Ecole Publique d'Ingénieurs en 3 ans

Projet encodage d'un message dans un parachute

RAPPORT DU PROJET

Le 04 avril 2021,

Version 1.1

Lamhamdi Mehdi, mehd,lamhamdi@ecole,ensicaren,fr

JOUILI Houssem houcem.jouili@ecole.ensicaen.fr

2A informatique

Une description des différentes classes:

-La classe MessageModel gére le message, elle ne contient que les attributs propre au message: le message lui même et le décalage par rapport au code ASCII. Elle offre la possibilité d'acceder au k éme bits et envoie un signal notify() si le message change.

-Les classes BinaryViewWidget et ParachuteView sont propre à la vue, elles ne contiennent que les attributs propre a leur représentation

-BinaryViewWidget contient comme attributs: color0 et color1 et un pointeur sur le modèle. Elle dispose d'une méthode paintEvent(QPaintEvent *) pour dessiner le codage binaire du message après la récéption d'un signal notify().

-ParachuteView contient comme attributs: nombre de pistes / secteurs et les couleurs des bits 1 et 0 et un pointeur vers le model. Elle dispose d'une méthode paintEvent(QPaintEvent *) qui se lance après un la récéption d'un notify() pour dessiner le codage sous la forme d'un parachute de message.

-mainWindow est la fenêtre principale qui va afficher notre logiciel. Elle contient l’interface qui va communiquer avec l’utilisateur pour choisir les différentes paramètres qu’il veut. Elle met à la disposition de l’utilisateur des actions pour enregistrer, ouvrir une nouvelle fenêtre et ouvrir un fichier enregistré.

Fonctionnalités implementees:

- Le message est géré par une classe MessageModel qui possède une méthode "getBinaryK" qui retourne le kéme bit.

- Il est possible de regler le nombre de secteurs et de pistes par un slider et par un spinbox

- La vue binaire est implementée, si il n y a plus de place dans une ligne pour la dessiner, l'application saute avec une marge aux 7 lignes suivantes pour continuer à dessiner, s'il n y a plus de lignes pour dessiner un message s'affiche entre les deux "sept derniers lignes".

- Si le message est très long pour les nombres de secteurs et pistes choisis, un message s'affiche pour demander a l'utilisateur d'augmenter les deux variables.

- L'application dispose d'actions suivantes: New, Save, Open et d'une description : New pour créer une nouvelle fenêtre, Save pour sauvgarder les paramètres choisies par l'utilisateur dans un fichier texte et open pour lire à partir d'un fichier texte les différentes données et les utiliser pour afficher le codage du message correspondant.

- On peut sauvegarder le message dans un fichier texte et lire d'aprés un fichier texte qui a été déjà enregistré par l'application.

- On peut changer le caractère de référence. On demande à l'utilisateur de choisir lui même le code de @ (par défaut il est mit à 0, ce qui correspond à un décalage de 64 par rapport au code ASCII).

- On peut choisir parmi une liste des coulours. On a choisit des couleurs claires pour le bit 0 et des couleurs sombres pour le bit 1.

Fonctionnalités non implémentées :

- Un mode de couleurs aléatoires

- Le choix d’afficher des blocs de 10 trapèzes

- Choisir des nombres de secteurs simplement parmi les multiples de 7 (ou de 10)

- Offrir le choix d’ajouter le motif 0001111111 1111111111 sur chaque piste

- Dessin plus fidèle, avec les dents de scie entre les pistes

Ressources utilisées :

-Cours de Qt

-Les sujets de tps

-Openclassroom

-StackOverflow

Gestion d’équipe :

Nous avons partagé les tâches entre nous qui nous a permis d’avancer dans le projet. Nous avons aussi travaillé hors des séances de tps pour faire le maximum des fonctionnalités. Pour se communique hors des séances nous, avons utilisé discord. Dans ce projet, nous avons utilsé git pour versionner notre travail.