

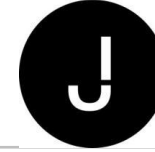


JAVACREAM

*Training
Consulting
Projectmanagement*

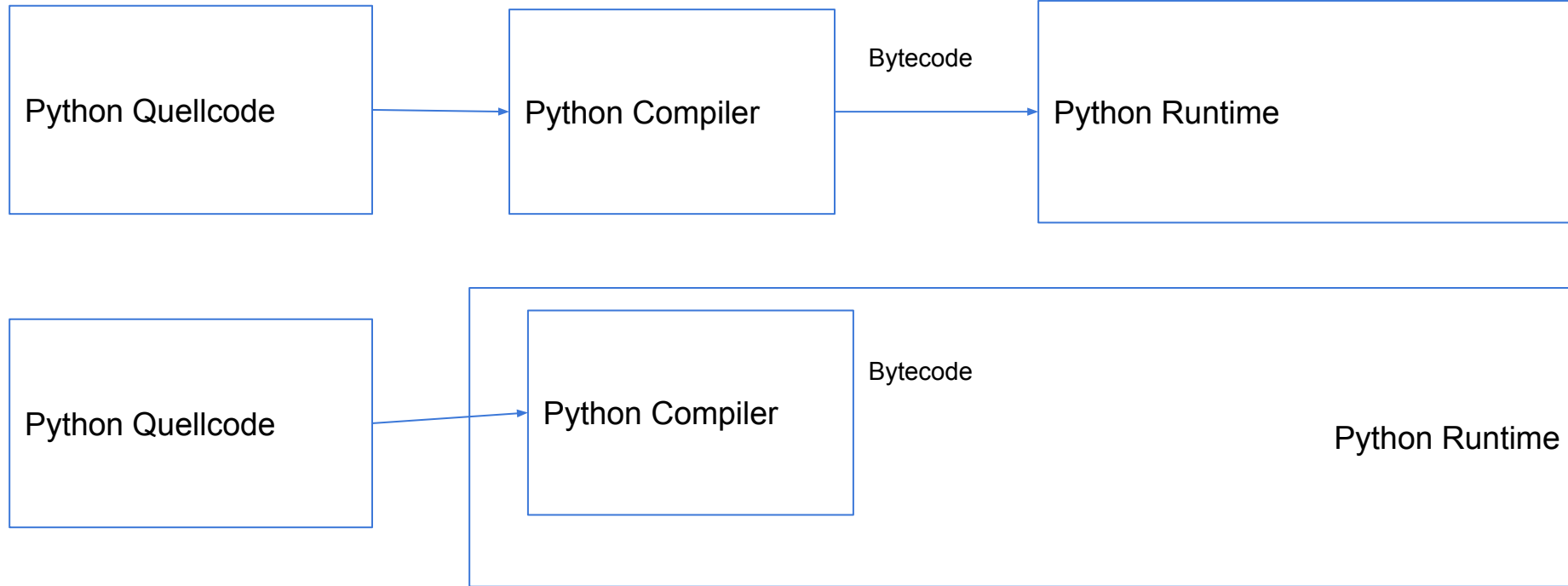
Python

- Name
- Rolle im Unternehmen
- Themenbezogene Vorkenntnisse
- Aktuelle Problemstellung
- Konkrete individuelle Zielsetzung



Übersicht

- Python Runtime
 - Häufig genannt “der Python Interpreter”
 - Implementierungen für verschiedene Plattformen / Betriebssysteme sind vorhanden
 - Damit ist Python “Plattform-unabhängig”
 - Es gibt im Gegensatz zu z.B. Java keine Spezifikation
 - Dieser Effekt spielt heute keine große Rolle mehr
- Programmiersprache Python
 - Eigentlich eine “ganz normale” Programmiersprache
 - Einordnung
 - Sequenzielle Programm-Ausführung
 - Laufzeit-Typisiert, optional: eine statische Typisierung ist möglich
 - Objekt-orientierte



