C-EXTENSIONS: WIE SIE PYTHON-CODE AUF DIE ÜBERHOLSPUR BRINGEN

DR. MAIK WEBER

11.03.2024

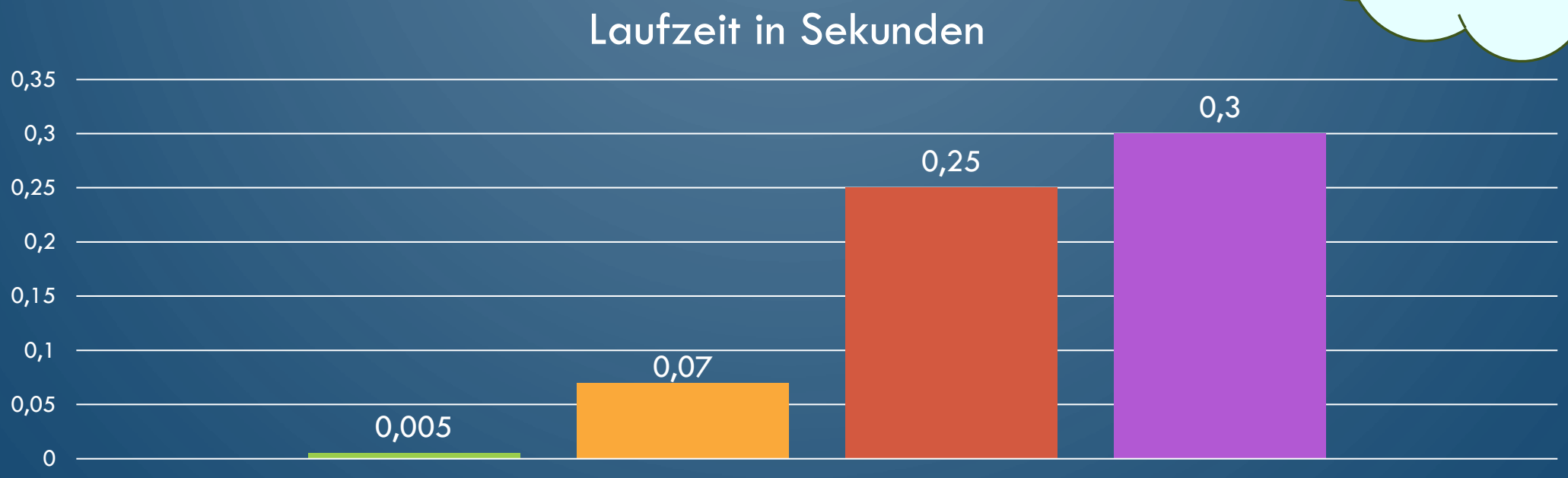
PROBELEHRVERANSTALTUNG AN DER HOCHSCHULE TRIER

PROGRAMMIERSPRACHEN IM WETTSTREIT

QUELLE: HTTPS://WWW.THOMASCHRISTLIEB.DE/ (2018)

Aufgabe: Approximation von π durch die Leibniz-Reihe (10^6 Iterationen)

