

# MuScaT

## Manuel d'utilisation

### Introduction (histoire)

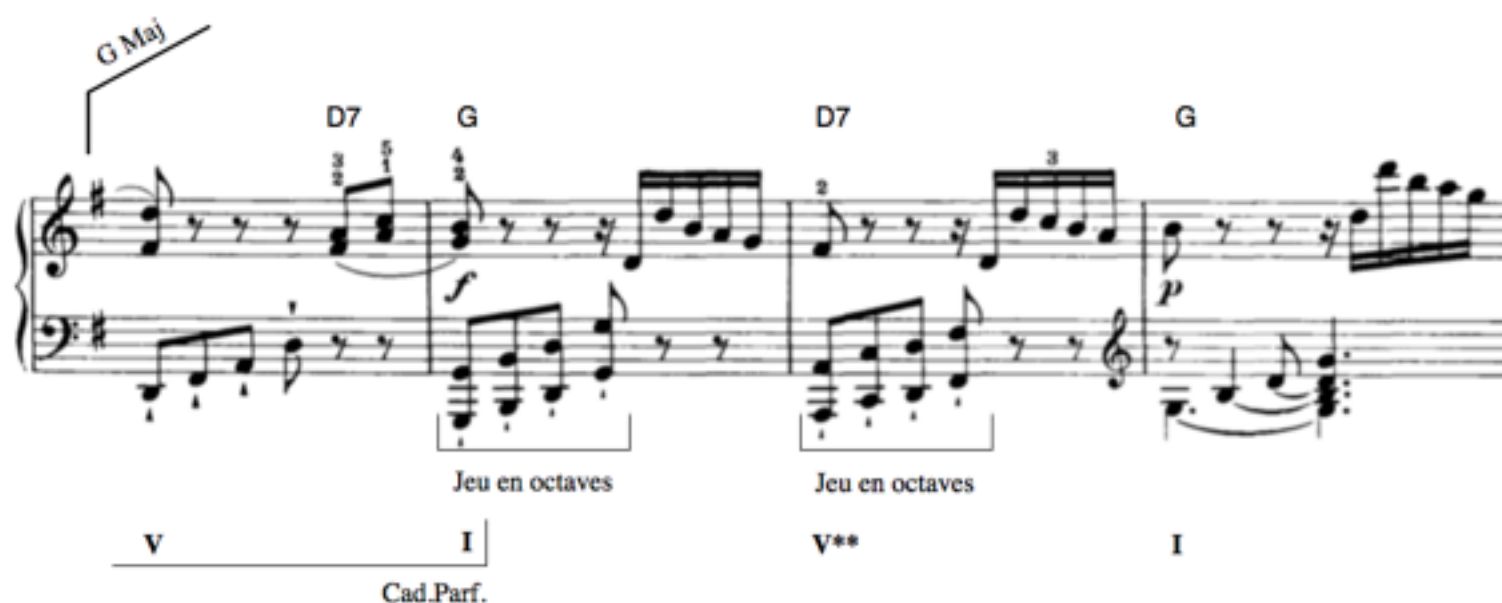
Suite à la diffusion de mon [Initiation à l'analyse musicale](#) — et son « succès » que je n'aurais jamais imaginé aussi grand — nombreux ont été les professeurs et autres pédagogues ou musicologues à me solliciter pour me demander le logiciel utilisé pour créer l'animation de cette initiation.

C'est malheureusement une application personnelle un peu trop... personnelle (comprendre : indomptable pour qui ne l'a pas créée), une usine à gaz ne fonctionnant qu'à la ligne de code (son créateur ne sait même pas toujours par quel bout la prendre).

Mais pour répondre à ces marques d'intérêts ainsi qu'à mes propres besoins, j'ai voulu concevoir un outil plus simple et plus pratique qui permettrait de réaliser rapidement des analyses de partitions de qualité.

C'est ainsi qu'est née l'idée de **MuScaT** — dont le nom se compose de « Mu » pour « Musique », « Sc » pour « Score » (« partition » en anglais) et « Ta » à l'envers pour *TAG*, le sens en français, comme les *TAGs* qu'on *tague* sur les murs.

En bref, **MuScaT** permet de **réaliser rapidement, de façon propre et pratique, des analyses de partitions musicales** de qualité, comme on peut le voir dans l'extrait ci-dessous.



Elle est semi-graphique, et permet d'ajuster très finement les *TAGs* — au pixel près — de façon visuelle et agréable.

- ⌚ Synopsis détaillé
  - ⌚ Charger de l'application **MuScaT**
  - ⌚ Créer du dossier de l'analyse
  - ⌚ Mettre l'analyse en analyse courante
  - ⌚ Découper la partition en « images-systèmes»
  - ⌚ Inscrire les images-systèmes dans l'analyse
  - ⌚ Préparer l'impression
    - ⌚ Ajout du titre, compositeur, etc.
  - ⌚ Créer les TAGs (accords, les chiffrages, les cadences et autres éléments d'analyse)
  - ⌚ Positionner les éléments graphiques
    - ⌚ Les lignes repères
    - ⌚ Note sur les coordonnées et dimensions
  - ⌚ Récupérer le code final
  - ⌚ Versions de l'analyse
  - ⌚ Imprimer en PDF
- ⌚ L'interface
  - ⌚ La Table d'analyse
  - ⌚ La boîte à outils
  - ⌚ Le champ de code
- ⌚ Composition détaillé d'un TAG
  - ⌚ Note sur le contenu du TAG (texte)
  - ⌚ Note sur les couleurs
- ⌚ Liste complète de tous les TAGs
  - ⌚ Les Images
    - ⌚ Définition de la taille d'une image
    - ⌚ Séquence d'images
  - ⌚ Les Accords
  - ⌚ Les Chiffrages (Harmonie)
  - ⌚ Les Cadences
  - ⌚ Les Modulations
  - ⌚ Autres types de textes
    - ⌚ Les Parties
    - ⌚ Les Mesures
    - ⌚ Les Degrés
    - ⌚ Les marques musicales diverses
  - ⌚ Autres éléments graphiques
    - ⌚ Les Lignes
    - ⌚ Les Boîtes
- ⌚ Opérations sur les TAGs
  - ⌚ Verrouillage des TAGs
  - ⌚ Grouper et dégroupier des TAGs
  - ⌚ Ligne de code du TAG
- ⌚ Animation d'une analyse
- ⌚ Les Options
- ⌚ Les Utilitaires

- ⌚ Changement du dossier des captures écran (Mac)
- ⌚ Renommage des fichiers images (Mac/Unix)
- ⌚ Création d'une nouvelle analyse (Mac)
- ⌚ Activation d'une analyse (Mac)
- ⌚ Pour aller plus loin
- ⌚ Annexe
  - ⌚ Application « Terminal »
  - ⌚ Raccourcis clavier

## Remerciements

Mes remerciements vont :

- ⌚ à Marion MICHEL pour la relecture attentive et patiente de ce manuel, et ces nombreuses corrections.

## Synopsis général de création d'une analyse

Commençons par un aperçu du processus général qui va permettre de produire une analyse musicale à l'aide de **MuScaT**. Noter que chaque item de cette liste est cliquable et permet de rejoindre la partie détaillée correspondante.

- ⌚ Chargement de l'application **MuScaT**
- ⌚ Création du dossier de l'analyse,
- ⌚ Mise de l'analyse en analyse courante,
- ⌚ Découpage de la partition en « images-systèmes »,
- ⌚ Inscription des images-systèmes sur la table d'analyse,
- ⌚ Préparation de l'impression
  - ⌚ Positionnement en fonction de l'aperçu d'impression
  - ⌚ Ajout des informations (titre, compositeur...)
- ⌚ Ajout de tous les éléments d'analyse,
- ⌚ Positionnement les éléments graphiques,
  - ⌚ Les lignes repères
- ⌚ Récupération du code final,
- ⌚ Impression en PDF.

## Synopsis détaillé

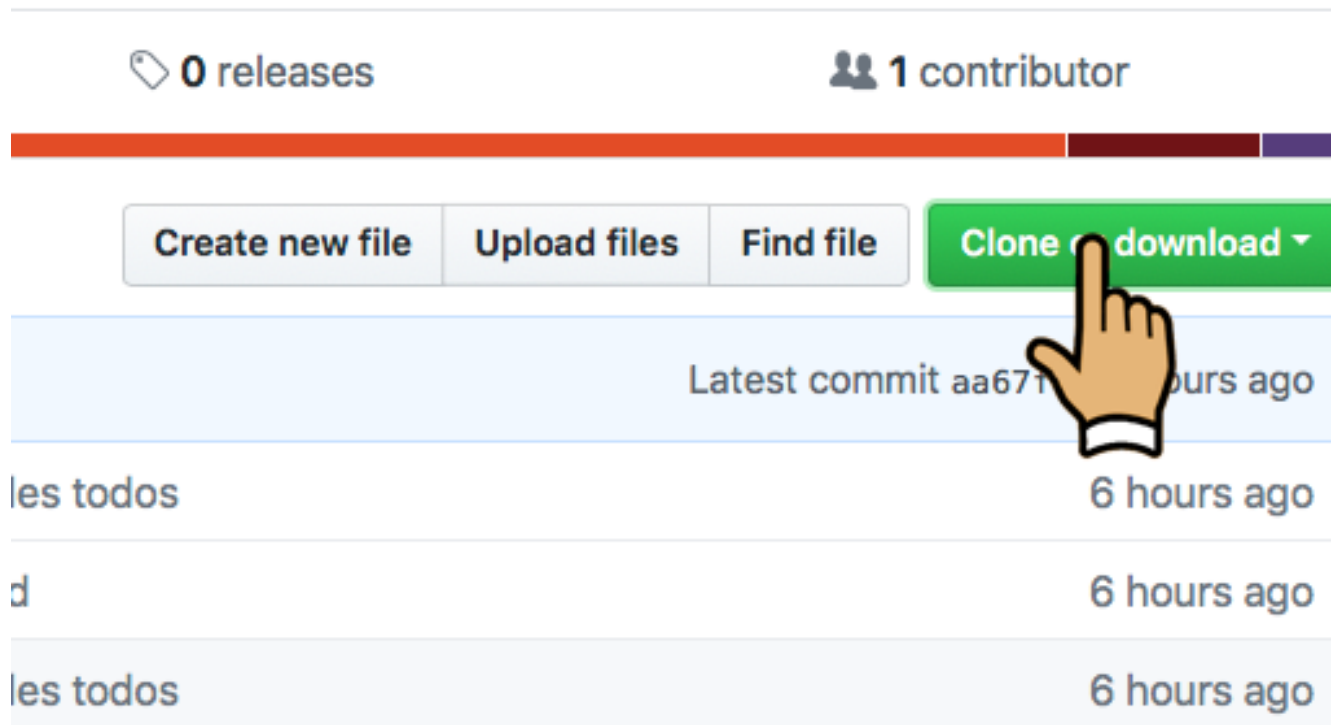
Après ce bref aperçu des étapes de la fabrication d'une analyse, abordons-en tous les aspects et tous les détails.

### Chargement de l'application MuScaT

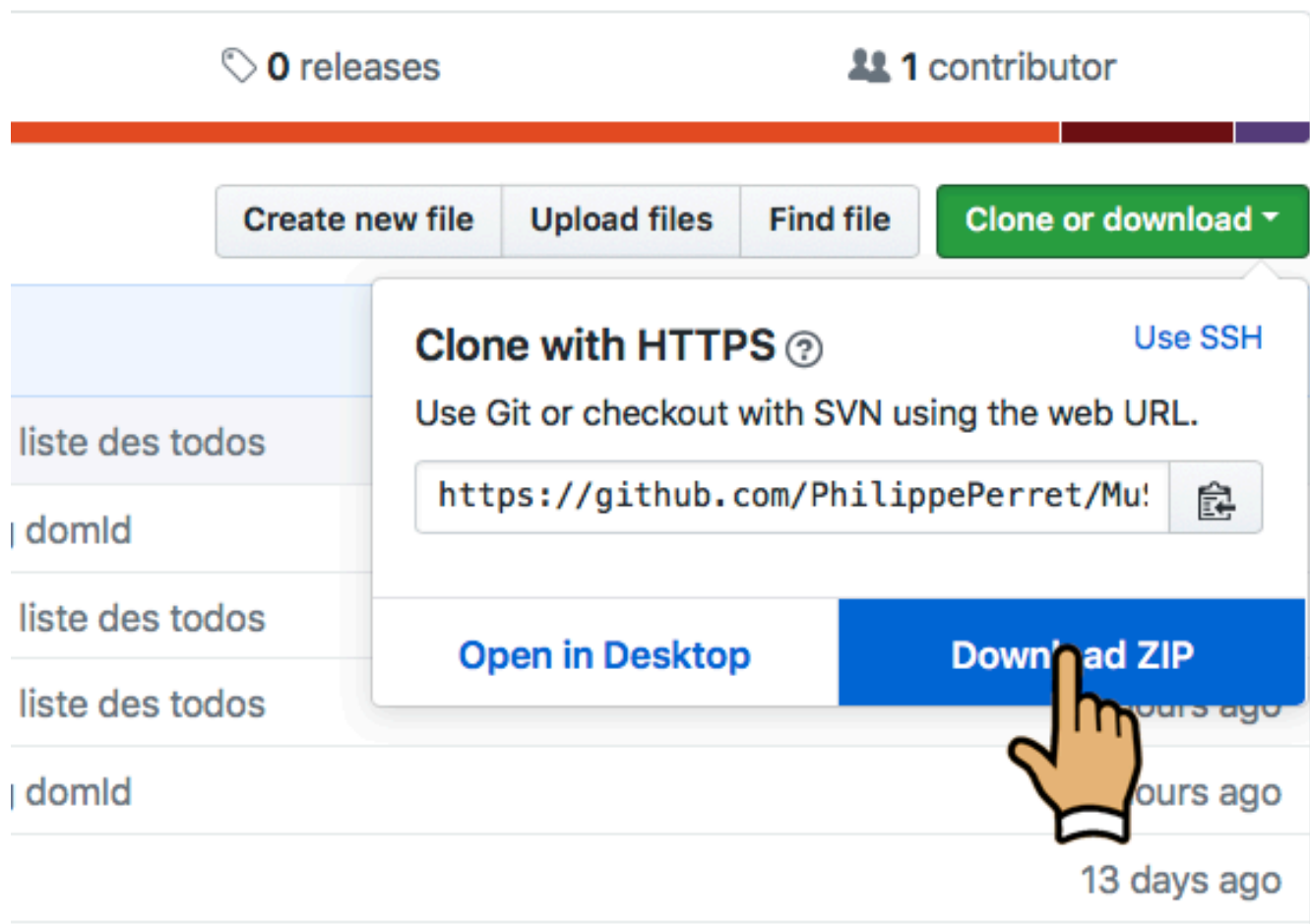
La toute première chose à faire, bien sûr, est de charger **MuScaT**. Pour le

moment, on peut le faire par le biais de son repository Github de **MuScaT**.

Il suffit de cliquer sur le bouton « Clone or download »...

