

MuScaT

Manuel d'utilisation

Introduction (histoire)

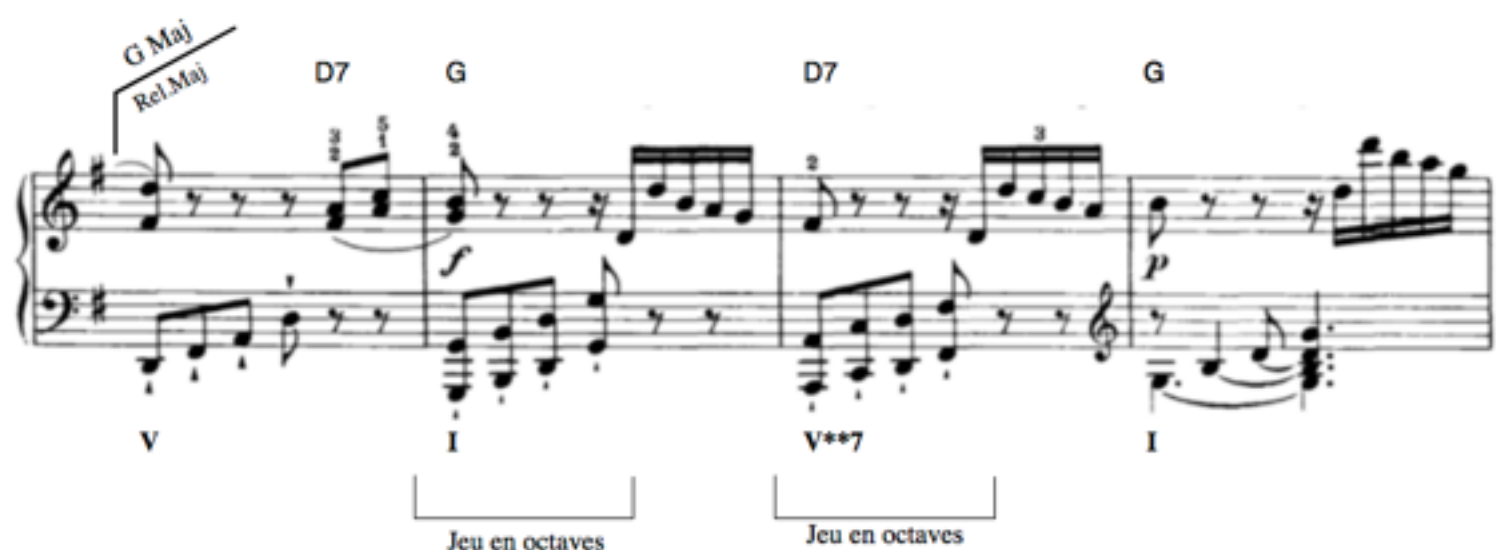
Suite à la diffusion de mon [Initiation à l'analyse musicale](#) — et son « succès » que je n'aurais jamais imaginé aussi grand — nombreux ont été les professeurs et autres pédagogues ou musicologues à me demander le logiciel que j'avais utilisé pour en créer l'animation.

C'est malheureusement une application personnelle un peu trop... personnelle (comprendre : indomptable pour qui ne l'a pas créé), 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 et à mes propres besoins, j'ai voulu concevoir un outil plus simple et plus pratique qui permettrait de réaliser rapidement des analyses de partitions (entendu que c'est toujours un peu fastidieux et que le résultat manque souvent d'homogénéité).

C'est ainsi qu'est née l'idée de **MuScaT** — dont le nom est composé 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 très propre et très pratique, des analyses de partitions musicales** comme 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 général de création d'une analyse
- ⌚ 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,
 - ⌚ Ajouter les accords, les chiffrages, les cadences, tous les éléments d'analyse,
 - ⌚ Positionner les éléments graphiques,
 - ⌚ Les lignes repères
 - ⌚ Récupérer du code final,
 - ⌚ Imprimer en PDF.
- ⌚ L'interface
- ⌚ Composition d'un tag
- ⌚ Les Images
 - ⌚ Définition de la taille d'une image
 - ⌚ Séquence d'images
- ⌚ Tous les types de tags (natures de tags)
 - ⌚ Second mot (contenu, accord)
 - ⌚ Autres données de la ligne
 - ⌚ Les types de textes
- ⌚ 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 loing

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.

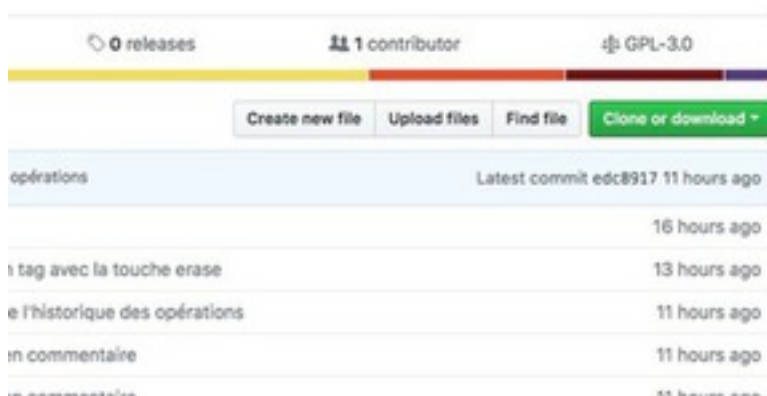
- ⌚ Charger de l'application **MuScaT**
- ⌚ Créer du dossier de l'analyse,
- ⌚ Mettre l'analyse en analyse courante,
- ⌚ Découper la partition en « images-systèmes»,
- ⌚ Inscrire les images-systèmes dans l'analyse,
- ⌚ Ajouter les accords, les chiffrages, les cadences, tous les éléments d'analyse,
- ⌚ Positionner les éléments graphiques,
- ⌚ Les lignes repères
- ⌚ Récupérer du code final,
- ⌚ Imprimer en PDF.

