Proyecto Final

Nombre del catedrático: Marco Waldo Ángeles Tena

Materia: Programación Concurrente

Nivel: Superior

Carrera: Ingeniería en Software

Nombre de los Alumnos: Fabricio Meneses Ávila, Jorge Ruiz Diaz, Josefa Francisco Hernández, Diego Daniel Magdaleno Medina, Ángel Gabriel Castillo Sánchez

Fecha de elaboración : 20/Noviembre/2024

Grupo: SW_07_03

Descripción general

Este programa es una interfaz gráfica de usuario (GUI) desarrollada con la biblioteca “tkinter” en Python. Su propósito es ejecutar varios scripts de Python relacionados con los temas vistos en clase de programación concurrente a través de una serie de opciones disponibles en un menú interactivo. Los usuarios pueden seleccionar diferentes módulos, como hilos, sockets, semáforos y patrones, para ejecutar tareas concurrentes específicas.

Características principales

- *Interfaz gráfica interactiva*: Usa tkinter para crear una ventana principal con botones que ejecutan acciones específicas.
- *Ejecución de scripts concurrentes*: Al seleccionar una opción, el programa ejecuta un script o función asociada.
- *Submenús dinámicos*: Dependiendo de la opción seleccionada en el menú principal, se despliegan diferentes submenús con opciones específicas para cada módulo (Hilos, Sockets, Semáforos, Patrones).
- *Ayuda e información*: Un menú de ayuda muestra los nombres de los integrantes del proyecto.
- *Documentación*: Una pestana con la documentación
- *Fondo personalizado*: Utiliza una imagen de fondo personalizada.

Requisitos Funcionales (RF):

Número de Requisito	Nombre del Requisito	Tipo	Fuente del Requisito	Prioridad del Requisito	Descripción
RF01	Interfaz gráfica con Tkinter	Funcional	Cliente	Alta	El sistema debe proporcionar una interfaz gráfica utilizando Tkinter para facilitar la interacción.
RF02	Menú interactivo principal	Funcional	Cliente	Alta	La ventana principal debe incluir un menú interactivo para seleccionar entre diferentes módulos.

RF03	Ejecución de scripts concurrentes	Funcional	Cliente	Alta	El sistema debe ejecutar scripts asociados a cada módulo seleccionado sin bloquear la GUI.
RF04	Submenús dinámicos	Funcional	Cliente	Media	El sistema debe desplegar submenús con opciones específicas dependiendo del módulo seleccionado.
RF05	Funcionalidad de ayuda	Funcional	Cliente	Baja	Debe incluir un botón de "Ayuda" que muestre los nombres de los integrantes del proyecto.

RF06	Pestaña de documentación	Funcional	Cliente	Media	La aplicación debe incluir una pestaña donde se pueda consultar la documentación del software.
RF07	Cierre seguro de la aplicación	Funcional	Cliente	Alta	La interfaz debe permitir cerrar la aplicación de manera segura desde un botón o el menú.
RF08	Fondo personalizado	Funcional	Equipo	Baja	La ventana principal debe tener un fondo personalizado (BG.png).

RF09	Diseño de botones consistente	Funcional	Equipo	Baja	Los botones deben tener un diseño uniforme con fondo azul y texto blanco.
RF10	Ejecución de scripts externos	Funcional	Equipo	Alta	Debe ejecutar scripts relacionados con los módulos utilizando la biblioteca subprocess.

Requisitos No Funcionales (RNF):

Número de Requisito	Nombre del Requisito	Tipo	Fuente del Requisito	Prioridad del Requisito	Descripción
RNF01	Interfaz intuitiva	No funcional	Cliente	Alta	La GUI debe ser intuitiva, clara y fácil de usar para cualquier usuario.
RNF02	Responsividad	No funcional	Cliente	Alta	El sistema debe mantener la responsividad de la GUI durante la ejecución de scripts en segundo plano.

RNF03	Modularidad del código	No funcional	Equipo	Media	El código debe estar estructurado de forma modular para facilitar su mantenimiento.
RNF04	Portabilidad	No funcional	Cliente	Alta	El sistema debe ser portable y ejecutable en cualquier plataforma que soporte Python 3.x y Tkinter.
RNF05	Tiempos de respuesta	No funcional	Cliente	Media	La interfaz gráfica debe responder en menos de 500 ms incluso con scripts en ejecución.

RNF06	Escalabilidad	No funcional	Equipo	Media	Debe permitir agregar nuevos módulos o scripts sin cambios significativos en la estructura.
RNF07	Seguridad en scripts externos	No funcional	Cliente	Alta	La ejecución de scripts no debe comprometer la seguridad del entorno del usuario.
RNF08	Diseño visual consistente	No funcional	Equipo	Baja	Debe mantener consistencia en colores, fuentes y tamaños en toda la interfaz.

RNF09	Código documentado	No funcional	Equipo	Media	El código debe incluir comentarios claros para facilitar su comprensión y mantenimiento.
RNF10	Compatibilidad con Python 3.x	No funcional	Equipo	Alta	El programa debe ser compatible con versiones recientes de Python 3.x y Tkinter estándar.