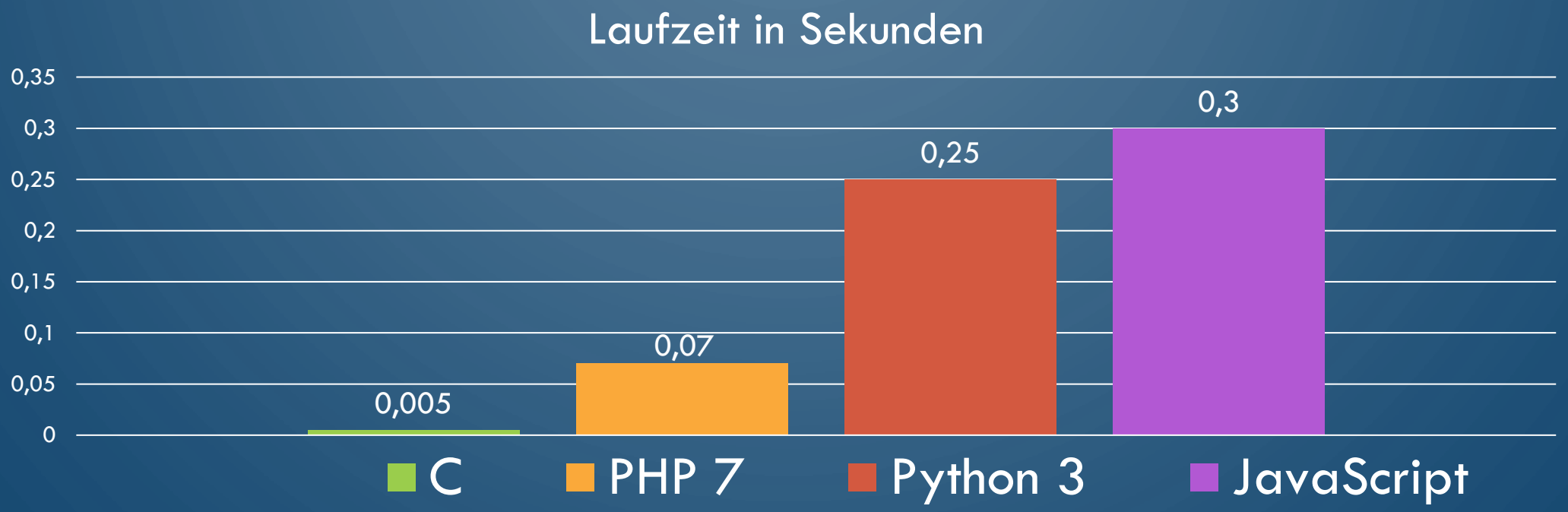
PHP7
JavaScript
C
Python3



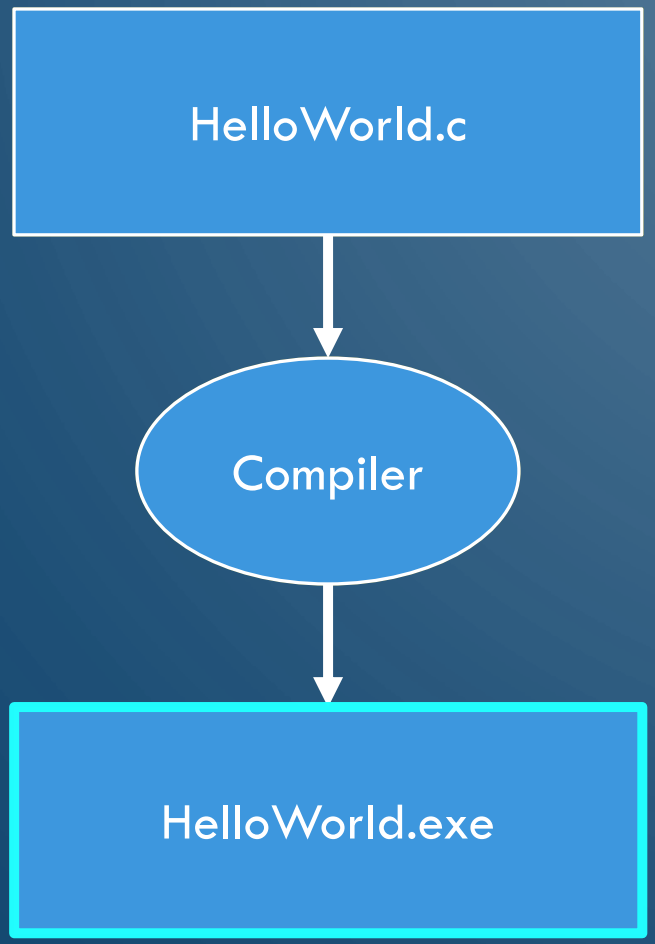
PROGRAMMIERSPRACHEN IM WETTSTREIT

QUELLE: HTTPS://WWW.THOMASCHRISTLIEB.DE/ (2018)