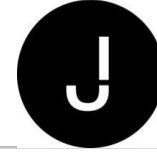
- Installation einer Python Runtime
 - Native Installation auf einem Betriebssystem



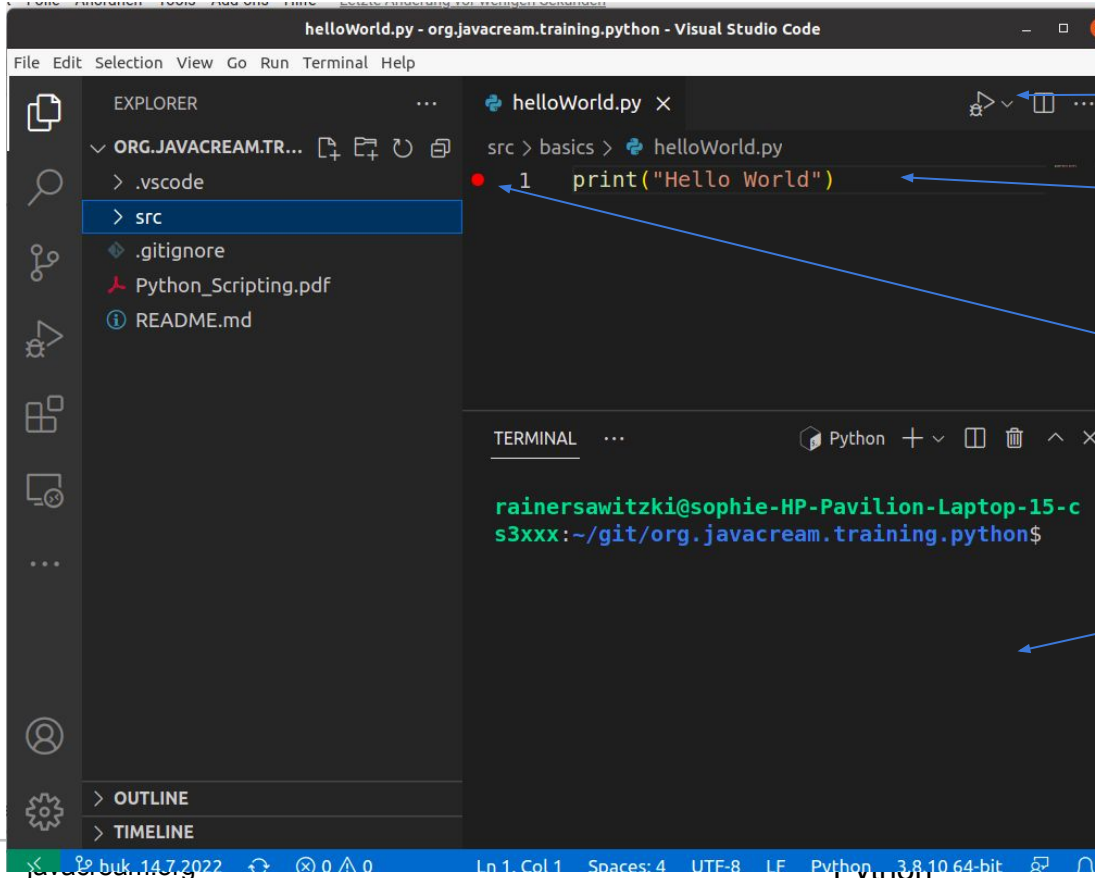
- Verwendung eines Docker-Containers

rainersawitzki@sophie-HP-Pavilion-Laptop-15-cs3xxx: ~

```
rainersawitzki@sophie-HP-Pavilion-Laptop-15-cs3xxx:~$ python3
Python 3.8.10 (default, Mar 15 2022, 12:22:08)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> 
```



- Entwicklungsumgebung
- **Visual Studio Code**
 - Alternativ: Eclipse-basierte Lösung, Visual Studio, IntelliJ
-