Estructura del código

1. Importación de módulos

Se importan varios módulos de Python y scripts personalizados que proporcionan funcionalidades adicionales:

- tkinter: Para la interfaz gráfica de usuario.
- canvas: Para la visualización del PDF
- *subprocess, os*: Para ejecutar scripts externos

  Módulos personalizados como *hilos_hilos, hilos_con_argumentos, mario_bros_ruleta*, entre otros.

2. **Funciones principales**

- *ejecutar_hilos_hilos():* Ejecuta el script *hilos_hilos.py* utilizando el módulo *subprocess.*

- *mostrar_submenu(titulo, opciones):* Muestra un submenú en la ventana principal. Se crea un nuevo frame con botones que ejecutan las funciones correspondientes a las opciones seleccionadas.

- *ejecutar_accion(opcion):* Ejecuta la acción asociada a la opción seleccionada, verificando si existe la función correspondiente en el diccionario opciones_funciones.

- *salir():* Cierra la ventana principal y termina la aplicación.

- mostrar_documentacion(): Muestra la documentación del software

- *mostrar_ayuda():* Muestra un submenú con la información de los integrantes del proyecto.

3. **Diccionario de funciones (opciones_funciones)**

Este diccionario mapea las opciones de los submenús a las funciones correspondientes, como por ejemplo:

- "Hilos-Hilos" ejecuta la función ejecutar_hilos_hilos().
- "Hilos con argumentos" ejecuta la función hilos_con_argumentos.ejecutar(), si está disponible.

4. **Menú principal y submenús**

El menú principal contiene botones que despliegan submenús con opciones específicas de programación concurrente:

- Hilos: Opciones relacionadas con el manejo de hilos en Python.
- Sockets: Opciones para manejar comunicación cliente/servidor con protocolos TCP/UDP.
- Semáforos: Opciones que incluyen problemas clásicos de semáforos como el Barbero Dormilón y condiciones de carrera.
- Patrones: Implementaciones de patrones de diseño como Futuro/Promesa y Reactor/Proactor.

5**. Configuración de la ventana principal**

La ventana principal se configura con:

- Un fondo personalizado (BG.png).
- Un marco de menú superior (menu_bar) con botones que permiten al usuario navegar entre las diferentes opciones.
- Cada botón en el menú principal tiene un estilo consistente con fondo azul y texto blanco.

Requisitos

- Python 3.x
- Biblioteca tkinter (generalmente incluida en las instalaciones de Python)
- Archivos adicionales como BG.png para la imagen de fondo.

Caso de Uso

- Ejecuta el script principal.
- En la ventana principal, selecciona una opción del menú:
- Hilos: Para trabajar con hilos y tareas concurrentes.
- Sockets: Para probar comunicaciones entre cliente y servidor.
- Semáforos: Para ejecutar soluciones relacionadas con sincronización y semáforos.
- Patrones: Para implementar patrones de diseño como Reactor y Proactor.
- Selecciona la opción deseada y el script asociado se ejecutará en segundo plano.
- Ayuda: Al seleccionar el botón "Ayuda" en el menú, se muestra la siguiente información:

Integrantes del Proyecto:

- Fabricio Meneses Ávila
- Jorge Ruiz Diaz
- Josefa Francisco Hernández
- Diego Daniel Magdaleno Medina
- Ángel Gabriel Castillo Sánchez

Conclusión

Podemos concluir que el programa creado en Python con Tkinter realiza correctamente las funciones que se describieron previamente. Es una GUI escalable que no solo cumple con los criterios de usabilidad, sino que también permite una interacción intuitiva y eficiente para el usuario. Además, su diseño modular y flexible facilita futuras expansiones y adaptaciones según las necesidades específicas.

El desarrollo del programa ha sido una excelente oportunidad para integrar conocimientos teóricos y prácticos adquiridos en la materia de Programación Concurrente de Séptimo Cuatrimestre. Se abordaron temas clave como el manejo de hilos, sincronización y procesamiento paralelo, destacando cómo estos conceptos pueden implementarse en un entorno gráfico interactivo.

En términos educativos, este proyecto refleja el aprendizaje de conceptos fundamentales y su aplicación en el desarrollo de herramientas útiles, reforzando competencias técnicas, analíticas y creativas. Este esfuerzo culmina en una solución funcional y profesional que podría servir como base para aplicaciones más complejas en el futuro.

Referencias Bibliográficas

- Ben-Ari, M. (2006). Principles of Concurrent and Distributed Programming (2nd ed.). Pearson Education.

- Rossainz López, M. (s.f.). Programación Concurrente y Paralela. Material académico disponible en el portal de la Universidad Autónoma de Puebla.

- Andrews, G. R. (1991). Concurrent Programming: Principles and Practice. Addison-Wesley.

- Silberschatz, A., Galvin, P. B., & Gagne, G. (2018). Operating System Concepts (10th ed.).