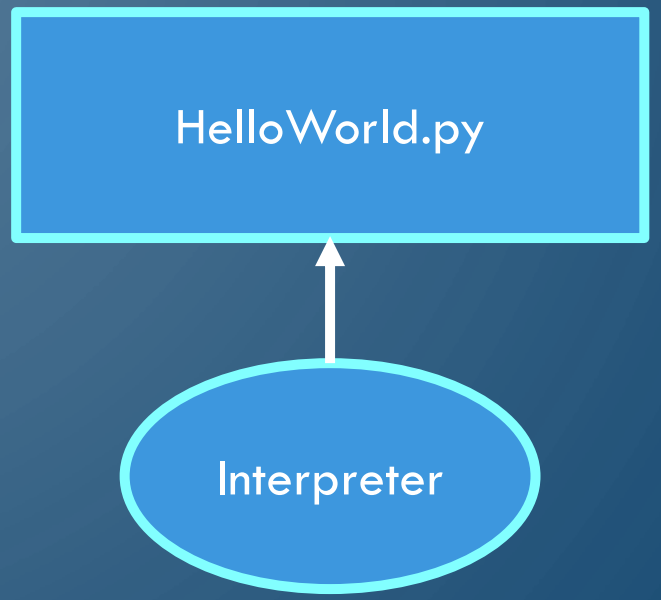
Aufgabe: Approximation von π durch die Leibniz-Reihe (10^6 Iterationen)