Synopsis détaillé

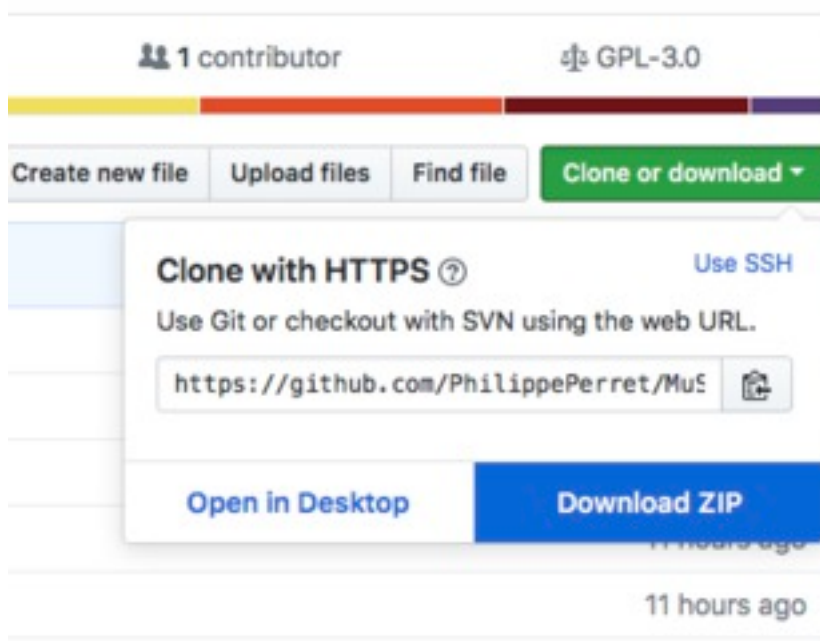
Chargement de l'application MuScaT

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 »,

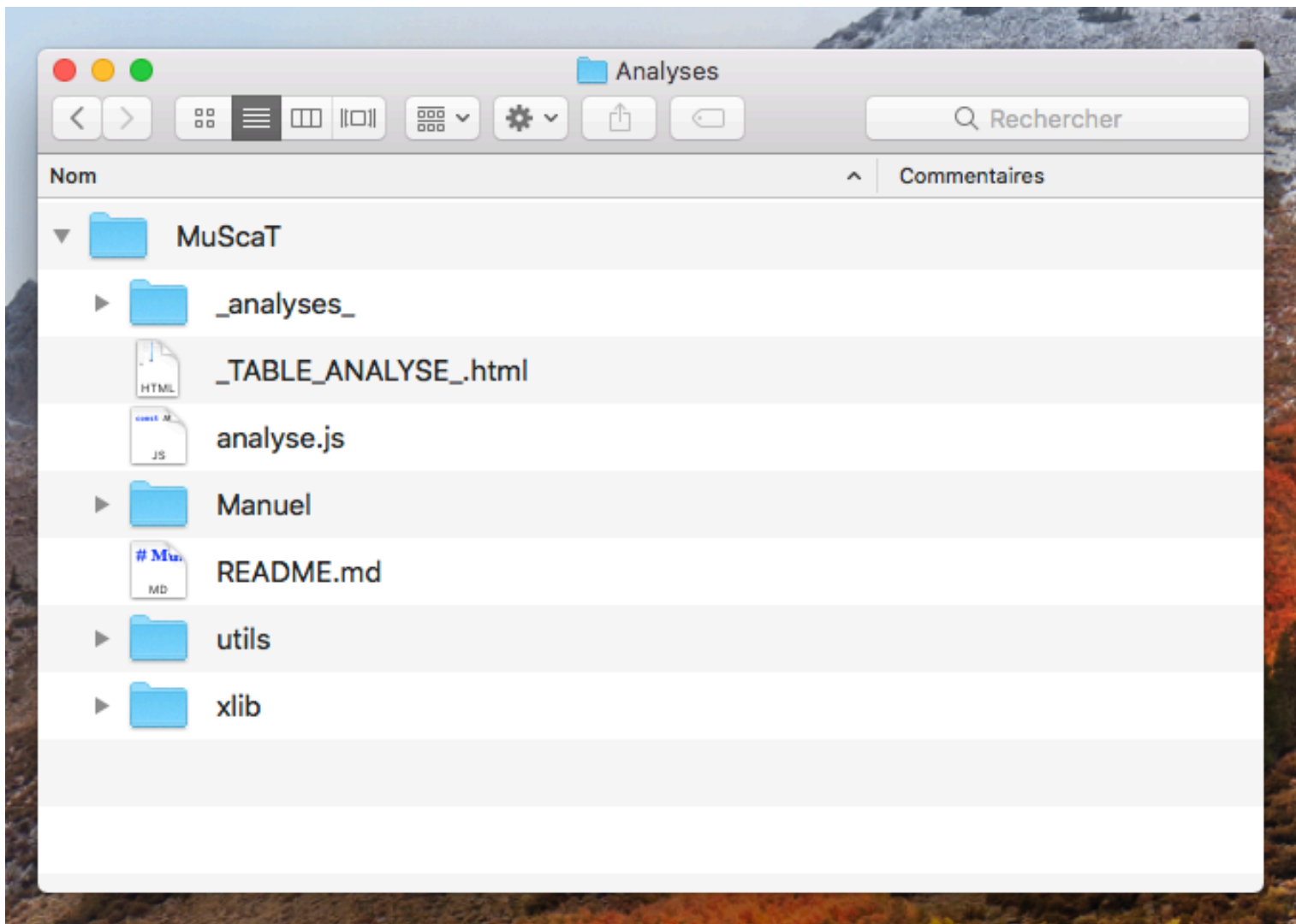


... de choisir « Download ZIP »



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

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



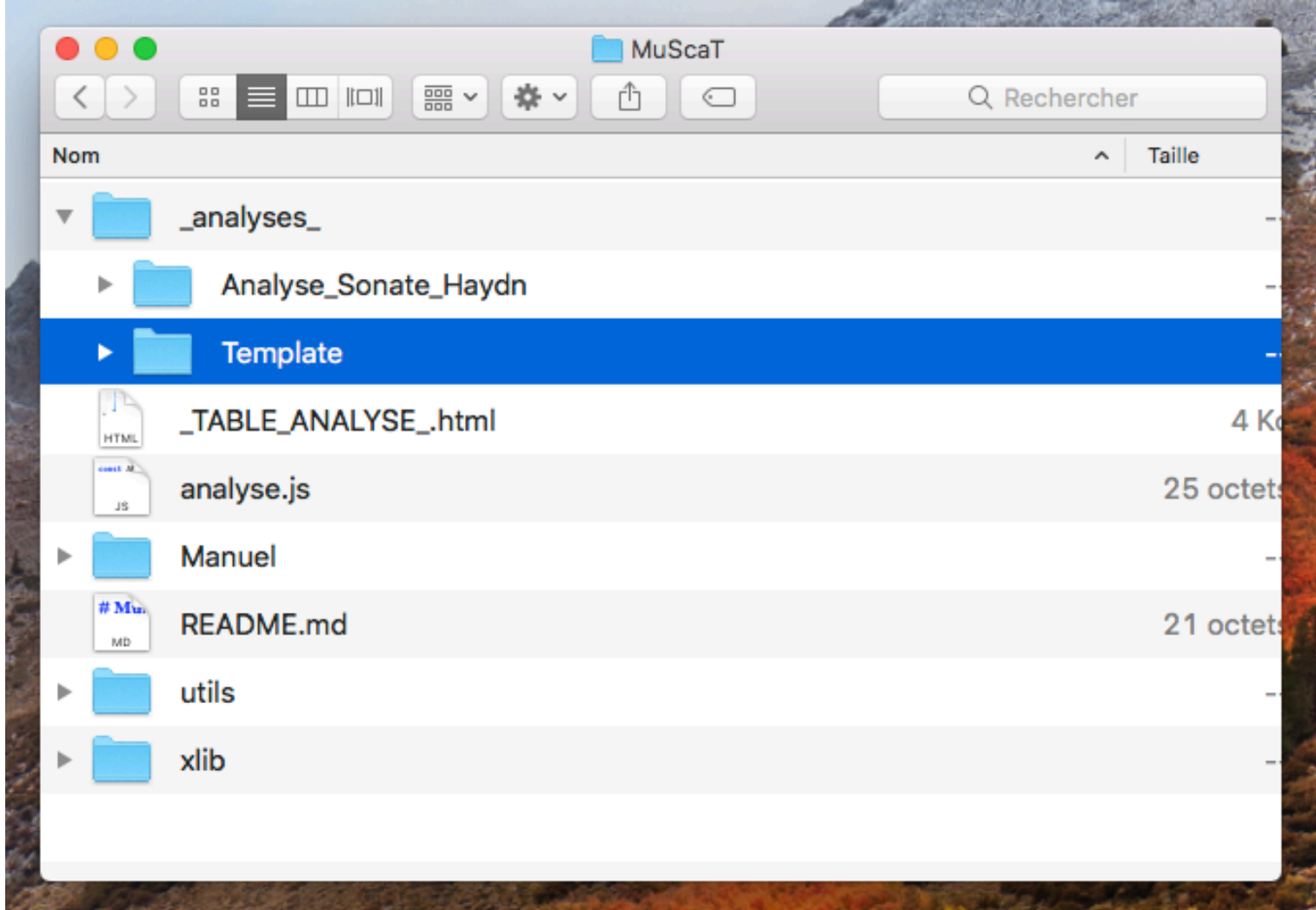
Création du dossier de l'analyse

Le plus simple pour créer une nouvelle analyse — et donc son dossier — est d'utiliser le script `create.rb` (ruby doit être installé sur votre ordinateur) qui fait tout le travail pour vous, simplement en lui donnant le nom de l'analyse.

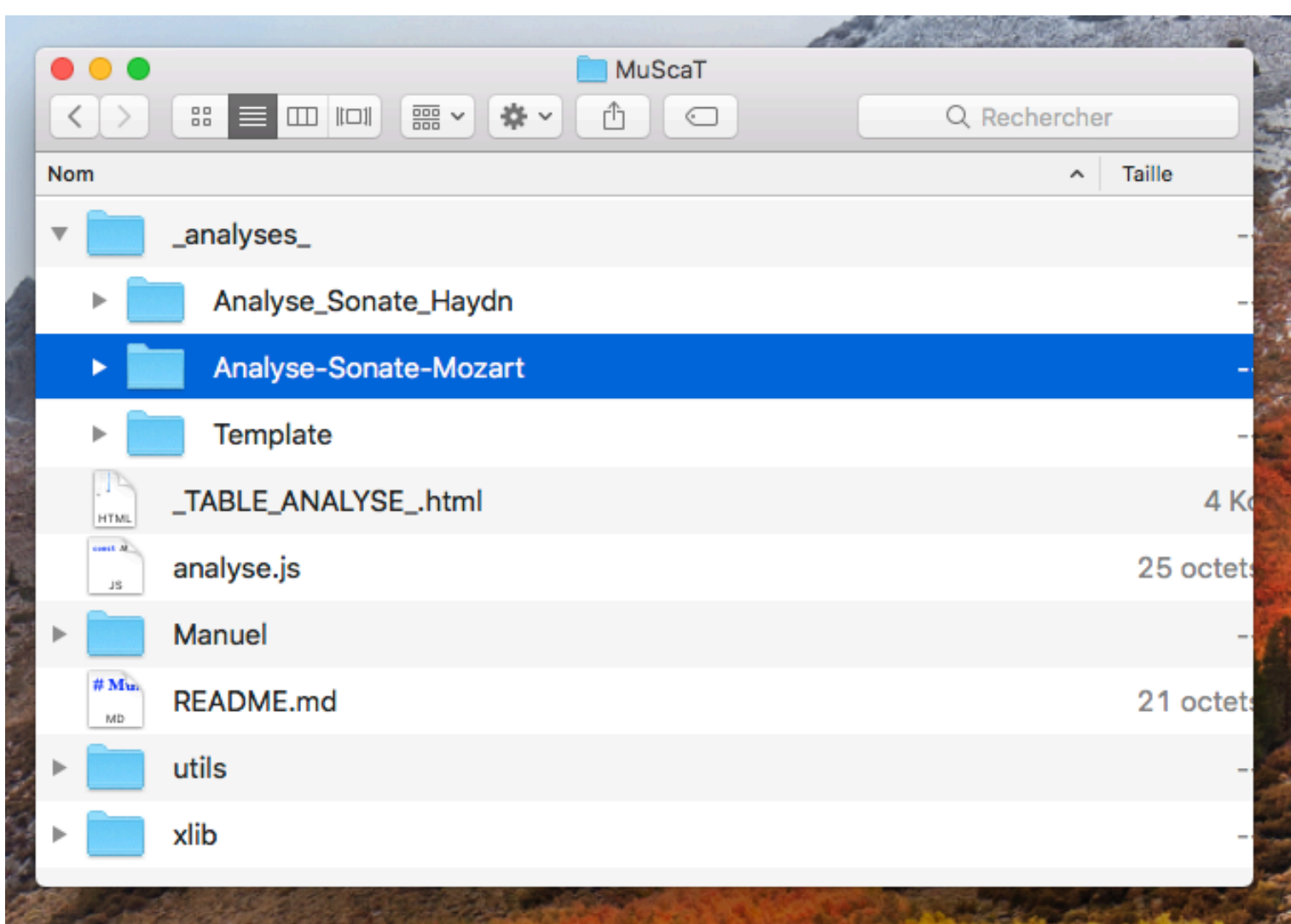
```
> cd /chemin/vers/dossier/MuScaT
> ./utils/create.rb "Ma première analyse"
```