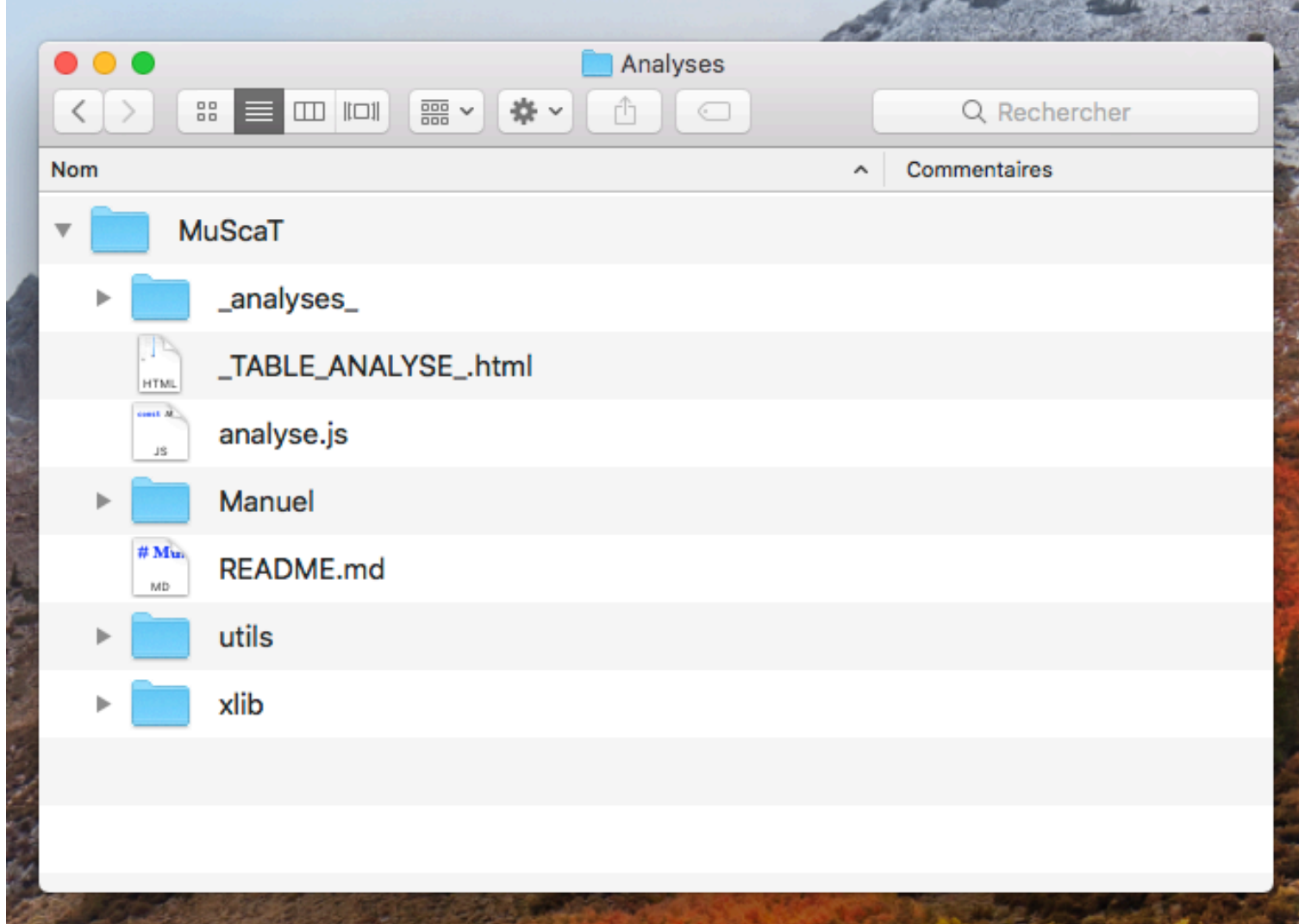


... de choisir « Download ZIP »...



... et d'attendre la fin du téléchargement (l'application fait plusieurs mégaoctets, donc suivant la qualité de votre connexion, l'opération peut être plus ou moins longue).

On se retrouve alors avec le dossier de l'application.



## Créer le dossier de l'analyse

### Création en ligne de commande

Si vous êtes à l'aise avec votre [Terminal](#) sur Mac, avec votre console sur Unix (`Application -> Accessories -> Terminal`) ou avec votre console dans Windows (`Start > Run`, puis `cmd` et `<Entrée>`), le plus simple est d'exécuter l'opération en ligne de commande.

Pour cela, vous utilisez le script `/utils/create.rb`.

Notez que comme l'extension le suggère, le langage Ruby doit être installé sur votre machine. Il l'est par défaut sur Mac, vous pouvez [l'installer facilement sur Windows](#).

```
> cd /chemin/vers/dossier/MuScaT
> ./utils/create.rb "Ma première analyse"
```

Note : les chemins ci-dessus correspondent aux OS de Mac et Linux. Il faut utiliser des balances arrières (« \ ») si vous êtes sur Windows.

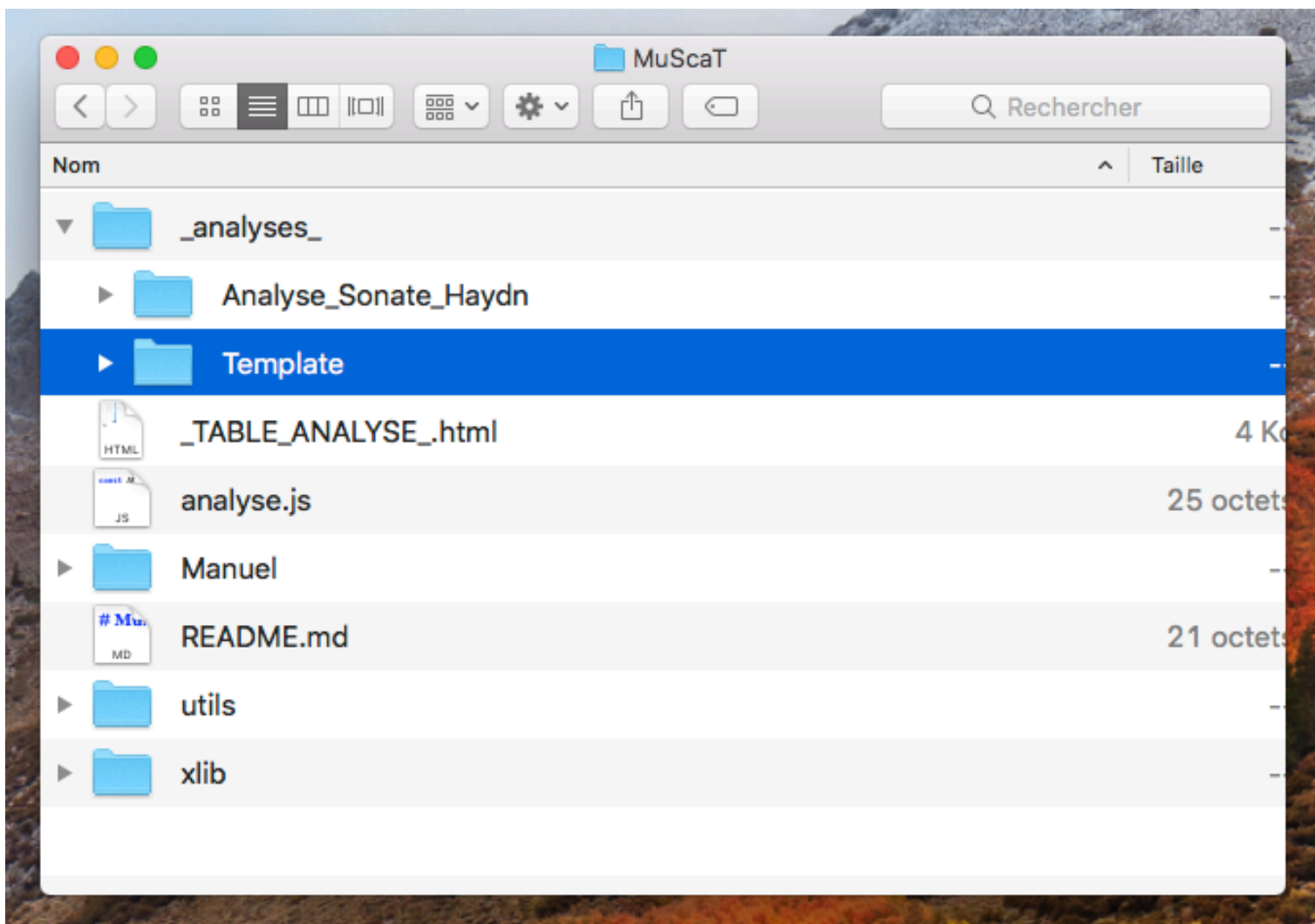
L'avantage de cette procédure en ligne de commande, c'est notamment qu'elle enregistre la version de **MuScaT** utilisée, ce qui sera très pratique pour les actualisations.

### Création par le Finder

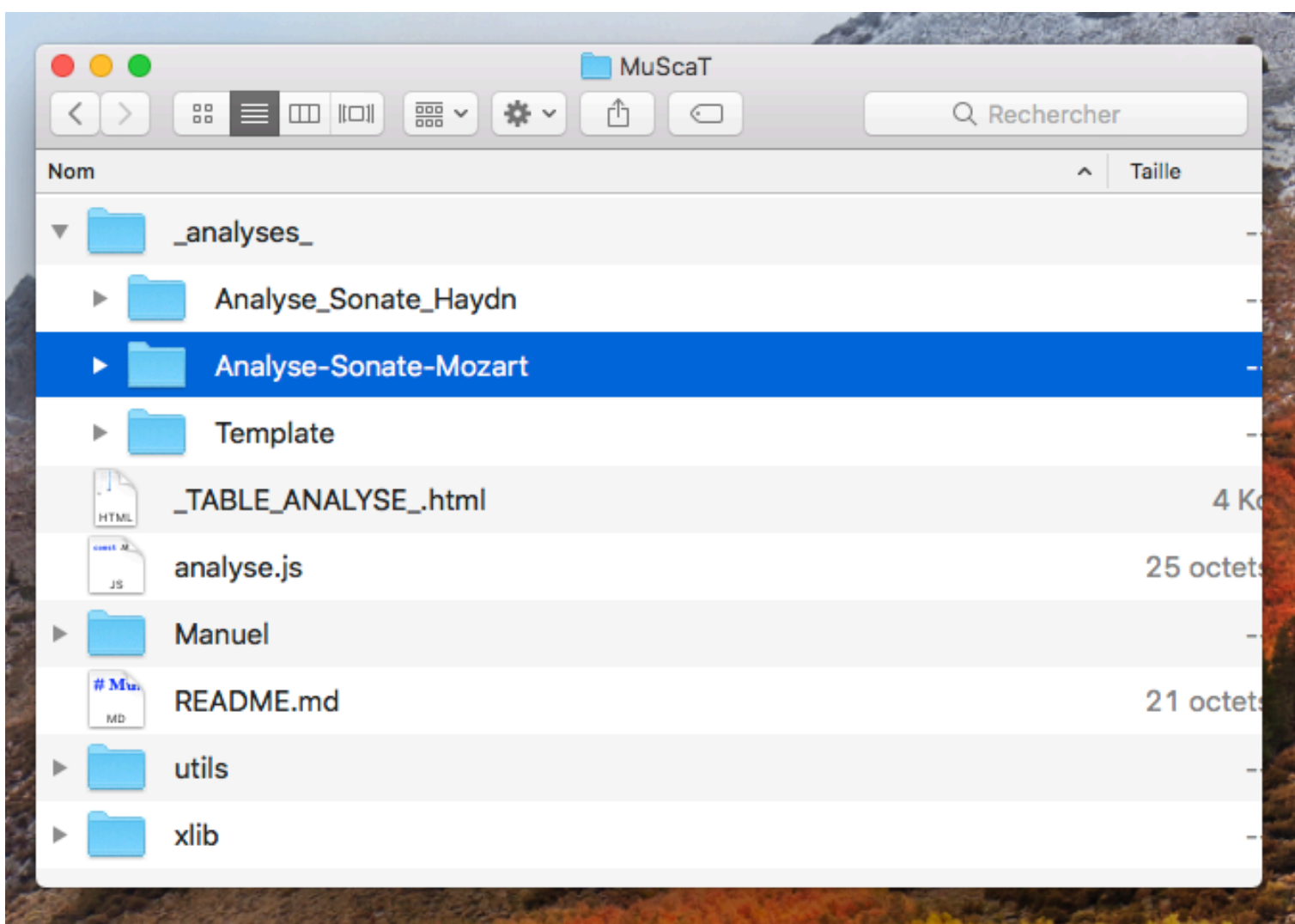


La procédure est à peine plus compliquée par le Finder, « à la main » :

- dupliquer le dossier `Template` qui se trouve dans le dossier `MuScaT/_analyses_` (ce dossier est le dossier qui peut contenir toutes les analyses),



- le renommer du nom de l'analyse, par exemple « Analyse-Sonate-Mozart ».



Note : il est vivement recommandé de ne pas mettre d'espaces vides dans les noms de dossiers ou de fichiers pour une raison qui sera expliquée plus tard. Personnellement, j'aime les remplacer par des traits plats (« Analyse\_Sonate\_Mozart »).

- 🌀 Éditer le fichier `analyse.js` (celui du dossier d'analyse que vous venez de créer) en texte simple, renseigner le nom de l'analyse (**ANALYSE**) :

```
const ANALYSE = "Analyse_Sonate_Mozart";
```

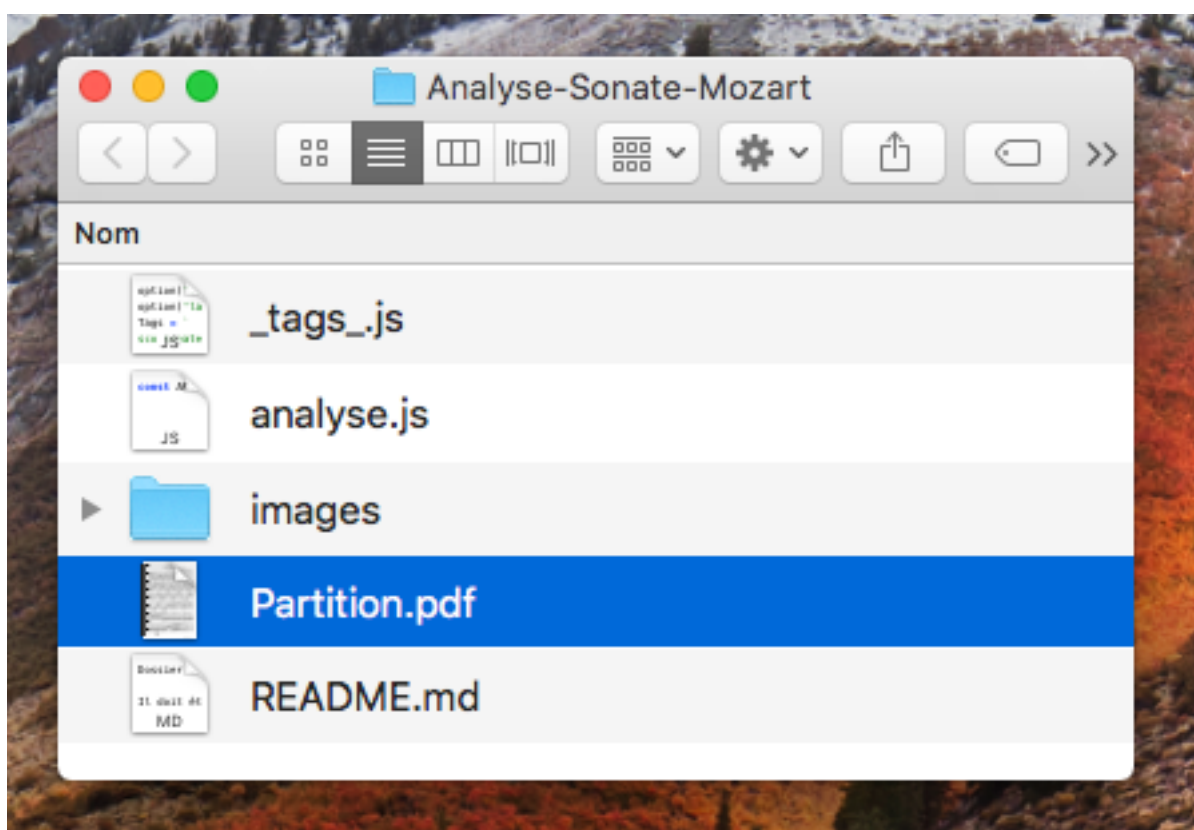
... et enregistrer ce fichier toujours en texte simple.

## Contenu d'un dossier d'analyse

Voyons rapidement le contenu du dossier d'analyse. On trouve :

- 🌀 le dossier « images » qui comme son nom l'indique va rassembler toutes les images utiles à l'analyse, c'est-à-dire les partitions, les *systemes*,
- 🌀 le fichier le plus important, le fichier `_tags.js` qui va contenir la définition précise de l'analyse,
- 🌀 un fichier `analyse.js` qu'il suffit de glisser — en le dupliquant, donc avec la touche **<ALT>** — à la racine du dossier **MuScaT** pour mettre cette analyse en analyse courante.

