

Jonathan Racaud

Music Sheet Writer

Music Sheet Writer

Guide D’installation

Objectifs du document

Résumé

Glossaire

***– M –***

Mot : Définition

***– S –***

Signification de l’Acronyme (SDA) : Définition

Description du document

|  |  |
| --- | --- |
| Titre | Music Sheet Writer : Guide D’installation |
| Date de création | 08/04/2016 |
| Date de publication | 08/04/2016 |
| Auteur | Jonathan Racaud |
| Responsable | Jonathan Racaud |
| E-mail | musicsheetwriter\_2017@labeip.epitech.eu |
| Sujet | Guide D’installation |
| Version du modèle | 1.0 |

Tableau des révisions

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Auteur | Section(s) | Commentaire |
| 04/08/2016 | Jonathan Racaud | Toutes | Création du document |
| 22/04/2016 | Julien Blondeel | Logiciel | 1ere ébauche |
| 23/04/2016 | Julien Blondeel | Logiciel | Modifications suite au 1er rendu + correction orthographe |

Sommaire

[1. Logiciel 1](#_Toc449184873)

[1.1. Qt 1](#_Toc449184874)

[1.2. GuidoEngine 1](#_Toc449184875)

[1.2.1. Windows 1](#_Toc449184876)

[1.2.2. MacOS 3](#_Toc449184877)

[1.3. GuidoQt 4](#_Toc449184878)

[1.3.1. Windows 4](#_Toc449184879)

[1.3.2. MacOS 6](#_Toc449184880)

[1.4. Midisharelight 7](#_Toc449184881)

[1.5. Ajouter une librairie à un projet dans Qt 8](#_Toc449184882)

[1.6. Fichier de police guido2 10](#_Toc449184883)

[2. Site internet 10](#_Toc449184884)

[3. Application Android 10](#_Toc449184885)

[4. Application iOS 10](#_Toc449184886)

[5. Application Windows Phone 10](#_Toc449184887)

Liste des Tableaux

**Aucune entrée de table d'illustration n'a été trouvée.**

Liste des Figures

[Figure 1: Contenu du dossier suite à l'exécution de CMake sous Windows 1](#_Toc449109200)

[Figure 2: Génération de la librairie depuis Visual Studio 2013 2](#_Toc449109201)

[Figure 3: Contenu du dossier Debug suite à la compilation sous Windows 2](#_Toc449109202)

[Figure 4: Contenu du dossier suite à l’exécution de CMake sous MacOS 3](#_Toc449109203)

[Figure 5: Lancer une compilation sous Xcode 3](#_Toc449109204)

[Figure 6: Contenu du dossier Debug suite à la compilation sous MacOS 4](#_Toc449109205)

[Figure 7: Qt Project file sous Windows 4](#_Toc449109206)

[Figure 8: Message d'erreur lors de la compilation de GuidoQt sous Windows 5](#_Toc449109207)

[Figure 9: Contenu du dossier suite à la compilation dans Qt 5](#_Toc449109208)

[Figure 10: Génération de la librairie GuidoQt dans Visual Studio 2013 6](#_Toc449109209)

[Figure 11: Qt Project File sous MacOS 6](#_Toc449109210)

[Figure 12: Fenêtre d'avertissement lors de la compilation de GuidoQt sous MacOS 7](#_Toc449109211)

[Figure 13: Ajouter une librairie externe 8](#_Toc449109212)

[Figure 14: Paramétrage de la librairie externe 9](#_Toc449109213)

[Figure 15: Exemple d'ajout de la librairie GuidoEngine sous Windows 9](#_Toc449109214)

[Figure 16: Librairie GuidoEngine dans le .pro 10](#_Toc449109215)

[Figure 17: Fichier de police guido2 10](#_Toc449109216)

# Logiciel

Le développement du logiciel requiert certains outils que voici :

* Qt version 5.5.1 ou plus
* Visual studio 2013 ou plus pour Windows
* XCode pour Mac OS

## Qt

Le développement du logiciel se fait à l’aide de l’API Qt qui permet le développement crossplateform. Pour installer Qt sur votre système, téléchargez l’installateur disponible ici : <https://www.qt.io/download-open-source/> . La procédure d’installation est relativement simple, il vous suffit de suivre les instructions.

## GuidoEngine

Le rendu graphique des partitions dans le logiciel ce fait à l’aide de la librairie GuidoEngine, vous pouvez retrouver une explication succincte ainsi que des exécutables utilisant cette API en suivant ce lien : <http://guidolib.sourceforge.net/> .

### Windows

1. Installer CMake : <https://cmake.org/download/> .
2. Décompresser le fichier rar qui se trouve sur le SVN dans la partie Software/guidolib
3. Créer un dossier dans source/build/Win64.
4. Ouvrir un terminal et accéder au dossier précédemment créé puis exécuter la commande suivante :



