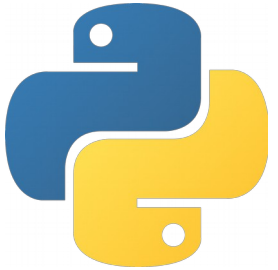
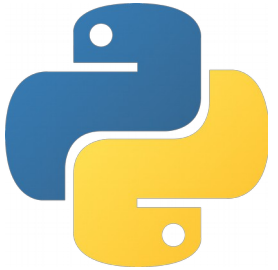
Python
Erstellen einer Distribution



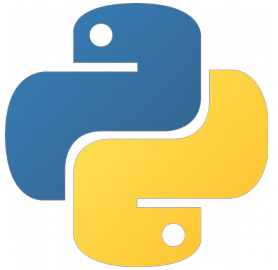
- Distribution von Python-Projekten
  - Erstellen einer Binärdistribution
    - Neben einer Quellcodedistribution ist das Erstellen einer Binärdistribution von besonderem Interesse, da diese den wenigsten Installationsaufwand hat
    - Umgekehrt bedeutet es allerdings mehr Arbeit für Sie, da für verschiedene Betriebssysteme ganz unterschiedliche Formate für Binärdistributionen erstellt werden müssen
    - Das prominenteste dieser Formate ist ein Windows Installer, aber auch RPM-Pakete für RPM-basierende Linux-Distributionen können erstellt werden

Python
Erstellen einer Distribution



- Distribution von Python-Projekten
  - Erstellen einer Binärdistribution
    - Beachten Sie, dass Sie neben einer Binärdistribution stets auch eine Quellcodedistribution Ihres Projekts veröffentlichen sollten, da es Betriebssysteme gibt, die weder mit einem RPM-Paket noch mit einem Windows Installer etwas anfangen können
    - Zum Erzeugen einer Binärdistribution wird das Installationsscript mit den Argumenten auf der folgenden Folie aufgerufen

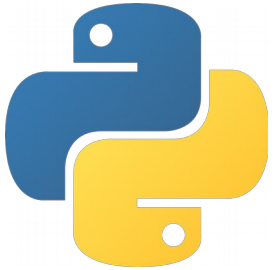
Python
Erstellen einer Distribution



- Distribution von Python-Projekten
  - Erstellen einer Binärdistribution

Argument	Bedeutung
`bdist_msi`	Erzeugt einen Windows MSI-Installer, der dazu da ist, ein Modul bzw. Paket auf einem Windows-System zu installieren. Es wird die Datei *Projektname-Version.win32.msi* im Unterordner *dist* erzeugt.
`bdist_rpm`	Erzeugt ein RPM-Paket für RPM-basierende Linux-Distributionen wie beispielsweise Fedora Core oder openSUSE. Es wird die Datei *Projektname-Version.src.rpm* im Unterordner *dist* erzeugt.
`bdist_wininst`	Erzeugt einen Windows Installer, der dazu da ist, ein Modul bzw. Paket auf einem Windows-System zu installieren. Es wird die Datei *Projektname-Version.win32.exe* im Unterordner *dist* erzeugt.

Python
Erstellen einer Distribution



- Distribution von Python-Projekten
  - Erstellen einer Binärdistribution

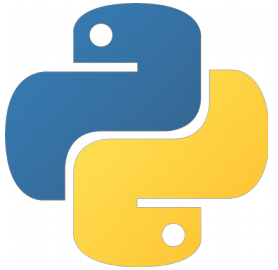
    - Da alle Informationen, die zum Erstellen der Binärdistribution benötigt werden, bereits im Installationsscript angegeben wurden, ist das Erzeugen einer Binärdistribution tatsächlich mit den folgenden Aufrufen von setup.py erledigt

      setup.py bdist_wininst
      setup.py bdist_rpm

    - Hinweis
      - Solange Ihr Projekt aus reinen Python-Modulen besteht, also weder Pakete noch Extensions umfasst, kann die Installationsdatei für Windows auch unter anderen Betriebssystemen, beispielsweise unter Linux, erzeugt werden

      - Sobald aber Pakete oder Erweiterungen enthalten sind, muss dafür ein Windows-System verwendet werden

Python
Erstellen einer Distribution



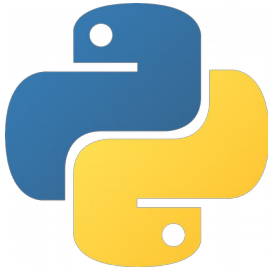
- Distribution von Python-Projekten
  - Distributionen installieren

    - Nachdem Sie jetzt das grundlegende Handwerkszeug zum Erstellen einer Binär- und Quellcodedistribution erlernt haben, sollen hier noch ein paar Worte zur Verwendung der Distributionen selbst folgen

    - Zu einer Binärdistribution brauchen wir dabei nicht viel zu sagen, denn die Installationsprozedur entspricht dem auf dem jeweiligen Betriebssystem üblichen Vorgehen

    - Wie eine Quellcodedistribution installiert wird, ist hingegen nicht ganz intuitiv und sollte bei Ihren eigenen Distributionen unbedingt in einer Readme-Datei erklärt werden

Python
Erstellen einer Distribution



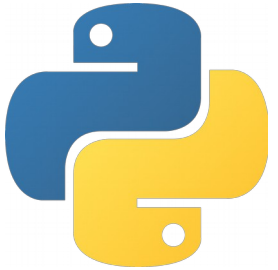
- Distribution von Python-Projekten
  - Distributionen installieren

  - Die Installation einer Quellcodedistribution verläuft nach dem folgenden Schema