Dans ce dossier, vous pouvez mettre enfin votre partition en PDF ou en image.



## Mettre l'analyse en analyse courante

Cette partie utilise une procédure inhabituelle, je vous demande donc toute votre attention.

**MuScaT** possède ce que l'on appelle une *table d'analyse*, c'est en fait la fenêtre du navigateur dans laquelle nous allons procéder à l'analyse de la partition. Cette table d'analyse, on l'ouvre en lançant le fichier

MuScaT/\_TABLE\_ANALYSE\_.html .

Sur cette [table d'analyse](#) est déjà posée une analyse. Pour savoir laquelle sans ouvrir la table, il suffit de lire le fichier `MuScaT/analyse.js`. C'est donc ce fichier `MuScaT/analyse.js` qui détermine quelle analyse nous sommes en train de lire, d'imprimer ou de travailler sur la table d'analyse.

Donc, pour que notre nouvelle analyse devienne l'analyse courante, nous devons entrer son nom dans le fichier `MuScaT/analyse.js` pour qu'il devienne :

```
// dans le fichier MuScaT/analyse.js  
const ANALYSE = "Analyse_Sonate_Mozart";
```

C'est ici qu'il faut bien comprendre : nous parlons bien du fichier `analyse.js` principal, celui qui se trouve à la racine de l'application, PAS celui qui se trouve dans notre dossier d'analyse.

Pour faire de notre analyse l'analyse courante, nous avons donc la solution d'éditer le fichier `MuScaT/analyse.js` (en texte simple, comme tous les fichiers **MuScaT**) et de changer la valeur d'`ANALYSE` comme nous venons de le faire.

Voici d'autres façons de procéder :

### Avec la commande `mus`

Si vous avez installé la commande `mus`, alors il vous suffit de jouer `mus analyse Analyse_Son` dans une fenêtre de [Terminal](#).

Le nom est volontairement abrégé, **MuScaT** retrouve d'elle-même le nom de l'analyse, si c'est la seule qui commence par « Analyse\_Son ». Si elle en trouve plusieurs, elle les affiche pour vous laisser choisir la bonne.

### Avec le script `analyse.rb`

Dans le [Terminal](#) toujours, on peut rejoindre le dossier de **MuScaT** (`cd ...`) et taper `./utils/analyse.rb "Analyse_Son"`.

### En dupliquant le fichier `analyse.js` de l'analyse

Vous pouvez aussi détruire le fichier `MuScaT/analyse.js` principal et le remplacer par celui qui se trouve dans le dossier de votre analyse (`MuScaT/_analyses_/Analyse_Sonate_Mozart/analyse.js`).

Attention, cette procédure ne fonctionne que si vous avez [préparé correctement votre dossier d'analyse](#).



À vous de choisir la solution qui vous semble la plus pratique.

## Découper la partition en « images-systèmes »

Très souvent, on part d'un fichier PDF contenant une partition où les systèmes sont trop rapprochés pour être « tagués » de façon lisible. Même s'il est tout à fait possible d'utiliser un tel fichier PDF avec **MuScaT**, il est infiniment plus pratique de travailler avec de « vraies » images et des systèmes séparés (donc une image par système).

La première opération consiste donc à transformer le fichier PDF en images-systèmes. Pour ce faire, vous pouvez passer par [Gimp](#), Photoshop ou tout autre logiciel de traitement de l'image. Je vous renvoie à leur manuel pour la procédure à adopter.

Mais vous avez plus simple, beaucoup plus pratique et extrêmement plus rapide : les fonctions de capture d'écran. Si vous êtes sur Mac, vous pouvez utiliser l'application [Aperçu](#). Si vous êtes sur PC/Windows, ce sera plutôt [LightShot](#) par exemple.

Pour une version détaillée et illustrée de la procédure, je vous renvoie à [ma chaîne YouTube](#) et plus précisément le [Découpage de la partition](#). Je l'explique rapidement seulement ici.

- 🌀 Sur Mac, vous pouvez modifier le dossier de capture (le dossier où seront automatiquement enregistrées les captures d'écran) en utilisant l'utilitaire `change_folder_captures.rb` de **MuScaT**. Dans le [Terminal](#), taper :

```
> cd /chemin/vers/application/MuScaT
> ./utils/change_folder_captures.rb
/dossier/des/captures/
```

Note : pour ne pas avoir à remplir les chemins à la main, il vous suffit de glisser les éléments (fichier ou dossier) depuis le Finder jusque sur la fenêtre de Terminal. Le chemin de l'élément est aussitôt inscrit ! Donc, ici, par exemple, pour la première ligne, taper seulement **cd** (sans oublier l'espace de fin) puis glisser le dossier **MuScaT** sur la fenêtre de Terminal. Taper ensuite **./utils/chan[TAB]** (sans oublier l'espace de fin) puis faire glisser le dossier où mettre les images sur la fenêtre de Terminal.

- 🌀 ouvrir le PDF dans Aperçu,
- 🌀 activer la combinaison de touches **<CMD MAJ 4>**,
- 🌀 sélectionner le système (ou la portion de partition à isoler),
- 🌀 recommencer ces opérations pour chaque système (ou chaque portion

de partition).

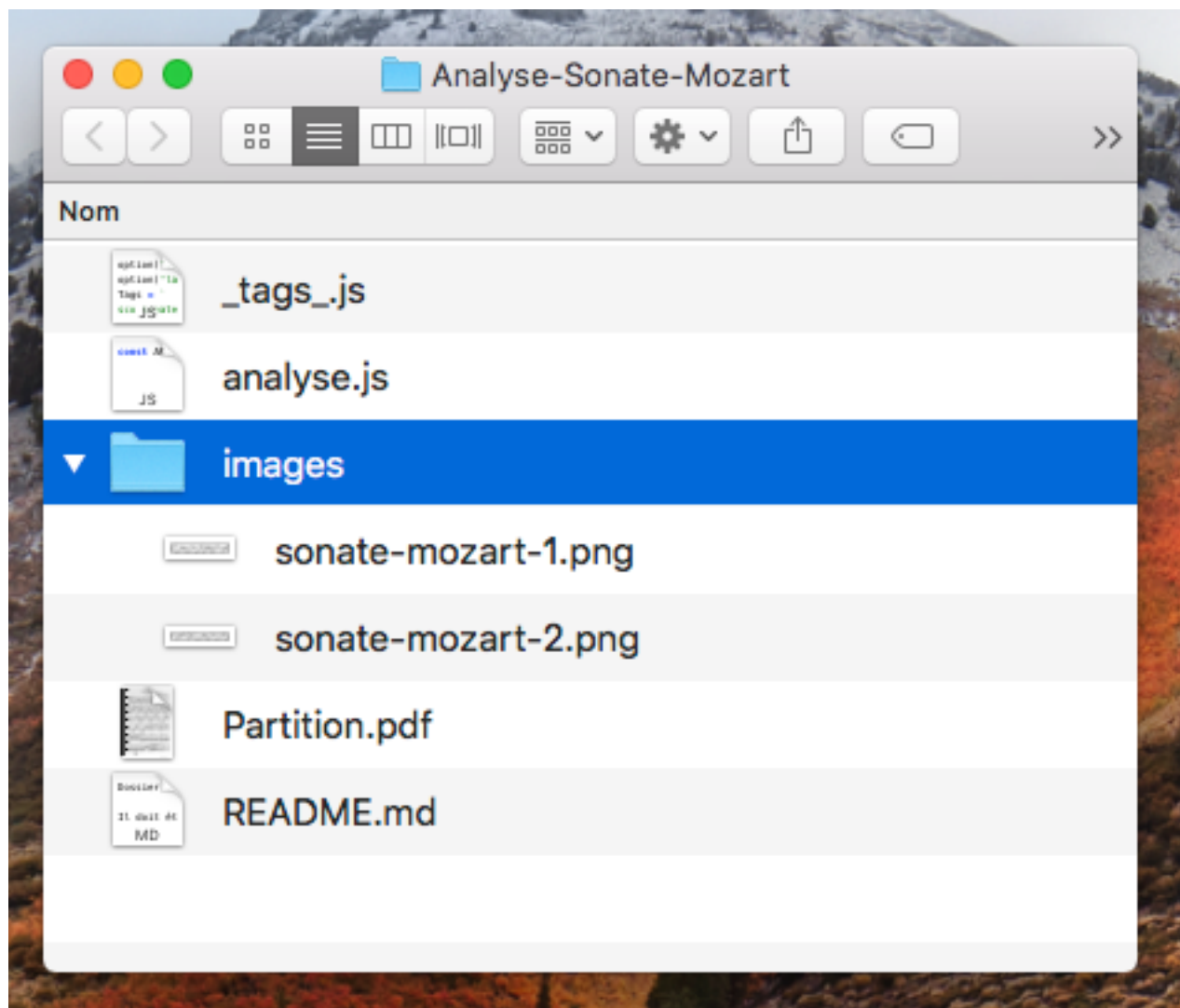
Pour renommer automatiquement toutes les captures produites — qui possèdent pour le moment des noms quelque peu ésotériques... — vous pouvez utiliser l'utilitaire `/utils/rename_images.rb`. Je vous renvoie à son mode d'utilisation qu'on peut obtenir, par la console (ou le Terminal), grâce à :

```
./utils/rename_images.rb -h
```

Noter que si vous avez mal découpé certains systèmes, il est extrêmement simple de les affiner dans un second temps :

- 🕒 ouvrez cette fois l'image dans Aperçu (pas le PDF de la partition, mais bien l'image à retoucher),
- 🕒 dessiner un rectangle à la souris,
- 🕒 régler les « poignées » de la sélection dessinée pour obtenir exactement la découpe voulue,
- 🕒 jouer la combinaison `<CMD K>`,
- 🕒 enregistrer l'image.

Quelle que soit la méthode adoptée pour découper la partition en systèmes, on place obligatoirement toutes les images dans le dossier `MuScaT/_analyses_/Analyse_Sonate_Mozart/images/`.



## Inscrire les images-systèmes

Maintenant que nos images sont prêtes, nous devons les « inscrire » dans l'analyse pour qu'elles apparaissent sur notre table d'analyse.

Pour ce faire, on ouvre le fichier `_tags_.js` de notre analyse (donc celui

qui se trouve exactement au chemin

`MuScaT/_analyses_/Analyse_Sonate_Mozart/_tags_.js` ). Ce fichier est **le fichier principal de l'analyse**, celui qui va définir tous les éléments, les images, les marques de modulations, les accords, les cadences, les parties, tout ce qui constitue l'analyse et que nous appelons les *TAGs* dans **MuScaT**.

```
2
3  Tags = `
4  // Définir ci-dessous les tags
5  // -----
6
7  // Premier système
8  // -----
9
10 sco sonate_haydn-0.png x=50 y=100 w=16cm
11
12 // Deuxième système
13 // -----
14
15 sco sonate_haydn-1.png x=50 y=140 w=16cm
16
17 // Troisième système
18 // -----
19
20 sco sonate_haydn-2.png x=50 y=1870 w=16cm
21
22 `;
23
```

**Attention ici !**

Quel que soit l'éditeur ou le traitement de texte que vous utilisez pour ouvrir ce fichier, LibreOffice, Word, TextEdit, TextWrangler, Atom, TextMate, etc., il est impératif de conserver son format « Texte simple » (« Simple Text ») lorsque vous l'enregistrez, tout en conservant précieusement l'extension `.js`. Ne l'enregistrez surtout pas en `.odt` ou autre `.docx`.

Dans ce fichier `_tags_.js` On définit d'abord les images de la partition, en ajoutant des commentaires pour pouvoir s'y retrouver plus tard, lorsque le fichier deviendra volumineux.

Le contenu d'un fichier `_tags_.js`, au départ, peut ressembler — parfois de loin — à l'illustration ci-dessous. L'important est de trouver le code `option(...)` et la définition de `Tags`.

```
// Dans _tags_.js
option('code');

Tags = `
// Premier système, les mesures de 1 à 10
partition system-1-mes-1-10.png

// Deuxième système, les mesures de 11 à 16
partition system-2-mes-11-16.png

// Troisième système
// ... etc.

`;
```

Note : l'option 'code', en haut du fichier `_tags_.js`, permet simplement de voir le code à côté de la table d'analyse.

## Préparer l'impression

### Dimensionner et positionner en fonction de l'aperçu d'impression

Avant de placer quelconque marque d'analyse sur la partition, nous vous invitons vivement à regarder ce que l'agencement des systèmes produit au niveau de l'impression ou de la sortie du PDF. Lorsque tous les TAGs seront placés, il sera extrêmement difficile et pénible de devoir les repositionner pour qu'ils soient correctement placés sur la page ou le PDF.

N'hésitez pas à [ajouter les titre, compositeur, etc.](#) avant de procéder à l'opération ci-dessus.

Pour ce faire :

- 🌀 ouvrir la table d'analyse dans Chrome (le fichier `MuScaT/_TABLE_ANALYSE_.html`),
- 🌀 disposer les systèmes en les déplaçant à la souris,
- 🌀 demander l'impression (**<CMD/Ctrl P>**),
- 🌀 faire les réglages nécessaires (taille du papier, marges, etc. — sur Chrome, l'idéal est d'utiliser le format « Portrait », des marges « minimum », un grossissement de 100 % et pas d'entête ni de pied de page),
- 🌀 noter les systèmes qui « débordent », qui passent d'une page à l'autre,
- 🌀 fermer l'aperçu et ajuster la position des systèmes,
- 🌀 recommencer ces dernières opérations jusqu'à un résultat satisfaisant.

## Ajout du titre, compositeur, etc.

Vous pouvez placer les titre, compositeur, date, opus, etc. aux endroits voulus grâce aux TAGs `titre`, `compositeur`, `analyste`, `date_composition`, `opus`, `date_analyse`, etc.

C'est-à-dire que vous pouvez placer, en haut de votre définition de `Tags` dans votre fichier `_tags_.js`, les informations suivantes :

```
// Dans _tags_.js
option('code', 'guides');

Tags=`
titre Sonate_n°34_en_Mi_mineur
compositeur Joseph_Haydn
opus Hob_XVI_34
analyste Philippe_Perret
date_analyse 27_janvier_2019
...
`;
```

Notez que pour ces TAGs il est inutile de préciser les positions. C'est le thème qui s'en charge (le thème par défaut est le thème « Muscat », évidemment). Mais vous pouvez tout à fait les déplacer pour les ajuster à votre guise ou [choisir un autre thème](#).

## Créer les TAGs (accords, les chiffrages, les cadences et autres éléments d'analyse)

L'élément graphique de base de l'application **MuScaT** est le « TAG » (comme on en trouve sur les murs des villes). Une analyse avec **MuScaT** consiste à « taguer » une partition (remarquez que les partitions elles-mêmes, ou les images de leurs systèmes, sont elles aussi des TAGs). C'est donc tout naturellement que le fichier qui va les définir, pour une analyse donnée, s'appelle `_tags_.js`. Ce fichier se trouve en haut de votre dossier d'analyse.

Il existe plusieurs moyens de définir ces TAGs. À vous de choisir celui qui vous convient le mieux, en sachant qu'on utilise en réalité plusieurs moyens au cours de l'analyse.

### Dans le fichier `_tags_.js` lui-même

On peut tout à fait définir/créer les TAGs dans le fichier du code lui-même, dans la variable dédiée `Tags` :



```
Tags = `  
// On peut définir les tags ici.  
`;  
;
```

On pourra, afin d'obtenir les coordonnées de ces *TAGs*, aller cliquer aux endroits voulus sur la table d'analyse — dans le navigateur. Les coordonnées **x** et **y** sont automatiquement collées dans le presse-papier.

## Dans le code sur la table d'analyse

Avec l'option **code** activée (`option( 'code' )` ; en haut du fichier `_tags_.js`), le code apparaît à côté de la table d'analyse, dans le navigateur.

