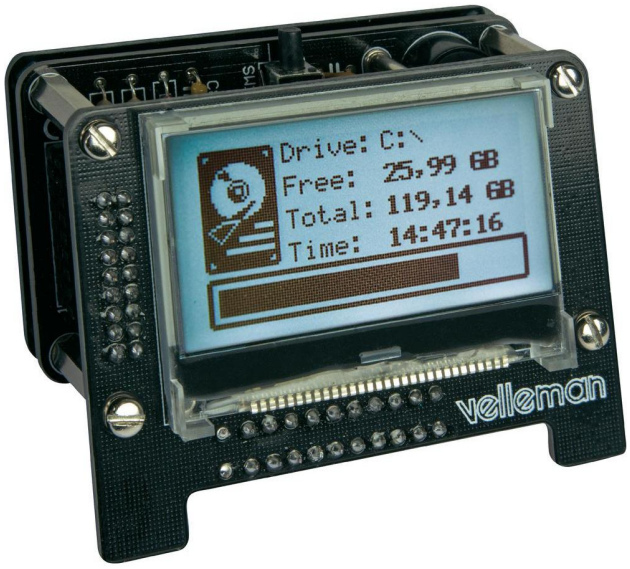

CFPT Ecole d'informatique - Technicien ES en informatique

Travail 1er semestre 2016-2017

Simulateur afficheur Velleman K8101

Elèves :
M. Alan Devaud
M. Dylan Wacker

Enseignants :
Mme Anne Terrier
M. Christophe Maréchal

Version 1.0 du
21 novembre 2016

1 Résumé

Notre projet a pour but de simuler un afficheur Velleman K8101 afin que des élèves puissent créer des applications qui fonctionnent pour le Velleman sans avoir besoin d'un appareils physique. Nous avons donc créé une librairie (.dll), semblable à celle que Velleman propose, même noms de méthodes, afin de juste changer la .dll pour (entre celle de Velleman et la nôtre) passer de la version physique à simuler et inversement. Le Velleman envoie les informations via serial et notre librairie via TCP/IP. Pour conclure, nous avons travaillé sur ce projet à deux pendant 44 périodes, nous avons fait ce projet en C#.

Our project is to simulate a Velleman K8101 display, so that students can create applications who operate for the Velleman whitout need a real devices. We created a library(.dll), similar to that of the Velleman, same names of methods, to just change the .dll to (between the library of Velleman and ours) move from the physical version to the simulate version and inversely. The Velleman send the information by serial and our library by TCP/IP. To conclude , we worked on this project during 44 hours, we did this project in C#.

Table des matières

2 Introduction

Dans le cadre de la formation de *Technicien ES en Informatique* un travail de semestre doit être effectué. Cette année, les sujets ont été imposés par les enseignants mais ils sont cependant réalisés par groupe. Le projet qui nous a été demandé est de réaliser un simulateur d'afficheur Velleman K8101.

Cet appareil permet d'afficher des messages venant de l'ordinateur par USB. Il existe différents programmes qui envoient des données à l'afficheur. Il peut, par exemple, afficher des messages *Twitter*, des emails ou encore les informations des disques (lettre, espace libre, taille totale).

L'intérêt de ce projet est de pouvoir avoir plusieurs de ces afficheurs, par exemple, pour des cours. L'afficheur est simulé sur un pc et intègre toutes les actions qui sont possibles avec la version physique.

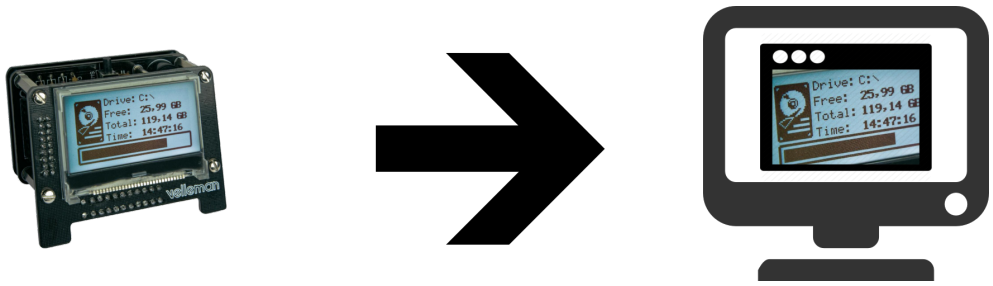

FIGURE 1 – Afficheur Velleman K8101

3 Cahier des charges

3.1 Sujet

Logiciel qui permet de simuler l'afficheur Velleman K8101.