Sans ce script, la procédure est à peine plus compliquée :

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

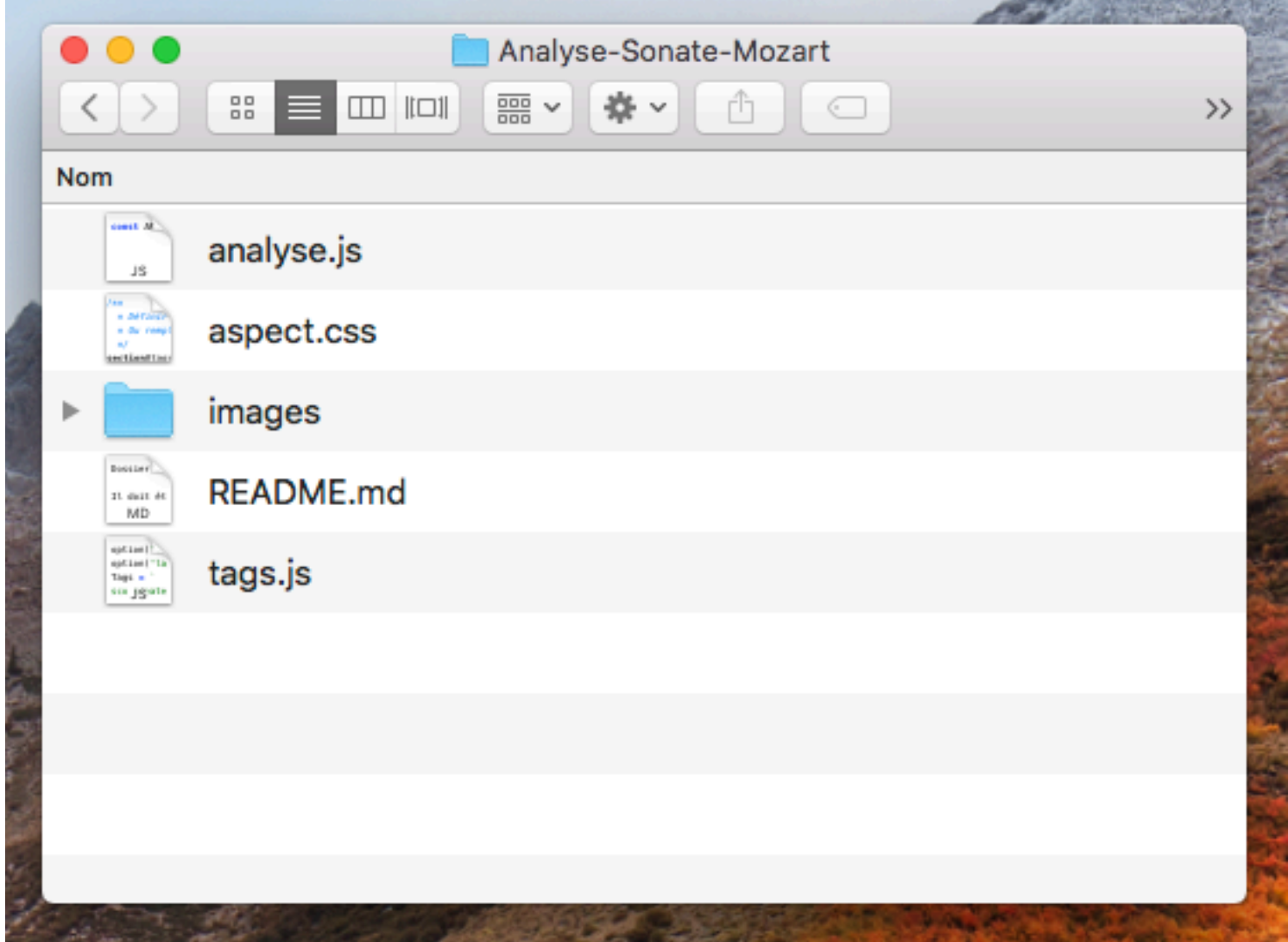


- 🔗 le renommer du nom de l'analyse, par exemple « Analyse-Sonate-Mozart ».

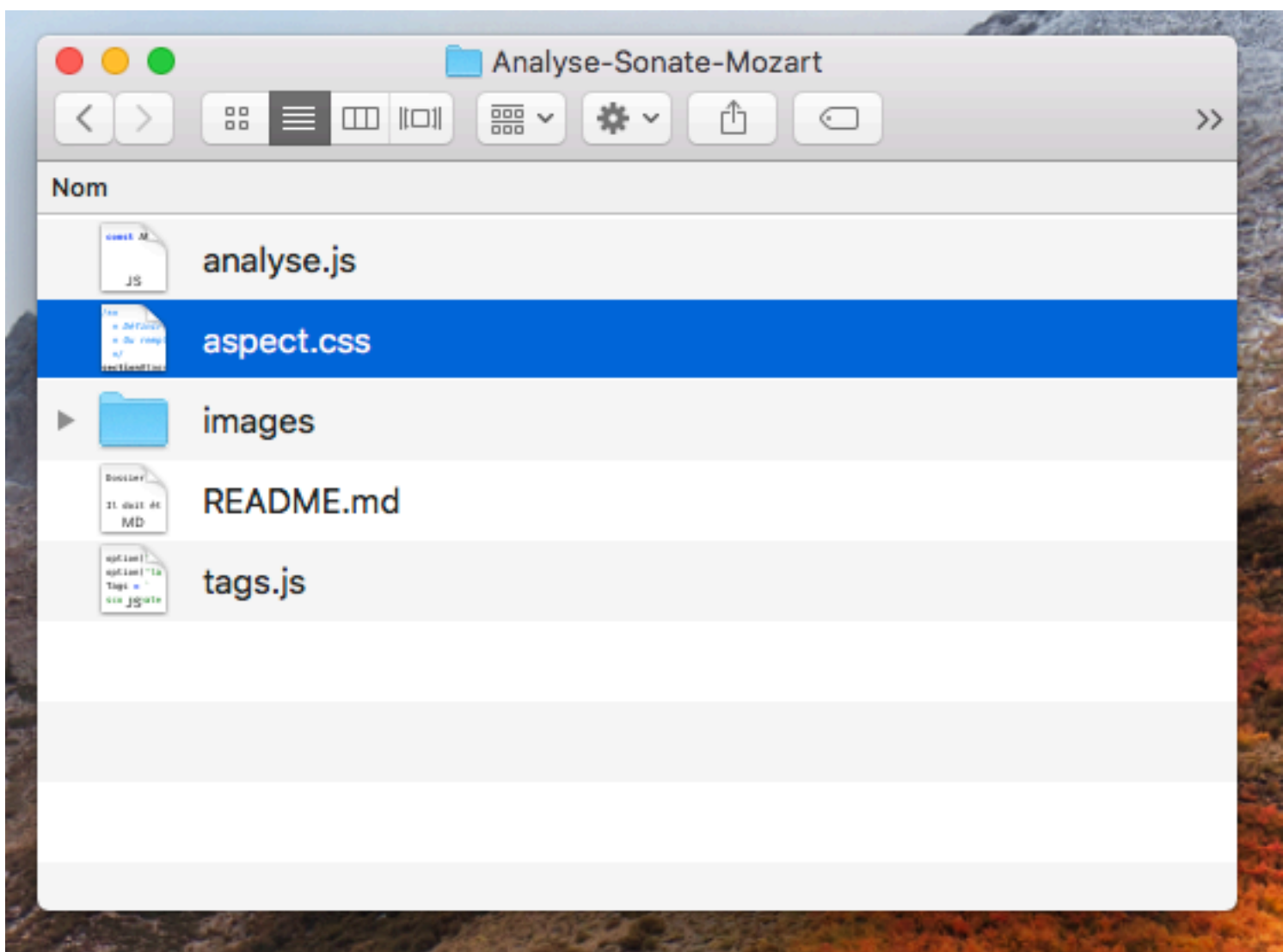


Note : il est vivement recommandé de ne pas mettre d'espaces vides dans les noms de dossier ou de fichiers pour une raison qui sera expliquée plus tard. Personnellement, j'aime les remplacer par des traits plats (« Analyse_Sonnate_Mozart »)