Plusieurs moyens permettent alors de créer un nouveau *TAG* :

- ⌚ soit : utiliser le bouton « + »,
- ⌚ soit : sélectionner une ligne de code et jouer **<CMD Entrée>**.

## En faisant des copies d'éléments existants

Un moyen très rapide — et donc très efficace — de créer de nouveaux éléments est de les copier depuis la table d'analyse.

Cela consiste à déplacer l'élément **en tenant la touche ALT pressée** puis de relâcher l'élément (la copie) à l'endroit voulu.

Une copie est alors créée, qui peut être réglée comme n'importe quel *TAG*.

## Versions ultérieures

Dans les versions ultérieures, il sera possible de créer le *TAG* de toutes pièces à partir de la [boîte à outils](#).

## Aperçu de la composition d'un TAG

Chaque *TAG* est représenté dans le code (la variable **Tags** du fichier `_tags_.js`) par une unique ligne.

Une image de système (**score**) peut être :

```
Tags = `  
  
score systeme-1.png x=50 y=3098  
  
`;  
;
```

Une modulation peut être inscrite par :

```
Tags = `

mod G_min x=150 y=539

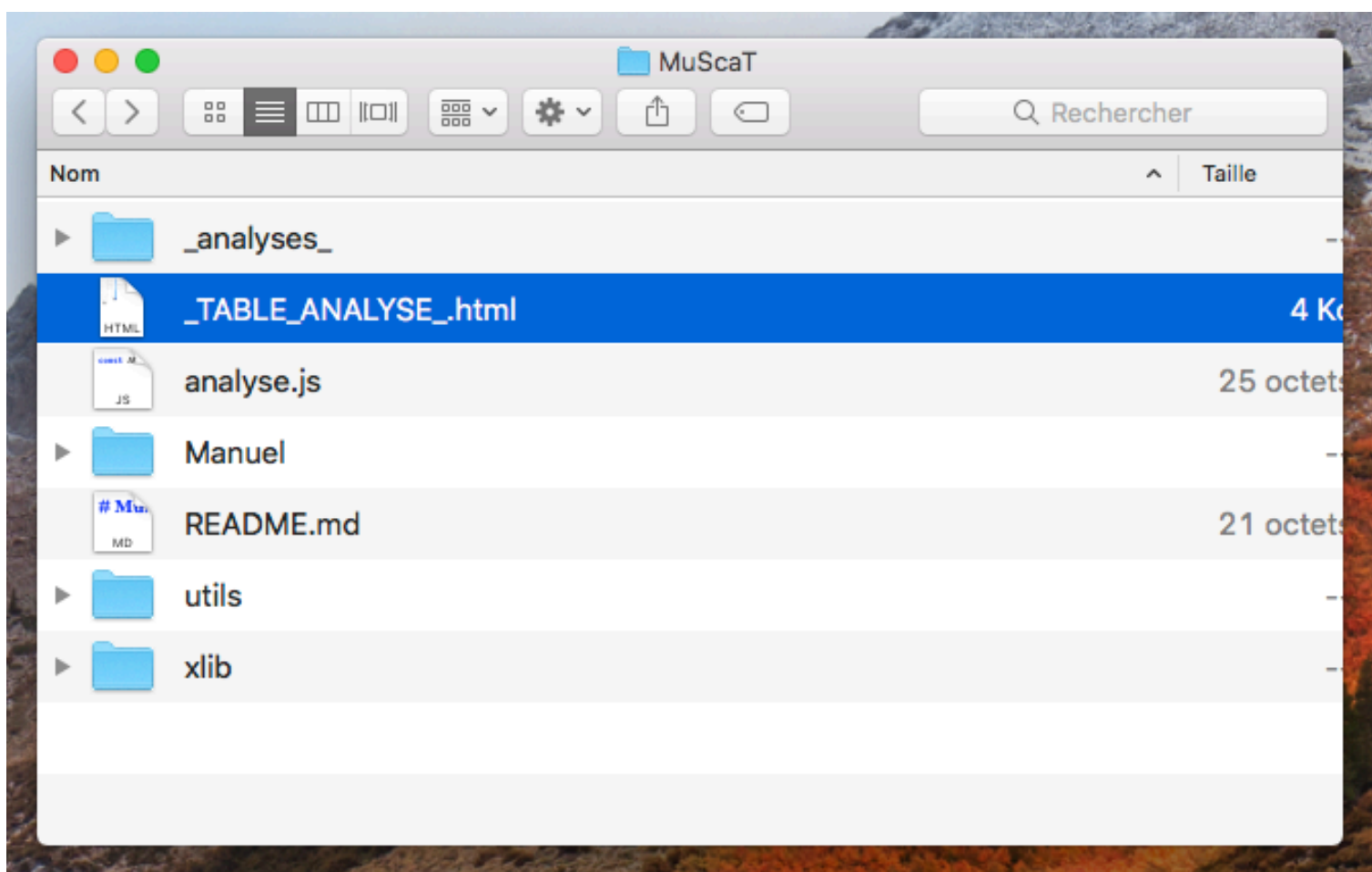
` ;
```

## Pour aller plus loin...

- 🔗 Composition détaillé d'un TAG,
- 🔗 Liste complète des TAGs.

## Positionnement et dimensionnement des éléments graphiques

Une fois l'analyse désignée comme analyse courante, on ouvre le fichier `_TABLE_ANALYSE_.html` dans Chrome (ou un autre navigateur, mais pas Firefox, qui échoue dans beaucoup d'opérations...).



On peut placer les éléments aux bons endroits de plusieurs manières :

- 🔗 en les déplaçant à la souris,



- ⌚ en utilisant les flèches de son clavier,
- ⌚ en jouant sur les touches **x** et **y**,
- ⌚ en modifiant leur coordonnées dans le champ de code.

Pour le détail, cf. [Positionnement des TAGs](#).

On peut modifier les tailles des éléments de plusieurs manières aussi :

- ⌚ en modifiant leur code,
- ⌚ en jouant sur les touches **w** et **h**.

Pour le détail, cf. [Dimensionnement des TAGs](#).

On peut en ajouter de nouveaux en dupliquant les lignes de code.

À tout moment on peut annuler une opération pour revenir en arrière en jouant **<CMD Z>** (sur Mac) ou **<Ctrl Z>** (sur Windows).

Sans l'option `option('code');` activée, il faut modifier le code directement dans le fichier `_tags_.js` puis recharger la page dans Chrome pour voir les changements.

## Lignes repères

Pour faciliter l'alignement des TAGs — par exemple l'alignement des dernières mesures de fin des systèmes — on peut utiliser des lignes repères. Pour cela, il suffit d'activer l'option **repères** (ou **reperes** ou **lines of reference**).

Cela ajoute deux lignes à l'écran, une verticale et une horizontale, qu'on peut déplacer à loisir à la souris.



Vous pouvez également définir leur emplacement exact avec les options `position repère vertical` (ou `vertical line offset`) et `position repère horizontal` (ou `horizontal line offset`) :

```
// Dans le fichier _tags_.js de l'analyse
option('code');
// à 120 pixels du haut et 200 de la gauche
option('vertical line offset', 120, 'horizontal
line offset', 200);
```

## Positionnement et dimensionnement des TAGs

Les positions **x** (horizontale) et **y** (verticale) s'indiquent toujours sans unité, en pixels :

```
x=13 y=200
```

Toutes les autres propriétés de dimension et de position peuvent s'indiquer sans ou avec unité ou pourcentage.

```
Tags = `
... w=200
... w=20%
... h=23mm
... fs=12pt
`;
```

## Obtenir des coordonnées

Pour obtenir les x/y d'une position quelconque, il suffit de cliquer à l'endroit de cette position sur la table d'analyse. Cela affiche les coordonnées en bas de l'écran, mais plus encore, ça colle un `y=134` `x=145` correspondant dans le presse-papier, valeur qu'il suffit ensuite de coller dans le code de la ligne du TAG (`<CMD V>` sur Mac ou `<Ctrl V>` avec Windows).

## Positionnement des TAGs

Pour **modifier la position d'un tag** (image, modulation, texte quelconque, etc.), on a plusieurs solutions :

- soit on la règle de façon explicite dans sa ligne de code, en définissant les valeurs de `x` (position horizontale) et/ou `y` (position verticale),
- soit on le sélectionne et on joue sur les flèches dans les quatre sens,
- soit on le sélectionne et on presse les touches `x` ou `y` pour modifier respectivement sa position horizontale et verticale. Avec la touche `<ALT>` (`<ALT x>`, `<ALT y>`), on inverse le déplacement. Avec la touche `<MAJ>` (`<MAJ x>`, `<MAJ y>`, `<ALT MAJ x>`, `<ALT MAJ y>`), on augmente le pas de déplacement, avec la touche `<CTRL>`, on peut régler la position pixel par pixel.

## Dimensionnement des TAGs

Pour **modifier les dimensions d'un tag** (comme une ligne, une cadence, une boîte, une image), on a plusieurs solutions :

- soit on les règle de façon explicite dans sa ligne de code en définissant les valeurs de `w` (largeur) et/ou `h` (hauteur),
- soit on sélectionne l'élément et on presse la touche `w` pour augmenter la largeur, `<ALT w>` pour diminuer la largeur, `h` (comme "hauteur") pour augmenter la hauteur, `<ALT h>` pour diminuer la hauteur. Tout comme pour les `x` et `y`, avec la touche `<ALT>` (`<ALT x>`, `<ALT y>`), on inverse le déplacement. Avec la touche `<MAJ>` (`<MAJ x>`, `<MAJ y>`, `<ALT MAJ x>`, `<ALT MAJ y>`), on augmente le pas de déplacement, avec la touche `<CTRL>`, on peut régler la position pixel par pixel.

## Récupérer le code final

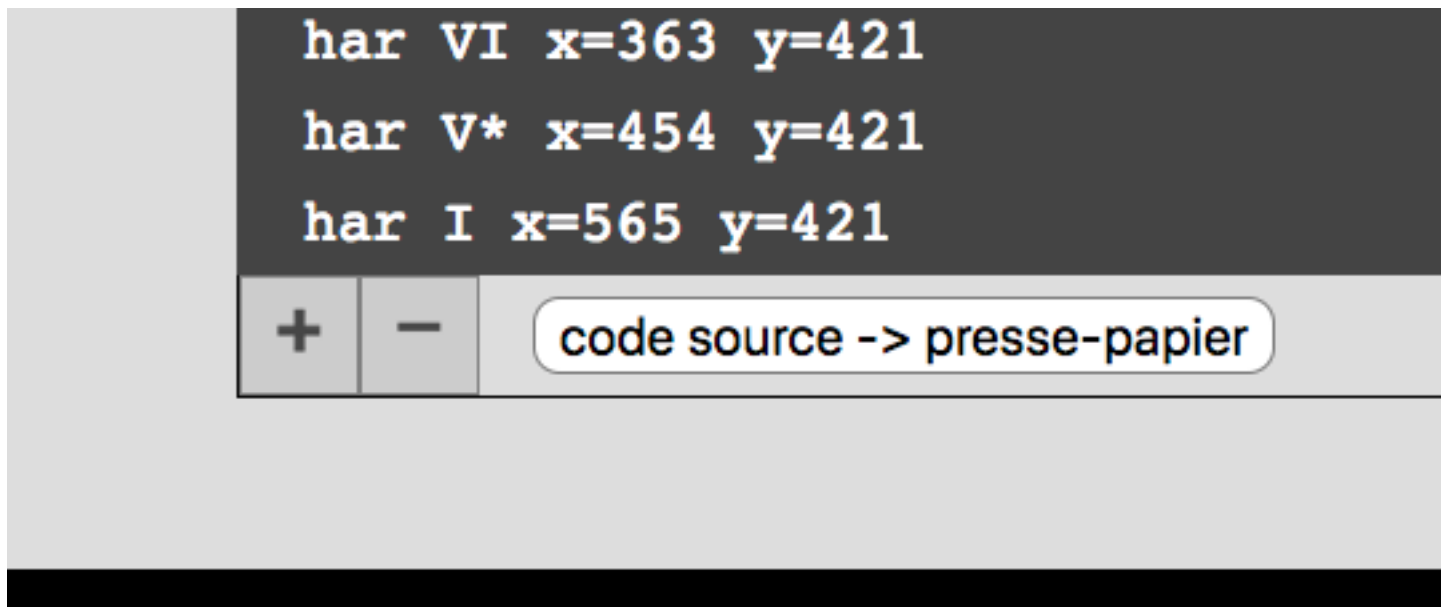
Si l'on a travaillé dans le champ de texte à côté de la table d'analyse (avec l'option `code` activée), on doit copier le code final dans le fichier `_tags_.js`, au risque de perdre tous les changements.

Noter que si vous avez préféré faire les changements directement



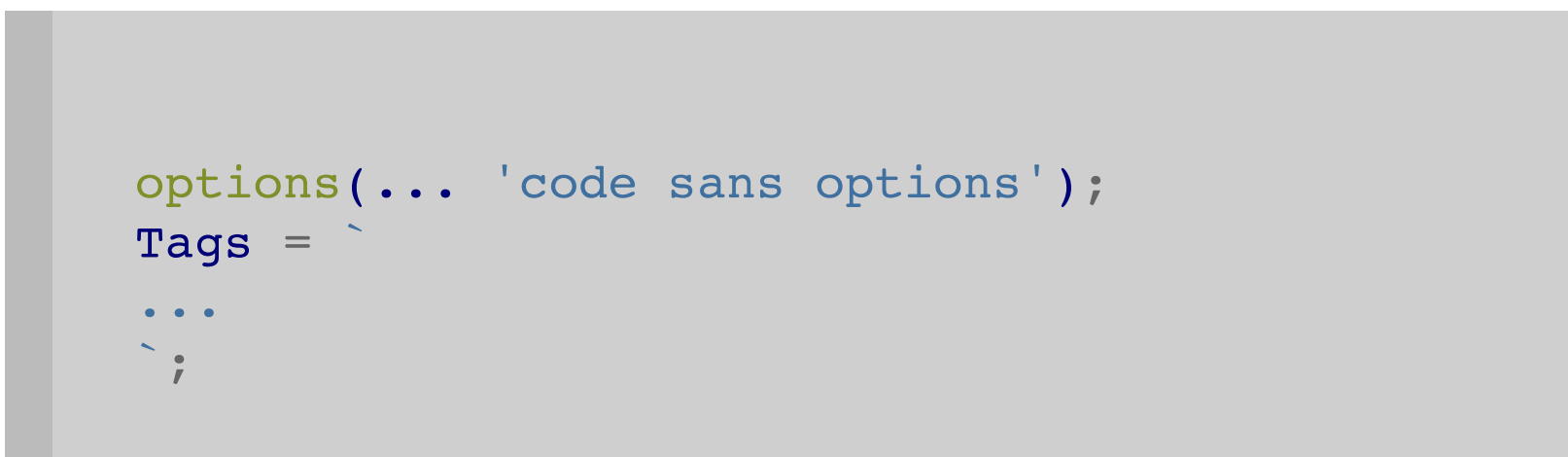
dans le fichier `_tags_.js`, il vous faut recharger la table d'analyse à chaque fois dans le navigateur pour voir les modifications.

Pour se faire, on clique sur le bouton « code source -> presse papier » qui se trouve sous le champ de code si l'option `code` est activée, sinon, dans les outils — en haut à gauche.



On colle ce code dans le fichier `_tags_.js`, en remplaçant l'intégralité de son contenu.

Par défaut, dans ce code sont ajoutées toujours les options. Si on garde plusieurs versions de l'analyse, ces divers appels aux options risquent de se parasiter. Pour palier ce problème, on peut ajouter l'option `code sans options` pour que seul le code des tags soient placés dans le presse-papier :



## Mesure d'urgence (ou de secours)

Parfois, le code de l'analyse (des TAGs) refuse de se copier dans le presse-papier. À la place, on trouve par exemple les coordonnées du dernier point cliqué sur l'analyse.

Il existe une solution de secours qui permet de ne pas perdre tout son code : utiliser le bouton « Code complet (secours) » de la boîte à outils, près du bouton d'aide.