1. Aller dans le dossier source/build/Win64 et exécuter :

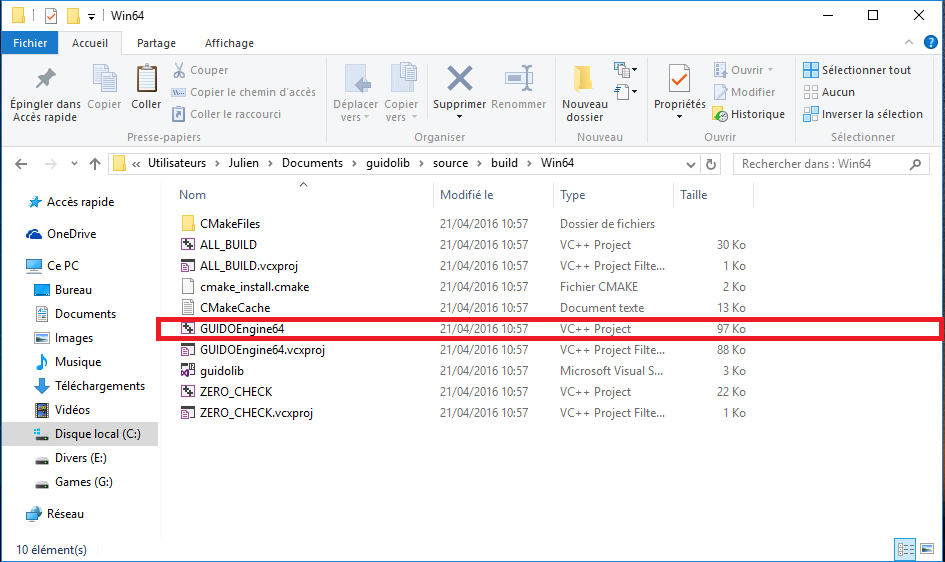


Figure 1: Contenu du dossier suite à l'exécution de CMake sous Windows

1. Dans Visual Studio, faites un clic droit sur votre projet, puis générer.

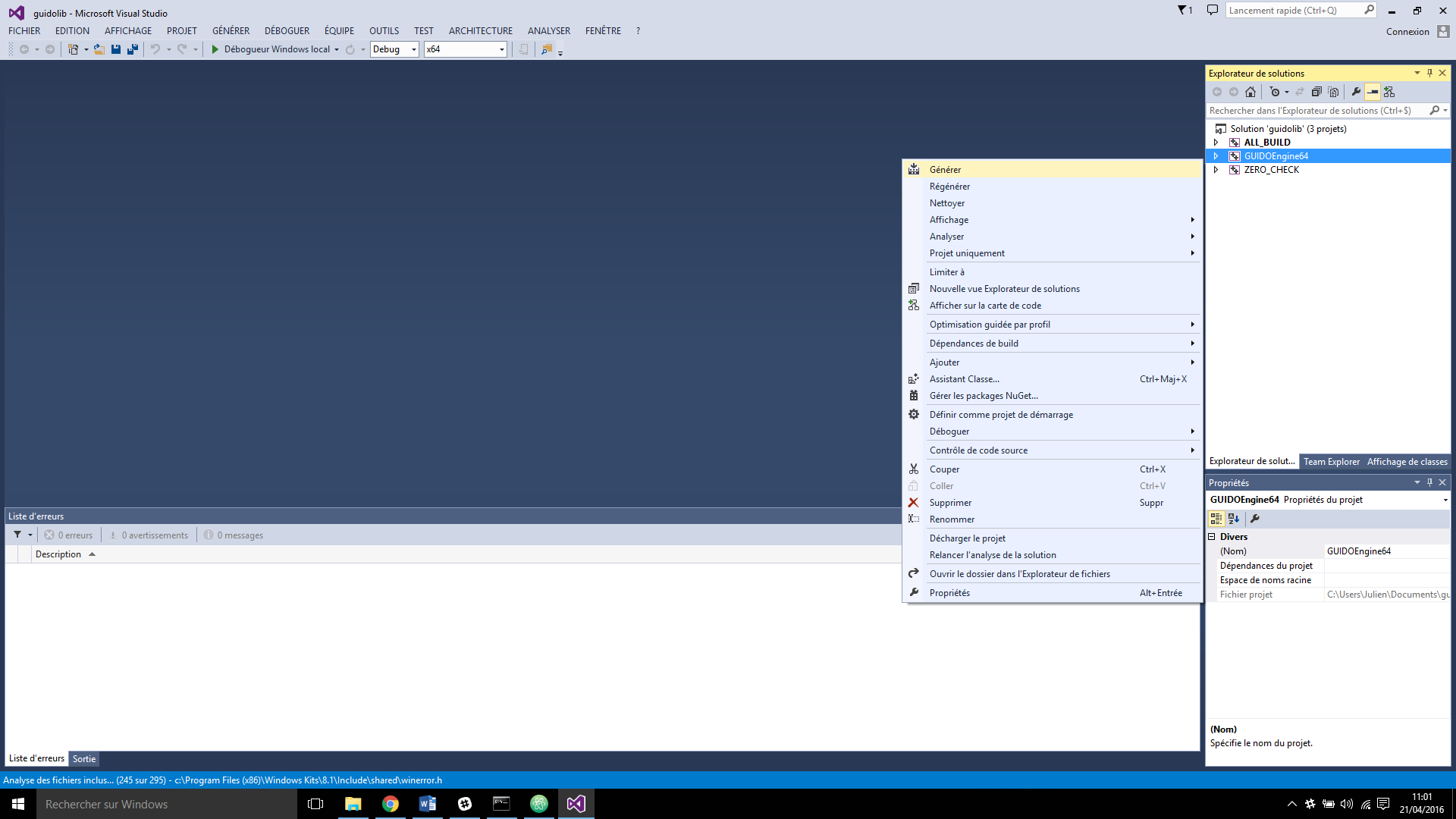


Figure 2: Génération de la librairie depuis Visual Studio 2013

1. Un dossier Debug est créé dans source/build/Win64, il contient la librairie compilée

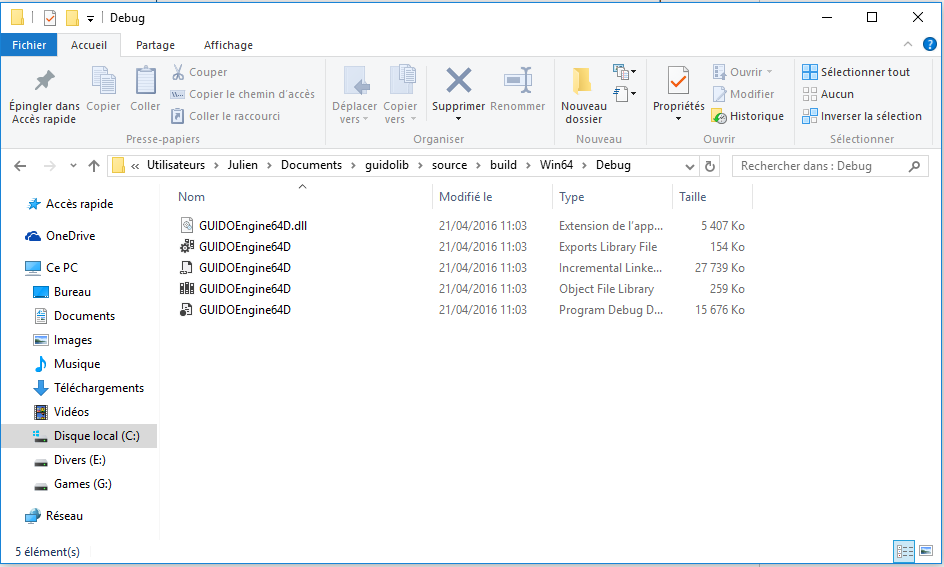


Figure 3: Contenu du dossier Debug suite à la compilation sous Windows

1. Créer un dossier du nom de GuidoEngine et y mettre le .lib et .dll ainsi que le dossier include se trouvant dans source/src/engine.

