

Programmation événementielle Python, événements, callbacks,...

Manuel Munier

Version 6 octobre 2025 (09:07)

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, et idéalement en programmation objet. 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, Internet regorge d'exemples pour s'initier à la programmation événementielle. Et trop d'étudiants se contenteraient de 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 sera une application de messagerie "WhatsApp like", ce qui devrait vous permettre d'exprimer toutes vos compétences... et uniquement les vôtres...

1. Programmation réseau en Python avec les sockets TCP/IP.
2. interface graphique en Tkinter $\leadsto$ partie "dessin", c'est-à-dire création des widgets et des conteneurs, gestion des placements, etc.
3. interface graphique en Tkinter $\leadsto$ partie événements, c'est-à-dire ajout de fonctions callback à certains widgets, fonctions callback qui mettent à jour d'autres widgets, etc.
4. multithreading $\leadsto$ plusieurs threads (légers) capables de lire et/ou modifier des widgets $\leadsto$ 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. Il existe déjà de nombreuses applications de messagerie avec des fonctionnalités très variées. Sans chercher à les copier entièrement, vous pouvez néanmoins vous en inspirer.
2. Votre application devra au minimum proposer des widgets pour :
 - mise en place d'un serveur de messagerie pour le routage des messages
 - envoyer et recevoir des messages $\leadsto$ nécessité pour les utilisateurs de se connecter au serveur
 - gérer une liste de contacts + informations sur les contacts (nom, prénom, alias, est-il connecté, depuis quelle adresse IP, etc.)
 - exporter le contenu d'une discussion dans un fichier texte et/ou dans un fichier JSON
3. Vous pourrez aussi ajouter la possibilité d'avoir des groupes de plusieurs utilisateurs avec les abonnements, etc. Et donc des discussions de groupe. . .
4. Pour afficher les messages reçus vous pouvez mettre en place des onglets ou des fenêtres flottantes pour séparer les différentes discussions.
5. Il serait intéressant de pouvoir également envoyer des pièces jointes dans vos messages (PDF, images, fichier ZIP, etc.).