IUT des Pays de l'Adour - Dpt RT - BUT2
R309 - Programmation événementielle

Programmation événementielle
Python, événements, callbacks,...

Manuel Munier

Version 18 novembre 2024 (11:29)

Objectifs

L'objectif de ce module est de comprendre le fonctionnement de la programmation par événements, c'est-à-dire de d'écrire un programme (en Python) capable de déclencher l'exécution de telle ou telle fonction *callback* pour réagir à la réception d'un événement.

Consignes du projet

- Le développement se fera en langage Python. Votre code devra impérativement être commenté.
- Vous devrez rédiger un court rapport fournissant la description de votre travail. Il ne s'agit pas de simplement fournir le code commenté ! À vous d'expliquer la logique de vos programmes et ce qu'apporte telle couche par rapport à la précédente. Bref, il vous faudra prendre un peu de recul...
- Une courte présentation ainsi qu'une démo seront exigées en fin de projet.

Travail à réaliser

Cette année ce sera de nouveau "figure imposée". En effet, si les applications "Messenger like" sont faciles à comprendre pour s'initier à la programmation événementielle, il en existe déjà malheureusement de trop nombreuses solutions sur Internet. Et trop d'étudiants se contenteraient se recopier le code trouvé sur Internet sans même chercher à comprendre ce qu'il fait ni comment il le fait... Du coup, pour cette année, l'application à développer servira à consulter les références bibliographiques, ce qui devrait vous permettre d'exprimer toutes vos compétences... et uniquement les vôtres...

1. interface graphique en Tkinter $\rightsquigarrow$ partie "dessin", c'est-à-dire création des widgets et des conteneurs, gestion des placements, etc.
2. interface graphique en TKinter $\rightsquigarrow$ partie événements, c'est-à-dire ajout de foncions callback à certains widgets, fonctions callback qui mettent à jour d'autres widgets, etc.
3. communications réseau $\rightsquigarrow$ programmation client/serveur via des sockets TCP/IP
4. multithreading $\rightsquigarrow$ plusieurs threads (légers) capables de lire et/ou modifier des widgets $\rightsquigarrow$ notion de Timer, gestion des connexions réseau, etc.

Pour vous aider voici quelques éléments d'information quant à l'application que vous devrez programmer :

1. Qu'appelle-t-on des références bibliographiques ? Vous trouverez "le nécessaire" sur cet ancien sujet dévaluation Java. Bien évidemment, à vous d'adapter en Python.
2. Quid de la programmation TCP/IP en Python ? Toutes les informations utiles sont disponibles dans le dossier de la SAÉ 302. Cela vous permettra :
   - d'écrire un programme serveur gérant une *Bibliography* avec des services d'ajout, de consultation, de recherche, etc.
   - d'écrire un programme client pour tester "simplement" les services de votre serveur
3. Votre application devra au minimum proposer des widgets pour :
   - renseigner une nouvelle *Publication* (classe mère) et l'enregistrer sur le serveur
   - récupérer la liste de tous les objets *Publication* depuis le serveur et les afficher
   - formulaire de recherche sur différents critères
   - exporter le résultat d'une recherche dans un fichier texte et/ou dans un fichier JSON
4. Vous pourrez aussi ajouter "ce qu'il faut" pour pouvoir créer des objets *Publication* d'une sous-classe précise (et donc avec ses attributs spécifiques).
5. Pour afficher le résultat de votre recherche bibliographique vous pouvez mettre en place des onglets, des tables ou des fenêtres flottantes pour afficher le détail des publications.

Document rédigé en LaTeX sous Linux.