### MacOS

1. Installer CMake : <https://cmake.org/download/> .
2. Décompresser le fichier rar qui se trouve sur le SVN dans la partie Software/guidolib.
3. Créer un dossier dans source/build/MacOS.
4. Ouvrir un terminal et accéder au dossier précédemment créé puis exécuter la commande suivante :



1. Aller dans le dossier source/build/MacOS et exécuter :

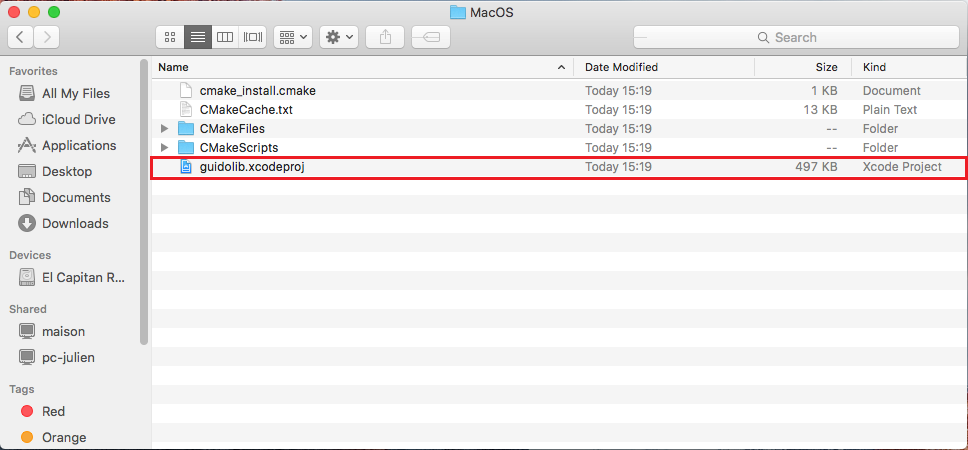


Figure 4: Contenu du dossier suite à l’exécution de CMake sous MacOS

1. Dans Xcode, lancer la compilation du projet.

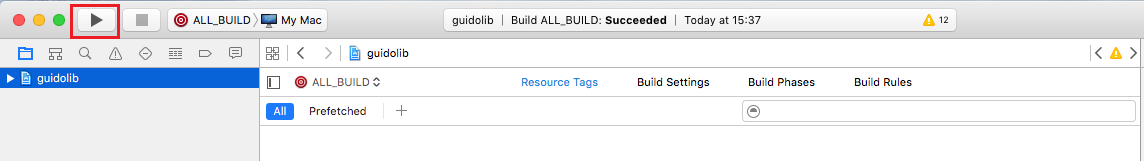


Figure 5: Lancer une compilation sous Xcode

1. Un dossier Debug est créé dans source/build/MacOS, il contient la librairie compilée.

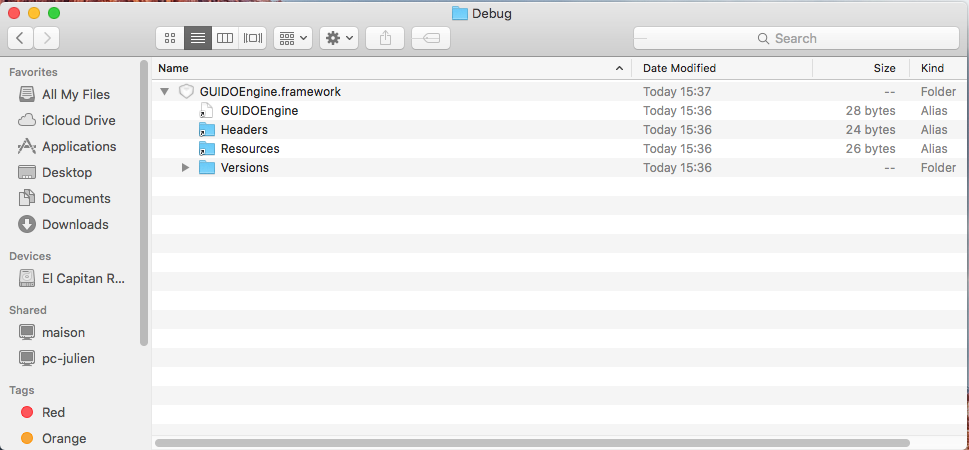


Figure 6: Contenu du dossier Debug suite à la compilation sous MacOS

1. Créer un dossier du nom de GuidoEngine et y mettre le GUIDOEngine.framework ainsi que le dossier include se trouvant dans source/src/engine.

## GuidoQt

Pour exécuter cette partie, Qt doit être installé sur votre système.

