Klassendiagrammprojekt für Python-Projekte

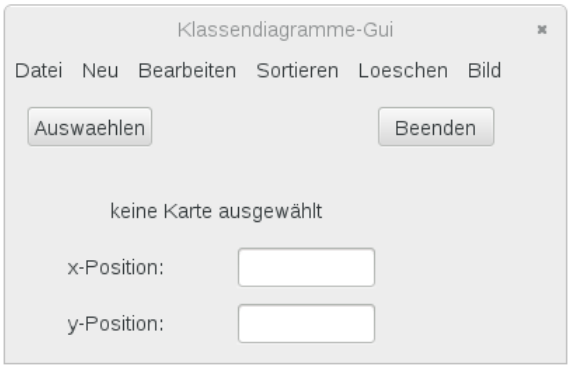
Voraussetzungen

Es handelt sich um ein ***Python***-Projekt, das als Grafiktool ***wxPython*** einsetzt. Es sollte sowohl unter Py2 als auch unter Py3 laufen. Versionsprobleme sind, soweit ich sie festgestellt habe, abgefangen.
Fehler bitte melden

Starten des Programms

Das "Hauptprogramm" heißt **`klassendiagrammApp.py`**.

Es startet die (gesonderte) Benutzeroberfläche (**`gui.py`**) und das Fenster mit der Zeichenfläche. Die Interaktion erfolgt über die als Controller dienende Klasse in **`klassendiagramm.py`**.



Dateien

Vorhandene Projektdateien können aus dem Verzeichnis **`Modelldaten`** geladen werden. Die Dateien sind reine Textdateien, so dass sie einerseits zur Not einfach mit einem einfachen Texteditor bearbeitet werden können und andererseits die Art der Speicherung auch leicht verständlich sein sollte. Die Möglichkeit von Python (***pickle***) Objekte direkt speichern und laden zu können, habe ich nicht eingesetzt.
Im Dateimenü wird als eine Möglichkeit angeboten, das Projekt ohne Attribute und Methoden abzuspeichern. Die ermöglicht einen einfachen Umbau zu übersichtlichen Diagrammen, bei denen es allein auf die Beziehungstypen ankommt.

Bilder

Die dargestellten Diagramme können (Menü **`Bild`**) abgespeichert werden (Ordner **`Bilder`**). Leider funktionieren nur die Dateiformate png und jpg.
Eine besondere Möglichkeit bietet das Umschalten der Darstellung eines Klassendiagramms auf die vergrößerte Darstellung einer einzelnen Klasse im Menü **`Bearbeiten`** mit den beiden Punkten **`Nur eine Klasse zeigen`** und **`Neu alle zeigen`**.

Klassenkarten

Im "Standardmodus" wird man bei einfachen Projekten die Klassen (zumindest teilweise) mit Attributen und Methoden entwickeln. Wegen der Qualität der Bilder von Klassenkarten kann es interessant sein, sie vergrößert abzuspeichern. Siehe dazu im Abschnitt Bilder.

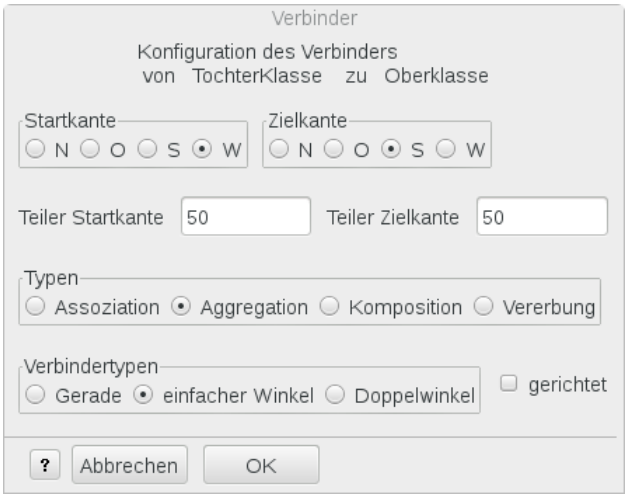
Typangaben

Python verlangt keine Typangaben, obwohl während der Laufzeit beim Zugriff eine strenge Prüfung erfolgt. Wegen dieser Problematik sollte man daher nicht auf die Typangaben verzichten. Lässt man sie weg (**`None`**), werden sie nicht dargestellt.

