) · `?decor=0` (sans chapeau ni dés) · `?apercu=1`

5. Les transitions

Les trois transitions mettent en scène **le bouffon**. 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	Pour
🎭 Rideau — le bouffon tire la corde du rideau	`rideau.webm`	`1400 ms`	Starting soon → Cam seule, et vers la Fin
⚔️ Coup d'épée — le bouffon tranche l'écran	`epee.webm`	`700 ms`	Cam seule → Jeu
🎲 Jet de dé — le bouffon marche sur le dé qui roule	`de.webm`	`1750 ms`	Jeu ↔ Pause, Contenu

Une transition par scène : clic droit sur la scène › **Remplacer la transition** › choisis-la.

Sans vidéo : ajoute la page (ex. `transitions/epee.html`) **tout en haut** de la scène d'arrivée, case *Actualiser…* cochée, et mets la transition d'OBS sur **Coupure**.

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

```
cd G:\Projets\Overlay\patagrain
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, 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/…` (ex. `patagrain`). Enregistre.
2. C'est tout : dans les scènes Cam seule, Contenu, Jeu et Pause, le chat s'affiche dès qu'un message arrive.

Ce que fait le chat :

- les **emotes Twitch** s'affichent en image, avec la couleur de pseudo de chacun (éclaircie si elle est trop sombre) ;

- des **badges façon JDR** devant les pseudos : ton chapeau (toi), une épée (les modos), un bouclier (les abonnés), un d20 (les VIP) ;
- les **bots** sont cachés (Nightbot, StreamElements… liste dans `reglages.html` › **Le chat**), ainsi que les **commandes** qui commencent par `!` ;
- un message **supprimé par un modo**, ou tous les messages d'un spectateur **banni** ou mis en sourdine, disparaissent aussi de l'overlay ;
- quand tu **changes de scène**, les messages des 10 dernières minutes sont réaffichés.

Ce qu'il ne fait pas : il ne montre pas les messages envoyés **avant** l'ouverture d'OBS, ni les emotes des extensions **7TV, BTTV ou FFZ** (elles s'affichent en texte).

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/patagrain/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, **il n'y a rien à régler par événement** : l'overlay écoute tout seul chacun d'eux.

Sur Twitch	Alerte (textes modifiables dans `reglages.html` › Alertes)	Et aussi	Condition
Follow	**Nouvel aventurier** — *Pseudo* rejoint la cour du roi !	bandeau « Aventurier », objectif (+1), le bouffon sort la tête en scène Jeu	aucune
Abonnement	**Adoubement !** — devient chevalier de la cour	bandeau « Chevalier » (et objectif si réglé sur les abonnements)	chaîne **affiliée** ou **partenaire**
Réabonnement (partagé dans le chat)	**Chevalier fidèle** — sert la cour depuis *X* mois	bandeau « Chevalier »	affilié
Abonnement offert	**Présent royal** — adoube *Destinataire*	bandeau « Chevalier »	affilié
Pluie d'abonnements offerts	**Largesse royale !** — offre *X* adoubements *(grande alerte)*	les cadeaux de la pluie ne font pas chacun une alerte	affilié
Bits	**Tribut au bouffon** — lance *X* pièces d'or	bandeau « Tribut »	affilié
Raid	**Une horde débarque !** — arrive avec *X* compagnons *(grande alerte)*	—	aucune
Don	**Offrande royale** — offre *X* au royaume	bandeau « Tribut »	un service de dons branché (6.5)
Objectif atteint	**Objectif atteint !** *(grande alerte)*	—	automatique quand le compteur atteint la cible

Les alertes passent **une par une** (file d'attente) : pendant un raid suivi de dix follows, rien n'est perdu.

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** : chaque page accepte `?test=1` (fausses alertes toutes les 9 secondes, qui ne touchent pas au vrai compteur). Le plus simple : `reglages.html` › **Tester** › Alertes.
- **Pour de vrai** : demande à un ami (ou à un deuxième compte à toi) de suivre la chaîne, pendant qu'OBS est ouvert. L'alerte « Nouvel aventurier » doit arriver dans les secondes qui suivent.

