Persistenz (1)

- Eines der konfusesten Themen in der Programmierung ist das Abspeichern von Daten in Dateien, manchmal auch Persistierung oder Serialisierung genannt. Dabei wäre alles so einfach.
- Beispiel: Drei Punkte (1.0, -2.0), (3.0, -8.0) (3.5, 5.0) bestehend aus x- und y-Koordinaten sollen abgespeichert werden.

  Lösung: Wie bereits in einem der letzten Kapitel behandelt, wird eine Textdatei geöffnet und werden 6 Floats getrennt durch Leerzeichen abgespeichert:

  ```
  1.0 -2.0 3.0 -8.0 3.5 5.0
  ```

  Wenn man die Datei z.B. in einem Editor öffnet, erkennt man aber nicht mehr, was die Zahlen bedeuten. Es könnten auch die Koordinaten von zwei Punkten im dreidimensionalen Raum oder ganz etwas anderes sein.

Persistenz (2)

- Dieses Problem könnte man durch Abspeicherung als csv-Datei beheben. Beispiel:

  1.0, -2.0
  3.0, -8.0
  3.5, 5.0

- Dieses Vorgehen funktioniert gut für gleichförmige Daten, z.B. in jeder Zeile zwei Floats. Für unterschiedliche Anzahlen und Typen von Daten in jeder Zeile ist dieser Ansatz eher ungeeignet.
- Deshalb speichert man Beschreibungstexte oder Tags mit ab.
- Der Nachteil ist, dass die Dateigröße deutlich steigt.
- Wichtig ist, dass nicht jede/r ihr/sein eigenes Format definiert, sondern sich an ein standardisiertes Format hält.
  - Bekannte Standards: JSON, XML

Persistenz (3)

- Beispiel:

```
{
   "description": "destinations of Cobra",
   "points": [
      {
         "x": 1.0,
         "y": -2.0
      },
      {
         "x": 3.0,
         "y": -8.0
      },
      {
         "x": 3.5,
         "y": 5.0
      }
   ]
}
```

JSON (1)

- **JSON** (Java Script Object Notation):
  - Stützt sich auf die Objektdarstellung der Programmiersprache Java Script
- Vergleich zu Python:

Python	Beispiel	JSON	Beispiel
list	["string1", "string2"]	Array	["string1", "string2"]
tuple	(1.5, -0.13)	Array	[1.5, -0.13]
str	"string1"	String	"string1"
int	101	Number	101
float	-0.13	Number	-0.13
(Konstanten)	True, False, None	(Konstanten)	true, false, null
dict	{ "string1": 1.5, "string2": -0.13 }	Object	{ "string1": 1.5, "string2": -0.13 }

JSON (2)

- Ein **Dictionary** (dict), manchmal auch Map genannt, ähnelt einer Liste. Die Elemente werden aber nicht nummeriert, sondern können über frei definierte Strings (Schlüssel) angesprochen.
- Beispiel:

list =	[	"Mustermann"	,	"Max"	,	85049	,	"Ingolstadt"	]
		list[0]		list[1]		list[2]		list[3]	

dict =	{	"name": "Mustermann"	,	"first name": "Max"	,	"code": 85049	,	"city": "Ingolstadt"	}
		dict["name"]		dict["first name"]		dict["code"]		dict["city"]	

- Als Schlüssel sind nicht nur Strings sondern alle unveränderlichen Datentypen erlaubt.