Sichtbarkeit

Attribute und Methoden sind nicht mit Kennzeichnungen der Sichtbarkeit dargestellt, da dies Projekt für Python-Projekte gedacht ist. Bei Python gibt es kein "übliches" Sichtbarkeitskonzept. Die Sichtbarkeit ist durch vorgestellte (einfache oder) doppelte Unterstreichungsstriche vor den Namen direkt erkennbar und muss daher nicht gesondert gekennzeichnet werden.

Verbinder



(Klasse **`KD_Verbinder`**) gehören zu den Beziehungstypen Vererbung (*ist ein*) und Assoziation (*hat*, *kennt*, *verwendet*, …), Aggregation (*besteht aus*) und Komposition. Die drei letzteren lassen sich mit einer Richtung kennzeichnen und sind untereinander bei Konfiguration austauschbar.
Zu den Verbindern sind jeweils zwei aktive Felder beim Startpunkt und beim Zielpunkt definiert, wo sie auf rechten Mausklick reagieren. Dadurch lassen sie sich konfigurieren.
Im Konfigurationsdialog können gewählt werden

- von Start- und Zielkarte die Kante des Ansatzpunkts,
- wie viel Prozent dieser auf der Kante verschoben sein soll,
- die Typen der Verbinder
- zwischen einer geraden, einfach rechtwinklig oder doppelt rechtwinkligen Linie
- und gegebenenfalls einer Richtung

Menüs

Menü Datei

Siehe oben im Text.

Menü Neu

Die in diesem Menü angebotenen Punkte sollten selbst erklärend sein.

Menü Bearbeiten

Um Klassenkarten, Attribute und Methoden bearbeiten zu können, muss eine Klassenkarte ausgewählt werden. Das geht neben dem Punkt in diesem Menü auch über einen Button auf der Oberfläche. Zur ausgewählten Klasse wird dort der Name und seine Position (editierbar) angezeigt.
Auch Verbinder können aus allen zum Konfigurieren ausgewählt werden.
Zur Darstellung einer einzelnen Klasse siehe oben.

Menü Sortieren

Dient dazu die Reihenfolge der Attribute, Methoden und deren Parameter zu ändern.

Menü Loeschen

Sollte selbst erklärend sein.

Menü Bild

Das jeweilige Bild lässt sich speichern (s.o.).

Mausaktionen

Mit Mausklick lassen sich Klassenkarten auswählen und mit drag-and-drop verschieben. Die Verbinder werden automatisch angepasst. Gegebenenfalls müssen sie neu konfiguriert werden.
Klick in einen freien Bereich löscht die Auswahl.
Doppelklick auf eine Klassenkarte ermöglicht die Bearbeitung von Attributen oder Methoden.
Rechter Mausklick auf eine Klassenkarte ermöglicht die Bearbeitung der Klassenkarte.
Rechter Mausklick auf den Anschluss eines Verbinders an eine Klassenkarte ermöglicht die Konfiguration des Verbinders (s.o.).

Generieren von Programmcode

Die gegebenenfalls notwendigen Parameter im Aufruf des Konstruktors vererbender Klassen müssen per Hand eingefügt werden,
Beachten Sie, dass Python Mehrfachvererbung zulässt, so dass gegebenenfalls mehrere Aufrufe erfolgen müssen.

Abstrakte Klassen

Die Köpfe abstrakter Klassen können leider nicht einfach für Python_3 und Python_2 gleich realisiert werden. Umgesetzt ist die Version für Python_3.
Für Python_2 darf die Metaclass-Angabe nicht als vererbende Klasse eingefügt werden, sondern muss unter der **`class...`**-Zeile durch

```
    __metaclass__ = ABCMeta                    # einfache Einrückung beachten
```

realisiert werden.
Die import-Anweisung bleibt wie bei Python 3.