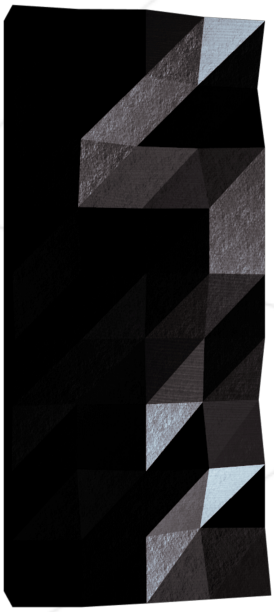

GUImp

Votre première interface graphique

Gadi gadi@42.fr
Jpeg jpeg@42.fr
Team Pedago pedago@42.fr

Résumé: *Ce projet aura pour but de créer votre première interface graphique afin de créer une librairie réulitisable dans tous vos projets par la suite !*

Table des matières

Chapitre I

Préambule

Le 9 Mai 1969, à deux kilomètres de la frontière nord du Laos, la 101ème Division d'Infanterie de l'armée américaine progresse vers la colline 937.

Pour eux, c'est une simple reconnaissance. Pour les Viêt-Cong, la colline 937 est une zone stratégique.

La dizaine d'apellés de la 101ème Division, peu formée au combat, devait effectuer cette mission de routine en moins de deux heures.

Ils résisteront héroiquement pendant neuf jours.

`Ce projet ne raconte pas leur histoire.`

FIGURE I.1 – GUI...mp

Chapitre II

Introduction

Une interface graphique - ou plus communément **GUI** *"Graphical User Interface"*, est aujourd'hui un standard pour tout bon logiciel qui se respecte, cependant cela n'a pas toujours été le cas ...

Au début de l'ère informatique, le seul type d'affichage existant était le **CLI** *"Commande-line Interface"* avec lequel vous êtes maintenant devenus plus que familier.

Le **CLI** n'ayant pas été créé pour rendre l'informatique simple d'utilisation pour le commun des mortels, c'est pour pallier ce problème que l'interface graphique fût créé dans les années 1970 par les ingénieurs du `Xerox Palo Alto Research Center`.
Mise pour la première fois sur le marché avec le `Star` de `Xerox`, elle fût grandement popularisée avec la commercialisation du `Macintosh` par `Apple` en 1984.

FIGURE II.1 – CLI vs GUI.

À vous aujourd'hui de créer votre propre libraire d'interface graphique pour créer de magnifiques logiciels simples d'utilisation pour le commun des mortels qui ne savent pas ce qu'est un terminal !

Chapitre III

Objectifs

L'objectif de ce projet sera pour vous de réaliser une librairie d'interface graphique. Cette librairie devra être la plus complète et modulable possible, le but étant de la réutiliser dans vos futurs projets, que ce soit dans le cadre de la branche graphique ou de vos autres projets.

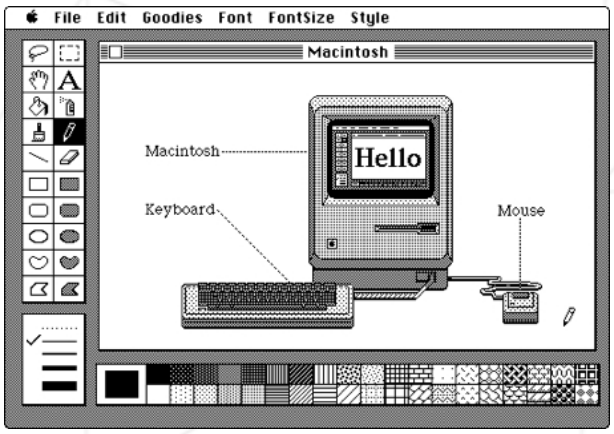

FIGURE III.1 – L'interface orignal de Paint - Macintosh en 1984.

Pour ce faire vous allez devoir réaliser un petit logiciel de retouche et de création d'images numériques sur le modèle de `The GIMP` pour montrer toutes les fonctionnalités ainsi que le bon fonctionnement de votre librairie.