> ⚠️ **Pas encore vérifié sur un vrai live** : l'overlay a été testé avec un faux serveur StreamElements, pas encore avec le vrai. Il affiche chaque événement reçu dans le journal (6.3). Si une alerte montre « Quelqu'un » ou « ? », fais une capture d'écran du journal (avec les données brutes) 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	quelques grelots et deux clochettes (ding-ding)
Abonnement	deux coups de tambour puis quatre clochettes qui montent
Réabonnement	grelots et petit air de clochettes qui fait un aller-retour
Abonnement offert	clochettes aiguës qui scintillent
Pluie d'abonnements	plein de grelots, la grande fanfare et une pluie d'étincelles
Bits	des pièces d'or qui tombent
Raid	roulement de tambour puis la grande fanfare
Don	une harpe qui monte
Objectif atteint	grelots, fanfare et un grand accord final

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).

6.9 Les Grelots : les spectacles du bouffon

À quoi ça sert : tes spectateurs dépensent leurs **grelots** (les points de fidélité StreamElements) pour offrir un spectacle au bouffon, qui le joue par-dessus la scène (source Alertes). Le fil rouge : sa rivalité avec **Darktagrain**, le bouffon maléfique. Les spectateurs ne connaissent que le **prix** : le spectacle est une surprise.

Commande	Prix	Spectacle (secret !)
`!spectacle10`	10	la peau de banane
`!spectacle25`	25	la tarte à la crème
`!spectacle50`	50	la sérénade
`!spectacle75`	75	le jet du destin
`!spectacle100`	100	la potion mystère
`!spectacle125`	125	pierre, feuille, ciseaux
`!spectacle150`	150	le coffre au trésor
`!spectacle175`	175	la catapulte
`!spectacle200`	200	le duel
`!spectacle300`	300	le dragon

Les spectacles tirés au sort (dé du destin, potion, chifoumi, coffre, catapulte) changent à chaque fois. Ils passent l'un après l'autre, dans la même file que les alertes : rien ne se chevauche. Pendant un spectacle, le bouffon des écrans **Starting soon**, **Pause** et **Fin** s'éclipse (pas de bouffon en double), puis revient tout seul quand c'est fini.

Comment ça marche : StreamElements retire les grelots et son bot répond dans le chat, par exemple « 🔔 Gwendal jette 75 grelots dans le chapeau du bouffon… 🎭 ». L'overlay lit le chat, reconnaît **cette phrase exacte** et lance le spectacle. Si le spectateur n'a pas assez de grelots, StreamElements ne répond rien : il ne se passe rien. Un spectateur qui recopie la phrase lui-même ne déclenche rien : seul le bot compte.

Créer les 10 commandes (une seule fois, 2 minutes) : StreamElements doit être connecté à ta chaîne (son bot est déjà là, puisque `!grelots` marche). Dans **ton** chat Twitch, colle ces lignes **une par une** (ce sont des commandes pour le bot, réservées au streamer et aux modos) :

```
!command add !spectacle10 🔔 ${user} jette 10 grelots dans le chapeau du bouffon… 🎭
!command options !spectacle10 -cost 10
!command add !spectacle25 🔔 ${user} jette 25 grelots dans le chapeau du bouffon… 🎭
!command options !spectacle25 -cost 25
!command add !spectacle50 🔔 ${user} jette 50 grelots dans le chapeau du bouffon… 🎭
!command options !spectacle50 -cost 50
!command add !spectacle75 🔔 ${user} jette 75 grelots dans le chapeau du bouffon… 🎭
!command options !spectacle75 -cost 75
!command add !spectacle100 🔔 ${user} jette 100 grelots dans le chapeau du bouffon… 🎭
!command options !spectacle100 -cost 100
!command add !spectacle125 🔔 ${user} jette 125 grelots dans le chapeau du bouffon… 🎭
!command options !spectacle125 -cost 125
!command add !spectacle150 🔔 ${user} jette 150 grelots dans le chapeau du bouffon… 🎭
!command options !spectacle150 -cost 150
!command add !spectacle175 🔔 ${user} jette 175 grelots dans le chapeau du bouffon… 🎭
!command options !spectacle175 -cost 175
!command add !spectacle200 🔔 ${user} jette 200 grelots dans le chapeau du bouffon… 🎭
!command options !spectacle200 -cost 200
!command add !spectacle300 🔔 ${user} jette 300 grelots dans le chapeau du bouffon… 🎭
!command options !spectacle300 -cost 300
```

Le bot répond à chaque ligne. Pour vérifier : tableau de bord StreamElements › **Chatbot** › **Commandes** › **Commandes personnalisées** : les 10 commandes y sont, avec leur coût. On peut aussi les créer ou les modifier là, à la souris (la réponse, et le coût dans les réglages avancés).