En cliquant, on ouvre un champ de texte dans lequel se trouve tout le code, sélectionné. Il suffit de le copier (avec **<X>** pour s'assurer de bien l'avoir copié) et de le coller dans le fichier `_tags_.js`.

## Plusieurs versions de l'analyse

Noter que si vous préférez garder plusieurs versions de votre analyse (ce qui peut être prudent), il suffit de copier le code à *la suite du précédent* (c'est-à-dire à la fin du fichier `_tags_.js`) plutôt qu'en *remplacement* de l'ancien code. Votre fichier contiendra alors :

```
// Dans _tags_.js
// Version 1.1
option('code','lang','en');
var Tags = `
    //... tags de la première verison
`;
// Version 1.2
option('lang','fr','reperes');
Tags = `
    //... tags de la version suivante
`;
// Version 2.0
option('lang','fr','reperes');
Tags = `
    //... tags de la version suivante
`;
// etc.
```

Bien entendu, pensez à supprimer de vieilles versions pour alléger un peu le fichier, s'il devient conséquent.

## Imprimer l'analyse en PDF

Enfin, on imprime la page HTML du navigateur en choisissant le format PDF :

- ⌚ dans Chrome, demander l'impression (<CMD/Ctrl P>),
- ⌚ dans la fenêtre qui s'ouvre, choisir, dans le menu en bas à gauche :  
« Imprimer au format PDF » ou autre indication similaire.

## Et voilà !

Et voilà, c'est fait ! Et vous pourrez retoucher votre analyse à n'importe quel moment en la remettant en analyse courante.

---

## L'interface

Faisons un tour rapide de l'interface, qui reste volontairement relativement simple, se concentrant sur les éléments essentiels.

- ⌚ La Table d'analyse
- ⌚ La boîte à outils
- ⌚ Le champ de code

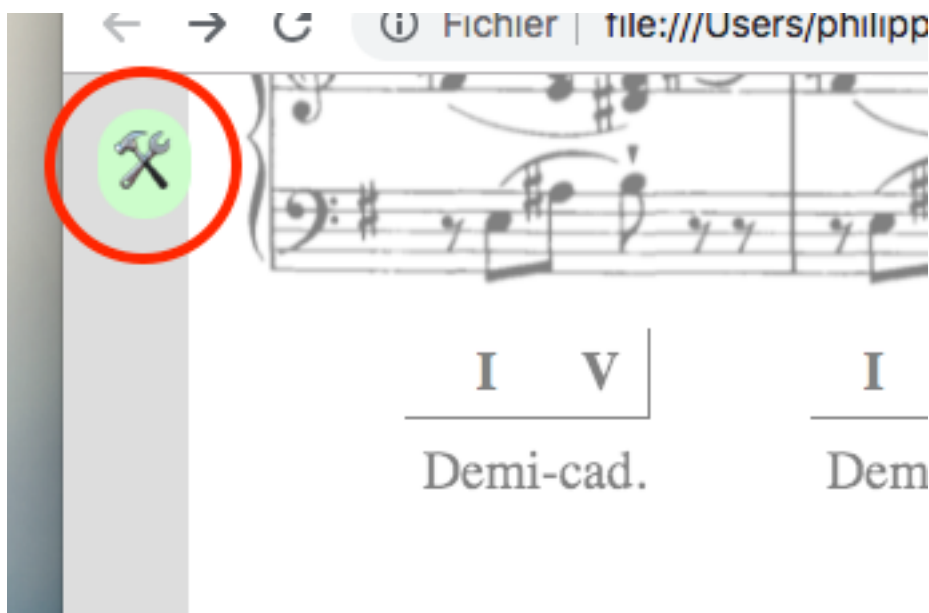
### La Table d'analyse

Cette table, c'est la surface de la page elle-même, la surface principale de la fenêtre de votre navigateur. Elle se présente en blanc sur une surface à peine grisée qui permet de repérer les dimensions de la page d'impression. Le mieux est de jouer sur la largeur des images (paramètre **w**, modifiable avec la touche **w**) pour toujours se trouver à l'intérieur de cette surface.

**Si vous dépassez la surface délimitée, l'impression en sera affectée de façon très aléatoire (au moins en apparence).**

### La boîte à outils

Sur la gauche en haut de l'écran, on trouve un petit picto qui permet d'ouvrir la boîte à outils.



Cette boîte à outils contient des outils pour regrouper ou dégroupier des

TAGs, pour les aligner, pour copier le code, etc.

## Le champ de code

Si l'option `code` est activée, un champ de code est ouvert à droite de la page, contenant le code défini dans votre fichier `_tags_.js` (seulement celui dans `Tags`, pas le code intégral).

En modifiant ce code, vous pouvez construire votre analyse (n'oubliez pas d'en copier ensuite le code intégral).

Note : la touche `<CMD>`, sur PC/Windows, doit être remplacée par la touche `<Ctrl>`.

<i>Action</i>	<i>Description</i>
<code>&lt;CMD Entrée&gt;</code> ou bouton « + »	Créer une nouvelle ligne/un nouveau TAG.
Bouton « - »	Suppression du TAG courant
<code>&lt;flèche haut&gt;</code>	Passer au TAG au-dessus
<code>&lt;flèche bas&gt;</code>	Passer au TAG en dessous
<code>&lt;CMD flèche haut&gt;</code>	Remonter le TAG courant
<code>&lt;CMD flèche bas&gt;</code>	Descendre le TAG courant
<code>&lt;Tabulation&gt;</code>	Se place dans le code du TAG sélectionné sur la table.

Notez que dès que vous sélectionnez un TAG dans le code, le TAG est aussitôt sélectionné sur la table d'analyse (la table est « scrollée » pour toujours rendre visible ce TAG). L'opération inverse est vraie : toute sélection de TAG sur la table d'analyse est aussitôt sélectionné et affiché dans le code.

---

## Composition détaillé d'un TAG

Le code de l'analyse est constitué simplement de *lignes*, les unes au-dessus des autres, qui déterminent les images et les TAGs. Chaque ligne est une image ou un TAG (exception faite des lignes vides et des lignes de commentaire).

### 1 ligne = 1 TAG

Chaque ligne de TAG est composée d'un nombre de propriétés souvent définies par le signe égal — exception faite des deux premiers « mots » qui déterminent le plus souvent la *nature* et le *contenu* du TAG.

Voyons plus en détail comment se compose une ligne du fichier

`_tags_.js`, une ligne définissant un *TAG* ou une partition.

Cette ligne a le format général suivant :

*nature*[ *contenu*][ *propriétés*][ *option/type*]

### ***nature***

C'est la *nature* du *TAG*, ce qui détermine ce qu'il est, cadence, modulation, boîte ou image, etc.

### ***contenu***

C'est le *contenu* du *TAG*, parfois son *type* (pour les lignes par exemple). Pour un *TAG* de texte, c'est le texte, pour une modulation, c'est la tonalité vers laquelle on module.

Cf. [Note sur le contenu du TAG](#)

### ***propriétés***

Les propriétés du *TAG*, à commencer par ses coordonnées **x** (position horizontale) et **y** (position verticale) ainsi que les dimensions h (hauteur) et w (largeur) du *TAG*.

Ces coordonnées et ces dimensions se notent simplement en donnant les valeurs à l'aide d'un signe égal (**=**) **sans espace** : **x=12**, **y=20**, **w=12%**, **h=12mm**, etc.

Cf. aussi [Note sur les coordonnées et dimensions](#)

On peut également trouver les propriétés de couleur. Cf. [Note sur les couleurs](#)

### ***option/type***

Des options ou des types en fonction de la nature du *TAG*. Nous y reviendrons.

Noter qu'à part les deux premiers éléments, tous les autres peuvent être donnés dans l'ordre qu'on veut, sans importance.

Voici quelques définitions de *TAGs* :

```
Tags = `

score ma_partition.jpg y=100 x=10

// => L'image 'images/ma_partition.jpg' placée à
10 pixels de
//      la marge gauche et 100 pixels du haut.
```



```
cadence I type=parfaite y=200 x=100 w=100

// => Une cadence de type parfaite, avec un
trait de
//      100 pixels de large (`w`).

modulation G_min x=200 y=100

// => Une modulation vers Sol mineur.

`;
```

Noter les lignes commençant par `//` qui permettent de laisser un commentaire. C'est très utile lorsque l'on veut s'y retrouver lorsque l'analyse devient conséquente.

Noter qu'une *nature* de TAG (le premier mot), peut toujours être exprimée par ses trois premières lettres (exception faite du terme « partition » qui rentrerait en conflit avec « partie »). Ainsi, on peut écrire le code ci-dessous :

```
Tags = `

sco ma_partition.jpg y=100 x=10

// sco => score (partition)

cad V_I type=parfaite y=200 x=100

// cad => cadence

mod G_min x=200 y=100

// mod => modulation

`;
```

L'intégralité des *natures* de TAG (et leur diminutif) [est détaillée ici](#).

## Forme raccourcie d'écriture

Pour la première définition du TAG, on peut utiliser une version raccourcie de définition qui la rend très simple et très rapide. Elle consiste à utiliser :

```
Tags = `  
<nature>[ <contenu>] <valeur y seule> <valeur x  
seule>  
`;  
;
```

Par exemple, pour une *modulation* vers la tonalité de SOL mineur (G min.) qui doit se situer à 200 pixels du haut et 450 pixels de la gauche, on pourra écrire simplement :

```
mod G_min 200 450
```

Il suffit de se souvenir que le premier nombre concerne la *hauteur*.

## Note sur le contenu du TAG (texte)

Le *contenu* du TAG, c'est-à-dire son deuxième *mot*, peut être de type très différent. Mais en règle générale, il s'agit d'un texte, d'un accord ou d'un chiffage.

Il est important de noter immédiatement ce point :

**Aucune espace ne doit se trouver  
dans ce contenu.**

On doit obligatoirement remplacer les espaces par des traits plats (MAJ tîret sur Mac).

Ainsi, si l'on veut écrire « Second couplet », on doit écrire :

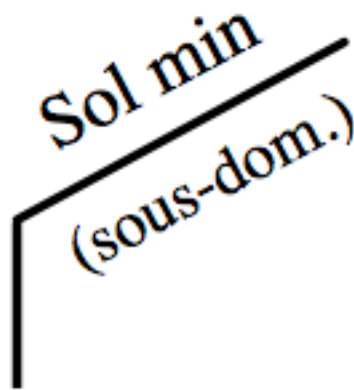
```
partie Second_couplet x=123 y=456
```

Si l'on veut écrire la tonalité « Sol mineur », on doit impérativement écrire :

```
modulation Sol_mineur x=23 y=344
```

En revanche, tous les autres caractères sont possibles, à l'exception des balances ( / ) dans les modulations, car elles indiquent le texte qui devra apparaître sous le trait biaisé :

```
// Une modulation vers la sous-dominante
```



### Note sur les couleurs

Tous les *TAGs* à part les images peuvent avoir une couleur d’écriture et une couleur de fond.

Ces couleurs sont définies par les propriétés suivantes au choix, suivant votre convenance. La première (*c* pour la *couleur* et *bgc* pour la *couleur de fond*) sont les moins difficiles pour *MuScaT* puisque ce sont celles qu’il utilise dans sa langue naturelle) :

<i>couleur</i>	<i>c</i> <i>couleur</i> <i>color</i>
<i>couleur de fond</i>	<i>bgc</i> <i>fond</i> <i>cf</i> <i>background-color</i>

Pour la valeur, on peut utiliser soit la forme littérale en anglais (*red*, *yellowgreen*, etc.) soit la forme hexadécimale (*12FF67*, *#12FF67*) soit la forme RVB (*rgb(100,23,45)*).

Dans la forme RVB/RGB, attention de ne laisser aucune espace !

### Liste complète de tous les TAGs

Trouvez ci-dessous la liste complète de tous les *TAGs*.

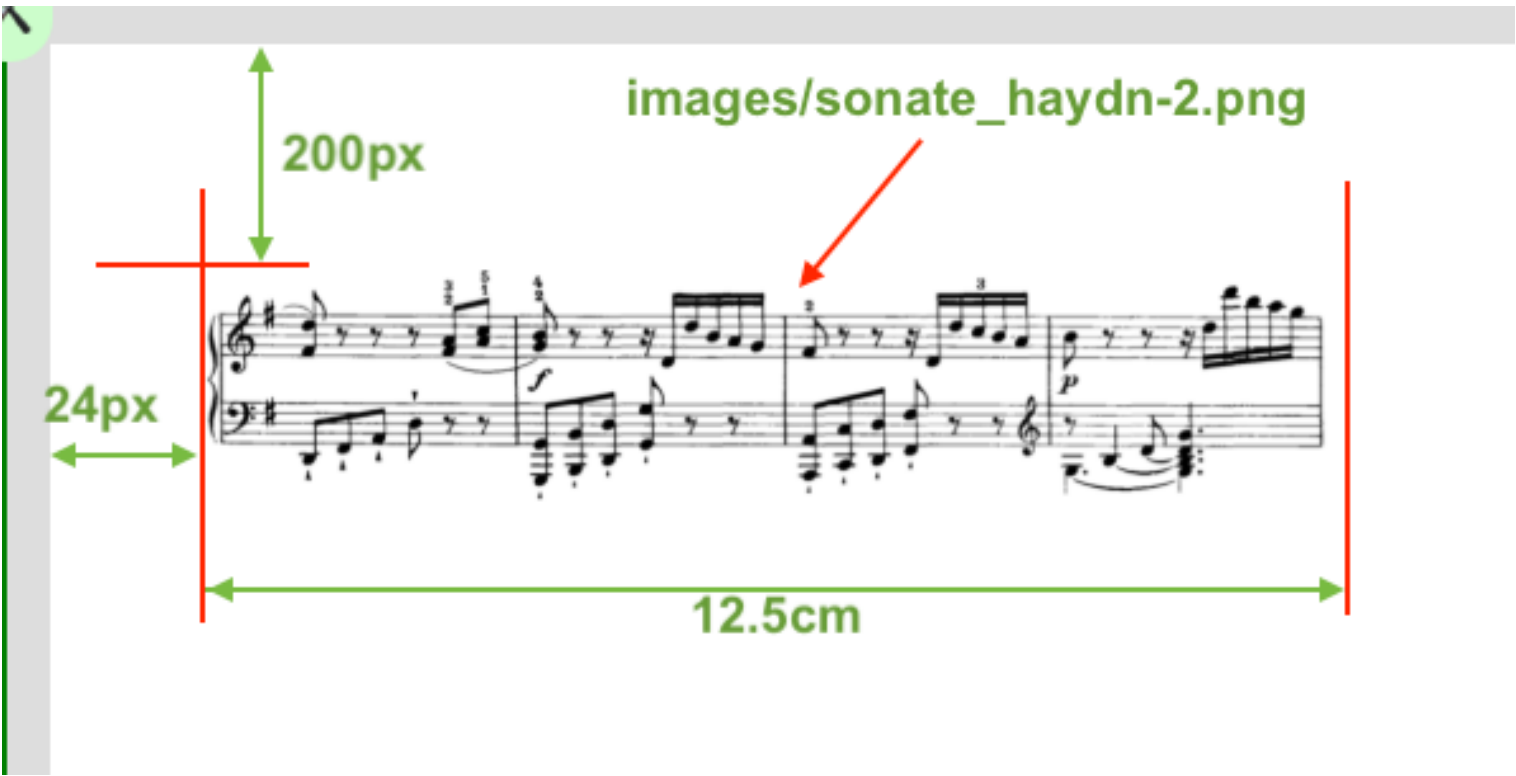
<i>TAG</i>	<i>Description</i>
partition score	Écriture d’une image, à commencer par celle de la partition ou du système à analyser.

sco  
image

usage : image <source> x=... y=... z=...

Exemple

sco sonate\_haydn-2.png x=24 y=200 w=12.5cm



Note

L'image doit se trouver dans le dossier "images" du dossier de l'analyse.

Détail

Les Images

accord  
chord  
acc

Écriture d'un accord au-dessus de la partition.

usage : accord <nom> x=... y=...