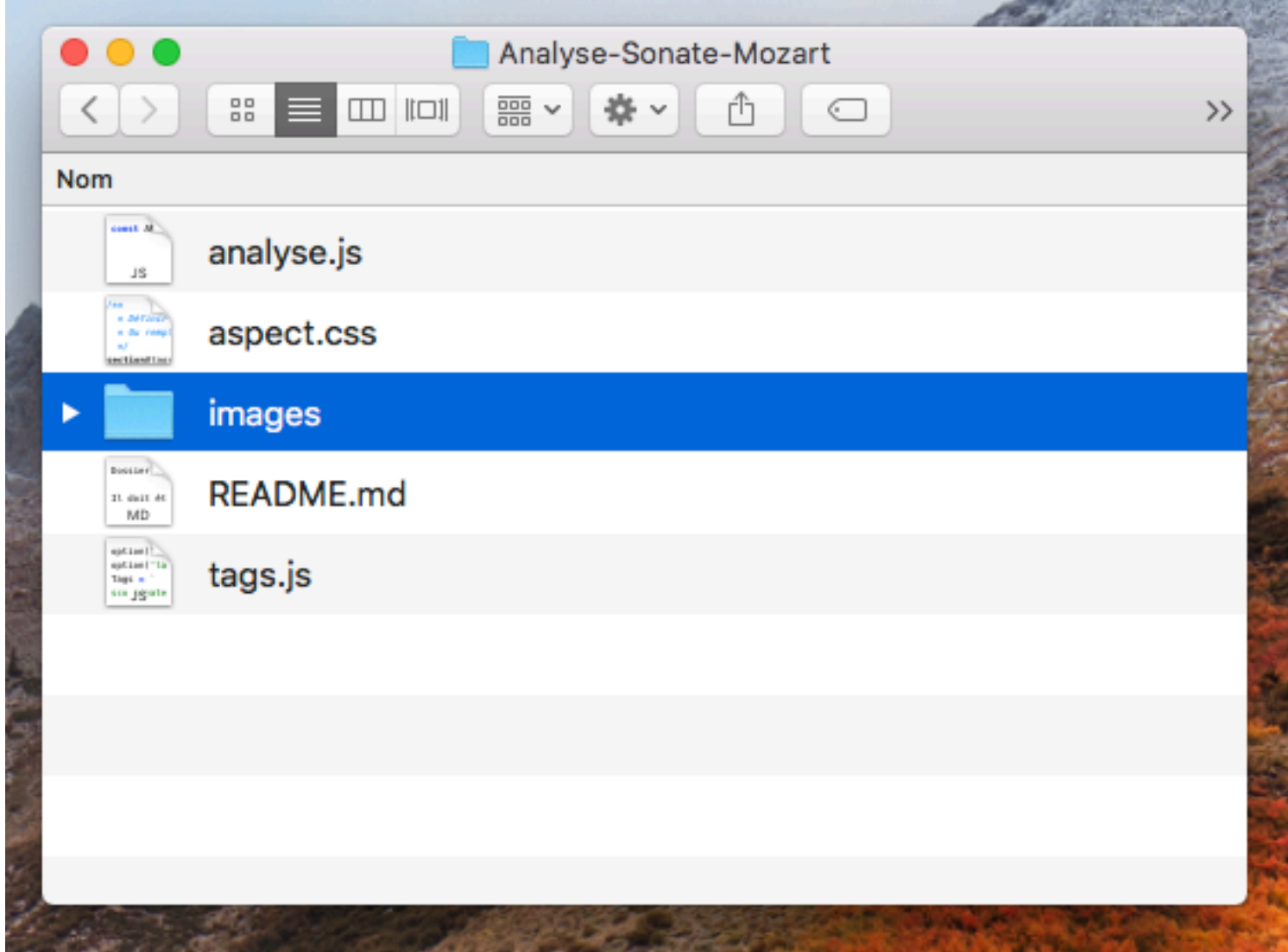
Voyons-en rapidement le contenu.



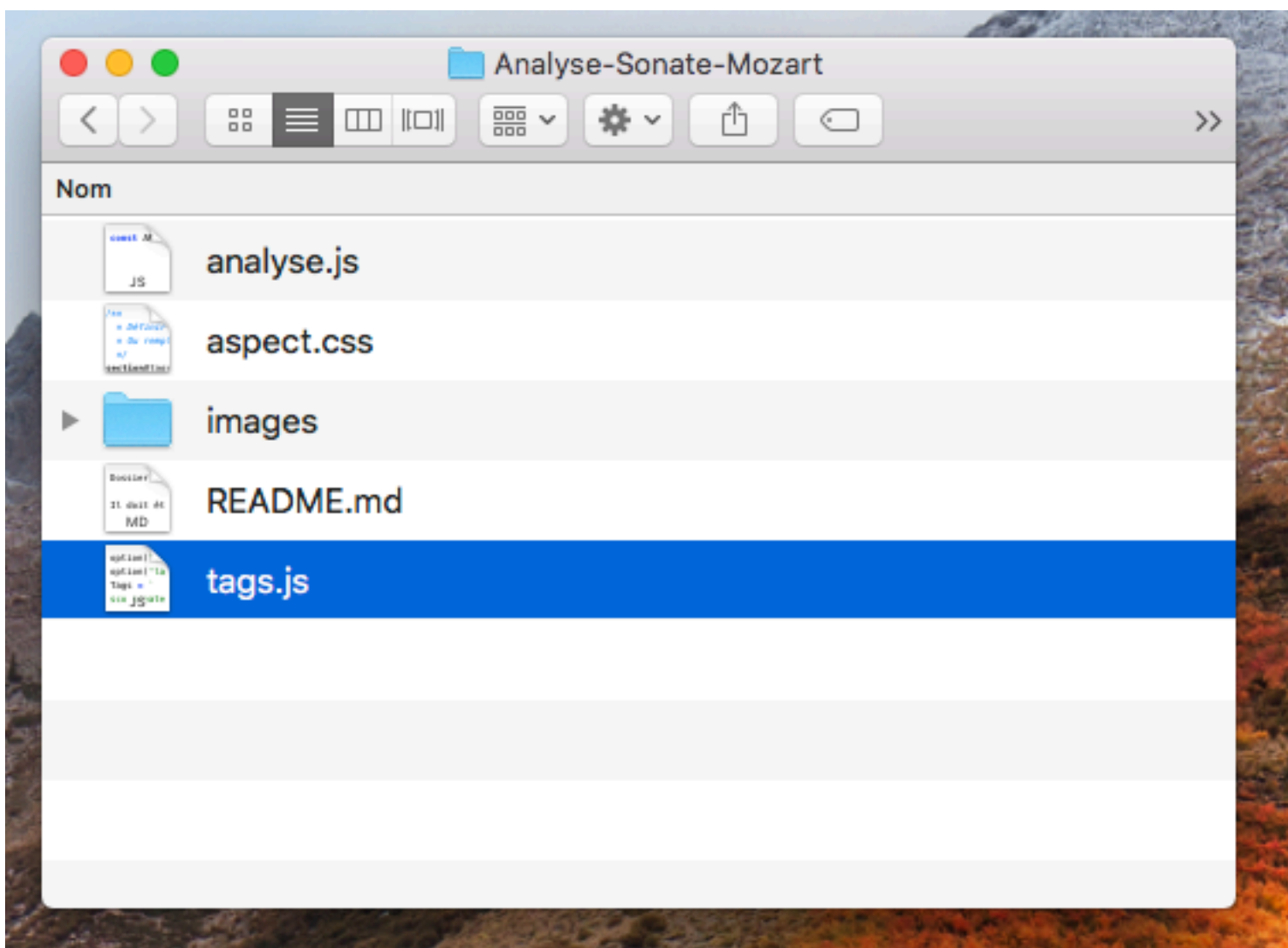
On trouve pour commencer un fichier « aspect.css », que vous ne toucherez pas au départ, et qui permet de rectifier l’aspect des analyses pour obtenir la présentation idéale souhaitée.



On trouve en dessous 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 systèmes.

