ChatGPT

Python Grundlagen: String- und Datentyp-Methoden (Erklärt wie für Anfänger)

Dieses PDF ist für dich gemacht, damit du ganz in Ruhe und von Grund auf verstehst, wie bestimmte Methoden in Python funktionieren. Du brauchst dafür **kein Vorwissen über andere Module**. Alle Erklärungen sind **klar, kurz, mit Beispielen und Anwendungstipps**.

---

1. `.strip()`

Was es macht:

Entfernt Leerzeichen (oder andere Zeichen) **am Anfang und Ende** eines Strings.

Beispiel:

```
text = "   Hallo Welt   "
sauberer_text = text.strip()
print(sauberer_text)  # Ausgabe: "Hallo Welt"
```

Wofür nützlich?

Wenn du z. B. Benutzereingaben verarbeitest und verhindern willst, dass versehentlich eingegebene Leerzeichen ein Problem verursachen.

---

2. `.lower()` und `.upper()`

Was sie machen:

- `.lower()` macht den Text klein
- `.upper()` macht ihn groß

Beispiel:

```
wort = "Python"
print(wort.lower())  # "python"
print(wort.upper())  # "PYTHON"
```

Wofür nützlich?

Vergleiche ohne auf Groß-/Kleinschreibung zu achten, z. B. bei Passwörtern oder Formatnamen.

---

3. `.isdigit()`

Was es macht:

Prüft, ob der String **nur aus Ziffern besteht** (0-9).

Beispiel:

```
wert = "123"
print(wert.isdigit())  # True

wert2 = "12a3"
print(wert2.isdigit())  # False
```

Wofür nützlich?

Wenn du testen willst, ob ein Wert als Zahl (z. B. für Einstellungen) übernommen werden kann.

---

4. `.replace()`

Was es macht:

Ersetzt einen Teil des Strings durch etwas anderes.

Beispiel:

```
text = "Max(100)"
neu = text.replace("Max(", "").replace(")", "")
print(neu)  # "100"
```

Wofür nützlich?

Sehr hilfreich zum "Aufräumen" von Strings, z. B. Menüs wie "Min(50)" oder "Max(100)" in reine Zahlen verwandeln.

---

5. `.split()`

Was es macht:

Teilt einen String an bestimmten Stellen (z. B. bei Leerzeichen oder Kommas).

Beispiel:

```
text = "Name, Alter, Beruf"
teile = text.split(", ")
print(teile)  # ['Name', 'Alter', 'Beruf']
```

Wofür nützlich?

Zerteilen von Listen, Benutzereingaben oder Datei-Inhalten.

---

6. `.startswith()` / `.endswith()`

Was sie machen:

Prüfen, ob ein String mit etwas beginnt oder endet.

Beispiel:

```
filename = "bild.jpeg"
print(filename.endswith(".jpeg"))  # True
```

Wofür nützlich?

Erkennung von Dateiformaten oder bestimmten Eingaben.

---

7. `.find()` / `.index()`

Was sie machen:

- `.find()` gibt den Index (also die Position) zurück, an der ein Teilstring vorkommt
- `.index()` ist wie `.find()`, **aber gibt einen Fehler, wenn nichts gefunden wird**

Beispiel:

```
text = "Hallo Welt"
print(text.find("W"))  # 6
print(text.find("Z"))  # -1
```

Wofür nützlich?

Zum Suchen und Positionieren von Daten in Strings.

---

8. `in` (kein Punkt davor!)

Was es macht:

Prüft, ob ein bestimmter Teil im String enthalten ist.

Beispiel:

```
text = "Max(100)"
if "100" in text:
    print("Zahl gefunden")
```

Wofür nützlich?

Allgemeine Prüfung von Inhalten, ohne kompliziertes Parsen.

Bonus: Kombination dieser Methoden

Beispiel:

```
wert = "Max(100)"
if wert.startswith("Max(") and wert.endswith(")"):
    zahl = wert.replace("Max(", "").replace(")", "")
    if zahl.isdigit():
        zahl = int(zahl)
        print(zahl)  # Ausgabe: 100
```

So könntest du sehr einfach auch Combobox-Werte verarbeiten und in richtige Zahlen verwandeln.

Fazit

Mit diesen Methoden kannst du **fast alle Strings in deinem Programm analysieren, umwandeln und kontrollieren**, ohne dass du andere Bibliotheken oder Module brauchst. Wenn du das verstanden hast, bist du bereit, deine GUI noch intelligenter zu machen!