Exemple

acc E\_min x=100 y=200



Détail

Les Accords

harmonie

harmony

chiffre

chiffage

har

Écriture d'un chiffre sous la partition. C'est en général un chiffre romain indiquant le renversement de l'accord.

*usage* : har[mony] <chiffre> x=... y=...

Exemple

har VII\*\*\* x=540 y=325



Détail

Les Harmonies

modulation

mod

Marque de modulation à placer en haut de la partition.

*usage* : mod[ulation] <tonalité> x=... y=...

Exemple

mod G\_Maj/Relatif\_maj. x=83 y=77 h=70px





## Détail

### Les Modulations

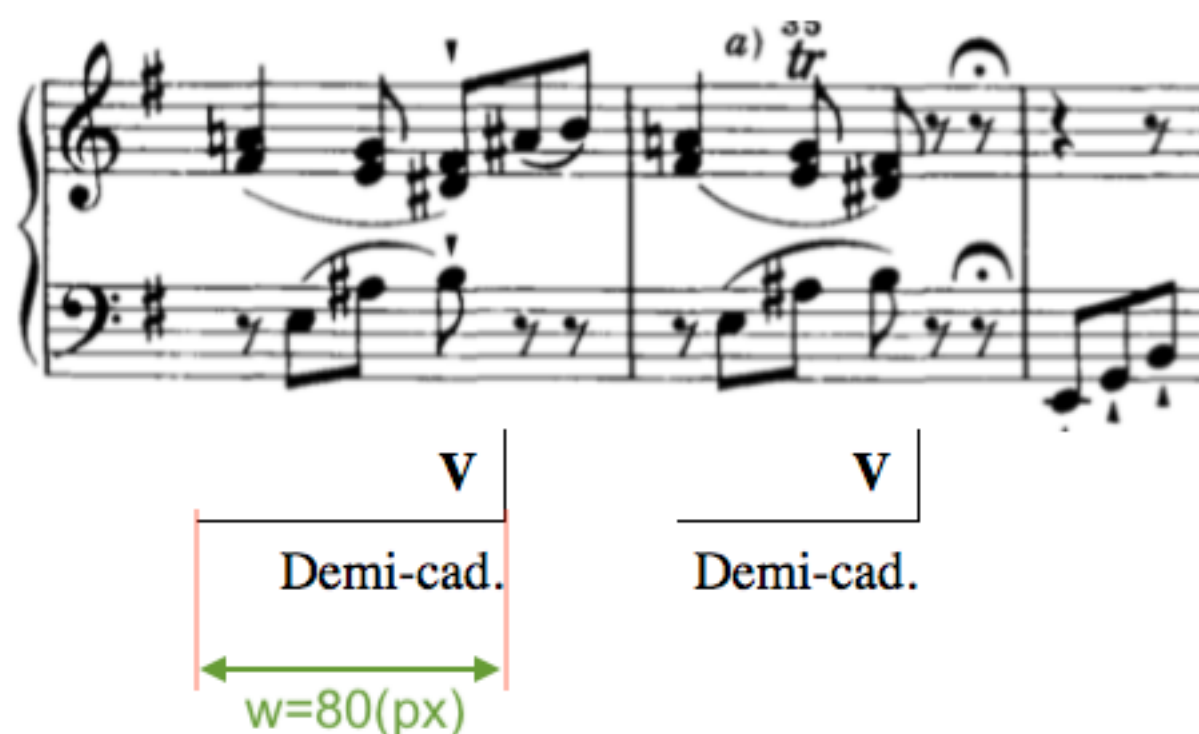
cadence  
cad

Marque de cadence à placer sous la partition.

*usage* : **cad[ence]** <acc> type=<type cad> x=. y=. w=...

## Exemple

cad V type=demi x=23 y=3456 w=80



## Note

Noter ci-dessus l'emploi 1) d'un *type* (type de la cadence) et 2) d'une largeur *w* (longueur du trait).

Détail

Les Cadences

mesure  
measure  
mes

Marque de mesure à placer à l’endroit voulu.

```
usage : mes[ure] <numéro> x=... y=...
```

Exemple

```
measure 15 x=434 y=171
```



Note

Noter que le style dépend du thème choisi.

Détail

Les Mesures

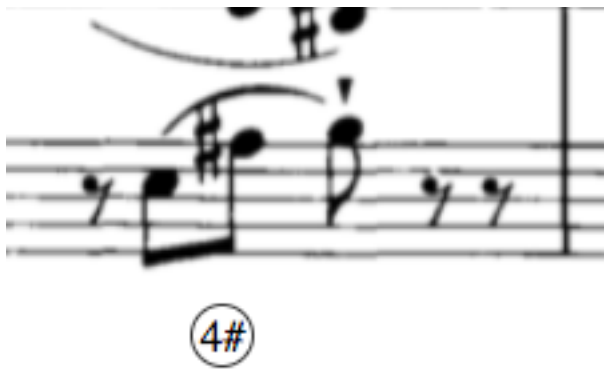
degree  
degre  
deg

Marque du degré de la note dans la gamme, à placer à l’endroit voulu.

```
usage : degre <degré> x=... y=...
```

Exemple

```
degre 4# x=23 y=1200
```



Détail

Les Degrés

ligne  
line  
lig  
lin

Marque une ligne, par exemple pour indiquer la poursuite d'un chiffrage sur plusieurs temps.

```
usage : ligne <type> x=... y=... w=...
```

Exemple

```
line | _____ | x=242 y=323 h=15 w=110
```



Détail

Les Lignes

box  
boite  
boi

Dessine une boite à l'écran. Ces boites permettent de masquer des éléments de la partition. Bien que ces boites apparaissent en gris sur la table d'analyse, elles seront invisibles dans le document PDF final ou à l'impression.

Exemple

```
boite x=45 y=152 h=180 w=295
```



Note

La hauteur (**h**) et la largeur (**w**) sont ici très importantes. Noter également que cette boite est visible sur la table d’analyse, mais elle sera invisible dans le document imprimé.

Détail

Les Boites

texte  
text  
tex

Pour marquer un texte quelconque à l’écran.

```
usage : tex[te] <le_texte> x=... y=...[ type=<type>]
```

Exemple

```
tex Opposition_marquée x=296 y=138 w=73
```

## Opposition marquée



### Notes

- ⌚ Mettre des traits plats à la place des espaces dans le code. Ils seront remplacés par de vraies espaces lors de l'inscription du texte sur la table d'analyse.
- ⌚ Grâce à la définition de la largeur (w), on met le texte sur deux lignes.

### Détail

#### Les Textes

## Les Images

Il existe trois mots clés pour indiquer la nature d'une image, mais ils produisent en réalité la même sorte de TAG : **image**, **score** ou **partition**. C'est le premier mot à trouver sur la ligne d'une image. Juste après, on doit trouver le nom de cette image, ou son chemin relatif depuis le dossier `images` du dossier de l'analyse.

```
partition premier_mouvement/image-12.png [...]
```

Ci-dessus, l'image `image-12.png` doit donc se trouver dans le dossier `_analyses_/monAnalyse/images/premier_mouvement/`.

## Définition de la taille d'une image

On peut définir la taille d'une image à l'aide du paramètre **w** (ou **width**, « largeur », en anglais). Sa valeur peut être explicite avec une unité, explicite sans unité ou en pourcentage. Par exemple :

```
// Dans _tags_.js  
Tags = `
```



```
sco image-0.png
sco image-1.png w=200
sco image-2.png w=10cm
sco image-3.png w=50%
`;
```

Avec le code ci-dessus, l'image 0 aura sa taille normale, `image-1.png` fera 200 pixels de large, `image-2.png` fera 10 centimètres de large et `image-3.png` sera mise à 50 % de sa largeur.

Pour voir en détail toutes les façons de modifier la taille ou la position d'une image, cf. [Dimensionnement des TAGs](#) et [Positionnement des TAGs](#).

## Séquence d'images

Bien souvent, une analyse n'est pas constituée d'une seule image pour toute la partition. Il y a trop peu d'espace entre les systèmes. On conseille donc fortement de découper les partitions en autant de systèmes qu'elles en comportent (vous trouverez des indications sur la [procédure de découpage de la partition](#) ci-dessous).

Mais il serait fastidieux d'entrer la ligne de chaque image de système dans notre fichier `_tags_.js`. Une partition même courte peut très vite comporter de 20 à 30 systèmes et ce serait autant de lignes de partition qu'il faudrait introduire dans le code...

Au lieu de ça, si les images des systèmes ont été nommées en respectant une règle simple (avec des suites de nombres), une seule ligne suffira pour entrer tous les systèmes de la partition. Par exemple :

```
score mouvement_1/image-[1-35].png
```

Le texte ci-dessus indique qu'il y a 35 images de système dans ce mouvement. Le code qui en résultera sera :

```
score mouvement_1/image-1.png
score mouvement_1/image-2.png
score mouvement_1/image-3.png
score mouvement_1/image-4.png
...
...
score mouvement_1/image-35.png
```

Si vous indiquez une taille — ce qui est mieux pour être sûr de tenir dans la page — cette taille sera appliquée à toutes les images.

```
Tags=`  
score mouvement_1/image-[1-35].png w=17.5cm  
`;  
;
```

Nous vous invitons vivement à commencer par cette opération — l’inscription des systèmes par séquence — avant l’insertion de toute autre marque sur la partition. Comme nous l’expliquons plus haut déjà, il est recommandé, pour s’éviter ensuite un travail fastidieux de repositionnement, de placer en tout premier lieu les systèmes correctement sur chaque feuille, de façon définitive, en se servant de l’aperçu d’impression.

Noter que lorsque **MuScaT** place les images sur la table d’analyse, il les répartit pour obtenir l’aspect original de la partition. On peut modifier ce comportement en définissant explicitement un espace (vertical) entre chaque système ou chaque image, grâce à l’option **espacement images** :

```
// Code intégrale du fichier _tags_.js  
option('code');option('espacement images', 50);  
  
Tags=`  
sco haydn/mouvement_1-[1-35].png  
`;  
;
```

Noter la version raccourcie de la nature du TAG : **sco** pour **score**.

Noter également l’usage de l’option **code** qui permet d’afficher le code à côté de la table de l’analyse, pour pouvoir le modifier dans le navigateur lui-même.

Grâce à l’option **espacement images** définie ci-dessus, chaque image (chaque système) sera séparée de 50 pixels.

Une fois ce code établi, vous pouvez déplacer les images dans la page pour les ajuster à vos besoins. Cela créera automatiquement les **x** et les **y** des coordonnées spatiales de chaque système au bout des lignes de score.

Astuce : si votre écran est assez grand et que vous adoptez l’option **code beside** (ou **code à côté**), vous pourrez voir en direct votre code s’actualiser.

## Les Accords

Les accords, placés en général au-dessus de la portée, se définissent par les natures **accord**, **chord** ou **acc** en version raccourcie.

On peut les indiquer en version anglosaxonne (**A**, **B**, ...) ou en version italienne (**Do**, **Ré**, ...), peu importe. L'important est de comprendre que comme tout texte **MuScaT** impose de remplacer les espaces par des traits plats. Ainsi, pour indiquer un accord de Si bémol 7e diminuée, on pourra utiliser dans les deux systèmes de langue :

```
Tags=`  
chord SIb_min_7edim  
acc Bb_min_7edim  
`;  
;
```

Le code ci-dessus produira :

**SIb min 7edim    Bb min 7edim**

La taille du texte peut se régler de façon générale avec l'option **chord** **size** ou, pour un accord particulier, avec la propriété **fs**.

Cf. [Options de taille pour les textes](#), pour des renseignements complets sur les options de tailles.

## Les Chiffrages (Harmonie)

On indique un chiffrage d'accord, sous la partition, à l'aide de la *nature* **harmonie**, **harmony**, **chiffrage** ou **har**.

Les recommandations sont les mêmes que pour les accords : aucune espace.

La taille du texte peut se régler de façon générale avec l'option **harmony** **size** ou, pour un chiffrage particulier, avec la propriété **fs**.

Cf. [Options de taille pour les textes](#), pour des renseignements complets sur les options de tailles.

## Les Cadences

On indique une cadence, sous la partition, à l'aide de la *nature* **cadence** ou **cad**.

```
Tags=`  
  cadence I type=parfaite x=100 y=200 w=150  
`;  
;
```

Remarquer que deux nouvelles propriétés apparaissent ici : le **type**, qui définit comme son nom l'indique le type de cadence (cf. la liste ci-dessous) et **w**, la largeur, qui détermine ici la longueur du trait.

On peut faire varier la longueur du trait en jouant sur la touche **w** (pour augmenter la longueur du trait) et **<ALT w>** (pour la diminuer). Les touches **<MAJ>** et **<CTRL>** servent respectivement à augmenter le pas ou à le diminuer (action plus précise).

La taille du texte peut se régler de façon générale avec l'option **cadence size** ou, pour une cadence particulière, avec la propriété **fs**.

Cf. [Options de taille pour les textes](#), pour des renseignements complets sur les options de tailles.

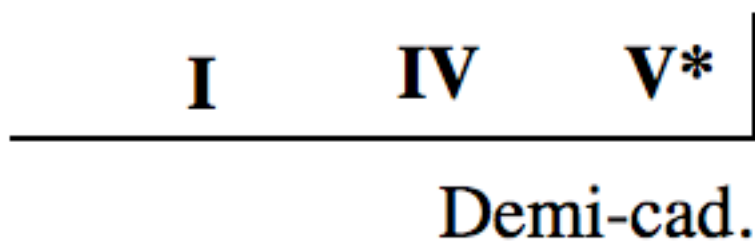
### Types de cadence

Les types de cadence qu'on peut utiliser comme argument de la propriété **type** sont :

- ⌚ parfaite
- ⌚ imparfaite
- ⌚ demi
- ⌚ italienne
- ⌚ rompue
- ⌚ plagale
- ⌚ faureenne (pour la cadence Fauréenne)
- ⌚ baroque

Exemple :

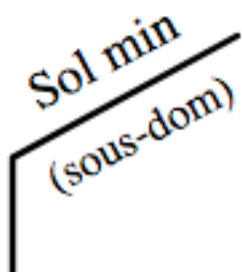
```
Tags=`  
  har I x=110 y=264  
  har IV x=155 y=262  
  cad V* type=demi x=77 y=263 w=147px  
`;  
;
```



## Les modulations

On peut mettre un texte au-dessus de la barre inclinée (en général la tonalité vers laquelle on module) et un texte en dessous (en général la fonction de cette tonalité).

Pour séparer les deux textes, on utilise tout simplement la barre inclinée, appelée « balance ». Ainsi, pour obtenir :



... on utilisera simplement :

```
modulation Sol_min/(sous-dom.) x=200 y=300
```

ou

```
mod Sol_min/(sous-dom.) 200 300
```

On peut modifier la hauteur du trait vertical qui rejoint la partition en modifiant la propriété **h**. On peut donc la modifier en pressant la touche **h** (augmente la longueur du trait) ou les touches **<ALT h>** (diminue la longueur du trait).

La taille du texte peut se régler de façon générale avec l'option **modulation size** ou, pour une modulation particulière, avec la propriété **fs**.

Cf. [Options de taille pour les textes](#), pour des renseignements complets sur les options de tailles.

## Les autres types de textes

Ce que l'on appelle spécifiquement les « textes », ici, ce sont tous les textes hors des accords, modulations, chiffrages, etc. Ce sont vraiment des textes qu'on peut placer n'importe où. À commencer par la définition des grandes parties de la pièce (« Introduction », « Coda », etc.).



Dans un texte, il est impératif de remplacer toutes les espaces par des traits plats (on les obtient, sur mac, à l'aide de Maj+tiret).

Par exemple, pour écrire sur la partition :

Premier couplet

Il faut impérativement définir la ligne :

```
texte Premier_couplet y= 50 x=200
```

En dehors des textes « normaux » ou simples, on peut utiliser :

- 🌀 Les parties
- 🌀 Les mesures
- 🌀 Les degrés

## Les parties

Les marques de partie s'indiquent avec le TAG **partie** (ou **par** ou **part**).  
Ce sont des textes dans des boîtes inclinées qui ont cet aspect :



La taille du texte peut se régler de façon générale avec l'option **part size** ou, pour une partie en particulier, avec la propriété **fs**.

