Sistema de Plugins en EVAR Stat

1. Introducción

EVAR Stat es un software estadístico que permite la gestión, análisis y visualización de datos de manera intuitiva. Una de sus características más potentes es la capacidad de extender sus funcionalidades mediante plugins. Los plugins permiten a los usuarios y desarrolladores agregar nuevas herramientas, análisis, menús y automatizaciones sin modificar el núcleo del programa.

2. ¿Qué es un plugin en EVAR Stat?

Un plugin es un archivo Python (.py) que sigue una estructura específica y que, al ser cargado por EVAR Stat, añade nuevas funcionalidades al programa. Los plugins pueden interactuar con la interfaz gráfica, la tabla de datos, el área de resultados y otros componentes del software.

- Ventajas de los plugins:
  - Permiten personalizar y ampliar el software según las necesidades del usuario.
  - Facilitan la colaboración y el intercambio de herramientas entre usuarios.
  - Separan las funcionalidades adicionales del código principal, facilitando el mantenimiento y la actualización.

3. Gestión de Plugins en EVAR Stat

3.1. Importar un plugin

1. Haz clic en el menú de plugins (ícono de rompecabezas).
2. Selecciona Importar plugin.
3. Elige un archivo .py desde tu equipo.
4. Si el plugin requiere dependencias externas, EVAR Stat intentará instalarlas automáticamente.

3.2. Eliminar un plugin

1. Ve al menú de plugins y selecciona Gestionar plugins.
2. Aparecerá una lista de plugins instalados.
3. Selecciona el plugin que deseas eliminar y confirma la acción.

3.3. Ayuda sobre plugins

En el menú de plugins, selecciona Ayuda sobre plugins para acceder a una explicación sobre el sistema de plugins y cómo utilizarlos.

4. Estructura de un Plugin

Un plugin debe ser un archivo .py que contenga, al menos, una función llamada register_plugin(main_window). Esta función recibe como argumento la ventana principal de EVAR Stat y debe devolver un objeto QMenu (menú de Qt) que será añadido a la barra de menús.

4.1. Ejemplo básico de plugin

Código de ejemplo:

```
from PyQt6.QtWidgets import QMenu, QAction, QMessageBox

__name__ = "Ejemplo Básico"
__version__ = "1.0"
__author__ = "Equipo EVAR Stat"
__description__ = "Un plugin de ejemplo que muestra un mensaje."
__dependencies__ = []  # Si necesitas librerías externas, agrégalas aquí

def register_plugin(main_window):
    menu = QMenu("Ejemplo", main_window)
    accion = QAction("Mostrar mensaje", main_window)
    accion.triggered.connect(lambda: QMessageBox.information(main_window, "Mensaje", "¡Hola desde el plugin!"))
    menu.addAction(accion)
    return menu
```

4.2. Metadatos del plugin

Puedes incluir variables opcionales al inicio del archivo:

- __name__: Nombre del plugin.
- __version__: Versión.
- __author__: Autor.
- __description__: Descripción breve.
- __dependencies__: Lista de dependencias externas, por ejemplo, ["numpy, pandas"]).

Estos metadatos serán leídos por EVAR Stat para mostrar información relevante al usuario.

5. Interacción del Plugin con EVAR Stat

El objeto main_window que recibe la función register_plugin permite acceder a los principales componentes del programa:

- Tabla de datos: main_window.table (QTableWidget)
- Área de resultados: main_window.result_area (QTextEdit)
- Otros métodos y atributos: Puedes acceder a métodos públicos de la ventana principal para realizar tareas como importar/exportar datos, modificar la interfaz, etc.

6. Ejemplo avanzado: Plugin para sumar dos columnas

```
from PyQt6.QtWidgets import QMenu, QAction, QInputDialog,
QMessageBox, QTableWidgetItem

def register_plugin(main_window):
    menu = QMenu("Herramientas Avanzadas", main_window)
    accion = QAction("Sumar dos columnas", main_window)
    def sumar_columnas():
        cols = [main_window.table.horizontalHeaderItem(i).text() for i in
range(main_window.table.columnCount())]
        col1, ok1 = QInputDialog.getItem(main_window, "Columna 1",
"Selecciona la primera columna:", cols, 0, False)
        if not ok1: return
        col2, ok2 = QInputDialog.getItem(main_window, "Columna 2",
"Selecciona la segunda columna:", cols, 0, False)
        if not ok2: return
        idx1 = cols.index(col1)
        idx2 = cols.index(col2)
        nueva_col = f"{col1}_mas_{col2}"

main_window.table.insertColumn(main_window.table.columnCount())

main_window.table.setHorizontalHeaderItem(main_window.table.column
Count()-1, QTableWidgetItem(nueva_col))
        for row in range(main_window.table.rowCount()):
            try:
                v1 = float(main_window.table.item(row, idx1).text())
                v2 = float(main_window.table.item(row, idx2).text())
                main_window.table.setItem(row,
main_window.table.columnCount()-1, QTableWidgetItem(str(v1+v2)))
            except:
                pass
        QMessageBox.information(main_window, "Listo", f"Columna
'{nueva_col}' creada.")
```

accion.triggered.connect(sumar_columnas)
menu.addAction(accion)
return menu

---

7. Recomendaciones y Buenas Prácticas

a. Prueba tus plugins en un entorno seguro antes de usarlos con datos importantes.
b. No uses plugins de fuentes desconocidas para evitar riesgos de seguridad.
c. Documenta tu código y utiliza los metadatos para facilitar la gestión.
d. Declara correctamente las dependencias para que EVAR Stat pueda instalarlas automáticamente.

8. Ubicación y gestión de archivos de plugins

Todos los plugins importados se almacenan en la carpeta plugins dentro del directorio principal de EVAR Stat. Puedes acceder a esta carpeta para editar, eliminar o compartir tus plugins.

9. Conclusión

El sistema de plugins de EVAR Stat es una herramienta poderosa para personalizar y ampliar el software según las necesidades de cada usuario. Gracias a su diseño sencillo y flexible, cualquier persona con conocimientos básicos de Python y PyQt6 puede crear sus propias herramientas y compartirlas con la comunidad.