### Windows

1. Lancer le Qt Project file qui se trouve dans source/environments/Qt/libs/GuidoQt.

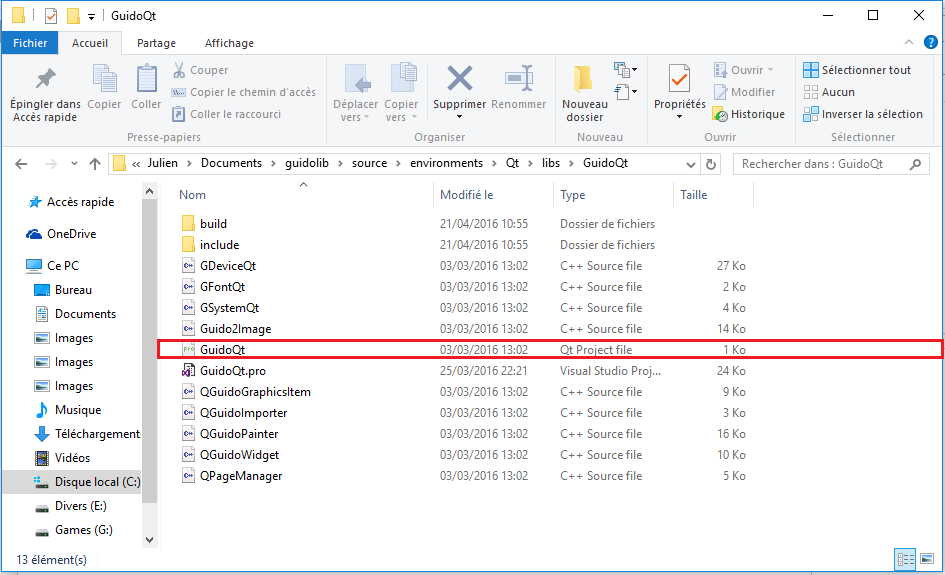


Figure 7: Qt Project file sous Windows

1. Exécuter le programme. Un message d’erreur indique qu’il n’est pas possible de compiler le programme néanmoins un dossier a été créé. Celui-ci se trouve normalement dans source/environments/Qt/libs et son nom commence par build-GuidoQT-… .

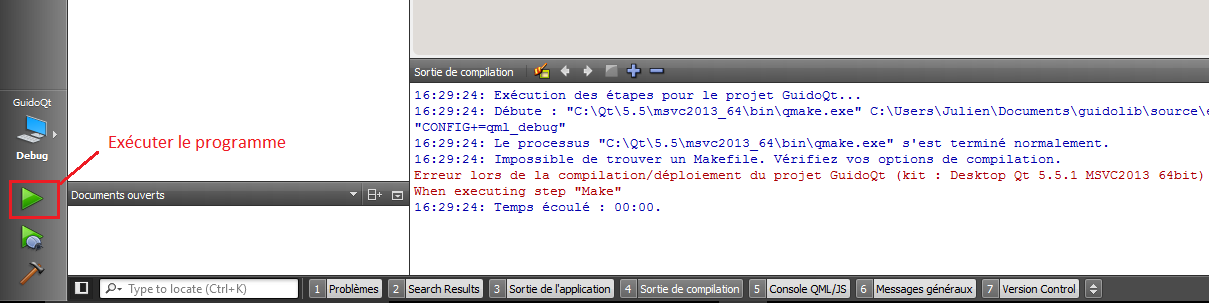


Figure 8: Message d'erreur lors de la compilation de GuidoQt sous Windows

1. Dans ce dossier, exécuter le VC++ Project du nom de GuidoQt.

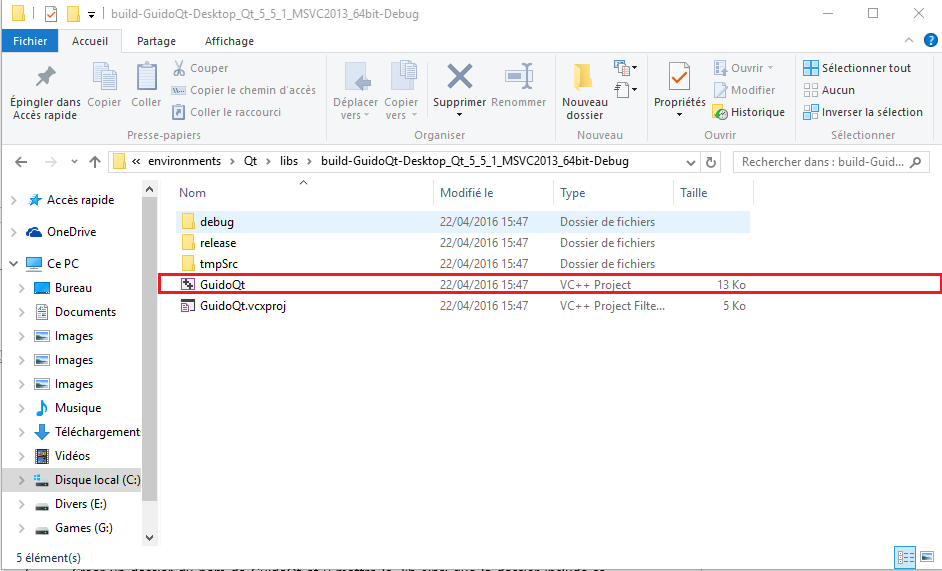


Figure 9: Contenu du dossier suite à la compilation dans Qt

1. Il ne vous reste plus qu'à faire clic droit sur le nom du projet puis générer.

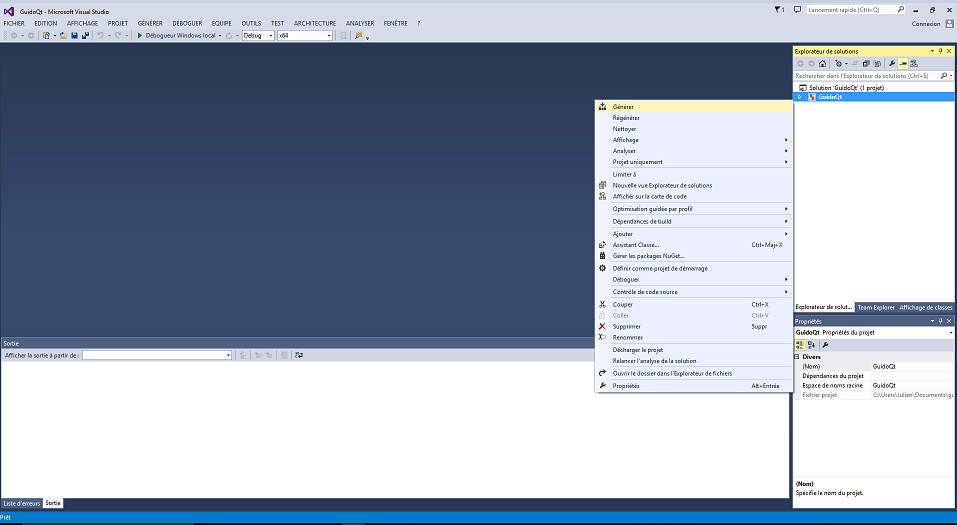


Figure 10: Génération de la librairie GuidoQt dans Visual Studio 2013

1. Le .lib générer se trouve dans source/environments/Qt/libs
2. Créer un dossier du nom de GuidoQt et y mettre le .lib ainsi que le dossier include se trouvant dans source/environments/Qt/libs/GuidoQt