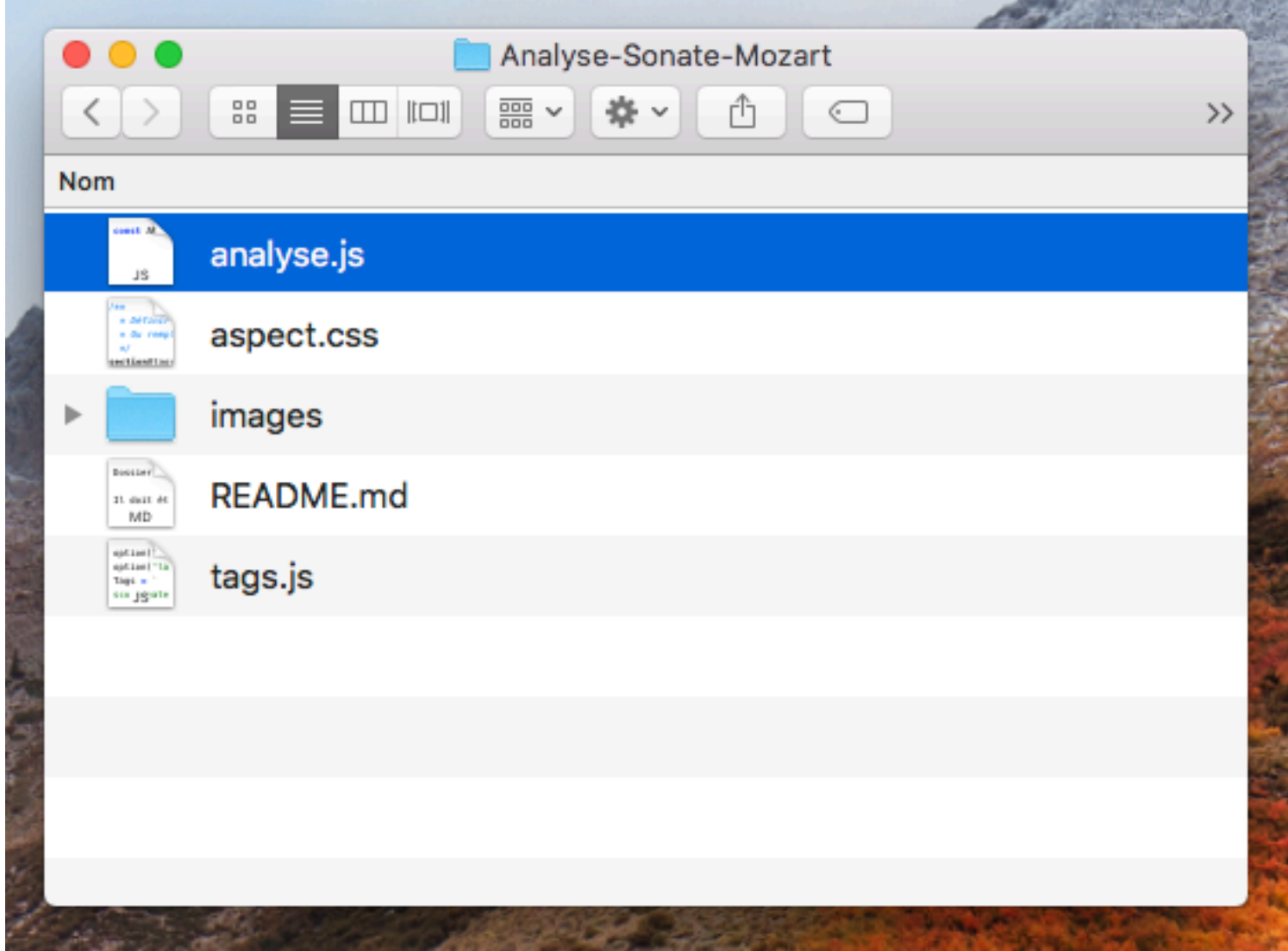


On trouve le fichier le plus important, le fichier « *tags.js* » qui va contenir la définition précise de l'analyse.

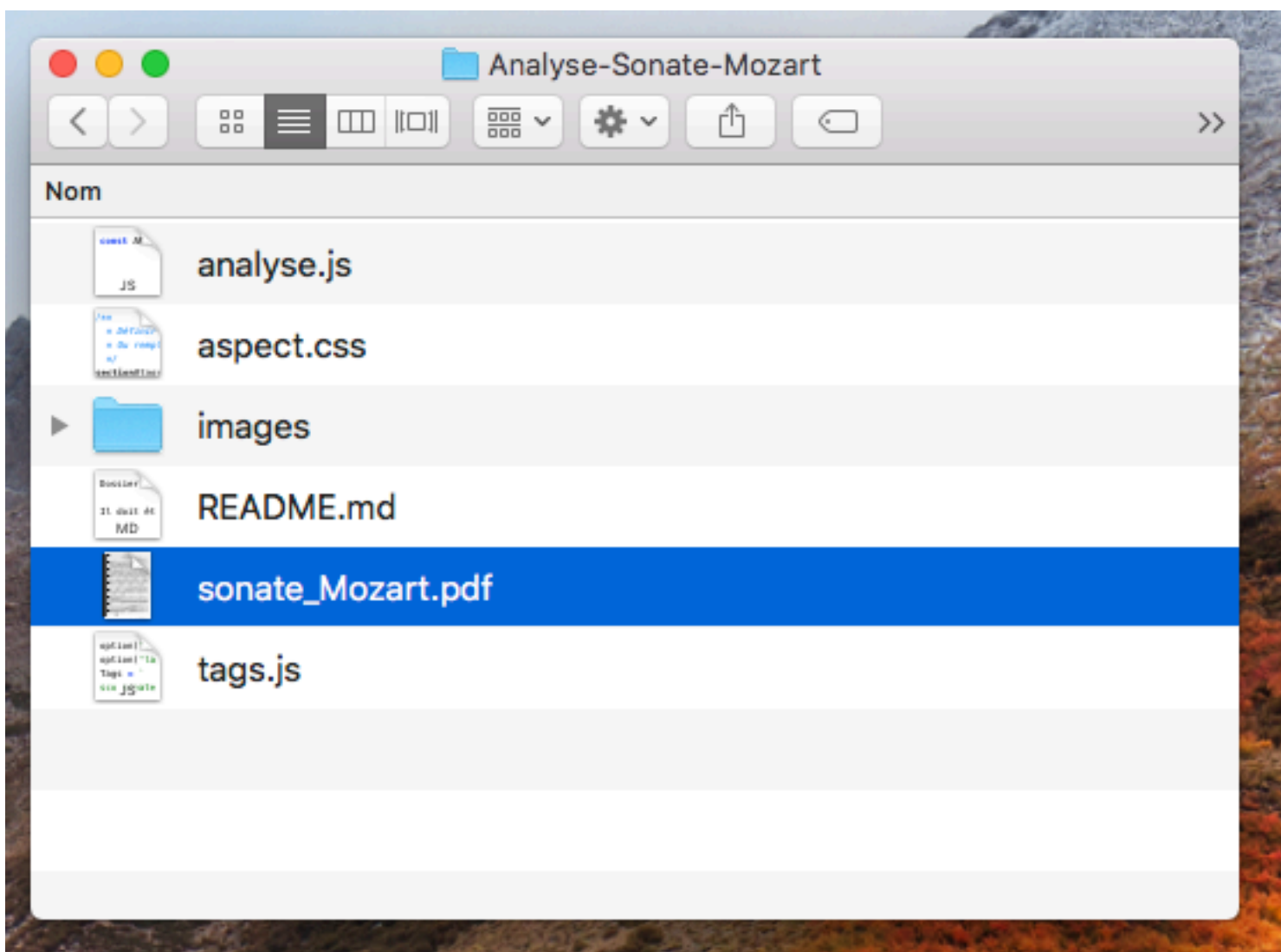


Nous aurons à y revenir en détail très vite.

On trouve aussi un fichier `analyse.js` qu'il suffit, pour activer cette analyse, de glisser à la racine du dossier **MuScaT** en remplacement du fichier qui s'y trouve déjà.



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



Mettre l'analyse en analyse courante

Pour faire de cette nouvelle analyse l'analyse courante, il y a deux solutions.

Soit on édite le fichier `MuScat/analyse.js` et on met le nom de notre dossier d'analyse :


```
const ANALYSE = "Analyse-Sonate-Haynd" ;
```

Soit on glisse le fichier `analyse.js` qui se trouve dans notre dossier d'analyse à la racine du dossier de l'application, en remplacement du fichier qui s'y trouve déjà.

À vous de choisir la solution qui vous semble le plus pratique. En tout cas, la seconde est la plus sûre, les erreurs de typo sont impossibles.

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 « taggués » 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 si vous êtes sur Mac, vous avez beaucoup plus simple : utiliser l'application Aperçu et la capture d'image par portion (CMD MAJ 4).

Pour une version détaillée et illustrée de la procédure, je vous renvoie à [ma chaîne YouTube](#) [[TODO: Mettre adresse de la vidéo]]. Je l'explique rapidement seulement ici.

- 🌀 modifier le dossier de capture (le dossier où seront enregistrées les captures d'écran) en passant par le [Terminal](#) :

```
> 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) puis glisser le dossier **MuScaT** sur la fenêtre de Terminal. Ensuite, taper `./utils/chan[TAG]` (sans oublier l'espace) puis faire glisser le

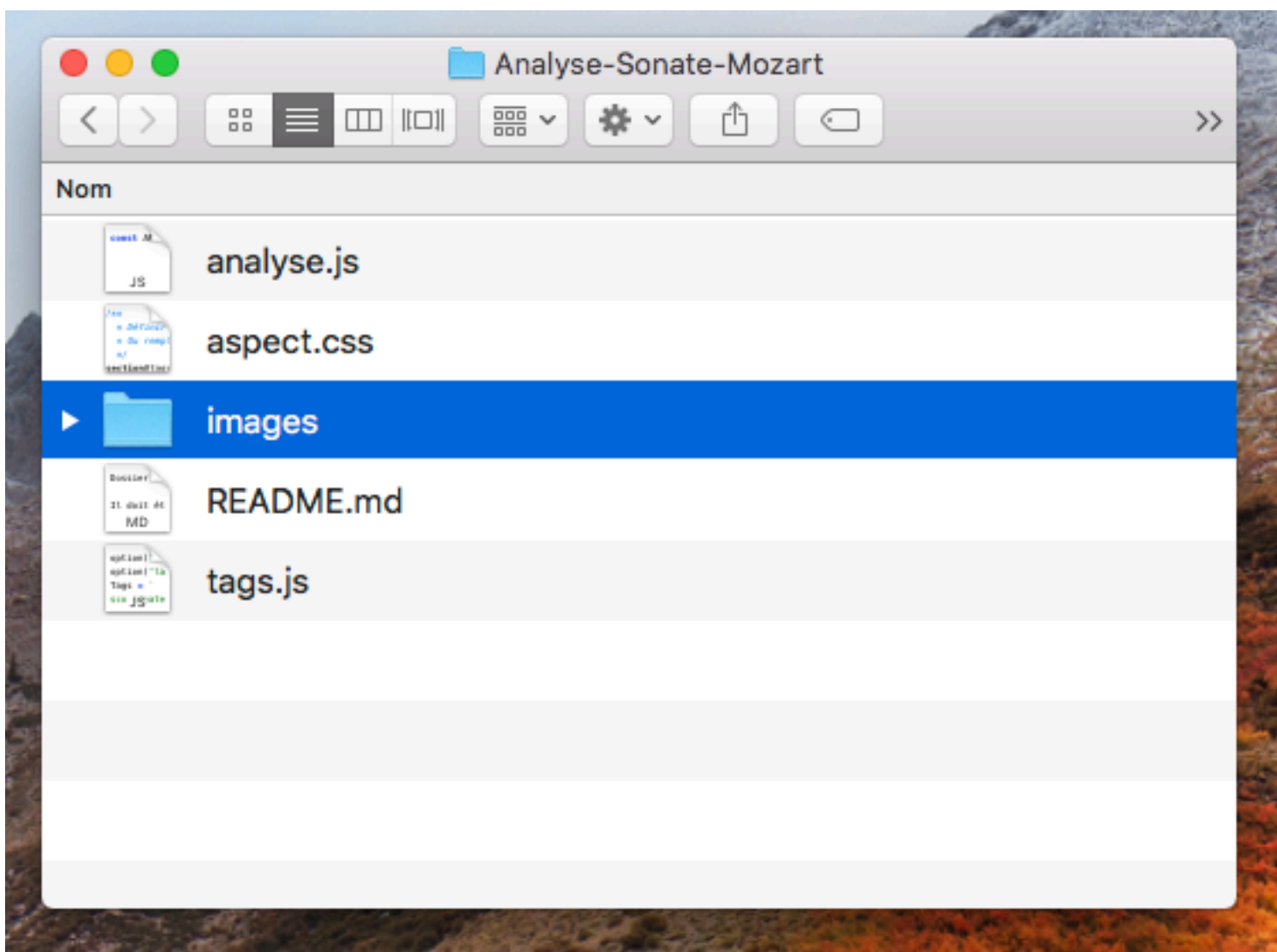
dossier où mettre les images sur la fenêtre de Terminal.

- ⌘ ouvrir le PDF dans Aperçu,
- ⌘ activer la combinaison de touches CMD-MAJ-4,
- ⌘ sélectionner le système,
- ⌘ recommencer ces opérations pour chaque système.

Noter qu'il est extrêmement simple d'affiner très précisément le découpage d'une image :

- ⌘ ouvrir l'image dans Aperçu,
- ⌘ dessiner un rectangle à la souris,
- ⌘ régler ses "poignées" pour obtenir exactement la découpe voulue,
- ⌘ jouer la combinaison CMD-K,
- ⌘ enregistrer l'image.

Quelle que soit la méthode adoptée, on place obligatoirement la ou les images dans le dossier `_analyses_ /<Nom analyse> /images/`.



Inscrire les images-systèmes

On ouvre ensuite son fichier `monAnalyse/_tags_.js`. C'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.

