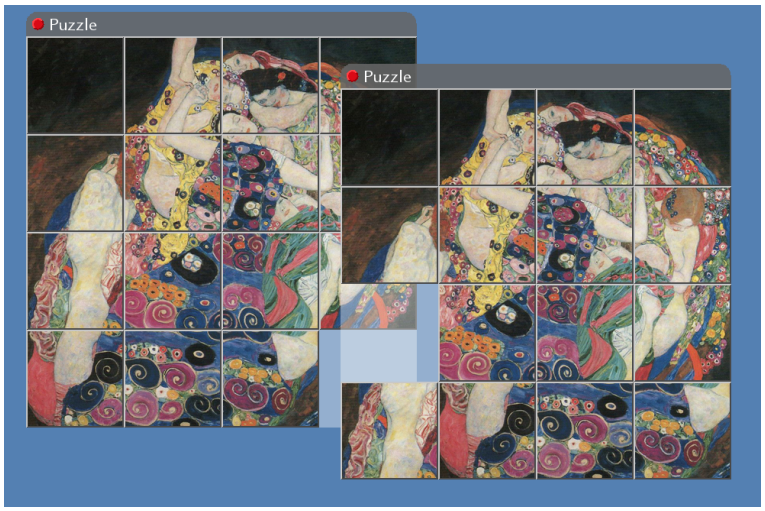
Ensimag — Printemps 2023

Projet Logiciel en C

Sujet : Interfaces Utilisateur Graphiques



Version : 2.702 (6 mai 2021)
Auteurs : F. Bérard, P. Reignier, JS. Franco, E. Frichot, N. Gesbert,
D. van Amstel, C. Ramisch, A. Shahwan.

Chapitre 1

Introduction

Ce chapitre présente les objectifs du projet, puis un rapide survol des grandes familles d'algorithmes que vous aurez à programmer.

1.1 Objectifs

1.1.1 Langage C, Projet

Tout informaticien doit connaître le langage C. C'est une espèce d'espéranto de l'informatique. Les autres langages fournissent en effet souvent une interface avec le langage C (ce qui leur permet en particulier de s'interfacer plus facilement avec le système d'exploitation) ou sont eux-mêmes écrits en C. D'autre part c'est le langage de base pour programmer les couches basses des systèmes informatiques. Par exemple, on écrit rarement un pilote de périphérique en Ada ou Java. Le langage C est un excellent langage pour les programmes dont les performances sont critiques, en permettant des optimisations fines, à la main, des structures de données ou des algorithmes. Par exemple, les systèmes de gestions de base de données et d'une manière générale les logiciels serveurs sont majoritairement écrits en C. La programmation graphique interactive, c'est à dire nécessitant le calcul immédiat de nouvelles images en fonctions des actions de l'utilisateur, nécessite à la fois performance et accès au matériel (cartes graphiques et dispositifs d'interaction), c'est donc un domaine où la connaissance du C est indispensable.

En outre, le projet C a pour objectif de vous confronter au premier projet logiciel un peu conséquent, que vous devez développer dans les règles de l'art : mise en œuvre de tests, documentation, démonstration du logiciel, partage du travail, etc.

1.1.2 Bibliothèque de programmation des interfaces utilisateur graphiques

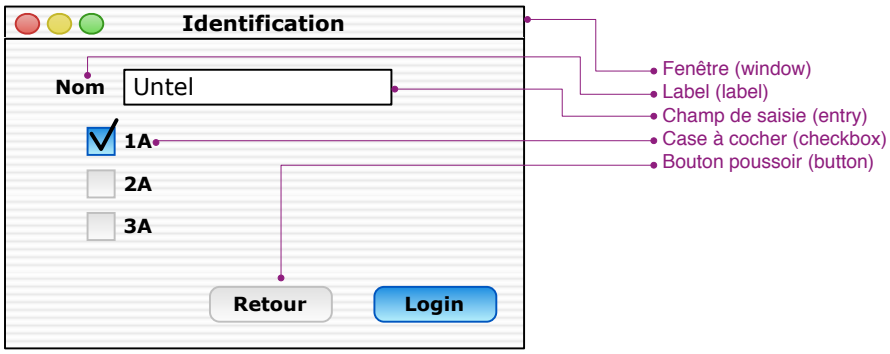


FIGURE 1.1 – Une fenêtre d'une interface utilisateur graphique.