3.2 But

Cette application permet de simuler l'affichage de l'afficheur Velleman K8101. L'appareil permet d'afficher des messages provenant de l'ordinateur par le connecteur USB. Il y a actuellement cinq programmes fournis par le fabriquant pour tester l'afficheur. Au sein de l'école, il y a présentement un seul afficheur à disposition de tous. Le but du projet est de créer un simulateur de l'afficheur qui a les mêmes fonctionnalités que l'original mais sur un pc. Cela permet d'avoir à disposition autant de simulateurs que le besoin nécessaire pour un cours.

3.3 Spécifications

L'application permettra de simuler cet afficheur. Il pourra donc simuler :

1. La bibliothèque(.dll) du K8101 fournit par Velleman
2. interprétation des données reçues par les programmes
3. afficher les données venant des programmes
4. le buzzer
5. le bouton (pression simple et longue)

3.4 Restrictions

— Création de nouvelles applications
— Implémentation de fonctionnalités divers (Non existant sur l'appareil physique)

3.5 Environnement

— Un pc - Système d'exploitation Windows 7 x64
— Un afficheur Velleman K8101
— Langage de programmation : C# .NET 4.5

3.6 Livrables

— Poster
— Code source du projet
— Documentation
— Présentation
— Journal de bord

3.7 Redditions

— 3 octobre 2016 : rendu du poster
— 17 octobre 2016 : reddition du rapport intermédiaire
— 21 novembre 2016 : reddition finale
— 28 novembre 2016 : exposé oral

3.8 Planification

La planification du projet est décrite ci-dessous. Il faut noter que les dates indiquées sont celles des cours, tous les lundis, mais les tâches pourront être effectuées tout le long de la semaine, en plus des cours consacrés au projet.

— 5 septembre 2016 :
  — Présentation du cours
  — Cahier des charges et planification
— 12 septembre 2016 :
  — Remplir la planification dans le fichier excel
  — Recherche sur le produit
— 19 septembre 2016 :
  — Réflexion sur les possibles bibliothèques à utiliser
  — Maquettes de l'application
— 26 septembre 2016 :
  — Faire des Structogrammes
  — Rédaction du Poster
— 3 octobre 2016 :
  — Finalisation du poster
  — **Rendu Poster**
— 10 octobre 2016 :
  — Rédaction du rapport intermédiaire
— 17 octobre 2016 :
  — Développement des fonctions permettant de se connecter aux différents programmes
  — **Reddition du rapport intermédiaire**

- 24 octobre 2016 :
  - Finaliser le développement permettant de se connecter aux différents programmes
- 31 octobre 2016 :
  - Développement des fonctions permettant de simuler le buzzer
  - Développement des fonctions permettant de simuler le bouton
- 7 novembre 2016 :
  - Développement de l'interface
- 14 novembre 2016 :
  - Finaliser le développement de l'interface
- 21 novembre 2016 :
  - **Reddition finale**
- 28 novembre 2016 :
  - **Exposé oral**

4 Analyse de l'existant

A ce jour, il n'existe aucun simulateur pour l'afficheur Velleman K8101. Seul l'appareil physique est à disposition pour pouvoir réaliser le projet. Cet appareil permet d'afficher des données reçues par un programme compatible. Le programme envoie des données via un connecteur USB jusqu'à l'afficheur qui interprète ces données et les affiche à l'écran.

