MICROCONSULT

Dauerbrenner Python: Eine Einführung in die beliebte Multiparadigmen-Sprache

Unter den vielen Programmiersprachen, die bei Entwicklern beliebt sind, gibt es eine, die sich wie ein Chamäleon an die unterschiedlichsten Anforderungen anpassen kann: Python. Schon lange ist Python als eine der am weitesten verbreiteten Programmiersprachen bekannt. Doch was macht das Coden mit Python so besonders? Werfen wir einen Blick darauf, warum es sich lohnt, Python in Projekten einzusetzen.

Python hat sich bereits in einer Vielzahl von Anwendungsgebieten bewährt. Sei es in der Datenanalyse, im Machine Learning, im Web-Development, beim Testen, Skripten oder in der Prototypen-Entwicklung – Python ist heute nahezu überall im Einsatz. Ob in der Wissenschaft, der Finanzwelt, dem Bildungsbereich oder in der Softwareentwicklung, Python hat sich in den wichtigsten Branchen als eine äußerst flexible Lösung erwiesen.

Was steckt hinter der Beliebtheit von Python?

Doch was macht Python so attraktiv für Entwickler auf der ganzen Welt? Überzeugende Alleinstellungsmerkmale sind die einfache Syntax und die gute Lesbarkeit der Sprache. Viele bezeichnen sie sogar als „quasi Pseudocode“, was bedeutet, dass der Code fast wie eine natürliche Sprache daherkommt und dadurch für viele Einsteiger sofort leicht zu verstehen ist. Darüber hinaus ist Python eine Hochsprache, die plattformunabhängig arbeitet und auf verschiedenen Betriebssystemen und Architekturen ausgeführt werden kann. Ein weiterer großer Pluspunkt: Python ist kostenlos und verfügt über ein riesiges Ökosystem, das ständig wächst und sich weiterentwickelt.

Die Multiparadigmen-Sprache: Ein wesentliches Merkmal von Python ist die Fähigkeit, verschiedene Programmierparadigmen zu unterstützen. Ob objektorientierte Programmierung, funktionale Programmierung oder prozedurale Programmierung – Python bietet ein hohes Maß an Flexibilität und unterstützt problemlos unterschiedliche Programmierstile.

Objektorientierung: Python unterstützt alle wichtigen Konzepte der objektorientierten Programmierung, einschließlich Abstraktion, Datenkapselung, Vererbung und Polymorphie. Dies ermöglicht es Entwicklern, komplexe Systeme auf effiziente und gut strukturierte Weise zu entwerfen und zu implementieren.

Funktionale Programmierung: Neben der objektorientierten Programmierung bietet Python auch Unterstützung für funktionale Programmierkonzepte. Dazu gehören Immutability, Higher-Order-Functions, Generatoren, Lambda-Funktionen und zahlreiche eingebaute Funktionen wie *map*, *filter*, *enumerate*, *zip*, *all* und *any*. Diese Funktionen eröffnen den Entwicklern immer neue Möglichkeiten zur Gestaltung und Implementierung ihrer Programme.

Ein weiterer wichtiger Aspekt von Python ist, dass es keinen konkreten Programmierstil erzwingt. Der Nutzer kann je nach Bedarf und Anforderungen wählen, welcher Stil oder auch welche Kombination am besten zur aktuellen Aufgabe passt. Im Gegensatz dazu erzwingen einige andere Sprachen wie beispielsweise C# einen bestimmten Stil, zum Beispiel die objektorientierte Programmierung.

Ökosystem bringt eine Fülle von Möglichkeiten

Das Python-Ökosystem ist riesig und bietet eine breite Palette von Bibliotheken und Frameworks, welche die tägliche Arbeit von Entwicklern vereinfachen und beschleunigen. Die Standardbibliothek allein bietet bereits zahlreiche Module für Aufgaben wie File-Handling, Datum und Uhrzeit, Textanalyse, Serialisierung und Deserialisierung sowie Concurrency. Darüber hinaus gibt es eine Fülle von Third-Party-Modulen, die über die Python Package Index (PyPI) verfügbar sind. Diese Module decken fast jeden denkbaren Anwendungsfall ab, darunter den Umgang mit Webservices, numerische Berechnungen, Datenanalyse, Datenvisualisierung, Machine Learning, Desktopentwicklung und Testing. Viele dieser Module gehören sprachübergreifend zu den am weitesten verbreiteten Frameworks und Bibliotheken, wie aus dem StackOverflow Developer Survey hervorgeht.

Beispiel für das vielfältige Ökosystem: Nutzung von pandas und plotly

Ein hervorragendes Beispiel, das die Vielseitigkeit des Python-Ökosystems unterstreicht, ist die Verwendung von pandas und plotly, um Daten zu analysieren und visuell darzustellen. Angenommen, Sie haben eine CSV-Datei mit stündlichen Daten und möchten diese in tägliche Werte zusammenfassen, um einen besseren Überblick zu erhalten. Hier kommt pandas ins Spiel. Mit nur wenigen Zeilen Code können Sie mithilfe von pandas die Daten einlesen, aggregieren und in das gewünschte Format bringen. Zum Beispiel können Sie die Summe der stündlichen Werte für jeden Tag berechnen.