Cf. [Options de taille pour les textes](#), pour des renseignements complets sur les options de tailles.

## Les mesures

Les numéros de mesure, s'ils ne sont pas indiqués sur la partition elle-même, peuvent être ajoutés à l'aide du TAG **measure** (ou **measure**, ou **mes**), suivi du numéro de mesure puis des coordonnées.



La taille du texte peut se régler de façon générale avec l'option `mesure` `size` ou, pour un numéro de mesure en particulier, avec la propriété `fs`.

Cf. [Options de taille pour les textes](#), pour des renseignements complets sur les options de tailles.

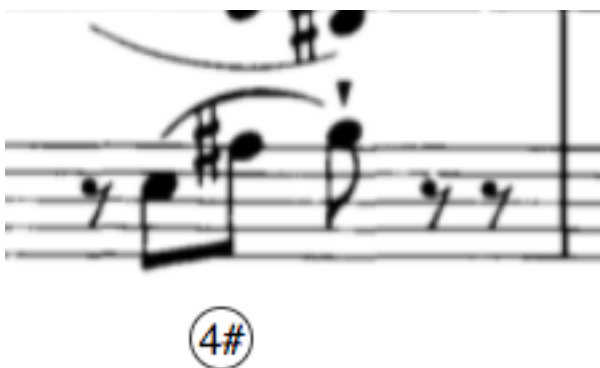
## Les degrés

Parfois il est intéressant de marquer les degrés des notes dans la gamme. On peut le faire grâce à la *nature* `degre`, `degree` ou le diminutif `deg`.

```
Tag= `
```

```
degree 4# x=123 y=678
```

```
` ;
```



La taille du texte peut se régler de façon générale avec l'option `degre` `size` ou, pour un degré en particulier, avec la propriété `fs`.

Cf. [Options de taille pour les textes](#), pour des renseignements complets sur les options de tailles.

## Les marques musicales diverses

Plusieurs marques musicales seront introduites dans les versions suivantes.

# Autres éléments graphiques

## Les Lignes

Les lignes se définissent par `line` ou `ligne`.

Le premier élément définit le `type` de la ligne. Bien noter que le `type` dont on parle ici ne se définit pas avec la propriété `type` — comme c’est le cas avec les cadences par exemple —, mais simplement en second mot. Dans le code suivant :

```
Tags = `
  line |____| x=100 y=120 w=230
`;
```

... `|____|` est le *type* de la ligne.

Les *types* sont les suivants :

		Description
U	____	Ligne inférieure, trait vertical avant et après
N	—	Ligne supérieure, trait vertical avant et après
L	____	Ligne inférieure, trait vertical avant
K	—	Ligne supérieure, trait vertical avant
V	____	Ligne inférieure, trait vertical après
^	—	Ligne supérieure, trait vertical après

On peut ensuite définir sa taille et sa position avec les lettres habituelles `x` (position horizontale), `y` (position verticale) et `w` (largeur en pixels).

## Les Boîtes

Les boîtes permettent aussi bien d’entourer un élément que de le masquer. Elles se définissent avec les propriétés `x`, `y`, `w` pour la largeur, `h` pour la hauteur et `bgc` pour la couleur de fond (cf. [Note sur les couleurs](#)).

Noter qu’une boîte sans couleur, sur la table d’analyse, apparaîtra toujours grisée — pour être visible et manipulable —, mais qu’elle sera invisible à l’impression ou dans le PDF.

Noter également qu’on peut utiliser de la transparence pour les boîtes. Il suffit pour cela de leur donner une valeur de `bgc` (background-color) à l’aide de `rgba` (où « a » final signifie « alpha », la transparence). Par exemple:

```
Tags=\  
// ...  
box x=100 y=233 w=50 h=400 bgc=rgba(0,255,0,0.3)  
// ...  
`;  
;
```



Rappelez-vous bien qu'il ne doit y avoir aucune espace dans la définition de `bgc` (comme dans la définition de toute valeur).

## Opérations sur les *TAGs*

- 🔒 Verrouiller les *TAGs*
- 🔗 Grouper et dégroupier des *TAGs*
- 📄 Ligne de code du *TAG*

### Verrouillage des *TAGs*

On peut « verrouiller » un *TAG*, c'est-à-dire empêcher totalement ses modifications, aussi bien sa position que son contenu, en ajoutant une astérisque, un rond (<ALT #>) ou même un 🔒 au tout début de sa ligne (suivi ou non par une espace).

Les trois lignes suivantes verrouillent leur *TAG* :

```
* sco systeme1.png id=14 x=103 y=567  
• sco systeme1.png id=15 x=103 y=678  
🔒 sco systeme1.png id=16 x=103 y=874
```

**MuScaT** ajoutera un vrai cadenas (🔒) qui rendra ce verrouillage très visuel.

Une fois verrouillé, le *TAG* ne peut plus être déplacé à la souris. En revanche, il peut tout à fait être modifié dans le code (sa position, son contenu, etc.) pour un ajustement très précis.

Pour déverrouiller un *TAG* et le rendre à nouveau mobile, il suffit tout simplement de retirer cette marque de verrouillage dans le code.

## Grouper et dégroupier des TAGs

« Grouper » des *TAGs* permet de les considérer comme un seul élément. On peut de cette manière les déplacer ensemble ou les supprimer tous ensemble.

Pour grouper :

- 🌀 sélectionner les *TAGs* les uns après les autres en maintenant la touche MAJ appuyée,
- 🌀 activer le bouton « Grouper » dans [la boîte à outils](#) ou jouer la combinaison clavier **<CMD G>** (**<Ctrl G>** sur Windows).

Pour dégroupier :

- 🌀 sélectionner un groupe en sélectionnant l'un de ses éléments,
- 🌀 activer le bouton « Dégroupier les *TAGs* » dans [la boîte à outils](#) ou jouer la combinaison clavier **<CMD G>** (**<Ctrl G>** sur Windows).

---

## Procédure de découpage de la partition

Voyons quelques méthodes de découpage de la partition en « images-systèmes ». Je les présente ici de la meilleure à la moins bonne. Cette qualité a été définie en fonction des deux critères suivants :

- 🌀 rapidité d'exécution,
- 🌀 précision du découpage.

### Avec capture sélection dans Aperçu (Mac)

Méthode la plus rapide, mais également la moins précise. Ce manque de précision oblige parfois à reprendre des systèmes pour mieux les découper. Cependant, elle est tellement plus rapide que les autres que je la privilégie sans problème, d'autant que le redécoupage est aussi simple.

- 🌀 Ouvrir la partition PDF dans l'application Aperçu,
- 🌀 jouer **<CMD Maj 4>** pour activer la sélection par souris,
- 🌀 sélectionner la zone de la partition à capturer — un système — (ne pas



- avoir peur de « prendre large », il est facile d'affiner ensuite),
- ⌚ recommencer l'opération pour tous les systèmes,
- ⌚ récupérer les captures sur le bureau — sauf si l'astuce ci-dessous (1) a été utilisée — et les mettre dans le dossier `images` de votre analyse,
- ⌚ modifier les noms des fichiers — sauf si vous avez utilisé l'astuce ci-dessous (1) — en les indiquant de 1 (ou 0) à N pour les insérer plus facilement dans l'analyse.

Pour affiner le découpage :

- ⌚ ouvrir l'image dans Aperçu,
- ⌚ choisir si nécessaire la sélection rectangle (p.e. Outils > Sélection rectangulaire),
- ⌚ sélectionner la partie à conserver,
- ⌚ affiner à l'aide des poignées,
- ⌚ jouer `<CMD K>` pour « cropper » l'image,
- ⌚ l'enregistrer.

1. Astuce : pour aller encore plus vite, vous pouvez :

- ⌚ utiliser l'utilitaire Muscat `change_folder_captures` pour définir le dossier des captures écran ou consulter la [procédure décrite ici](#). Vos captures iront directement dans ce dossier,
- ⌚ effectuer les captures,
- ⌚ utiliser l'utilitaire Muscat `rename_images` pour renommer instantanément vos fichiers.

Note : vous pouvez voir ou revoir la procédure dans les tutoriels consacrés sur [ma chaine YouTube](#).

## Avec sélection rectangulaire dans Aperçu (Mac)

La méthode suivante ressemble à la précédente mais permet d'être plus précis. Mais cette précision se fait au détriment du temps, notamment pour l'enregistrement des fichiers images.

- ⌚ Ouvrir la partition PDF dans Aperçu,
- ⌚ choisir la sélection rectangle (p.e. Outils > Sélection rectangulaire),
- ⌚ sélectionner le système grossièrement,
- ⌚ ajuster parfaitement la sélection à l'aide des poignées,
- ⌚ copier la sélection (`<CMD C>`),
- ⌚ activer la combinaison `<CMD N>` pour créer une nouvelle image à partir du presse-papier,
- ⌚ enregistrer l'image (`<CMD S>`) avec le nom voulu, dans le dossier voulu, en choisissant le format voulu.

## Avec Aperçu, sélection souris et rectangle (Mac)

On peut bien entendu imaginer une méthode intermédiaire qui reprendrait les deux méthodes précédentes. Lorsque la découpe est facile, on utilise la première, lorsque la découpe demande plus de précision, on privilégie la seconde.

## Avec MuScaT et **convert**

C'est une méthode qui souffre parfois d'un manque de qualité de rendu.

On tire déjà les images du PDF à l'aide de la commande à jouer dans le Terminal (adapter la qualité du traitement en fonction du résultat) :

```
# Se trouver dans le dossier contenant la partition
(cd ...)
convert[ options] partition.pdf partition.jpg # ou
.png
```

Autant d'images que de pages sont produites.

On insert la première dans le code du fichier `_tags_.js`, avec l'option **crop image** :

```
# Dans _tags_.js
option('crop image')
Tags=`
partition partition-0.jpg
`;
```

On ouvre le fichier `TABLE_ANALYSE.html` dans Chrome.

Maintenant, il suffit de sélectionner, à la souris, la zone de l'image à prendre puis de coller le code du presse-papier dans la console du Terminal. Puis de jouer ce code.

Répéter l'opération avec chaque système, puis avec chaque page de la partition.

## Avec Gimp/Photoshop (ou autre logiciel de traitement de l'image)

Si un logiciel de traitement d'images présente une précision de découpage inégalable, il offre en revanche la méthode la plus chronophage, même avec l'habitude du logiciel.

- 🌀 Ouvrir le PDF dans Gimp,
- 🌀 sélectionner chaque système en le découpant,
- 🌀 le placer en haut,
- 🌀 « cropper » l'image à la taille du plus haut système,
- 🌀 exporter chaque image-système (avec le bon nom).

Ce mode d'emploi n'étant pas destiné à maîtriser Gimp, Photoshop ou autre, je vous renvoie au manuel d'utilisation de ces applications.

## Ligne de code du TAG

On peut obtenir la ligne de code d'un TAG ou même de plusieurs TAGs de cette manière :

- 🌀 sélectionner sur la table d'analyse le ou les TAGs dont on veut les codes,
  - 🌀 jouer la combinaison **<ALT C>**,
  - 🌀 coller le code mis dans le presse-papier.
- 

## Animation d'une analyse

- 🌀 Démarrage de l'animaton
- 🌀 Boutons de l'animation
- 🌀 Réglage de l'animation

Serait-ce la cerise sur le gâteau de **MuScaT** ?... L'application ne permet pas seulement de faire une analyse statique, elle permet aussi de créer une animation qu'on peut utiliser pour YouTube ou pour donner un cours physique à la manière d'un power-point.

Les fonctionnalités de l'animation sont limitées cependant, puisqu'on ne peut que faire apparaître les éléments les uns après les autres. On ne peut pas (ou pas encore) les déplacer, les coloriser, etc. Avec un peu d'imagination et en exploitant toutes les possibilités de **MuScaT**, on peut cependant parvenir à des choses assez complexes.

Vous pouvez en trouver des illustrations sur les vidéos de ma chaîne :  
<https://www.youtube.com/channel/UCX3XhJw9x1RsVx1s3GNYceA>.

## Démarrage de l'animation

Pour lancer une animation, il n'y a rien de plus simple à faire que d'ajouter le commentaire **// START** à l'endroit où l'on veut qu'elle démarre.

À partir de ce **START**, tous les groupes de TAGs non espacés seront affichés ensemble et l'animation fera une pause lorsqu'elle rencontrera une ligne vide.

Tout ce qui précède ce commentaire **// START** sera affiché d'un seul coup.

Ensuite, chaque « groupe de TAGs » est affiché en laissant une pause entre chacun d'eux. Un « groupe de TAGs » est une suite de TAGs qui ne sont

séparés d'aucune ligne vide. Par exemple, ci-dessous, on trouve deux groupes de *TAGs*, qui s'afficheront donc en deux temps lors de l'animation :

```
// dans _tags_.js
Tags = `
// ...

// START

// Premier groupe de TAGs
acc C x=100 y=23
cad I x=100 y=233
sco mon-systeme-d.png x=3 y=200

// Second groupe de TAGs
mod G/(sous-dom) x=34 y=340
sco mon-systeme-e.jpg x=5 y=400

// ... suite...
`;
```

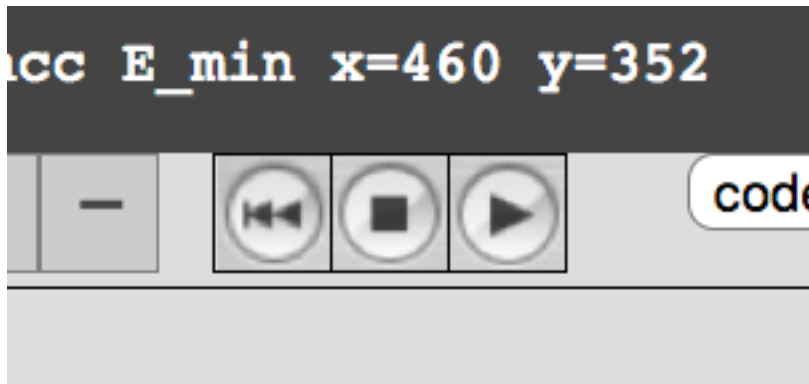
On peut régler la vitesse générale de l'animation en option mais on peut également définir des temps plus ou moins longs entre l'affichage des différents *TAGs*, par exemple pour aménager un temps plus long d'explication entre deux *TAGs*. Pour ce faire, on joue simplement sur le nombre de lignes vides entre ces *TAGs*.

Par exemple, ci-dessous, il y aura deux fois plus de temps entre la ligne `acc D x=100 y=200` et la ligne `acc E x=120 y=200` qu'entre la ligne `acc E x=120 y=200` et la ligne `acc F x=140 y=200`.

```
// Dans le fichier _tags_.js
Tags= `
1 | // ...
2 |
3 | // START
4 |
5 | acc D x=100 y=200
6 |
7 |
8 |
9 | acc E x=120 y=200
10|
11| acc F x=140 y=200
12|
```

## Boutons de l'animation

Des boutons permettent d'interagir sur l'animation pour la mettre en pause, remonter les pas ou l'arrêter et la reprendre grâce à des boutons « Retour en arrière », « Stop » et « Jouer/Pause ».



Comme vous pouvez le voir sur l'image ci-dessus, si le code de l'animation est activé, les boutons se trouvent en dessous du code.

## Réglage de l'animation

Plusieurs options permettent de régler les paramètres de l'animation.