4.1 Appareil physique

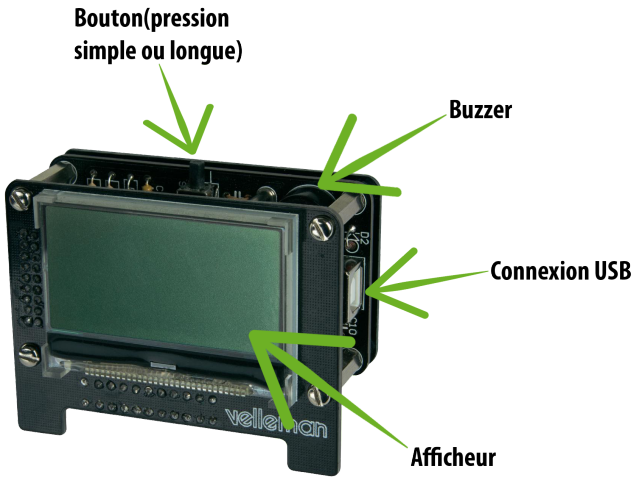

FIGURE 2 – Afficheur Velleman K8101

Fonctionnalités de l'afficheur :

— **Bouton** : le bouton permet à l'afficheur d'interagir avec le programme. Il donne à l'utilisateur la possibilité d'envoyer au programme une *instruction* pour effectuer une action tels que l'exécution d'une validation. Le bouton a "deux états", il a un pression longue et une pression courte.
— **Buzzer** : le buzzer permet d'émettre un son.
— **Connexion USB** : l'USB permet la communication entre les différents programmes et l'afficheur. Le programme envoie les instructions à l'afficheur qui les interprète pour les afficher par la suite.
— **Afficheur** : affiche les données reçus par l'application.

Commandes disponibles :

— Envoyer des images (.bmp 128x64) à l'écran
— Dessiner ou supprimer un carré
— Dessiner ou supprimer des pixels
— Dessiner ou supprimer une ligne
— Activer le buzzer en réglant la valeur pour le signal sonore (0-255)
— Activer le rétro-éclairage en réglant le temps ON (0-254s. / 255 = toujours allumé)
— Envoyer un texte court ou long
— Ajuster le contraste
— Affichage inversé
— Supprimer tout sauf l'image d'arrière-plan

— Supprimer tout
— Attribuer une action lors d'un appui court
— Attribuer une action lors d'un appui long

Informations technique :

— Ecran rétro-éclairé à LED (blanc)
— Résolution d'écran : 128 x 64 pixels
— Alimentation par USB
— Max. consommation de courant : 35 mA
— Fichier .DLL inclus pour écrire vos propres applications en VB.net ou C#
— Code source pour toutes les applications d'exemple
— Dimensions : 77.5 x 60.5 x 38 mm (3.05" x 2.4" x 1.5")