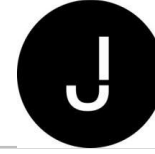


Starten / Debuggen eines
Python Programms

Editor mit
Syntax-Highlighting
und Code Assist

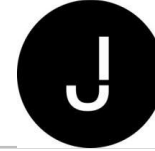
Haltepunkt gesetzt "im
Quellcode"

Konsole / Terminal



Python First Contact

- Literale
 - Zeichenketten
 - "...", "
 - Zahlen
 - 9, 4.2, 4.2e-9
 - Logische Werte, Booleans
 - True, False
 - + ein paar mehr...
-



- Operatoren
 - Mathematische Operatoren
 - +, -, *, /, %
 - Zeichenketten-Konkatination
 - +
 - Vergleichs-Operatoren mit logischem Rückgabewert
 - >, <, >=, <=, ==, !=
 - Logische Operatoren
 - and, or, not, is, is not
- Kontrollstrukturen
 - If - Else
 - Arbeitet mit Einrückungen
 - if, else, elif
 - ab Python 3.10: match, das entspricht einem switch-case

- **Tupel**
 - Eine geordnete, unveränderbare Liste mit erlaubten Duplikaten
 - Literal
 - `tuple = (element1, element2, element3)`
 - `value = tuple[1]` # -> element2
 - `#tuple[3] = element4` #Fehler, Tupels sind unveränderbar
- **List**
 - Eine geordnete, veränderbare Liste mit erlaubten Duplikaten
 - Literal
 - `list = [element1, element2, element3]`
 - `value = list[1]`
 - `list[3] = element4`
 - `list[1] = element5`

- Set
 - Eine ungeordnete, veränderbare Liste ohne Duplikate
 - Literal
 - `set = {element1, element2, element3}`
 - Zugriff auf Elemente nur über Iteration
- Dictionary
 - Eine ungeordnete, veränderbare Liste mit key=value-Paaren ohne Key-Duplikate
 - `dictionary = {"spring": "ok", "summer": "great", "autumn": "not so great", "winter": "depressive"}`
 - Zugriff auf Elemente erfolgt mit dem key
 - `value = dictionary["summer"] #-> great`

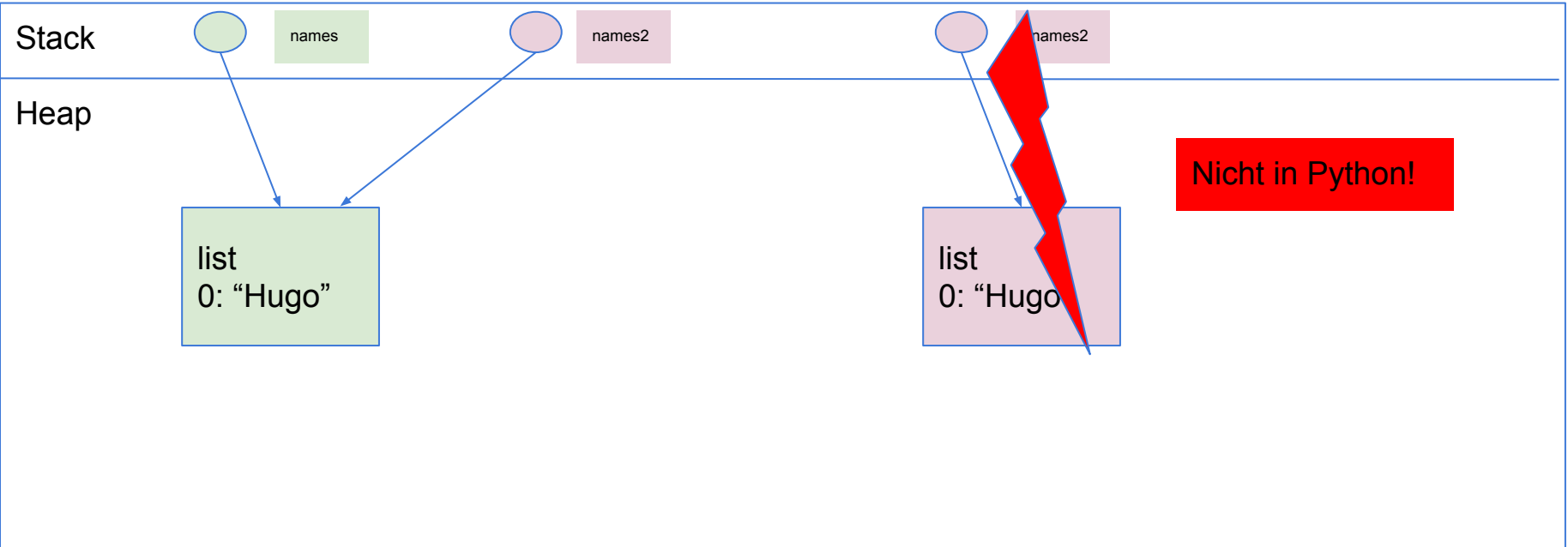
ab Python 3.x bleibt die
Ordnung der keys
erhalten

```
names = ["Hugo"]
```

