L3.2 Properties

Die *kivy.properties* sind eine wichtige Komponente des Kivy-Frameworks, die speziell dafür entwickelt wurden, dynamische Eigenschaften (**Properties**) in Widgets und anderen UI-Komponenten zu definieren und zu verwalten. Mit diesen Properties können Sie automatisch Änderungen in Attributen (wie Position, Text, Farbe, etc.) effizient zu verarbeiten, was eine reaktive Benutzeroberfläche ermöglicht.

3.2.1 Definition von Properties in der Python-Datei

Properties ist eine Klasse in Kivy, die verwendet wird, um dynamische Attribute zu verarbeiten.
Eine ausführliche Dokumentation des Properties-Packages ist hier zu finden:
https://kivy.org/doc/stable/api-kivy.properties.html#module-kivy.properties

Zu den gebräuchlichsten Properties gehören:

Property-Typ	**Beschreibung**
StringProperty	Speichert einen String-Wert.
NumericProperty	Speichert Zahlenwerte (int oder float).
BooleanProperty	Speichert einen Booleschen Wert (True oder False).

```
1  from kivy.properties import StringProperty
2
3  ...
4
5  class MyGridLayout(GridLayout):
6      info = StringProperty()
7
8  ...
9
10     def newGame(self):
11         self.countMatch = randint(10,30)
12         self.info = "neues Spiel\nAnzahl Streichhölzer = " + str(self.countMatch)
13
14 class StreichholzSpielApp(App):
15     def build(self):
16         self.title = 'Streichholz Spiel'
17         return MyGridLayout(info="Spiel beginnt")
```

main.py

Zeile 1: Um Properties nutzen zu können, muss aus dem Package kivy.properties das passende Modul geladen werden.

Zeile 6: Um Properties zu verwenden, müssen diese auf Klassenebene deklariert werden. Das heißt, direkt in der Klasse, nicht in einer Methode der Klasse

Zeile 12: Sobald ein Objekt der Klasse existiert, kann mit self.*name_der_property* innerhalb der Methoden auf die Properties zugegriffen werden.

Zeile 17: Properties können mit entsprechenden Keyword-Argumenten im Konstruktor der Klasse (hier `MyGridLayout(info="Spiel beginnt")` oder direkt als Parameter beim Erzeugen des Properties (z. B. `info = StringProperty("Spiel beginnt")` initiale Werte erhalten.

3.2.2 Nutzung von Properties in der .kv-Datei

Die in der Python-Datei erstellte Property kann nun in der .kv-Datei genutzt werden. Sobald sich während der Laufzeit des Programms der Wert der Property ändert, wird durch automatische Bindung direct die reactive Synchronisation zwischen der Property und der graphischen Benutzeroberfläche ermöglicht.

```
1 ...
2
3     Label:
4         text: root.info
5
6 ...
```

main.kv

Zeile 3: Das Label wird angelegt, in dem später die Property genutzt werden soll.
Zeile 4: Das Attribut text des Labels erhält mit root.info eine dyanamische Eigenschaft. Hier bezieht sich root auf die Wurzelklasse, die das Layout definiert, z. B. ein GridLayout oder eine benutzerdefinierte Klasse, die in der .kv-Datei verwendet wird.
info ist die Property, die vorher in der Python-Klasse definiert wurde. Sie stellt dynamische Daten bereit, die im Label angezeigt werden.