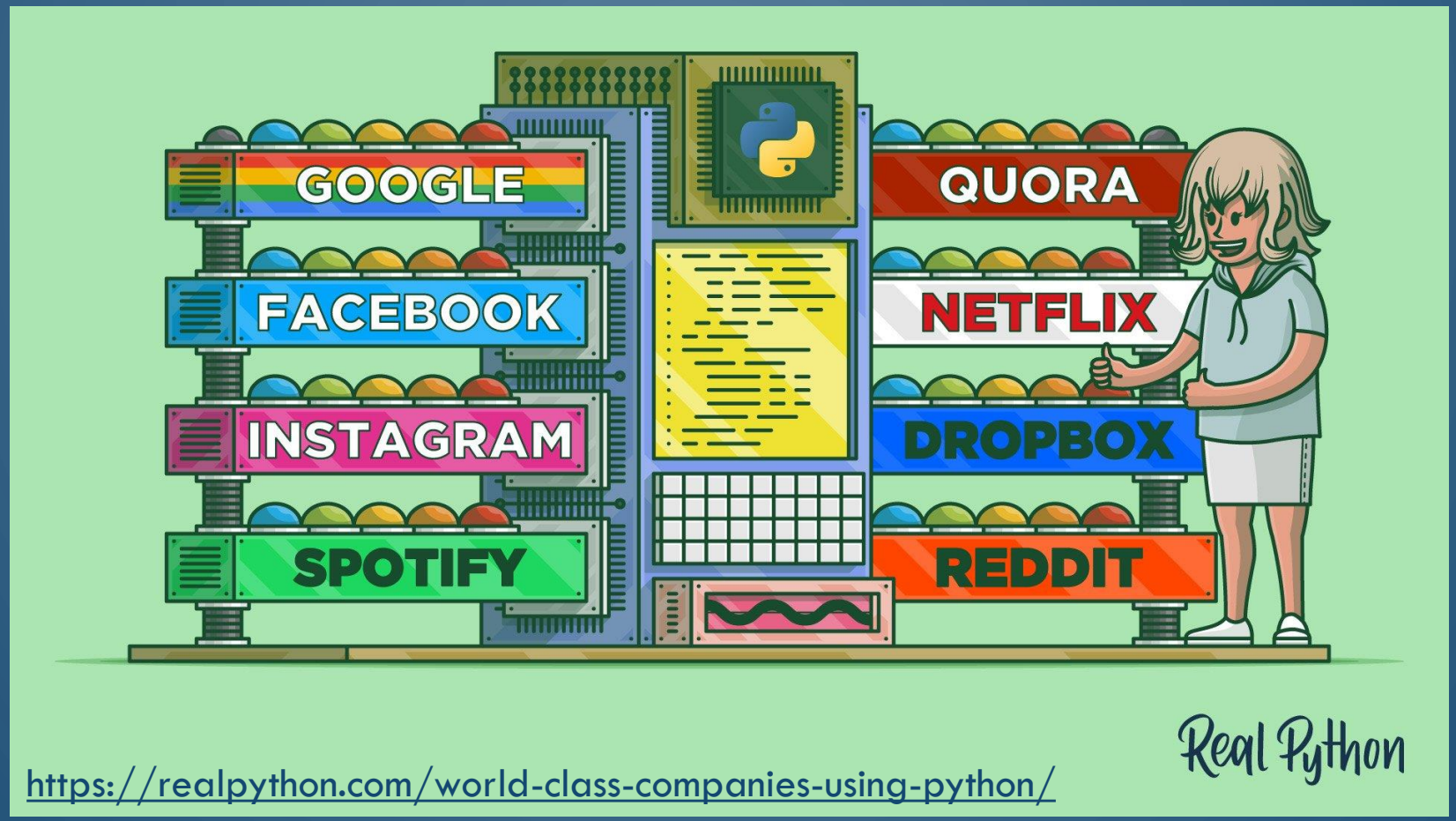


C


HelloWorld.c
Compiler
HelloWorld.exe

PYTHON


HelloWorld.py
Interpreter



https://realpython.com/world-class-companies-using-python/

*"We initially chose to use Python because of its reputation for **simplicity and practicality**, which aligns well with our philosophy of 'do the simple thing first.'"*

Min Ni, Instagram

C-EXTENSIONS: WIE SIE PYTHON-CODE AUF DIE ÜBERHOLSPUR BRINGEN

- Motivation ✓
- Praktisches Beispiel: Bildbearbeitung
  - Pure Python
  - C-Code als Shared Library
  - C-Code als Python-Modul (C-Extension)
- C-Extensions in der Praxis
- Quiz

5 SCHRITTE ZUR C-EXTENSION

1. **API einbinden:** `#include<Python.h>`
2. **Wrapper schreiben:**

   Input: Python-Objects zu C-Typen konvertieren — `PyArg_ParseTuple(…)`

   Funktionalität ausführen: C-Code

   Output: Ergebnis(se) zu Python-Object konvertieren — `Py_BuildValue(…)`
3. **„Buchhaltung“:** Method Definitions und Module Definition für Interpreter anlegen
4. **Module Initialization Function anlegen**
5. **Packaging:** Modul bauen und installieren bzw. verteilen (mehrere Möglichkeiten)

   direkte Installation: `setup.py` → `pip install ./`

   (zum Verteilen: `python -m build` → `Wheel-Datei`)

Link zur Dokumentation: https://docs.python.org/3/extending/extending.html

C-EXTENSIONS IN DER PRAXIS

Bekannte Python-Module, die in C oder C++ implementiert wurden:

- Numpy (aufbauend: SciPy)
- Pandas
- TensorFlow und PyTorch
- Matplotlib
- PyCrypto
- …

Praxistipps:

1. Verwenden Sie stets bereits verfügbare, optimierte Module in Ihren Programmen!
2. Falls nicht verfügbar: Identifizieren Sie langsame Routinen und lagern Sie ggf. einzelne Funktionen als C-Code in eine Bibliothek aus.
3. In größeren Projekten und bei häufiger Wiederverwendung: Erstellen Sie eine C-Extension als eigenständiges Python-Modul.

DOKUMENTE UND ÜBUNGSAUFGABE

Auf GitHub: https://github.com/Maik-Weber/C-Extensions-Trier

- Slides und alle Files zur Vorlesung.
- **Übungsaufgabe** „Caesar Verschlüsselung“ im Ordner `Uebung-Caesar`. Das Jupyter-Notebook `caesar-cipher.ipynb` führt Sie durch alle Aufgaben.

Ich kam, sah und siegte!

Lfk ndp, vdk xqg vlhjwh!