Vous réalisez une bibliothèque logicielle qui facilite la programmation des Interfaces Utilisateur Graphiques (IUG). En utilisant cette bibliothèque, un programmeur pourra facilement créer une interface gra-

phique composée de fenêtres et d'interacteurs tels que boutons, champs de saisie, etc [1]. Un exemple d'interface graphique est donné sur la figure 1.1. Vous allez donc réaliser une *bibliothèque logicielle* (en gros, un ensemble de fonctions C) destinée à des programmeurs, et non une *application* destinée à des utilisateurs.

Il vous est donné des fonctions pour :

— l'accès aux pixels de l'écran,
— le dessin de texte,
— la réception des actions de l'utilisateur sur le clavier et la souris (événements d'appuis de touche, de déplacement de souris, etc.).

Vous devez réaliser les algorithmes :

— de dessin de primitives graphiques (dessin de lignes, de polygones),
— de configuration et de dessin des interacteurs (boutons, fenêtre, etc.),
— de gestion de la géométrie (position, taille) des interacteurs à l'écran, en particulier lors du changement de taille de la fenêtre,
— de gestion des événements des utilisateurs (exécution des fonctions en réaction aux actions de l'utilisateur sur la souris et le clavier).

Le but du projet étant d'écrire du code en langage C, le principe des algorithmes ci-dessus vous est donné dans ce document. Votre rôle est de *programmer* ces algorithmes. Afin de vous simplifier le problème de *conception* de la bibliothèque, nous vous fournissons les fichiers d'en-têtes (.h) qui contiennent les spécifications des fonctions C correspondant à ces algorithmes, c'est à vous de programmer ces fonctions.

1.2 Survol du projet

1.2.1 Bibliothèque

La bibliothèque logicielle que vous réalisez offre les services suivants :

— **Dessin des primitives graphiques**. Vous réalisez une fonction de dessin de lignes brisées et une fonction de dessin de polygones *pleins* quelconques. Ces deux fonctions doivent être fortement optimisées car elles sont coûteuses en temps d'exécution, mais sont appelées de nombreuses fois par l'application.
— **Création et configuration des interacteurs**. Différentes fonctions permettent de créer différents interacteurs (fenêtres, boutons, labels) et de définir leurs attributs (texte ou image d'un bouton, par exemple).
— **Gestion de la géométrie**. La taille et la position des interacteurs dans leur fenêtre est calculée par un gestionnaire de géométrie. Cela permet de libérer le programmeur des calculs de géométrie quand un bouton s'agrandit, par exemple, suite à un changement de son label, ou à un redimensionnement de sa fenêtre.
— **Gestion des événements**. Votre bibliothèque doit faire en sorte, par exemple, qu'une fenêtre soit déplacée lorsque l'utilisateur clique sur son titre et déplace la souris. Clic et déplacement de la souris génèrent des "événements" que vous recevez et que vous devez traiter. De plus, le programmeur qui utilise votre bibliothèque peut lier une fonction "sauvegarde_document" à un bouton "Save". La bibliothèque est chargée d'appeler cette fonction quand l'utilisateur clique sur ce bouton.

1.2.2 Applications

Afin de tester votre bibliothèque, nous vous fournissons le code source de différents programmes (dessins de formes, affichage de bouton, de fenêtre, jeu de puzzle et jeu 2048). Ces programmes sont présentés dans la section 4.2.

Au début du projet, ces programmes ne peuvent pas compiler : ils ont besoin de votre implémentation de la bibliothèque. Ces programmes vous permettent de tester si votre bibliothèque fonctionne selon les spécifications données. Bien sûr, il serait très imprudent d'attendre la fin du projet pour tester vos développements. Nous vous conseillons de développer vos propres petits programmes de test au fur et à mesure de vos développements. Nous vous donnons également en annexe A des indications pour mener les premières étapes de votre projet.

1. Le nom anglais "widget" (pour WIndow gaDGET) est souvent utilisé à la place d'interacteur.