```

38
39
40  Tags = `
41  # Définir ci-dessous vos tags (aide: [TODO])
42  # -----
43
44  # Premier système
45  # -----
46
47  score ligne-01.png x=2 y=100
48
49  # Deuxième système
50  # -----
51
52  score ligne-02.png x=2 y=203
53
54  # Troisième système
55  # -----
56
57  score ligne-03.png x=2 y=269
58
59  `;
60
61
62
63
64
65
66
67
68

```

Vous devez ouvrir ce fichier en texte simple, c'est-à-dire ne surtout pas utiliser de traitement de texte, ou alors en s'assurant d'exporter le fichier final au format « SimpleText » (.txt).

Dans ce fichier `_tags_.js` On définit d'abord les images de la partition, en ajoutant des commentaires pour pouvoir se retrouver, plus tard, lorsque le fichier deviendra conséquent. Par exemple :

```

// 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.

Définir tous les éléments de l'analyse

L'élément graphique de base de l'application **MuScaT** est le « TAG » (comme on en parle sur les murs des villes). Une analyse avec **MuScaT** consiste donc à « tagguer » une partition (remarquez que les partitions elles-mêmes, ou les images de leurs systèmes, sont aussi des « TAGs »). C'est la raison pour laquelle le fichier qui va les définir s'appelle `_tags_.js`.

On définit tous les autres éléments graphiques, tous les *tags* (cf. pour le détail de la procédure, voir [Composition d'un tag](#)) : marque de parties, accords, chiffrages, numéros de portée, degrés de la gamme, cadences, textes divers, etc.

Chacun des éléments, chaque « tag », va être représenté dans le code par une unique ligne.

Une image de système pourra être :

```
Tags = `  
// ... d'autres données ici  
  
score systeme-1.png x=50 y=3098  
  
// ... d'autres données là  
`;  
`;
```

Une modulation pourra être inscrite par :

```
Tags = `  
// ... d'autres données ici  
  
mod G_min x=150 y=539  
  
// ... d'autres données là  
`;  
`;
```


Le mieux est de s'arranger pour définir ces tags à peu près en fonction des positions sur la table d'analyse (i.e. sur l'analyse). Si une cadence doit se produire sur le troisième système, il vaut mieux la définir après la ligne insérant l'image de ce troisième système (remarquez cependant qu'il n'y a aucune obligation là-dessus, vous pouvez aussi, rassembler tous les accords d'un côté, toutes les cadences de l'autre, etc. à votre guise).

Activer l'analyse

Pour activer cette nouvelle analyse, nous allons donc copier-coller son fichier `analyse.js` à la racine du dossier de l'application **MuScaT**. Assurez-vous d'abord que dans ce fichier, c'est bien votre analyse qui est indiqué. Si vous avez simplement dupliqué le dossier Template, l'analyse s'appellera `Template` et c'est elle qui s'ouvrira.

Si votre analyse s'appelle (son dossier) `monAnalyse`, alors le fichier `analyse.js` doit impérativement contenir :

```
const ANALYSE = "monAnalyse";
```

Vous pouvez ensuite, par exemple, sélectionner le fichier `analyse.js` dans le Finder ou sur votre bureau, sélectionner le dossier **MuScaT** et coller. Le bureau vous demandera de confirmer le remplacement (un autre fichier `analyse.js` existe déjà à cet endroit, celui de l'analyse courante).

Pensez en tout cas à faire une duplication du fichier, ne le glissez pas, il risquerait d'être perdu.

Seconde solution, vous pouvez également éditer le fichier **MuScaT/analyse.js** principal et mettre le nom de votre dossier dans la constante `ANALYSE`.

```
// Dans Le fichier MuScaT/analyse.js principal
```

```
const ANALYSE = "  
<METTRE_ICI_LE_NOM_DU_DOSSIER_DE_VOTRE_ANALYSE>"
```

Attention ! Il est capital de ne pas mettre d'espaces dans ce nom, ou ça ne fonctionnera pas. Il en va de même qu'une adresse dans votre navigateur.

Noter que les heureux possesseurs de Mac peuvent utiliser un script

permettant d'activer très simplement n'importe quelle analyse. Il suffit de rejoindre, dans l'application Terminal, le dossier de **MuScaT** et de taper `> ./utils/analyse.rb`. Nous y reviendrons en parlant des [utilitaires](#).

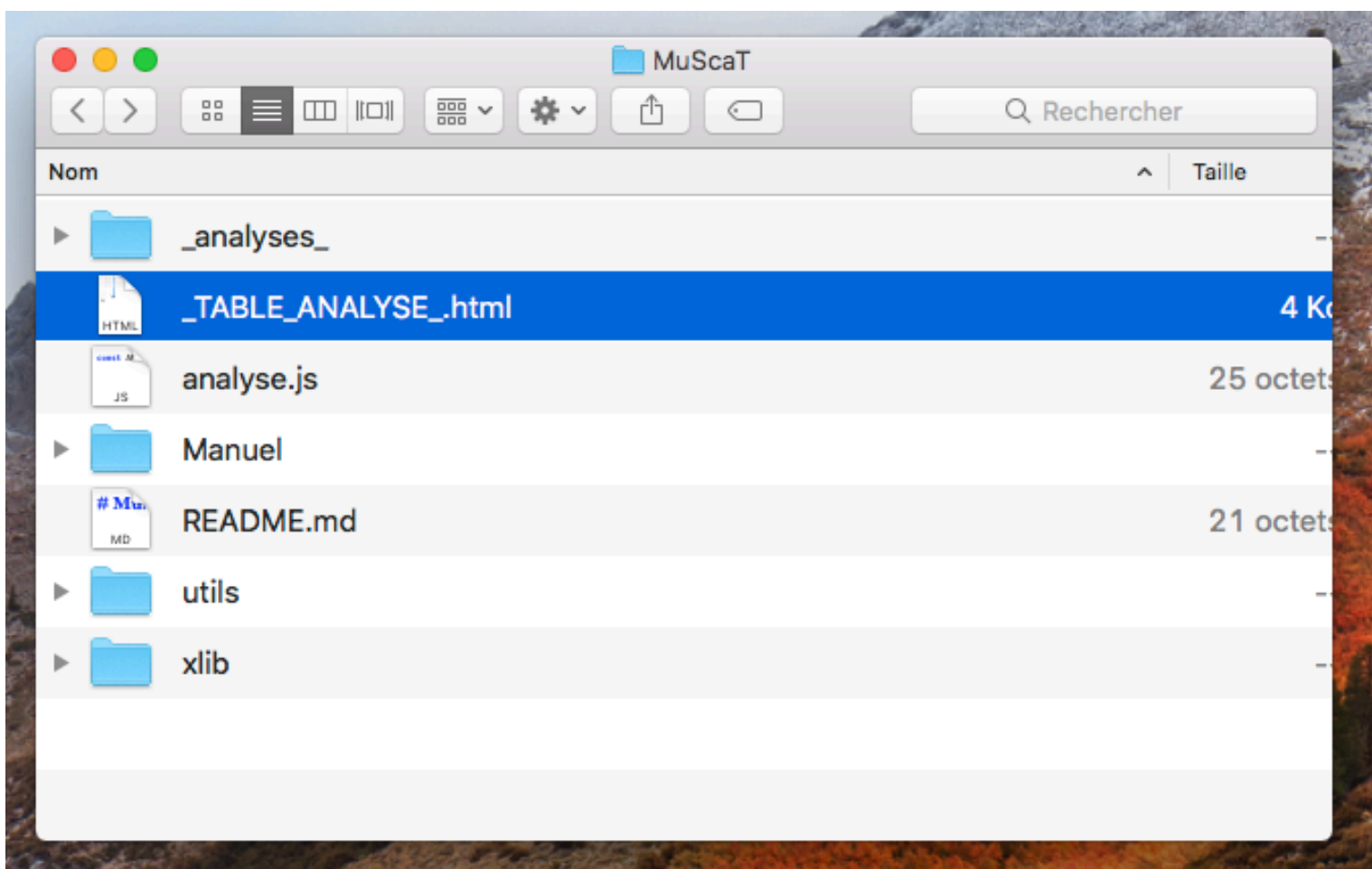
Et, plus loin, si [vous avez installé un alias](#) (par exemple `mus`), il vous suffit de taper :

```
> mus analyse
```

... et de choisir dans la liste l'analyse à ouvrir.