Chapitre IV

Consignes générales

- Ce projet ne sera corrigé que par des humains. Vous êtes donc libres d'organiser et de nommer vos fichiers comme vous le désirez, en respectant néanmoins les contraintes listées ici.
- L'exécutable doit s'appeller `guimp`.
- Vous devez rendre un `Makefile`. Ce `Makefile` doit compiler le projet, et doit contenir les règles habituelles. Il ne doit recompiler et relinker le programme qu'en cas de nécessité.
- Si vous êtes malin et que vous utilisez votre biliothèque `libft` pour votre `guimp` ou votre `libui`, vous devez en copier les sources et le `Makefile` associé dans un dossier nommé `libft` qui devra être à la racine de votre dépôt de rendu. Votre `Makefile` devra compiler la bibliothèque, en appelant son `Makefile`, puis compiler votre projet.
- Vous devez rendre votre librairie `libui` dans un dossier séparé, à la racine de votre rendu, nommé `libui`.
- Voud devez fournir un Makefile, contenant les règles habituelles, qui compilera une `libui.a`. Cette lib devra être linkée à votre projet de la même façon que votre `libft`.
- Les variables globales sont interdites.
- Votre projet doit être en `C` et à la Norme. La norminette fait foi.
- Vous devez gérer les erreurs de façon raisonnée. En aucun cas votre programme ne doit quitter de façon inattendue (segmentation fault, bus error, double free, etc...).
- Votre programme ne doit pas avoir de fuites mémoire.
- Vous devez rendre, à la racine de votre dépôt de rendu, un fichier `auteur` contenant vos logins suivi d'un '\n' :

```
$>cat -e auteur
xlogin$
ylogin\$
```

- Vous devez utiliser pour ce projet la `SDL2`. Soit la version installée sur les machines en cluster, soit directment via ses sources dans votre rendu, et elle devra se compiler automatiquement exactement comme la `MinilibX` ou votre `libft` sur vos autres projets, sans autre action que de compiler votre rendu.
- Dans le cadre de votre partie obligatoire, vous avez le droit d'utiliser les fonctions suivantes de la libc :
  - `open`
  - `read`
  - `write`
  - `close`
  - `malloc`
  - `free`
  - `perror`
  - `strerror`
  - `exit`
  - Toutes les fonctions de la lib math (`-lm` et `man 3 math`)
  - Toutes les fonctions de la `SDL 2`, `SDL_ttf` et `SDL_image`.
- Vous avez l'autorisation d'utiliser d'autres fonctions, voire d'autres bibliothèques, dans le cadre de vos bonus à condition que leur utilisation soit dûment justifiée lors de votre correction. Soyez malins.
- Vous pouvez poser vos questions sur le forum, sur slack...

Chapitre V

Partie obligatoire

Cette partie obligatoire sera séparée en deux parties :

- Une libraire nommée `libui` contenant toutes vos fonctions d'interface utilisant la `SDL 2`.
- Un logiciel de retouche/dessin nommé `guimp` semblable à `The GIMP` ou `Paint` utilisant votre `libui` et uniquement celle-ci pour toute votre gestion graphique.

`Votre guide de référence pour la SDL 2` https://wiki.libsdl.org/

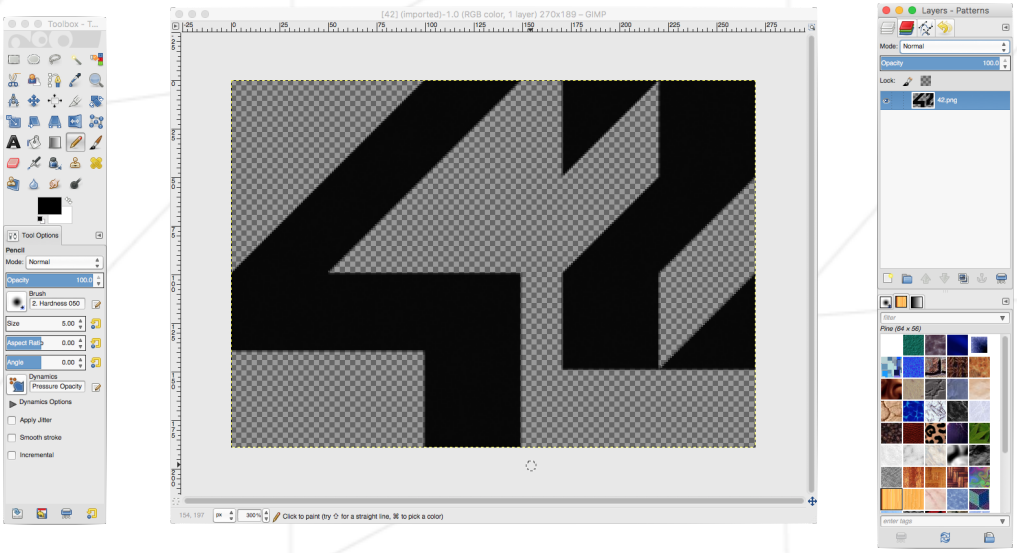

FIGURE V.1 – Interface de The GIMP.