Eine Variable in Python ist immer eine Referenz auf ein Objekt im Heap

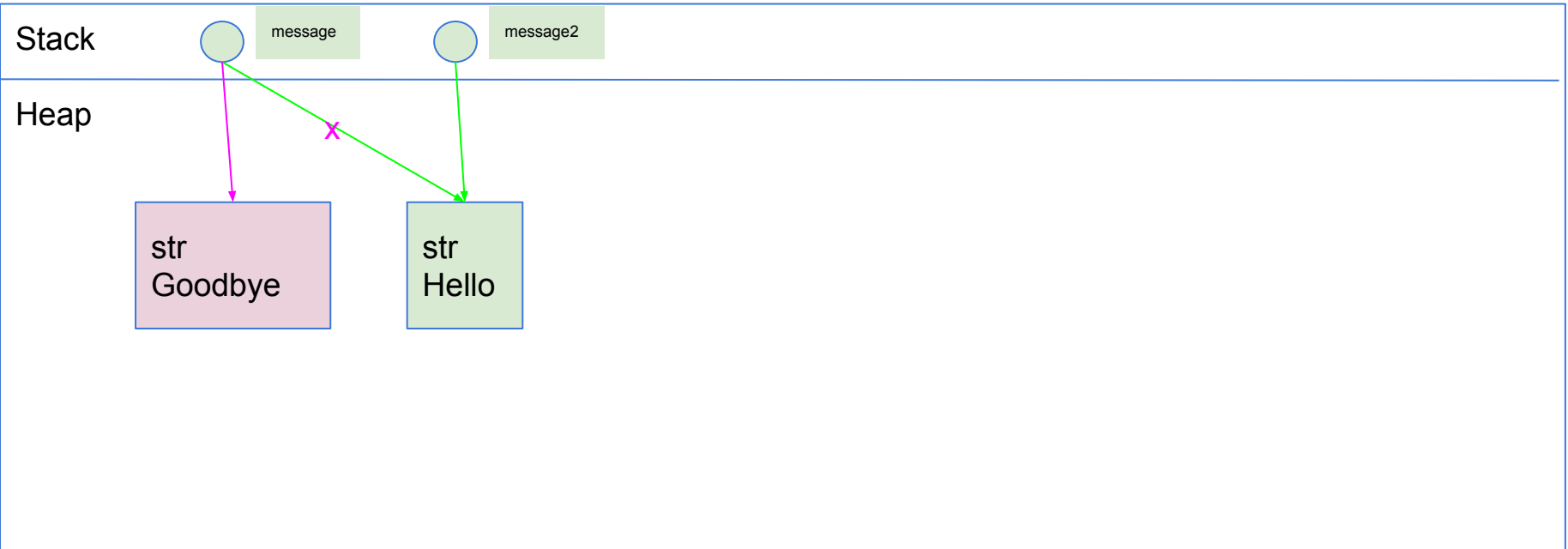
```
names2 = names
```

Eine Zuweisung in Python ist immer eine Kopie des Wertes der Referenz



```
message = "Hello"  
message2 = message
```

```
message = "Goodbye"
```