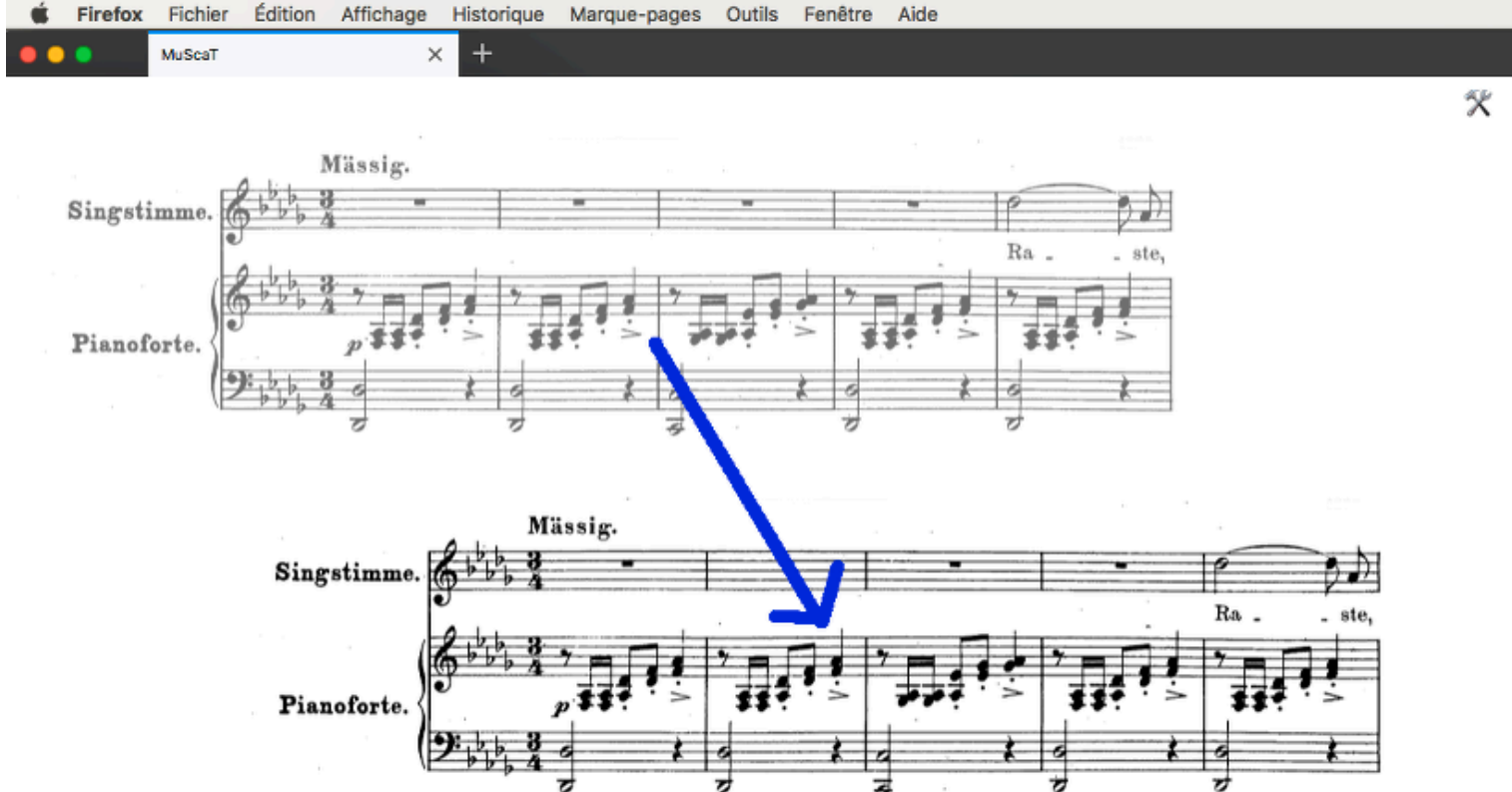
Positionnement des éléments graphiques

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



Firefox est tout à fait déconseillé : vous ne parviendrez pas à imprimer votre analyse si elle tient sur plusieurs pages. Chrome est l'application idéale.

On peut placer les éléments aux bons endroits simplement en les déplaçant à la souris, ou avec les flèches de son clavier. On peut en ajouter des nouveaux en dupliquant les lignes de code ou les ajoutant explicitement dans le 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.

Ajuster les éléments en fonction de la sortie

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

Donc demandez l'impression, faites les réglages nécessaires (taille du papier, marges, etc.) et ajustez la position des systèmes en fonction des sauts de page que vous voyez dans l'aperçu de l'impression (pour ce faire, Google Chrome est idéal).

Ajout du titre, compositeur, etc.

Vous pouvez même placer dès à présent les titres et compositeur 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` :

```
// 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 (le fichier `aspect.css` général) qui s’en charge. Mais vous pouvez tout à fait les déplacer pour les ajuster à votre guise, ou choisir un autre thème.

Lignes repères

Pour faciliter l’alignement des TAGs — par exemple l’alignement des dernières mesures de fin des systèmes — on peut utiliser des lignes de repère. 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 );
```

Récupérer le code final

Si l’on a travaillé dans le champ de texte à côté de la table d’analyse, on doit copier le code final dans le fichier `_tags_.js`, au risque de perdre tous les changements. Pour se faire, on clique sur le bouton des outils

— en haut à gauche — et on demande à mettre le code complet dans le presse-papier. On colle ce code dans le fichier `_tags_.js`, en remplaçant l'intégralité de son contenu.

Imprimer l'analyse en PDF

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

- 🌀 dans Chrome, demander l'impression,
- 🌀 dans la fenêtre qui s'ouvre, choisir, dans le menu en bas à gauche : « Imprimer au format PDF » ou autre indication similaire.

(sur PC, enregistrer la page au format HTML et utiliser un outil de transformation des pages HTML en PDF)

Et voilà

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

L'interface

- 🌀 La boîte à outils
- 🌀 Le champ de code

Voyons un peu de quoi est constitué l'interface de **MuScaT**, que nous appelons la « table d'analyse ».

Cette table, c'est déjà la surface de la page elle-même.

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.



Le champ de code

Si l'option **guides** 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).

Composition d'un tag

Voyons plus en détail comment se compose une ligne du fichier `_tags_.js`, une ligne définissant un *tag* ou une partition.

Un *TAG* — image de la partition comprise — se compose d'une ligne dans le fichier de données.

Cette ligne a le format général suivant :

```
<nature>[ <contenu>][ <coordonnées>][ <options, type>]
```

Par exemple, pour une cadence (nature = 'cadence') de « V I » (contenu = 'V_I') qu'on veut placer à 200 pixels depuis le haut (coordonnée y = 200) et 100 pixels de la gauche (coordonnées x = 100), de type « cadence parfaite » (type = 'parfaite'), on insèrera dans son fichier `_tags_.js`, sous la définition de l'image (« score ») :

Tags = `

```
score ma_partition.jpg y=100 x=10
```

```
cadence V_I type=parfaite y=200 x=100
```

```
modulation G_min x=200 y=100
```

```
`;  
;
```

Une « nature » de TAG (le premier mot), peut toujours être exprimé 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  
  
cad V_I type=parfaite y=200 x=100  
  
mod G_min x=200 y=100  
  
`;  
;
```

L'intégralité des natures de TAG [est détaillé ici](#).

Vous observerez que tout de suite après la création, un identifiant est ajouté à toutes les lignes, mêmes les lignes vides. Il contient de ne pas y toucher, sous peine de voir son travail réduit à néant.

Ainsi, le code ci-dessous, au final, donnera :

```
// Contenu intégral du fichier _tags_.js  
option('code'); // pour voir ce code à côté de la  
partition
```

```
Tags = `  
sco ma_partition.jpg id=2 y=100 x=10  
#3#  
cad V_I type=parfaite id=4 y=200 x=100  
#5#  
mod G_min id=6 y=100 x=200  
  
`;  
;
```

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 = `  
;
```

```
<version 3 lettres|normale> <contenu|source> <valeur x>  
  <valeur y>  
`;  
`;
```

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
```

Les Images

Il existe trois mots clés pour indiquer la nature d'une image, mais ils sont identiques en réalité : `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 `MuScaT/_analyses_<mon analyse>/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` fera 200 pixels de large, `image-2` fera 10 centimètres de large et `image-3` sera mise à 50% de sa largeur.

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 10 à 15 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 nommés 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
```

Nous vous invitons vivement à commencer par cette opération avant l'insertion de toute autre marque sur la partition.

