LES TROPHÉES NSI

Édition 2024

DOSSIER DE CANDIDATURE

PRÉSENTATION DU PROJET

nom de votre projet :	Saturne
membres de l'équipe :	Ywan Gerard
membres de l'équipe :	Clément Coirier
membres de l'équipe :	Lilou Gourdel
niveau d'étude :	première / terminale
établissement scolaire :	Lycée Naval
enseignante/enseignant de NSI :	M. Yohann Pronost

> PRÉSENTATION GÉNÉRALE :

Saturne est une application de génération de code pour la librairie Custom Tkinter (qui sera abrégé par la suite en « Tkinter »). Son but est de permettre de créer des interfaces Tkinter graphiquement, de pouvoir les tester, et de récupérer le code généré pour l'implémenter dans des projets très divers. Ce projet est né de la volonté d'implémenter des interfaces plus rapidement, Tkinter est une librairie complexe à manipuler et comprendre, et l'implémentation manuelle du code est quelque chose de long et fastidieux.
Ainsi, Saturne propose les widgets de bases de Tkinter, accompagnés de tous leurs paramètres, pour permettre à des utilisateurs de tous niveaux en programmation de pouvoir créer leur propre interface avec Custom Tkinter.

> ORGANISATION DU TRAVAIL :

L'équipe est constitué de 3 personnes, de niveaux différents. IL y a tous d'abord Lilou Gourdel en première, en charge de l'implémentation des fonctionnalités de l'application. Clément Coirier, élève aussi en première, en charge de l'implémentation de la génération du code. Enfin il y a Ywan Gerard, en charge de la création des interfaces de l'application.
Nous avons réparti le travail en trois pôles : les interfaces, les interactions au sein des modules, et la génération du code. Chacun a donc pris ce qui lui convenait en fonction de ses capacités et affinités.
Au total une centaine d'heures ont été nécessaires pour réaliser le projet dans la phase présentée, l'intégralité a été réalisé en dehors des heures de cours. Pour communiquer dans le groupe, nous avons utilisé WhatsApp pour discuter au sein du groupe et Github pour partager le code.

> LES ÉTAPES DU PROJET :

Le projet est passé par différentes étapes. Tout d'abord, il y a eu la recherche d'une idée. Une fois l'idée trouvée, nous avons fait une phase de réflexion sur la forme que l'application devait prendre. Puis nous avons réfléchi à l'architecture de l'application, sur le plan des modules, comme des fichiers et dossiers ressources. Nous sommes ensuite entrés dans la phase de développement de l'application, puis du débogage.

> FONCTIONNEMENT ET OPÉRATIONNALITÉ :

A l'heure actuelle, le démonstrateur présenté fonctionne comme il devrait. En d'autres termes, l'ajout de widgets, leur modification, la génération du code et les possibilité d'aperçues et de copie du code fonctionnent. Cependant, les bugs sont encore fréquents, et de nombreux défauts dans le fonctionnement de la génération du code rendent l'application capricieuse. A l'avenir, une refonte du système de génération de code semble nécessaire. Aussi, le fonctionnement des tooltips (les petites boites d'information affichées sous certaines fonctionnalités) est très aléatoire, Une nouvelle forme de boite d'aide est à ajouter.

Pour limiter les bugs Nous avons massivement utilisé le « try/except » de python, couplé à un affichage des erreurs dans la console si il devait y en avoir. Aussi, une phase de test a été faite pour trouver et résoudre les erreurs qui peuvent l'être. Dans l'architecture du programme, la limitation de bugs se fais via un module passerelle entre les interfaces et le cœur du programme. Ce module passerelle (« intermediateLayer.py ») s'occupe d'appeler les différents modules lors

de chaque requête d'un interface. Ainsi, Les données qui sont entrées dans le module passerelle sont traitées en amont pour ne pas avoir d'erreur dans le cœur du programme. De même si une erreur intervient entre le cœur et une interface du programme, on peut savoir plus facilement d'où elle vient car tout passe par ce module.

La principale difficultés a été la transmission des données générales du programme entre les modules (comme le nom du projet, du widget en cours d'utilisation ou encore les données de ce dernier). La solution mise en pratique a été de transmettre les données via les paramètres des fonctions des différents modules.

> OUVERTURE :

Il est important de noter que le démonstrateur présenté n'est pas la version définitive de l'application, et ne sera pas modifié. De nombreuses fonctionnalités nécessitent des connaissances supérieures au niveau terminal. C'est pour cela qu'une autre version du projet existe, incorporant des notions hors terminale.

Parmi les nouvelles fonctionnalités, il est à ajouter de petites boites d'aide pour expliquer et tester les fonctionnalités de chaque widget ; l'ajout de logs pour le débogage ; de nouveaux raccourcis clavier ; les fonctions de retour en arrière/en avant ; et de nouveaux paramètres pour améliorer la personnalisation de l'application.

Aussi, la génération du code doit être modifiée pour la rendre plus optimale. La structuration des widgets dans l'application doit aussi être changée, pour permettre une meilleure lisibilité, et de nouvelles possibilité, pour arriver à terme sur le plus gros ajout de l'application, c'est-à-dire l'ajout des commandes et des variables.

Actuellement, le démonstrateur fonctionne comme il est sensé le faire. Cependant, il reste capricieux. Si nous devions le refaire, nous le referions probablement de la même manière, c'est-à-dire via un travail de conception de l'application, de ses modules, et de son architecture, puis via l'implémentation des fonctionnalités.

Ce concours nous a permis de développer nos connaissances sur les classes, l'utilisation de modules, la gestion de fichiers, et sur le développement de projets d'une taille bien supérieure à ce que l'on peut faire seul en cours de NSI.

Enfin notre application permet à tous de créer des interfaces. Saturne est en effet développée pour permettre autant aux grands débutants d'appréhender la librairie Tkinter, qu'aux codeurs plus expérimentés de créer des interfaces plus simplement et rapidement. Grâce à cela, n'importe qui peut débuter python avec des interfaces.