  → Herunterladen der Distribution
  → Entpacken des Archivs
  → Ausführen der Programmdatei setup.py mit dem Argument install

  - Sie sehen, dass auch für die Installation einer Distribution die Programmdatei setup.py verantwortlich ist
    setup.py install

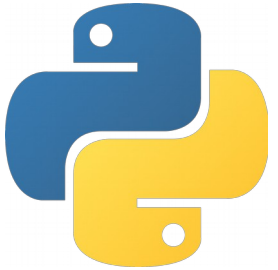
Python
Erstellen einer Distribution



- Distribution von Python-Projekten
  - Distributionen installieren
    - Wenn die Programmdatei setup.py mit dem Argument install ausgeführt wird, installiert sie die Distribution in die Python-Umgebung, die auf dem System installiert ist
    - Beachten Sie, dass dafür, je nach System, Administrator- oder Root-Rechte erforderlich sind
    - Hinweis
      - Die Distribution wird in das Standardverzeichnis für Python-Drittanbieterbibliotheken des Systems installiert
      - Wenn Sie dies nicht wünschen, können Sie über das Argument --prefix ein Zielverzeichnis vorgeben

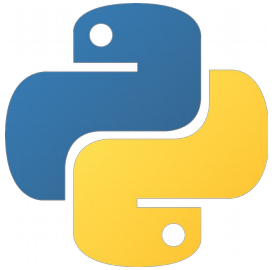
        python setup.py install --prefix="Pfad/Zum/Zielverzeichnis"

Python
Erstellen einer Distribution



- Distribution von Python-Projekten
  - Erstellen von EXE-Dateien – cx_Freeze
    - Mit dem distutils-Paket lassen sich Distributionen aus Python-Projekten erzeugen, die dann auf dem Zielsystem im Kontext einer existierenden Python-Umgebung installiert werden können
    - Besonders unter Windows ist es manchmal wünschenswert, ein Programm als einfache ausführbare Datei auszuliefern, die ohne weitere Voraussetzungen auch auf Systemen läuft, auf denen keine Python-Umgebung installiert ist
    - Eine solche Distibution kann mit dem Drittanbietermodul cx_Freeze erstellt werden, das Sie unter http://www.cx-freeze.sourceforge.net herunterladen können

Python
Erstellen einer Distribution



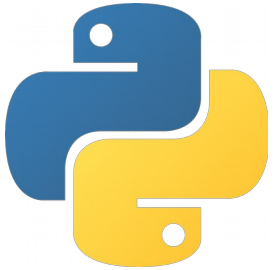
- Distribution von Python-Projekten
  - Erstellen von EXE-Dateien – cx_Freeze
    - Alternative wäre dann py2exe (http://www.py2exe.org/)
    - In diesem Abschnitt verwenden wir das Modul cx_Freeze, um aus einem kleinen Beispielprogramm eine ausführbare Datei zu schnüren
    - Quelltext des Beispielprogramms

```
import sys

if len(sys.argv) > 2:
    print("Ergebnis: {}".format(int(sys.argv[1]) +  int(sys.argv[2])))
```

    - Es handelt sich dabei um ein einfaches Programm, das zwei als Argument übergebene Zahlen addiert und das Ergebnis ausgibt

Python
Erstellen einer Distribution



- Distribution von Python-Projekten
  - Erstellen von EXE-Dateien – cx_Freeze

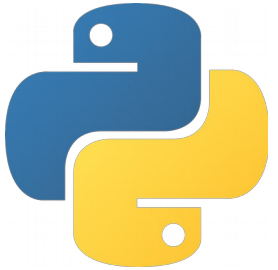
    - Es gibt zwei Möglichkeiten, das Modul cx_Freeze zu verwenden
      → über das Skript cxfreeze, dem der Pfad zur zu bearbeitenden Programmdatei als Argument übergeben werden muss

      → alternativ in Kombination mit distutils
      Dazu schreiben wir im Kontext des obigen Beispielprogramms die folgende Programmdatei setup.py

```
from cx_Freeze import setup, Executable

setup(
        [...]
        executables = [Executable("calc.py")]
        )
```

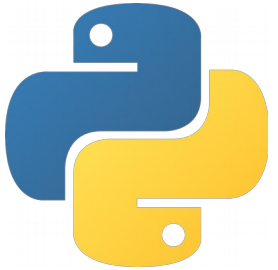
Python
Erstellen einer Distribution



- Distribution von Python-Projekten
  - Erstellen von EXE-Dateien – cx_Freeze
    - Um mit cx_Freeze eine ausführbare Datei zu erstellen, muss der Schlüsselwortparameter executables angegeben werden
    - Für diesen wird eine Liste von Executable-Instanzen übergeben, die jeweils eine zu erzeugende EXE-Datei repräsentieren
    - Nachdem das Installationsscript fertig ist, kann die ausführbare Datei erzeugt werden
    - Dazu muss das Installationsscript setup.py mit dem Argument build aufgerufen werden

      setup.py build

Python
Erstellen einer Distribution



- Distribution von Python-Projekten
  - Erstellen von EXE-Dateien – cx_Freeze
    - Der Rest geschieht automatisch
    - Nachdem sich das Installationsscript beendet hat, finden Sie im Programmverzeichnis den Unterordner dist, der die fertige Distribution Ihres Python-Programms enthält
    - Diese beinhaltet nicht nur die ausführbare Datei selbst, in diesem Fall calc.exe, sondern noch weitere für das Programm benötigte Dateien
    - So sind beispielsweise der Python-Interpreter in der DLL python32.dll und die benötigten Teile der Standardbibliothek im Archiv library.zip ausgelagert