### MacOS

1. Lancer le Qt Project file qui se trouve dans source/environments/Qt/libs/GuidoQt.

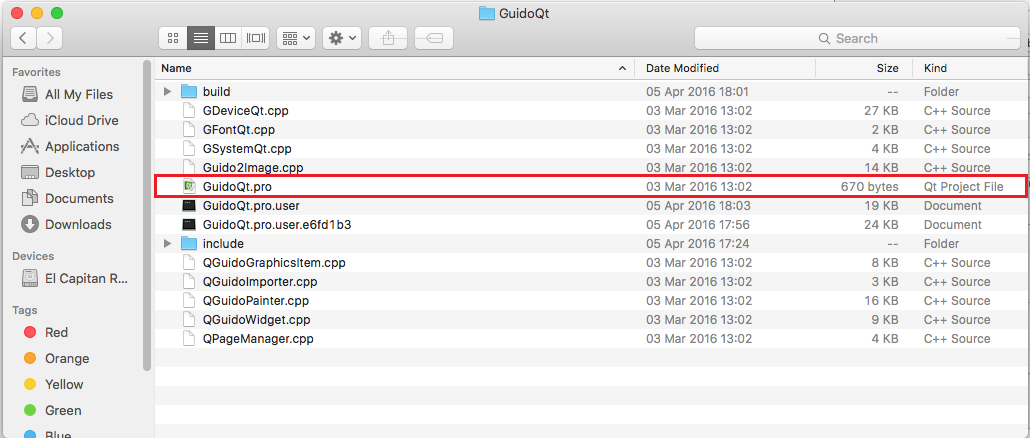


Figure 11: Qt Project File sous MacOS

1. Exécuter le programme. Une fenêtre s’ouvre indiquant qu’aucun chemin vers l’exécutable n’a été trouvé. C’est normal vous n’avez juste qu’à fermer cette fenêtre et Qt.

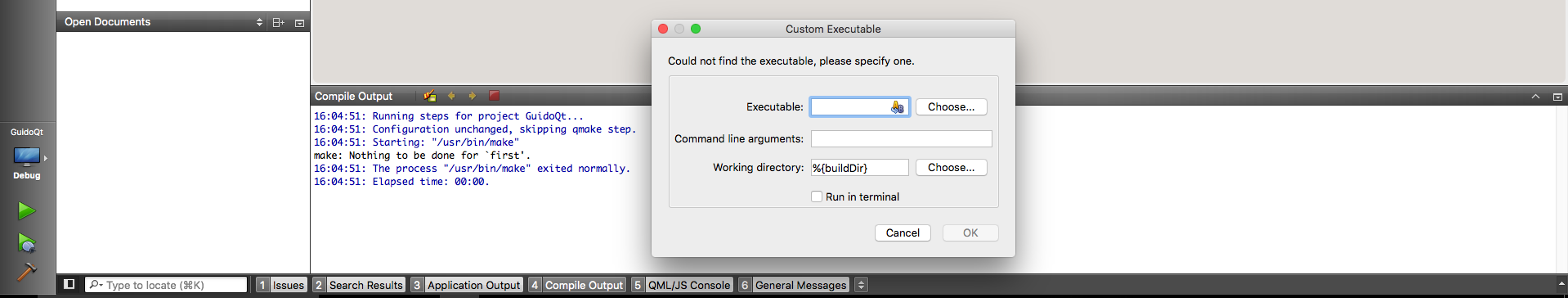


Figure 12: Fenêtre d'avertissement lors de la compilation de GuidoQt sous MacOS

1. Dans source/environments/Qt/libs se trouve un nouveau fichier, libGuidoQt.a, qui correspond à la librairie compilée.
2. Créer un dossier du nom de GuidoQt et y mettre le .a ainsi que le dossier include se trouvant dans source/environments/Qt/libs/GuidoQt

## Midisharelight

Pour l’export vers midi, GuidoEngine a besoin de la librairie midisharelight. Celle-ci est déjà compilée et se trouve dans source/src/midisharelight.

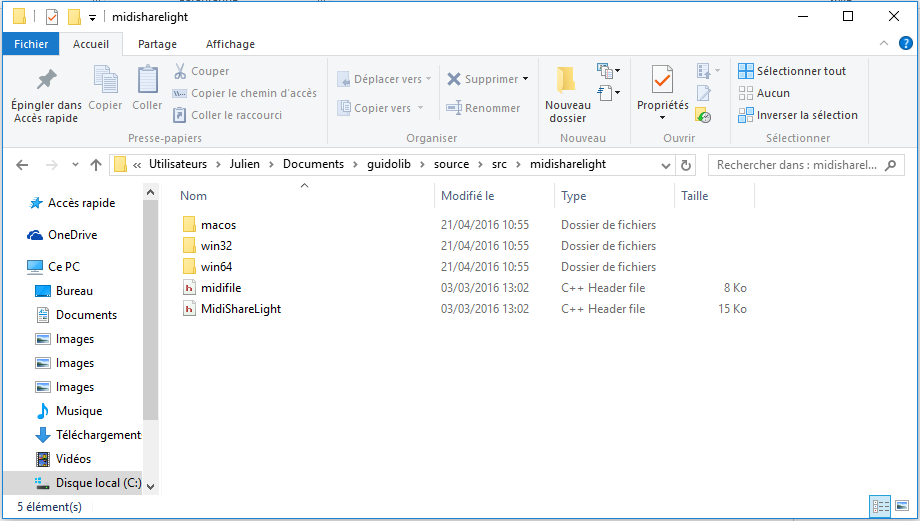


Figure 13: Contenu du dossier midisharelight

Créer un dossier dans lequel, vous allez mettre la librairie qui correspond à votre plateforme, ainsi qu’un dossier include contenant les deux .h contenus dans le dossier midisharelight.