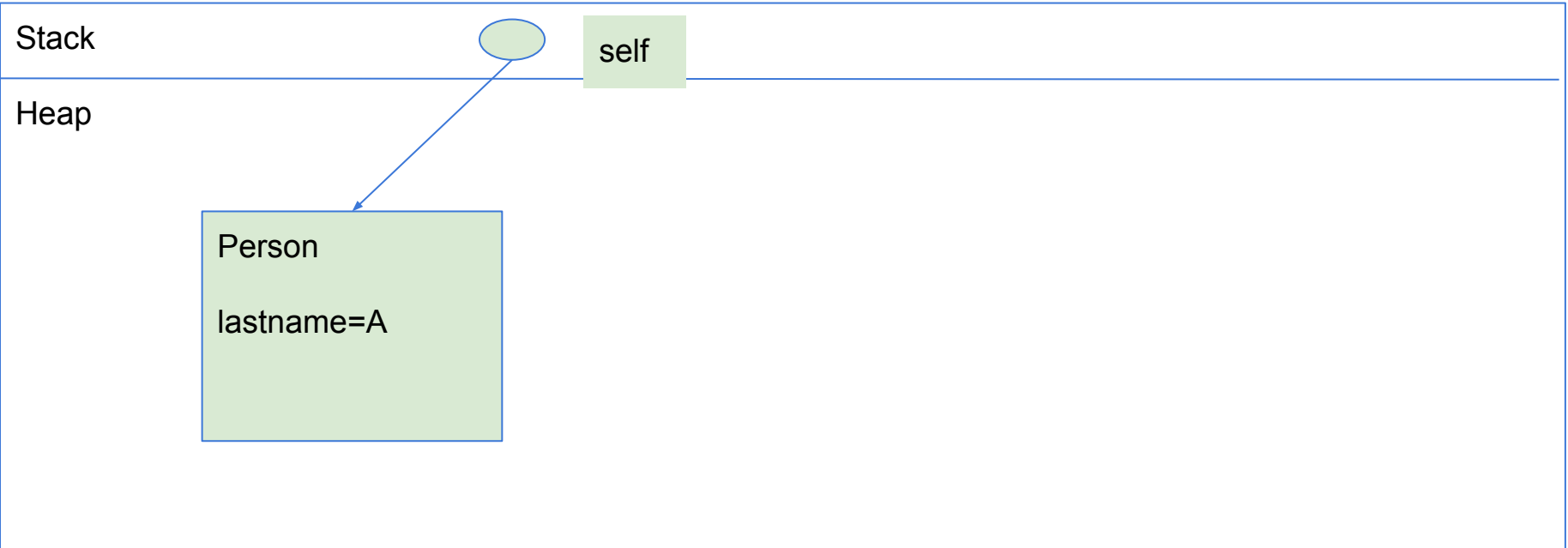
- Variablen und Funktionsnamen nur Kleinbuchstaben
- Zur besseren Lesbarkeit: snake-Konventionen
 - nicht personname, sondern person_name
 - Andere Sprachen: camel-Case, personName
- Benutzerdefinierte Datentypen (-> etwas später) beginnen mit einem Großbuchstaben und sind Came Case

Person

- + lastname
- + firstname

Person("A", "B")

```
__init__(lastname, firstname):  
    self.lastname = lastname
```



Person

- + lastname
- + firstname
- + say_hello()