> ⚠️ La **réponse** de chaque commande doit être **exactement** la phrase de `reglages.html` › **Les Grelots** (`${user}` compris). Si tu changes une phrase ou un prix, change-le aux **deux** endroits : dans StreamElements et dans `reglages.html`.

Pourquoi pas `!duel` : StreamElements a déjà un `!duel` (un duel de points entre deux spectateurs, celui de ta bio). D'où les noms `!spectacle10`, `!spectacle25`…

Essayer sans dépenser de grelots :

- dans ton chat, écris `!essai catapulte` : le spectacle se joue tout de suite. Ça ne marche que pour toi, et pour les modos si tu coches la case dans `reglages.html` › **Les Grelots**. Noms : `chute`, `tarte`, `serenade`, `destin`, `potion`, `chifoumi`, `coffre`, `catapulte`, `duel`, `dragon` ;
- sans être en live : `sources/alertes.html?grelots=catapulte` dans le navigateur, ou `reglages.html` › **Tester**.

Pour ta bio Twitch (dans la description du panneau **Commandes**, section 7, ou dans « À propos » sous « Les Commandes de la Taverne ») :

> **Les Spectacles du Bouffon** : `!spectacle10`, `!spectacle25`, `!spectacle50`, `!spectacle75`, `!spectacle100`, `!spectacle125`, `!spectacle150`, `!spectacle175`, `!spectacle200`, `!spectacle300`. Jette tes grelots dans le chapeau du bouffon… Plus tu donnes, plus le spectacle est grand. Lequel ? Surprise !

> ⚠️ **Pas encore vérifié sur un vrai live** : testé avec un faux chat où un faux bot StreamElements répond. Si un spectacle ne part pas, regarde la phrase exacte du bot dans le chat et compare-la avec celle de `reglages.html`.

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` : le bouffon (800 × 800) · autre choix : `profil-embleme.png` (le d20)	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-nat20`, `nat1`, `gg`, `grelot`, `epee`, et le bouffon : `bouffon`, `bouffon-clin`, `bouffon-rire`, `bouffon-choc` (-112, -56, -28)	Tableau de bord › **Récompenses des spectateurs** › Emotes *(affilié ou partenaire)*
Badges d'abonné	`badge-mois-1` (d4), `-3` (d6), `-6` (d8), `-9` (bouclier), `-12` (d20) (-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. Ajoutes-y les spectacles des Grelots (`!spectacle10` … `!spectacle300`) : le texte prêt à coller est en 6.9.

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\patagrain
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 (le bouffon sort aussi tout de suite en scène Jeu)
`?apercu=1`	affiche les zones de la cam et du jeu
`?minutes=0.2`	*(Starting soon)* compte à rebours de 12 s, pour voir le lancer de d20
`?mode=complet`	*(transitions)* animation entière
`?bouffon=0` · `?chat=0` · `?bandeau=0`	retire le bouffon, le chat ou le bandeau de la page
`?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)
`?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)

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`
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
Le bouffon (le cacher, son rythme en scène Jeu, sa bulle de fin)	`reglages.html` › Le bouffon
Les visuels de la chaîne	`reglages.html` › Kit de chaîne Twitch (puis OBS › Scripts › **Refaire les images du kit Twitch**)
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 (Halloween, Noël…)	`reglages.html` › Couleurs (voir ci-dessous) ; les couleurs d'origine sont au début de `css/theme.css` (palette « Royal bleu & or »)
Les polices	fichiers dans `assets/polices/` (déjà fournis : Grenze Gotisch et Nunito), déclarés au début de `css/theme.css`
Le dessin du bouffon ou de son chapeau	`outils/generer-bouffon.mjs` : `node outils/generer-bouffon.mjs` refait le bouffon, le chapeau et le logo, puis refaire les vidéos et le kit
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 (Halloween, Noël…) en un clic

1. Ouvre `reglages.html` › section **Couleurs** (le logo, le chapeau posé sur les cams et l'emblème sont des images : ils gardent leurs couleurs).