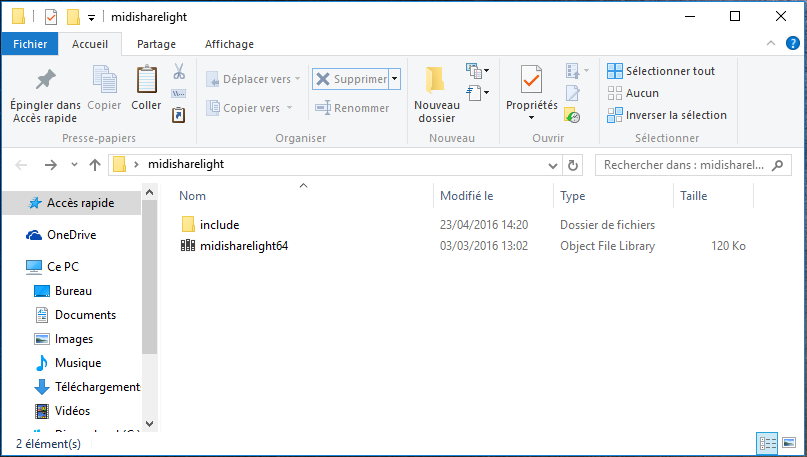


Figure 14: Exemple midisharelight pour windows

## Ajouter une librairie à un projet dans Qt

La procédure pour ajouter une librairie sous Qt est la même pour Windows et MacOS. Voici les étapes à suivre :

1. Copier le dossier contenant la librairie que vous souhaitez ajouter à votre projet dans le dossier de ce projet.
2. Clic droit nom du projet -> ajouter une bibliothèque -> choisir bibliothèque externe

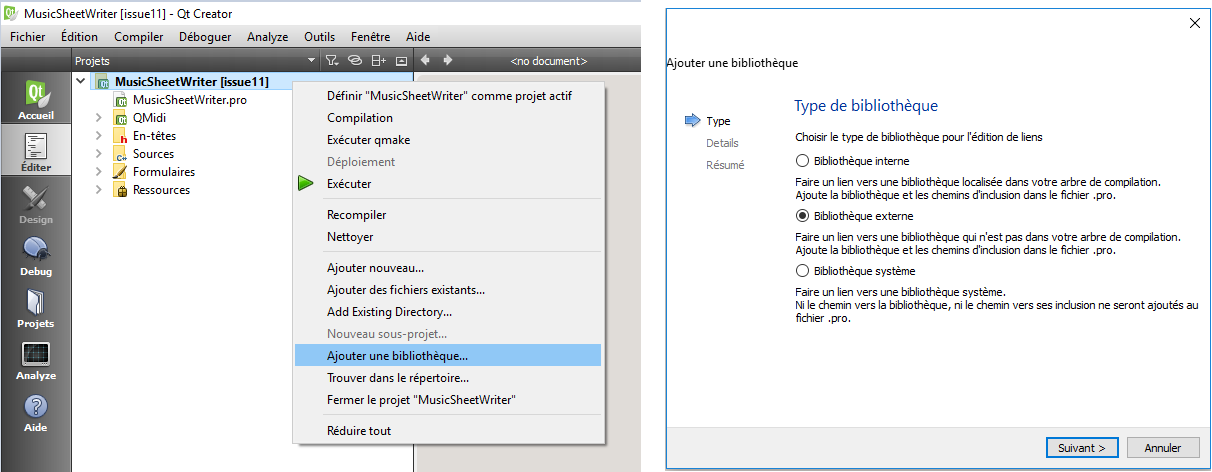


Figure 15: Ajouter une librairie externe

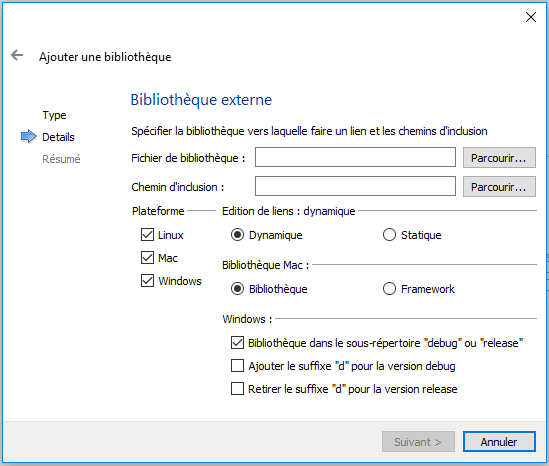
1. Une fois bibliothèque externe sélectionnée, une fenêtre similaire s’ouvre : 

Figure 16: Paramétrage de la librairie externe

Fichier de bibliothèque : correspond au fichier de la librairie qu’on souhaite ajouter à notre projet. À l’aide du bouton Parcourir, aller chercher le .lib (sous Windows) ou le framework (sous MacOS) de votre librairie.

Chemin d’inclusion : correspond au chemin du dossier include qui accompagne vos librairies. Utiliser le bouton Parcourir pour accédé à ce dossier.

Plateforme : Ici, l’utilisateur choisit sous quelle plateforme tourne la librairie.

Editions de liens et Bibliothèque Mac n’ont pas besoin d’être modifiés.

Windows : Pensez à tout décocher, cela évite d’avoir des problèmes avec le nom de la librairie.