5 Analyse fonctionnelle

5.1 Esquisses d'interface

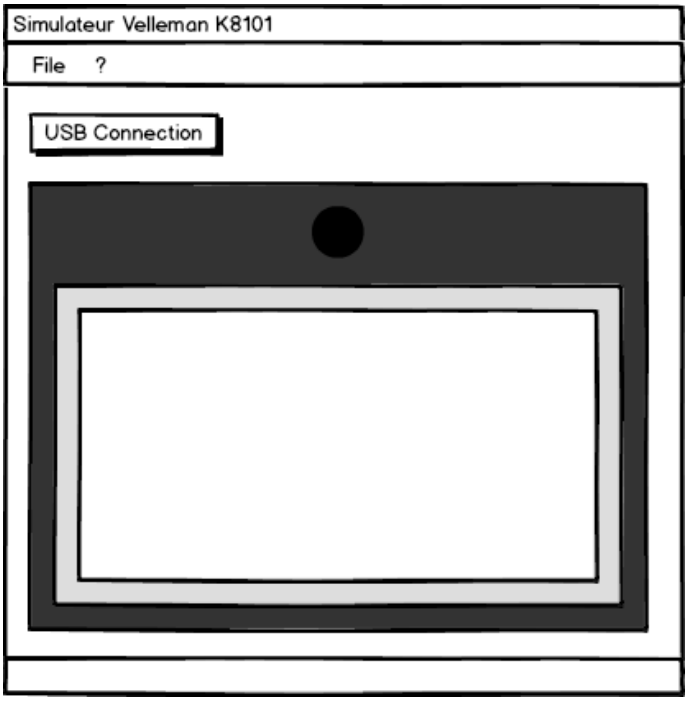

FIGURE 3 – Interface simulateur Velleman K8101

L'interface générale se présente comme la fig. 6 (page 11). On peut retrouver tout ce qui existe sur l'appareil physique. L'afficheur LCD, le bouton

pour l'interaction, la connexion USB (simuler pour un bouton de connexion) et le buzzer qui est simulé par le son de l'ordinateur.

Le bouton *USB Connection* permet à l'utilisateur de lancer la connexion usb entre les logiciels.

5.2 Conventions / Architectures

Pour ce projet, les conventions de codage utilisées sont celles que nous utilisons à l'école. Pour faciliter la compréhension de tous, notre application est développée en anglais.

Une architecture MVC [1] est mise en place. Nous avons choisi de mettre en place cette architecture pour que l'on puisse apprendre à bien la maîtriser.

Pour la gestion de notre projet et son versionnement, un répertoire git sur github a été créé et partagé : `https://github.com/Devaud/SimulateurAfficheurK8101.git`.

6 Analyse organique

Nous avons utilisé un décompilateur de fichiers .dll afin de savoir l'approche de Velleman, pour cela nous avons utilisé un programme qui se nomme "jetbrains dotpeek".

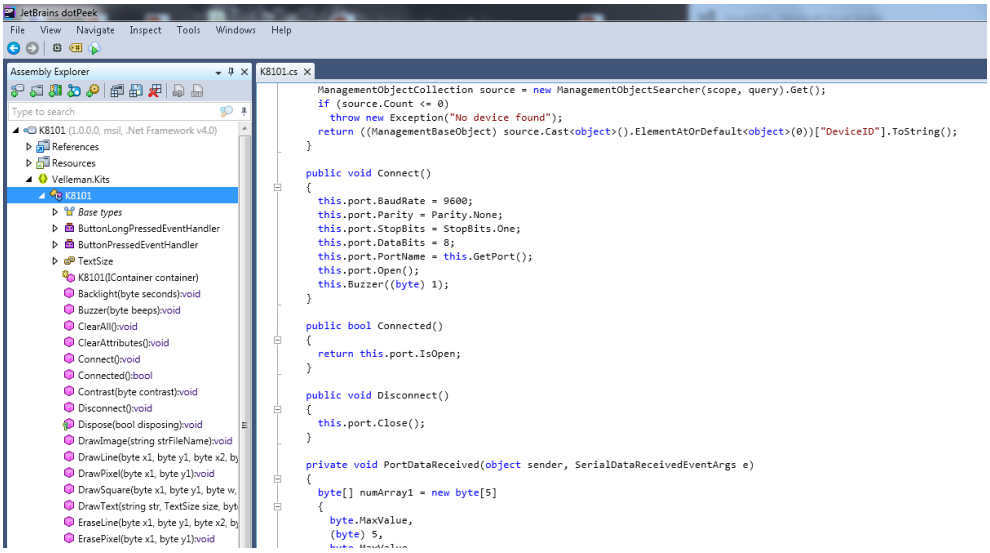

FIGURE 4 – Interface du décompilateur

1. *Model View Controller* (Modèle Vue Contrôleur)

Nous reprenons les mêmes noms de méthodes avec les mêmes paramètres afin que notre .dll fonctionne comme la leur sauf que le code à l'intérieur des méthodes est différent pour fonctionner avec notre simulateur.