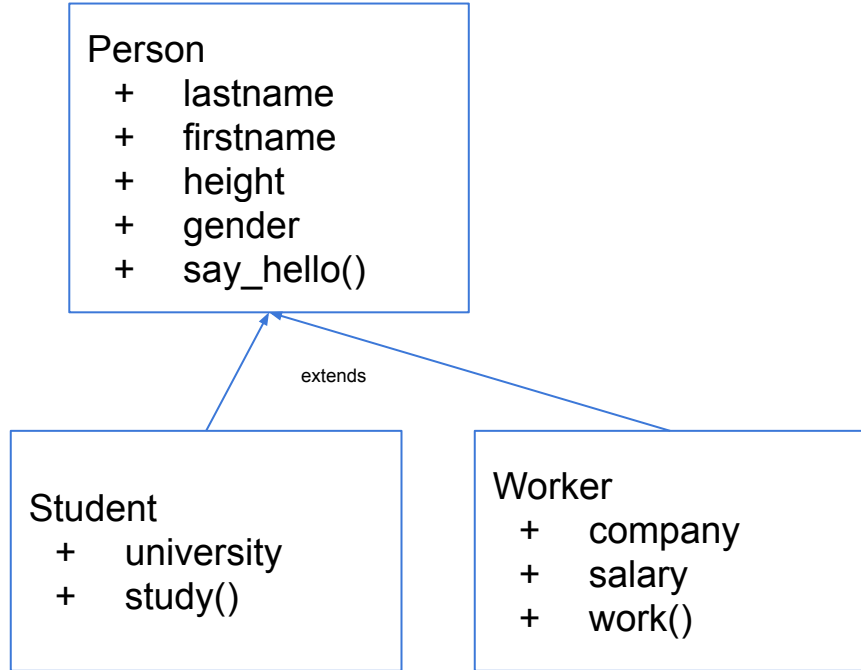
Attribute
Fields

Methods

```
Person
+ lastname
+ firstname
+ height
+ gender
+ say_hello()
```

say_hello soll auch
den Vornamen mit
ausgeben

ToDo: Umsetzung dieses Modells

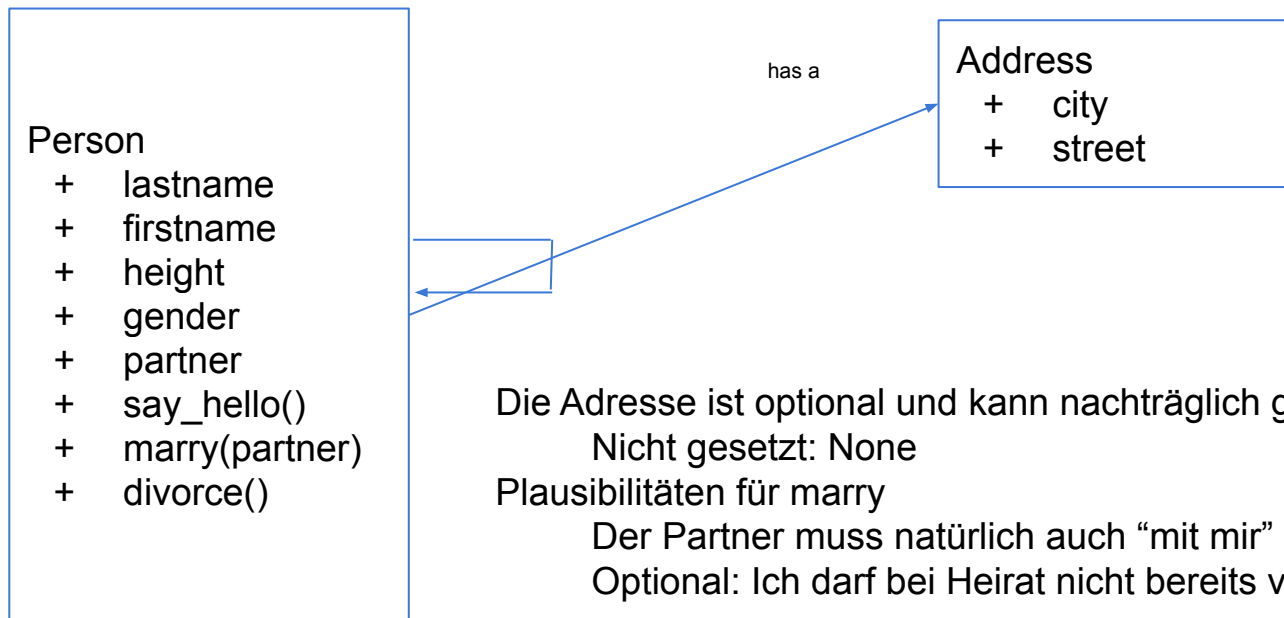


Python-Scratch

```
class Person:
    ...

class Student(Person):
    ...

class Worker(Person):
    ...
```