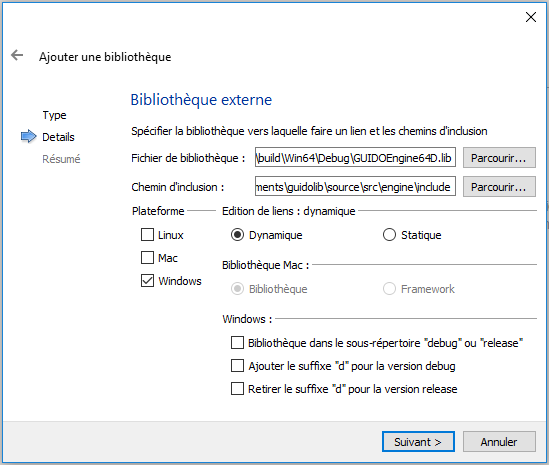


Figure 17: Exemple d'ajout de la librairie GuidoEngine sous Windows

La librairie est automatiquement ajoutée au .pro de votre projet Qt et se présente sous cette forme :

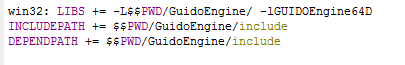


Figure 18: Librairie GuidoEngine dans le .pro

## Fichier de police guido2

La librairie GuidoEngine utilise une certaine police qui n’est pas de base reconnue par l’ordinateur, vous devez donc exécuter le fichier guido2 qui se trouve sur le SVN dans source/src.

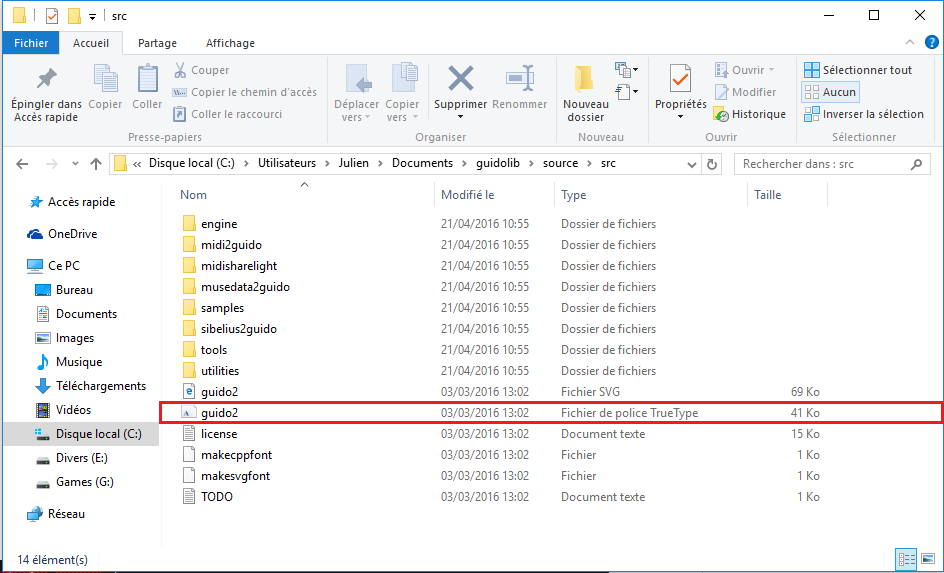


Figure 19: Fichier de police guido2

# Site internet

# Application Android

## Outils de développement

### Android Studio

Android Studio est un environnement de développement d’application Android. Il inclut tous les outils nécessaires et utilisés dans cette procédure. Installez Android Studio à partir de cette page : <http://developer.android.com/sdk/index.html>. Un fois téléchargé, il suffit d’exécuter l’installateur et de suivre les instructions d’installation.

### SDK Manager

A présent, il faut installer le SDK (bibliothèque) Android. Pour ce faire :

* lancez Android Studio et sélectionnez **Configure 🡺 SDK Manager**.
* Dans la fenêtre contextuelle, sélectionnez dans l’onglet **SDK Plateforms** les versions supérieures à la version 4.0.3 – API 15 (car c’est la version minimale sur laquelle l’application peut fonctionner).
* Sélectionnez dans l’onglet **SDK Tools**, tous les éléments.
* Cliquez sur **OK**.

### SVN

Afin de continuer le développement de l’application Android de Music Sheet Writer, il faut tout d’abord récupérer le code source du projet. Il est hébergé sur le SVN du lab EIP à l’adresse <https://labeip.epitech.eu/svn/2017/musicsheetwriter/Android>.

Pour cela, deux possibilités s’offrent à vous :

* Vous pouvez récupérer le code source à l’aide d’un exécutable Subversion ou d’un outil (comme par exemple TortoiseSVN sur Windows disponible sur <https://tortoisesvn.net/downloads.html>) et dans un second temps, ouvrir le projet avec Android Studio.
* Synchroniser Android Studio avec le dépôt SVN. Cette seconde option permet par la suite une simplification des interactions avec le dépôt distant car vous pourrez les gérer directement depuis Android Studio.

Voici les étapes à suivre pour synchroniser Android Studio avec le dépôt SVN :

* Sélectionnez **Check out project from Version Control** 🡺 **Subversion**
* Dans la fenêtre contextuelle, ajoutez un nouveau un dépôt en cliquant sur 2016-04-21_00h39_47.
* Saisissez ensuite <https://labeip.epitech.eu/svn/2017/musicsheetwriter/Android>.