FIGURE V.2 – Simple Directmedia Layer

V.1 Back : libui

Vous devez créer une librairie `libui`, qui de la même façon que votre `libft`, devra être capable de :

- Créer des fenêtres graphiques à l'écran, selon différents paramètres. (taille, redimensionable, couleur et/ou image de fond, effet particulier, thème ou style, etc...)
- Proposer différents type de fenêtres : générique à éléments, modale `OK`, modale `OK/CANCEL`..
- Les fenêtres génériques peuvent contenir divers éléments : bouton, menu, texte, image, texte éditable ligne et zone, checkbox, radio, sliders, liste déroulante, barre de progression, etc...
- Chaque élément dispose de paramètres de style ou de fonctionalités, par ex un bouton poussoir ou bien on/off, couleur et police pour du texte, thème ou style...
- Plusieurs éléments peuvent eux-même contenir des éléments, comme un menu dans un menu, une image dans un bouton, etc... Plusieurs "cascades" différentes doivent etre présentées.
- Votre `libui` gère les événements utilisateurs associés aux fenêtres et aux éléments. Click, focus, touche, etc... il est possible pour chaque élément, si cela a un sens, de spécifier un comportement ou une action spéciale lié à chaque événement. Cela peut être une action par défaut de l'élément (incrémenter un compteur par ex), et donc géré par votre libui, ou bien une action définie par l'utilisateur.
- La bonne cohabitation des 2 points précédents doit être mise en évidence.
- Il doit être possible d'avoir une interaction sur un élément qui entraine l'apparition et/ou la modification d'un autre élément, par ex un bouton mis sur ON fait apparaitre une option supplémentaire.
- Un système de `HotKeys` propre à chaque type d'élément est disponible par défaut.
- Une fenêtre peut être scrollable si nécessaire selon les paramètres choisis.

- Un élément peut être scrollable si applicable et nécessaire, selon les paramètres choisis.
- Créer des "préfabs" : des combinaisons toutes prêtes d'éléments composés pour des utilisations courantes, comme par ex une barre d'état avec images et infos, ou encore un menu par défaut avec open/save/quit etc..
- Vous devez avoir un préfab de séléction de fichier et un préfab du choix des polices de caractère disponible.
- Certains éléments et à défaut la fenêtre sont capable de gérer le drag-n-drop.
- Toutes ces features, si elles ne sont pas utilisées dans la partie suivante, devront être demontrées en soutenance.

FIGURE V.3 – The Gimp

V.2 Front : Guimp

Pour la partie `guimp` :

- Un bouton `QUIT` doit fermer les fenêtres et quitter le programme proprement.
- Cliquer sur la croix rouge sur la bordure de la fenêtre doit fermer la fenêtre correspondante et quitter le programme proprement si celle-ci est la fenêtre de rendu.
- Image de rendu dans une unique fenêtre.
- Palette d'outils dans une unique fenêtre contenant au minimum les éléments suivants :
  - Un bouton pinceau
  - Un bouton gomme
  - Un menu de tracé de traits, rectangles, carrés et cercles (vide `et` plein)
  - Un bouton loupe et un bouton main de déplacement
  - Un menu de gestion de l'épaisseur du trait
  - Un menu d'import de plusieurs images dans le rendu
  - Un menu spécifique à la selection de couleurs, cercle RGB et/ou sliders R-G-B
  - Un bouton pour effacer l'espace de travail
  - Un menu tampon/brush avec lequel on doit pouvoir séléctionner et poser des "stickers" sur le rendu et cela de façon `fluide`
  - Un bouton qui permet de remplir une forme, aka l'outil `pot de peinture`
  - Un menu permettant d'afficher du texte à un endroit donné, avec choix de la couleur, de la font et de la size
  - Un bouton pipette, permettant de sélectionner une couleur sur une image

- Changement du curseur de la souris en fonction de l'outil selectionné.
- Sauvegarde de l'image dans le format de votre choix.
- Import d'images en `JPEG` et `PNG`, mais surtout en **JPEG** (Pensez à regarder du coté de la `SDL_image`).

`Pour toute la gestion graphique, votre programme devra utiliser votre libui et uniquement celle-ci. Votre programme ne devra donc en aucun cas utiliser directement la SDL, le framework graphique système ou tout autre système de gestion graphique.`

Chapitre VI

Partie bonus

Les bonus ne seront comptabilisés que si votre partie obligatoire est PARFAITE. Par PARFAITE, on entend bien évidemment qu'elle est entièrement réalisée, et qu'il n'est pas possible de mettre son comportement en défaut, même en cas d'erreur aussi vicieuse soit-elle, de mauvaise utilisation, etc ... Concrètement, cela signifie que si votre partie obligatoire n'est pas validée dans son intégralité, vos bonus seront intégralement IGNORÉS.

- Undo/Redo (Cmd+z et Cmd+y)
- Copier-coller une sélection
- Gestion de calques
- Menus avec des onglets
- Un fichier de configuration de style d'interface tel que le ferait un fichier `CSS`
- Drag and drop interne. (ex : calques gimp)
- Interface resizable
- Interface responsive !
- D'autres trucs qu'on a même pas imaginé

Chapitre VII

Rendu et peer-évaluation

Rendez-votre travail sur votre dépot `GiT` comme d'habitude. Seul le travail présent sur votre dépot sera évalué en soutenance.