7 Conclusion

7.1 Problèmes rencontrés

Nous avons perdu beaucoup de temps, car nous sommes partis dans la mauvaise direction. Nous pensions devoir simuler la connexion physique(USB) de l'appareil K8101 pour que les programmes fournis avec l'appareil détecte notre simulateur comme le vrai. On a passé la plupart de notre projet à faire des recherches, essayer pleins de solutions différentes, mais sans résultat. Nous avons exposé le problème à M. Maréchal le 7 novembre, soit 2 semaines avant le rendu final, il nous a proposé de faire une bibliothèque qui a les mêmes noms de méthodes que celle du Velleman mais avec une approche différente. Si on crée un programme qui fonctionne avec notre bibliothéque, ce même programme devrait fonctionner avec le Velleman réel en changeant simplement la .dll choisit.

7.2 Retour sur la planification

La planification initial n'a pas pu être suivi, étant donné le gros problème rencontré nous avons dû passer la plupart de notre temps à faire des recherches, ce qui devait durer un seul jour maximum, a duré trop longtemps. Au final on a pas fait ce que la planification avait prévu, un changement de dernière minute a presque tout changé.

7.3 Retour sur l'application

Notre application comporte notre librairie, une application de test qui devait envoyé les données à une autre application qui comporte l'afficheur simulé.

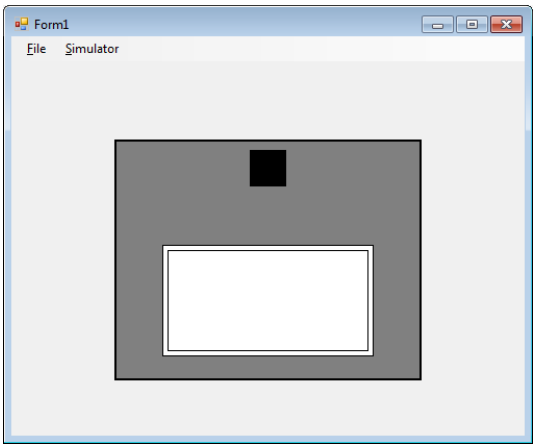

FIGURE 5 – Velleman simulé

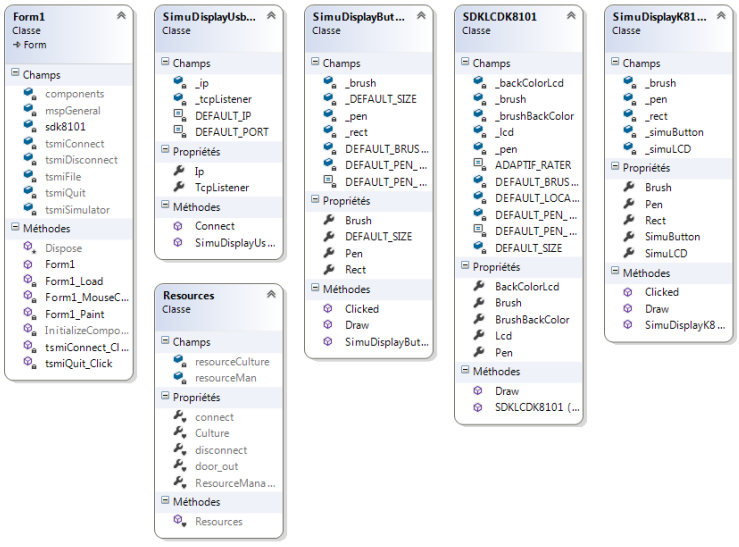

FIGURE 6 – Classe SimulatorDisplayerK8101.cs

7.4 Intérêt personnel

Bien que nous ayons perdu beaucoup de temps à la recherche et d'être parti dans la mauvaise direction, nous avons appris des choses intéressantes, par exemple de communiquer entre deux applications grâce au réseau TCP/IP. Nous n'avions jamais simulé un appareil physique en application auparavant, ce fut une bonne expérience, on ne savait pas trop ce qu'il était possible de faire pour simuler des appareils physique qui ont besoin d'une connexion direct au PC.

Table des figures