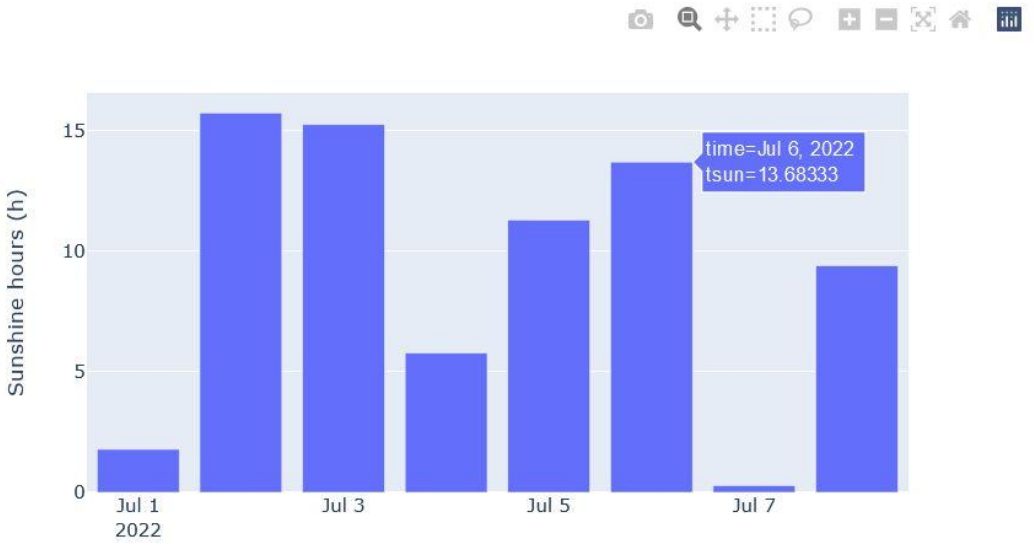
```
import pandas as pd
import plotly.express as px

raw_data = pd.read_csv("weather.csv", parse_dates=[0], index_col=0)

daily_data = raw_data.resample("1d").sum()

figure = px.bar(daily_data, x=daily_data.index, y="tsun")
figure.update_layout(yaxis_title="Sunshine hours (h)", xaxis_title="")
figure.show()
```

Mit plotly können Sie dann die aggregierten Daten schnell und einfach visualisieren, zum Beispiel als Balkendiagramm. Dieses Beispiel verdeutlicht, wie mühelos es ist, komplexe Datenanalysen durchzuführen und ansprechende Visualisierungen zu erstellen, dank der leistungsstarken Kombination von pandas und plotly in der Python-Welt.



Die vollständige HTML-Datei zu diesem Beispiel steht hier für Sie zum Download bereit.

Mit Python sind Ihnen keine Grenzen gesetzt

Dem Ruf als äußerst vielseitige und flexible Programmiersprache, die sich für eine Vielzahl von Anwendungen eignet, wird Python nach wie vor voll gerecht. Ihre einfache Syntax, gute Lesbarkeit, Unterstützung verschiedener Programmierparadigmen und das umfangreiche Ökosystem machen sie zu einer beliebten Wahl für Entwickler auf der ganzen Welt. Ob Sie nun Daten analysieren, Webanwendungen entwickeln, Machine Learning-Modelle trainieren oder einfach nur Prototypen erstellen möchten – mit Python sind Ihnen keine Grenzen gesetzt.

Das **MicroConsult Python-Training** schafft für Sie eine fundierte Basis für die Programmierung mit Python. Dazu gehört insbesondere der Umgang mit komplexen Datentypen und die objektorientierte Programmierung mit Python. Darüber hinaus werden die wichtigsten Python-Bibliotheken der Standardbibliothek (z.B. os, sys, pickle, json, datetime, tkinter, sqlite3) und von Drittanbietern (z.B. pytest, requests, numpy, pandas, Flask, plotly) in Theorie und Praxis vorgestellt. Durch eine große Zahl praktischer Übungen bekommen Sie ein Gefühl für die Sprache und das effektive Entwickeln und Testen von Programmen.

Weiterführende Informationen

MicroConsult Training: Python – Objektorientierte Skriptsprache

MicroConsult Fachwissen: Softwareentwicklung

Alle Trainings & Termine auf einen Blick

Der MicroConsult-Newsletter

Wir informieren Sie mehrmals jährlich über Trends und Best Practices im Embedded Systems Engineering.

Erhalten Sie wertvolles Fachwissen und Tipps aus erster Hand von unseren Embedded-Experten!

Jetzt abonnieren!

Autor

Dr.-Ing. Thomas Schütz ist als freier Mitarbeiter bei MicroConsult tätig. Er arbeitet seit 2013 im akademischen und anschließend im industriellen Umfeld an Softwarelösungen zur Simulation und Optimierung von physikalischen Systemen sowie Fragestellungen im Bereich der Datenanalyse. Er verfügt hierbei über Erfahrungen sowohl in der Software-Architektur als auch der konkreten Implementierung. Thomas Schütz ist Referent für Python- und C#-Trainings.