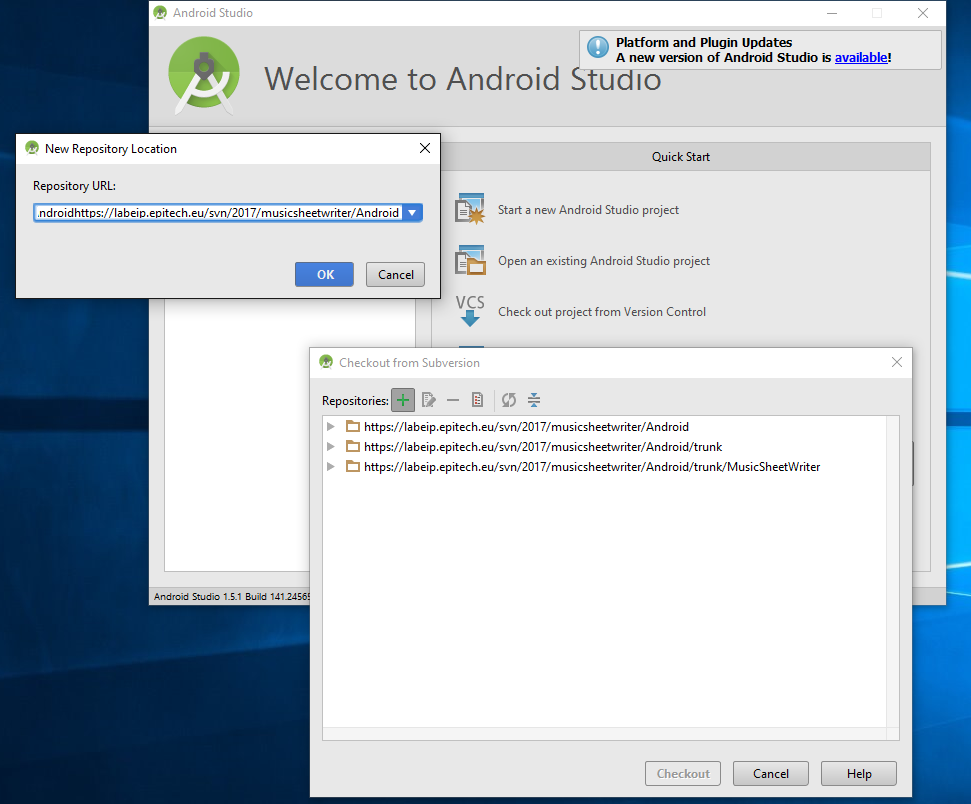


Figure 20. Récupération d'un dépôt SVN dans Android Studio

* Une fois que le projet est récupéré, importez le code source en tant que projet Gradle et sélectionnez le fichier *build.gradle* à la racine du projet.

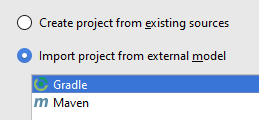




Figure 21. Import d'un projet Gradle dans Android Studio

## Configuration Android Studio

Le projet est à présent chargé dans Android Studio. Il reste maintenant à déclarer le module *app* au sein de l’application.

* Cliquez sur **File 🡺 Project Structure…**
* Dans l’onglet **Project 🡺 Project SDK**, sélectionnez le SDK correspondant à la version d’Android à utiliser.
* Dans **Modules**, cliquez surC:\Users\Jeremy\AppData\Local\Microsoft\Windows\INetCache\Content.Word\2016-04-21_00h39_47.png, et choisissez **Java**. Appellez le module « *app* ».

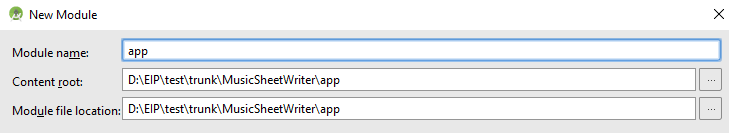
**

Figure 22. Création d'un module dans Android Studio

* Faites un clic-droit sur le module « *app* » dans la liste puis **Add 🡺 Android**
* Définissez les chemins d’accès des différents dossiers du projet.

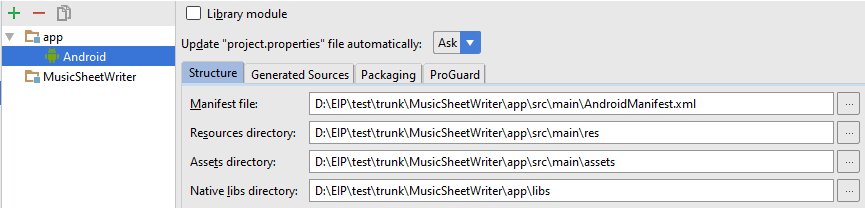


Figure 23. Définition d'un module dans Android Studio

## Compilation en mode production

Maintenant que le projet est correctement configuré, il est possible de générer l’exécutable (un fichier .apk).

* Cliquez dans la barre de menu sur **Build 🡺 Generate Signed APK…**.
* Il suffit de sélectionner l’endroit où le fichier .apk doit être placé
* Lancer la compilation

Une *Key Store*, c’est-à-dire un fichier chiffré contenant les informations de l’éditeur de l’application doit être généré. Ce fichier doit être gardé en sécurité. En effet, s’il est perdu, il ne sera plus possible d’effectuer des mises à jour de l’application. Ainsi, lors de la première sortie de l’application, la *key store* doit être créée en utilisant les informations de l’éditeur et, pour les autres mises à jour, le même fichier sera réutilisé.

Android Studio génère automatique ce fichier à la suite de ces étapes.

## Publication sur le Play Store

Tout d’abord, il faut se connecter à la console de gestion d’application pour développer sur <https://play.google.com/apps/publish/?hl=fr&dev_acc=11464000261115720484#AppListPlace>. L’accès à la plateforme nécessite un compte Android développeur. Un compte a déjà été créé pour l’application MusicSheetWriter.

Pour publier l’application pour la première fois, il suffit d’**Ajouter une nouvelle application**, de sélectionner le titre de l’application, d’uploader le fichier .apk généré dans la section précédente et d’éditer la fiche Google Play Store.

Cette fiche inclus les descriptions courte et complète de l’application, les captures d’écran et les vidéos qui seront visible sur la page de téléchargement de l’application, mais également les informations de restriction d’accès à l’application (langue, taille d’écran, téléphone ou tablette, coordonnées de l’éditeur, etc.). Ces informations doivent être remplies impérativement. La fiche Google Play Store pourra être mis à jour autant que désiré.

Pour mettre à jour l’application sur le Play Store, il suffit de sélectionner l’application Music Sheet Writer dans la liste des applications affichée dans la console et d’uploader un nouveau fichier .apk.

Le temps nécessaire à la publication sur le Play store de l’application est de moins de 24 heures.

# Application iOS

## Outils de développement

### Xcode

Le développement d’applications iOS se fait à l’aide de Xcode.

Voici le lien de téléchargement de XCode : [https://developer.apple.com/xcode/download/](https://www.google.com/url?q=https%3A%2F%2Fdeveloper.apple.com%2Fxcode%2Fdownload%2F&sa=D&sntz=1&usg=AFQjCNE9BfSEsNzbBpU7lXe0-RpGRE2ofA) .  
Vous pouvez aussi passer par l’iTunes Store pour télécharger l’outil.

### SVN

# Application Windows Phone