2. Clique 🎃 **Halloween** (ou 🎄 **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 `patagrain` 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)
`mettre-a-jour.cmd`	met l'overlay à jour depuis le serveur (double-clic), en gardant tes réglages ; c'est aussi ce que fait le bouton d'OBS	quand une nouvelle version est annoncée	
`generer-transitions.mjs`	refait les vidéos `transitions/videos/*.webm` (Stinger)	après un changement de couleurs ou du bouffon	`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`
`generer-pdf.mjs` (JDR)	refait `TUTO-JDR.pdf`	après une modification de `TUTO-JDR.md`	`node outils/generer-pdf.mjs TUTO-JDR.md`
`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)*
`generer-bouffon.mjs`	redessine le bouffon, son chapeau et le logo	seulement si on modifie le dessin (puis refaire vidéos et kit)	`node outils/generer-bouffon.mjs`

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).

« Mettre à jour l'overlay » ne marche pas
→ Le message est sous le bouton (OBS › Outils › Scripts) et dans **Journal des scripts**. « Serveur injoignable » : vérifie ta connexion, ou l'adresse dans `reglages.html` › **Mises à jour**. « Certains fichiers n'ont pas pu être remplacés » : OBS les utilise (souvent les vidéos de transition) → ferme OBS, double-clique sur `mettre-a-jour.cmd`. Si le dossier de l'overlay a des accents dans son chemin et que le bouton ne fait rien : utilise `mettre-a-jour.cmd`. Rien n'est perdu : l'ancienne version est dans `sauvegardes`.

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).

Une ancienne image s'affiche encore (ancien chapeau, ancienne couleur)
→ 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**.

Rien ne s'affiche / page blanche
→ Vérifie la taille de la source (celle du canevas), puis clic droit › **Actualiser**. Si tu as modifié `config.js` à la main, cherche un guillemet ou une virgule manquant.

`reglages.html` n'enregistre pas / je ne vois pas mes changements
→ Utilise **Edge** ou **Chrome**. La première fois, il faut bien choisir le fichier `config.js` du dossier `patagrain` (la page refuse celui d'un autre overlay), puis accepter qu'elle le modifie. Avec Firefox, le nouveau `config.js` est téléchargé : remplace l'ancien par celui-ci. Puis **Actualiser** les sources dans OBS.

L'overlay est décalé ou trop petit
→ La source doit faire **exactement** la taille du canevas. Clic droit › *Transformer* › *Réinitialiser la transformation*.

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.

Un spectateur a payé un spectacle (`!spectacle…`) mais rien ne se passe
→ Le bot a-t-il répondu dans le chat ? Sinon, le spectateur n'avait pas assez de grelots, ou la commande n'existe pas dans StreamElements (6.9).
→ S'il a répondu : sa phrase doit être **exactement** celle de `reglages.html` › **Les Grelots** (même texte,

mêmes émojis, `${user}` à la place du pseudo). Corrige l'une ou l'autre.
→ La source **Alertes** doit être dans la scène affichée, et l'identifiant Twitch rempli dans `reglages.html`. Essaie `!essai tarte` dans ton chat : si ça marche, c'est la phrase du bot qui diffère.

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

Le chat réaffiche de vieux messages
→ C'est la mémoire du chat (10 minutes). Règle-la (ou mets `0`) dans `reglages.html` › **Le chat**.

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.

On n'entend pas les grelots
→ Coche **Contrôler l'audio via OBS** sur la source d'alertes, puis vérifie son volume dans le mélangeur. Et vérifie « Tintement des grelots » dans `reglages.html` › Alertes.

Une alerte s'affiche mal (pseudo manquant, « ? »…)
→ Journal (6.3, `?journal=1`) : chaque événement reçu y est détaillé, avec ses données brutes. Fais-en une capture d'écran pour faire corriger l'overlay.

Le bouffon gêne sur une scène
→ Ajoute `?bouffon=0` à l'adresse de cette scène (geste C), ou décoche-le dans `reglages.html` › **Le bouffon**.

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 avec les chiffres de ton canevas, et vérifie que la webcam est bien **sous** l'overlay.

Le compte à rebours ne repart pas de zéro
→ Coche **Actualiser le navigateur quand la scène devient active** sur la source.

Bon live, et que les dés te soient favorables ! 🎲