Quand **MuScaT** place les images sur la table d'analyse, il les répartit pour obtenir l'aspect initial 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
`;
```

Notez la version raccourci de la nature du TAG : `sco` pour `score`.

Notez 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.

Grâce à l'option `espacement images` défini ci-dessus, chaque image (chaque système) sera séparé 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.

Nature des tags

Détaillons toutes les natures de TAGs qu'on peut utiliser.

Dans la ligne, le premier mot définit la `<nature>` du tag.

Tag	Description
<i>partition</i> <i>score</i> <i>sco</i> <i>image</i>	<i>image <source> x=... y=... z=...</i> Exemple : <i>score monimage.src x=200 y=20 w=170mm</i> <i>L'image doit se trouver dans le dossier <i>images</i> de l'analyse.</i> <i>Pour le détail : Les Partitions</i>
accord chord acc	Écriture d'un accord au-dessus de la partition. <code>accord <nom> x=... y=...</code> P.e. : <code>acc E_min x=100 y=200</code> Pour le détail : Les Accords
harmonie harmony chiffrage har	Écriture d'un chiffrage sous la partition.
box boite	Dessine une boite à l'écran. Ces boites permettent de masquer des éléments de la partition. Bien que

ces boîtes apparaissent en gris sur la table d'analyse, elles seront invisibles dans le document PDF final ou l'impression.

mesure `mesure <nombre> x=... y=...`
mes Exemple : `mes 13 x=100 y=234`
Alias : 'measure'

modulation `modulation <Ton[/sous-texte]> x=HH y=VV
h=HH`
mod Exemple : `modulation D_Maj/Sous-dom.
x=100 y=100 h=60`
« h », ici, permet de définir la
longueur du trait qui
rejoint la partition (le trait
vertical).

cadence `cadence <degré accord> type=<type
cadence> x=... y=... w=...`
cad Exemple : `cadence I type=italienne
w=200 x=12 y=100`

ligne `ligne <type ligne> x=... y=... w=...`
lig Exemples : `ligne U w=120 x=100 y=50`
 `line |---| w=50 x=100 y=50`
Alias : 'line'

degré `degre <indice> x=... y=...`
deg Exemple : `degre 5 x=100 y=120`
Alias : 'degree'

texte `texte <contenu> x=... y=... type=...`
tex Exemple : `texte Exposition x=100 y=50
type=partie`
Alias : 'text'

Contenu du tag (second mot)

Le seconde « mot » définit le plus souvent le contenu textuel ou, pour les images, le nom du fichier dans le dossier `images` de l'analyse. C'est aussi, souvent, un accord ou son chiffrage.

On peut par exemple écrire un texte quelconque à une position quelconque avec la ligne :

Tags = `

texte Et_si_j'étais_un_texte_quelconque x=300 y=400

^ ;

Remarquez comme les espaces ont été remplacées par des tirets plats (qu'on obtient sur Mac avec la combinaison de touches Maj— touche majuscule et tiret).

Ce deuxième « mot » de la ligne sert aussi par exemple à définir le type des lignes à obtenir (cf. [Dessiner des lignes](#)).

Autres données de la ligne

Les deux autres informations capitales sont les positions verticale et horizontale du tag à poser (ou de la partition).

NOTE IMPORTANTE : dans votre fichier `_tags_.js`, ces valeurs peuvent dans un premier temps être approximatives, et seront affinées directement à l'écran.

On définit la position verticale avec `y=` et la position horizontale avec `x=`, comme nous l'avons vu dans les exemples précédents. Le nombre est exprimé en pixels.

Pour les lignes et les cadences par exemple, on peut définir aussi la largeur avec la lettre « `w` » qui signifie « `width` » (largeur) en anglais : `w=200`. Le nombre correspond là aussi au nombre de pixels, mais il peut être exprimé avec une autre unité, notamment le pourcentage — ce qui n'est pas possible avec `x` et `y`.

Ensuite, on peut définir certaines choses comme le « type » du tag. On l'a vu pour la cadence, par exemple. Les autres tags pouvant définir leur type sont le `texte` ou la `ligne` (bien que la `ligne` se définit plutôt par son contenu).

Écrire des textes

Ce que l'on appelle les « textes », ici, ce sont tous les textes hors des accords, modulations, chiffage, 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
```

Les types de textes

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

- ⌚ Les parties
- ⌚ Les modulations
- ⌚ Les mesures

type	anglais	Description

partie	part	Titre de partie, comme
Exposition ou Coda		
mesure	measure	Numéro de mesure, dans un
carré.		
modulation (id.)		Marque de modulation, en
haut de partition,		inclinée.

Les parties

Les marques de partie s’indiquent avec le tag `partie` (ou `par` ou `part`).
Ce sont des textes dans des boites inclinées qui ont cet aspect :



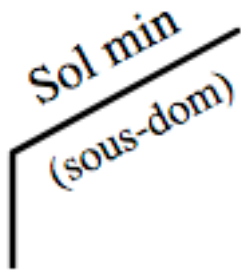
Les mesures

Les numéros de mesure, s’il ne sont pas indiqués sur la partition elle-même, peuvent être ajoutés à l’aide du tag `mesure` (ou `measure`, ou `mes`), suivant du numéro de mesure puis des coordonnées.

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
```

Dessiner des lignes

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

Le premier élément définit le `type` de la ligne. On trouve les types suivants.


U ou ____ avant/après	Ligne inférieure et trait vertical
N ou --- avant/après	Ligne supérieure et trait vertical
L ou ____ avant	Ligne inférieure et trait vertical
K ou --- avant	Ligne supérieure et trait vertical
V ou ____ vertical après	(Virgule) Trait inférieur et trait
^ ou --- et trait vertical après	(Virgule inversée) Trait supérieur

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

Opérations sur les tags

- 🌀 Verrouiller les tags
- 🌀 Grouper et dégroupier des tags

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 un astérisque, un rond (ALT #) ou même un  au tout début de sa ligne (suivi ou non par une espace).

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 modifiée dans le code (sa position, son contenu, etc) pour un ajustement très précis.

Pour deverrouiller 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 les x tags sélectionnés » dans [la boîte à outils](#) ou jouez la combinaison clavier CMD G (Ctrl G sur Windows)

Pour dégroupier :

- 🌀 sélectionner un groupe en sélectionnant un de ses éléments
- 🌀 activer le bouton « Dégroupier les tags » dans [la boîte à outils](#) ou jouez 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 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.

- ⌚ 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),
- ⌚ 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.

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 instantannément vos fichiers.

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

Avec sélection rectangulaire dans Aperçu (Mac)

Une méthode qui ressemble à la précédente et 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 Gimp présente une précision de découpage inégalable, l'application 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, je vous renvoie au manuel d'utilisation de l'application.

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'animation
- 🌀 Pause de l'animation
- 🌀 Réglage de la vitesse 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 complexe.

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.

Pause de l'animation

Pour une utilisation « en live », comme un power-point, il peut être intéressant de mettre l'animation en pause, c'est-à-dire de l'arrêter jusqu'à ce qu'une touche soit pressée. Pour cela, on utilise tout simplement la ligne `// PAUSE` à l'endroit où l'on veut que ça se fasse.

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).

Options

- ⌚ Options de la langue
- ⌚ Option « code à côté »
- ⌚ Option « découpe image »
- ⌚ Option « lignes de repère »
 - ⌚ Position des lignes repères
- ⌚ Option « espacement images »
- ⌚ Option « marge haut »
- ⌚ Option « marge gauche »
- ⌚ Vitesse de l'animation

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 place des autres, ou utiliser plusieurs fois la méthode `option`. Les deux 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 );
```

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 );
```

Option « langue »

Option : `lang`, `langue`

Type : les deux lettres de la langue, par exemple `fr` (français) ou `en`

(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 beside`, `code à côté`

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 : `crop image`, `découpe image`

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 « 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 (#position_lignes_reperes)

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`

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 « Espacement entre images »

Option : `espacement images, space between scores`

Type : non booléen, la valeur est le 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 : non booléen, la valeur est le 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.

Option « marge gauche »

Option : `marge gauche`, `left margin`

Type : non booléen, la valeur est le 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.

Vitesse de l'animation

Option : `vitesse animation`, `animation speed`

Type : un nombre de 1 à 100.

1 correspond au plus lent, 100 au plus rapide.

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

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 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 (commande `cd`) le dossier de l'application MuScaT,
- 🌀 se placer dans le dossier utilitaires (`cd utils`)
- 🌀 taper `./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.

Changement du dossier des captures écran (Mac)

Par défaut, les captures d'écran sont enregistrés 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 (commande `cd`) le dossier `utils` de l'application MuScaT,
- 🌀 taper `./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 (commande `cd`) le dossier `utils` de l'application MuScaT,
- 🌀 puis, au choix :

- ⌚ taper `./create.rb -h` et la touche Entrée pour tout savoir du script,
- ⌚ taper `./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 `./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 (commande `cd`) le dossier `utils` de l'application MuScaT,
- ⌚ puis, au choix :
 - ⌚ taper `./analyse.rb -h` et la touche Entrée pour tout savoir du script.
 - ⌚ taper `./analyse.rb` pour obtenir la liste des analyses et en choisir une,
 - ⌚ taper `./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.

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 bash profile 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.

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 ».

correspondances linguistiques

Quand on utilise l'alias ci-dessus, on peut utiliser des termes dans sa langue.

<i>Anglais</i>		<i>Français</i>
create open		créer ouvrir
rename_images		renommer_images
change_folder_captures		change_dossier_captures
<i>Espagnol</i>	<i>Allemand</i>	<i>Mandarin</i>

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. CMD ESPACE, puis tapez les premières lettres.