Das Singleton Entwurfsmuster

Singleton

Begründung

- Die Anwendung greift zum Zeichnen mehrfach auf das Zeichenflaeche-Objekt zu.
- Ziel ist dabei, alle Möbelobjekte in der selben Zeichenfläche darzustellen.

Singleton

Realisierung bei Java:

- Versehe den Konstruktor der Klasse Zeichenflaeche mit dem Sichtbarkeitsattribut private.
- Stelle in der Klasse eine Klassenmethode gibZeichenflaeche() bereit, die genau einmal intern den Konstruktor aufruft, danach immer das vorhandene Objekt zurück gibt.

Singleton

Realisierung bei Python:

- Beim Konstruktor geht nicht *private*.
- Eine Möglichkeit ist, einen Fehler auszulösen, wenn der Konstruktor extern aufgerufen wird.
- Stelle in der Klasse eine Klassenmethode GibZeichenflaeche() bereit, die genau einmal intern den Konstruktor aufruft, danach immer das vorhandene Objekt zurück gibt.

Singleton

Python Beispielcode in Zeichenflaeche:

- Attribut und Methode:

```
 __zeichenflaeche=None
@staticmethod
def GibZeichenflaeche(parent=None):
    if Zeichenflaeche.__zeichenflaeche == None:
        if parent==None:
            return None
        else:
            Zeichenflaeche.__zeichenflaeche=Zeichenflaeche(parent)
    return Zeichenflaeche.__zeichenflaeche
```

Singleton

Python Beispielcode in Zeichenflaeche:

- Konstruktor:

```
def __init__(self, parent):
    """Konstruktor
    !nicht direkt aufrufen!"""
    if Zeichenflaeche.__zeichenflaeche != None:
        raise Exception('Konstruktor nicht direkt aufrufen')
    wx.Panel.__init__(self, parent, -1)
    ...
```