### ⌚ Réglage de la vitesse

## Réglage de la vitesse de l'animation

On peut régler la vitesse de l'animation à l'aide de l'option `vitesse animation` ou `animation speed`. C'est un nombre de 1 à 100. Plus il est élevé et plus l'animation est rapide (i.e. plus les pauses sont courtes). Par exemple :

```
// Dans le fichier _tags_.js
option('vitesse animation', 80);
Tags = `
// Premiers tags à afficher d'un coup

// START

// Tags qui s'afficheront progressivement.
`;
```

---

## Options

### ⌚ Options de la langue



- ⌚ Option « code à côté »
- ⌚ Option « code sans options »
- ⌚ Option « lignes de repère »
  - ⌚ Position des lignes repères
- ⌚ Option « Rectangle de sélection »
- ⌚ Options de taille pour les textes
- ⌚ Option « espacement images »
- ⌚ Option « marge haut »
- ⌚ Option « marge gauche »
- ⌚ Thème
- ⌚ Option « découpe image »
- ⌚ Vitesse de l'animation
- ⌚ Option « Viseur de position »

Comme les TAGs et les partitions, les options se règlent dans le fichier `_tags_.js`. On utilise tout naturellement la fonction `option` (ou `options`) avec en argument les options à activer.

Ci-dessous, par exemple, on active l'option `guide` qui affiche deux lignes repère déplaçables pour aligner des éléments à la souris (ou par magnétisation).

```
// Dans _tags_.js
option('guide');
Tags=`
    ...
`;
```

Dans la méthode `option`, on peut passer toutes les options les unes à la suite des autres, ou utiliser plusieurs fois la méthode `option`. Les trois formulations suivantes sont équivalentes :

```
// Dans _tags_.js
option('guide', 'code', 'marge haut', 100);
```

... équivaut à :

```
// Dans _tags_.js
option('guide');option('code');option('marge
haut', 100);
```

... qui équivaut à :

```
// Dans _tags_.js
option('guide');
option('code');
option('marge haut', 100);
```

Note : les points-virgules sont optionnels.

Vous noterez qu'il existe deux types d'options. Les options dites « booléenne » qu'on active simplement en indiquant leur nom en argument (par exemple `guide` ou `code`) et il y a les options non booléennes qui attendent une valeur précise (par exemple `marge haut` attend la valeur de cette marge haut).

Dans les arguments de la méthode `option`, la valeur des options non booléennes doit suivre immédiatement le nombre de l'option :

```
// Dans _tags_.js
option('marge haut', 100);
```

Ci-dessus, la valeur `100` sera appliquée à l'option `marge haut`.

## Option « langue »

Option	<code>lang</code> , <code>langue</code>
Type	les deux lettres de la langue, par exemple <code>fr</code> (français) ou <code>en</code> (anglais).

Pour définir la langue parlée par l'application. Pour le moment, l'application ne sait que parler français et anglais, mais nous espérons rapidement voir d'autres langues se développer. Avis aux amateurs traducteurs même inexpérimentés !

## Option « code à côté »

Option	<code>code beside</code> , <code>code à côté</code>
Type	booléen

L'option « code à côté » permet d'avoir le fichier contenant le code juste à

côté de la partition, ce qui est très pratique pour le modifier sans avoir à changer d'application. On le voit ci-dessous dans la boîte noire.



## Option « découpe image »

Option	<code>crop image</code> , <code>découpe image</code>
Type	booléen

Cette option fait passer dans un mode d'utilisation qui va permettre de découper l'image de façon aisée (par simple copié-collé).

## Option « code sans options »

Option	<code>code sans option</code> , <code>code sans options</code> , <code>code no option</code>
Type	booléen

Par défaut, le code de l'analyse copiée dans le presse-papier contient l'intégralité de ce que doit contenir le fichier `_tags.js`, c'est-à-dire la définition des options et la valeur de la variable `Tags`.

Mais lorsque l'on veut garder plusieurs versions de son code, en les mettant au bout les unes des autres plutôt qu'en les remplaçant, des définitions d'options peuvent se contredire.

Avec l'option `code sans option` activée, seule la définition de la variable `Tags` est copiée dans le presse-papier, ce qui permet de ne garder que la ligne en haut du fichier pour définir les options. Impossible dans ce cas de s'y perdre :

```
// Dans _tags.js
option('code', 'code sans options', 'reperes');

// Version 1
```

```
Tags = `  
// ... tags de la version 1  
`;  
  
// Version 2  
Tags = `  
// ... tags de la version 2  
`;  
  
// Version 3  
Tags = `  
// ... tags de la version 3  
`;  
  
// etc.
```

Option « lignes de repère »

Option	repères , reperes , lines of reference , guides
Type	booléen

Ajoute une ligne horizontale et une ligne verticale qu’on peut déplacer et qui peuvent servir de guide, de repère, pour placer les TAGs.

Position des lignes repères

Pour la position de la ligne verticale :

Option	position repère vertical , vertical line offset
Type	nombre de pixels

Pour la position de la ligne horizontale :

Option	position repère horizontal , horizontal line offset
Type	nombre de pixels

Exemple :

```
// Dans le fichier _tags_.js de l'analyse  
// à 120 pixels du haut et 200 de la gauche
```

```
option('vertical line offset', 120, 'horizontal  
line offset', 200);
```

## Option « Rectangle de sélection »

Option	<code>rectangle selection</code> ,
Type	booléenne

Le « rectangle de sélection » permet de sélectionner plusieurs éléments à la fois en les incluant dans un rectangle dessiné à la souris.

Mais si cette fonctionnalité est tout à fait opérante dans la plupart des applications que nous connaissons, elle est très loin d’être pleinement satisfaisante dans les navigateurs (pour ceux qui connaissent : même avec `DragSelect`).

Aussi a-t-elle été mise en option dans **MuScaT**.

Si vous l’utilisez, pensez également que vous devez attendre une seconde avec la souris pressée avant qu’elle se mette en route.

Astuce : si le rectangle vert de sélection reste actif après avoir relâché la souris, il vous suffit de cliquer n’importe où sauf sur lui pour le faire disparaître.

## Options de taille pour les textes

Grâce aux options, on peut définir la taille par défaut de tous les types de texte de l’analyse (accords, cadences, etc.).

```
// Dans _tags_.js  
option('<type> size', '<valeur>');
```

Par exemple :

```
// Dans _tags_.js  
option('harmony size', '11pt');
```

Voici la table de toutes les options et ce qu’elles affectent.

<i>L’option...</i>	<i>affecte...</i>
<code>chord size</code>	Les accords
<code>harmony size</code>	Les chiffrages

harmonie size	idem
cadence size	Les cadences
modulation size	Les modulations
measure size	Les numéros de mesures
mesure size	idem
degre size	Les degrés de gamme
degree size	idem
part size	Les noms de parties
text size	Tous les autres textes

Noter que même si une taille est définie par les options, on peut fixer individuellement la taille des *TAGs* à l'aide de la propriété **fs** (ou **font-size**) dans la ligne de code du *TAG*.

### Option « Espacement entre images »

Option	espacement images , space between scores
Type	nombre de pixels

Permet de régler l'espacement en pixels entre deux images lorsque l'écriture séquentielle des images a été adoptée.

```
// Dans _tags_.js
option('espacement images', 100);
Tags=`
    ...
`;
```

Avec le code ci-dessus, l'espace entre les différents systèmes sera de 100 pixels.

### Option « marge haut »

Option	marge haut , top first score
Type	nombre de pixels

Lors de l'écriture séquentielle des images, cette valeur permet de déterminer à quelle hauteur doit être placée la première image (le premier système ou la partition).



```
// Dans _tags_.js
option('marge haut', 200);
Tags=`
    ...
`;
```

Avec le code ci-dessus, la première image de partition sera placée à 200 pixels du haut.

Penser à laisser de la place pour le titre.

## Option « marge gauche »

Option	<code>marge gauche</code> , <code>left margin</code>
Type	nombre de pixels

Lors de l'écriture séquentielle des images, cette valeur détermine la marge gauche où placer l'image (son `x`).

```
// Dans _tags_.js
option('marge gauche', 50);
Tags=`
    ...
`;
```

Avec le code ci-dessus, toutes les images de la séquence seront placées à 50 pixels de la gauche.

## Thème

L'option `theme` permet de choisir le thème, c'est-à-dire l'apparence générale de la partition. Pour choisir ce thème, utilisez, en haut de votre fichier `_tags_.js` :

```
option('theme', '<nom du thème>');
//...
```

Le thème par défaut est le thème **MuScaT**. On peut trouver les autres thèmes dans le dossier `/xlib/css/themes/`. Il suffit de retirer `.css` à leur nom pour obtenir le nom du thème.

muscat.css => thème 'muscat'

## Liste des thèmes

Voici une liste complète des thèmes, mais ceux-ci devraient rapidement s'étoffer :

- 🌀 muscat
- 🌀 fantasy
- 🌀 serioso

## Vitesse de l'animation

---

Option	<code>vitesse animation</code> , <code>animation speed</code>
Type	nombre de 1 (très lent) à 100 (très rapide)

Pour le détail, cf. [animation d'une analyse](#).

## Option « Viseur de position »

C'est plutôt un outil de développement (pour implémenter l'application), qui affiche un rectangle orange à l'endroit du clic (un peu plus en haut et à gauche).

Mais il peut être utile parfois pour signaler un problème : lorsque vous cliquez à un endroit de la table d'analyse et que ce « viseur » se positionne trop loin de la position cliquée, c'est qu'il y a un problème de positionnement. Si le problème persiste, vous pouvez soumettre une « issue » (un problème, une erreur) sur le [Github de MuScaT](#).

---

## Utilitaires

L'application **MuScaT**, comme tout bon vin, est fournie avec quelques utilitaires pour se faciliter la vie, en tout cas sur Mac. En voici la liste avec leur mode d'utilisation.

### Renommage des fichiers images (Mac/Unix)

Ce script, qui se trouve comme tous les scripts dans le dossier `utils` de l'application, permet de renommer les images d'un dossier de façon cohérente et indexée.

Pour utiliser ce script :

- 🌀 ouvrir l'application Terminal,
- 🌀 rejoindre le dossier de l'application **MuScaT** (commande `cd` ),

- ⌘ taper `./utils/rename_images.rb -h` et la touche Entrée pour tout savoir du script.

Noter que l'option `-h` ou `--help` permet toujours d'obtenir l'aide de la commande jouée.

## Changement du dossier des captures écran (Mac)

Par défaut, sur un Mac, les captures d'écran sont enregistrées sur le bureau. Ça n'est pas gênant en soit, il suffit de les glisser ensuite dans le dossier `images` de l'analyse. Mais si on veut encore gagner du temps, ce script permet de changer le dossier de destination.

Voici la procédure :

- ⌘ ouvrir l'application Terminal,
- ⌘ rejoindre le dossier de l'application **MuScaT** (commande `cd`),
- ⌘ taper `./utils/change_folder_captures.rb -h` et la touche Entrée pour tout savoir du script.

Pour remettre la valeur par défaut (le bureau), jouer simplement `./utils/change_folder_captures.rb` sans aucun autre argument.

## Création d'une nouvelle analyse (Mac)

Le script `create.rb` permet de créer une nouvelle analyse dans le dossier `_analyses_` de **MuScaT**.

- ⌘ Ouvrir l'application Terminal,
- ⌘ rejoindre le dossier de l'application **MuScaT** (commande `cd`),
- ⌘ puis, au choix :
  - ⌘ taper `./utils/create.rb -h` et la touche Entrée pour tout savoir du script,
  - ⌘ taper `./utils/create.rb "Ma nouvelle analyses" -o` pour créer l'analyse et l'ouvrir dans le finder.

Notez que pour l'activer, il faut l'ouvrir dans le navigateur avec le script `./utils/analyse.rb`.

## Activation d'une analyse (Mac)

Le script `analyse.rb` permet d'activer une analyse se trouvant dans le dossier `_analyses_` de **MuScaT**.

- ⌘ Ouvrir l'application Terminal,
- ⌘ rejoindre le dossier de l'application **MuScaT** (commande `cd`),
- ⌘ puis, au choix :
  - ⌘ taper `./utils/analyse.rb -h` et la touche Entrée pour tout

savoir du script.

- 🔗 taper `./utils/analyse.rb` pour obtenir la liste des analyses et en choisir une,
- 🔗 taper `./utils/analyse.rb "Mon_analyse"` pour ouvrir l'analyse qui commence par ce titre.

## Pour aller plus loin

Pour aller plus loin, si vous êtes sur Mac et que vous vous sentez à l'aise avec le Terminal, vous pouvez créer un alias dans votre `profil bash` pour ne pas avoir à rejoindre chaque fois le dossier de l'application et même utiliser les commandes plus simplement.

Grâce à cet alias, vous pouvez jouer tous les scripts ci-dessus sans autre forme de procès. Par exemple, si vous utilisez l'alias `mus`, alors il suffit d'ouvrir une nouvelle fenêtre de Terminal et de taper :

```
> mus analyse "Ma_Dernière_analyse"
```

... pour ouvrir cette analyse.

Il suffit de taper :

```
> mus rename_images "MonAnalyse" "systeme"
```

... pour renommer toutes les images du dossier `images` de « MonAnalyse ».

L'autre avantage de l'utilisation de cet alias, c'est qu'on peut utiliser les termes de différentes langues. Voir les [correspondances linguistiques](#).

## Création de l'alias

Pour créer cet alias, il suffit d'éditer le fichier de profil bash et d'ajouter la ligne `alias mus="/path/to/dossier/MuScat/utils"` en remplaçant "mus" par le mot que vous voudrez et "/path/to/dossier" par le chemin d'accès réel à votre dossier MuScaT.

Chez moi, cela revient à faire :

```
vim ~/.bash_profile
```

... pour éditer mon profil bash avec [Vim](#).

Dans ce fichier `.bash_profile`, j'ajoute la ligne :

```
alias
```

```
mus="/Users/philippeperret/Programmation/MuScaT/utils/run.rb"
```

Note : pour obtenir facilement la ligne ci-dessus sans aucune erreur, il suffit par exemple de glisser le fichier ou le dossier dans une fenêtre de Terminal. Le chemin d'accès s'y inscrit aussitôt. On peut également utiliser les fonctions puissantes d'autocomplétion.

J'enregistre le fichier avec la combinaison traditionnelle `:wq` et j'ouvre une nouvelle fenêtre de Terminal (ouvrir une nouvelle fenêtre de Terminal est indispensable pour prendre en compte les changements du profil bash).

Et maintenant, je peux, sans me trouver dans le dossier **MuScaT**, taper :

```
mus analyse "Analyse Sonate Haydn"
```

... pour ouvrir l'analyse « Analyse Sonate Haydn » qui se trouve dans le dossier `_analyses_/Analyse_Sonate_Haydn` (noter que les espaces sont automatiquement remplacées).

## Correspondances linguistiques

Quand on utilise l'alias ci-dessus, on peut utiliser ces termes :

### *Anglais*

### *Français*

create

créer

open

ouvrir

rename\_images

renommer\_images

change\_folder\_captures

change\_dossier\_captures

### *Espagnol*

### *Allemand*

### *Mandarin*

à venir...

à venir...

à venir...

## Annexe

### Application « Terminal »

Le Terminal est une application des plus puissantes, sur Mac, qui permet de travailler directement avec le noyau unix du Mac. En d'autres termes, elle permet de tout faire — attention : le pire comme le meilleur.


Cette application se trouve dans le dossier

`/Applications/Utilitaires`





mais vous pouvez l'utiliser plus facilement en passant par Spotlight. Jouer `<CMD ESPACE>`, puis taper les premières lettres « Term ».

# Raccourcis clavier

## Général

	Passer de la table d'analyse au champ de code s'il est ouvert.
---	--

## Sur la sélection

   	Déplacement à droite, à gauche, vers le bas, vers le haut
+ MAJ	+ fortement
+ ALT	+ finement
w	Augmente la largeur
+ MAJ	+ fortement
+ CTRL	+ finement
ALT w	Diminue la taille
+ MAJ	+ fortement
+ CTRL	+ finement
h	Augmente la hauteur
+ MAJ	+ fortement
+ CTRL	+ finement
ALT h	Diminue la hauteur
+ MAJ	+ fortement
+ CTRL	+ finement
x	Augmente la position H
+ MAJ	+ fortement
+ CTRL	+ finement
ALT x	Diminue la position H
+ MAJ	+ fortement
+ CTRL	+ finement
y	Augmente la position V
+ MAJ	+ fortement
+ CTRL	+ finement
ALT y	Diminue la position V
+ MAJ	+ fortement
+ CTRL	+ finement



