

Übung 6a: Einführung in die GUI Programmierung mit PyQt4

Ziel dieser Übung ist das Erstellen eines Dialoges zur Steuerung der Simulation des Pendels. Dabei sollen die Parameter des Modells (Masse, Seillänge) in der graphischen Oberfläche anpassbar sein, ebenso wie die Simulationsdauer und die Simulationsschrittweite. Auf Knopfdruck soll die Simulation ausgeführt werden.

Eine häufig vorkommende Aufgabe ist das Speichern von Programmeinstellungen oder Parametern in Konfigurationsdateien (ini-Files). Dazu existiert in der Python Standardbibliothek das Modul ConfigParser, mit dessen Hilfe die Daten aus der Eingabemaske geladen und gespeichert werden sollen.

Übungsaufgabe 1:

Graphische Oberfläche für die Simulationssteuerung (simgui.py)

1. Erstellen Sie Labels und Eingabefelder für die Parameter Massen m_1 , m_2 und die Pendellänge
1. Belegen Sie die Felder mit sinnvollen Werten als Voreinstellung.
2. Wiederholen Sie 1. für die Simulationsschrittweite und die Simulationsdauer.
3. Geben Sie jedem QLineEdit einen Validator für Zahlenwerte.
4. Stellen sie die Textausrichtung in den LineEdits auf rechtsbündig.
5. Legen Sie zwei Buttons für das Simulieren und das Beenden des Programms an.
6. Fügen Sie alle Widgets in ein GridLayout (7x2) ein und ordnen Sie die Buttons rechtsbündig an.
7. „connecten“ Sie den Simulationsbutton mit der simulate-Funktion; alle folgende Aufgaben sind innerhalb der simulate-Funktion zu bearbeiten
8. Holen Sie alle Werte aus den LineEdits und wandeln Sie diese in Fließkommazahlen um.
9. Legen Sie mit arange (aus scipy) eine Zeitachse an (Parameter aus GUI verwenden!).
10. Simulieren Sie das System mit odeint:
 $res = odeint(rhs, y0, t, args = (m1, m2, l))$
Die Startwerte $y0$ können frei gewählt werden.
11. Erzeugen Sie zwei Subplots auf <fig> und stellen Sie die Ergebnisse jeweils für Laufkatze und Last in einem Subplot graphisch dar. Achten Sie auf Achsenbeschriftung, Legende, Hilfslinien etc.

Übungsaufgabe 2: Speichern und Laden der Konfiguration

1. Legen Sie zwei weitere Buttons zum Laden und Speichern an.
2. Verbinden Sie diese mit den Funktionen openFile und saveFile.
3. Kompletieren Sie beide Funktionen (mit saveFile beginnen!) mit Aufrufen der entsprechenden Dialoge für Dateinamen und schreiben/lesen Sie die Daten mit dem ConfigParser Modul:
Schreiben:

```
c = ConfigParser.SafeConfigParser()
c.add_section("Parameter")
c.set("Parameter", "m2", str(mass2Edit.text()))
```

Lesen:

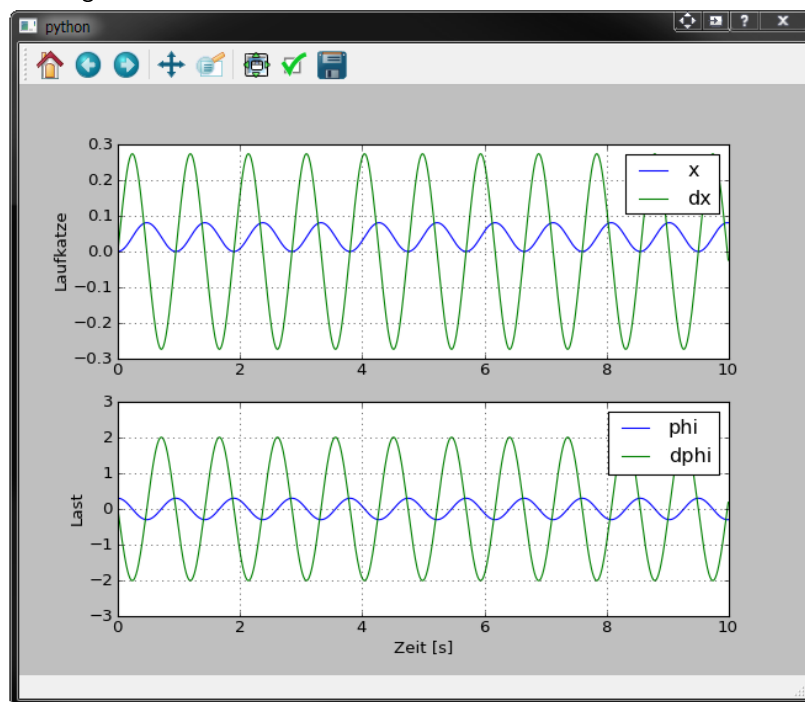
```
mass1Edit.setText(c.get("Parameter", "m1"))
```

Ergebnis und Beispielkonfigurationsdatei siehe Abb. 2

Zusatzaufgabe

- Erweitern Sie das Programm um die Eingabe der Startwerte der Simulation.

Zu Aufgabe 1



A small Python window titled 'python' containing a parameter input form. The parameters and their values are:

Parameter	Value
Masse Laufkatze	0.8
Masse Last	0.3
Pendellaenge	0.5
Schrittweite	0.01
Simulationsdauer	10

Buttons: Simulieren, Exit

Zu Aufgabe 2

A Python window titled 'python' containing a parameter input form. The parameters and their values are:

Parameter	Value
Masse Laufkatze	0.8
Masse Last	0.3
Pendellaenge	0.5
Schrittweite	0.01
Simulationsdauer	10

Buttons: Simulieren, Oeffnen, Speichern, Exit

A 'setup.ini - Editor' window showing the configuration parameters for the simulation. The parameters are:

```
[Parameter]
m1 = 1
m2 = 1
l = 1

[Simulation]
dx = 0.02
tend = 20
```