Die Adresse ist optional und kann nachträglich gesetzt werden

Nicht gesetzt: None

Plausibilitäten für marry

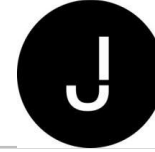
Der Partner muss natürlich auch “mit mir” verheiratet werden

Optional: Ich darf bei Heirat nicht bereits verheiratet sein

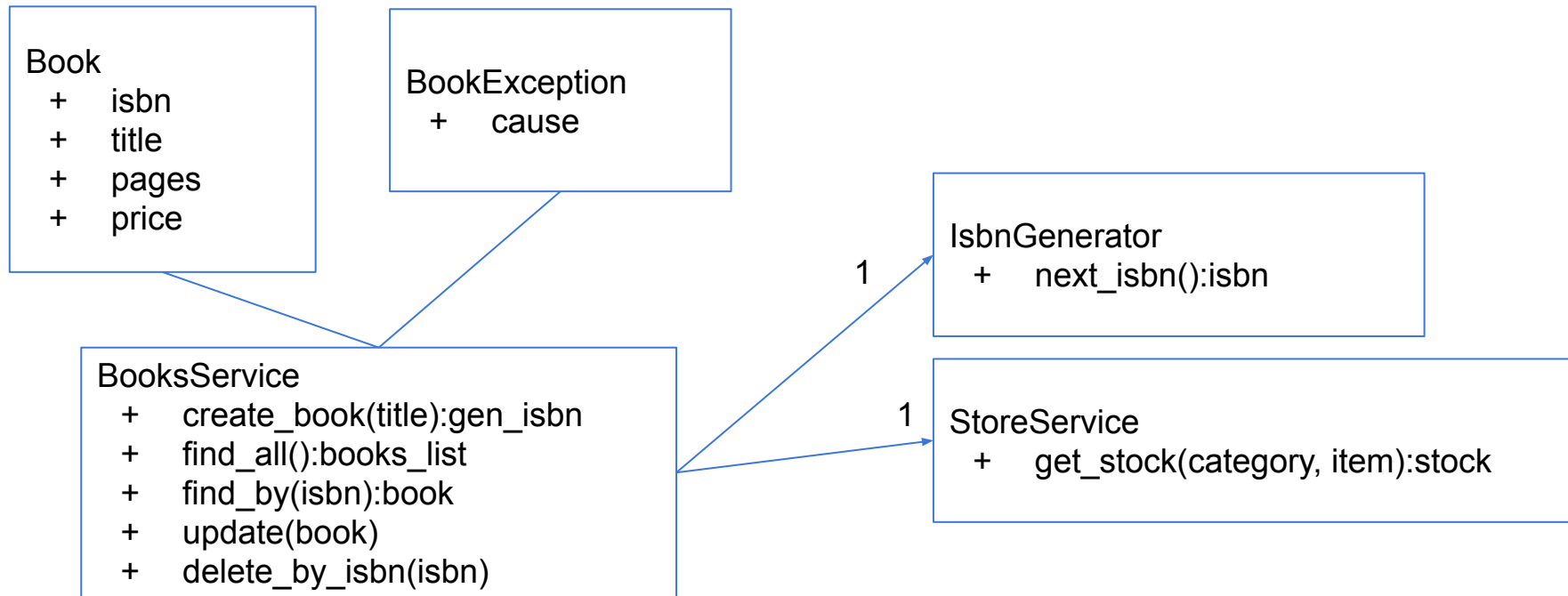
...

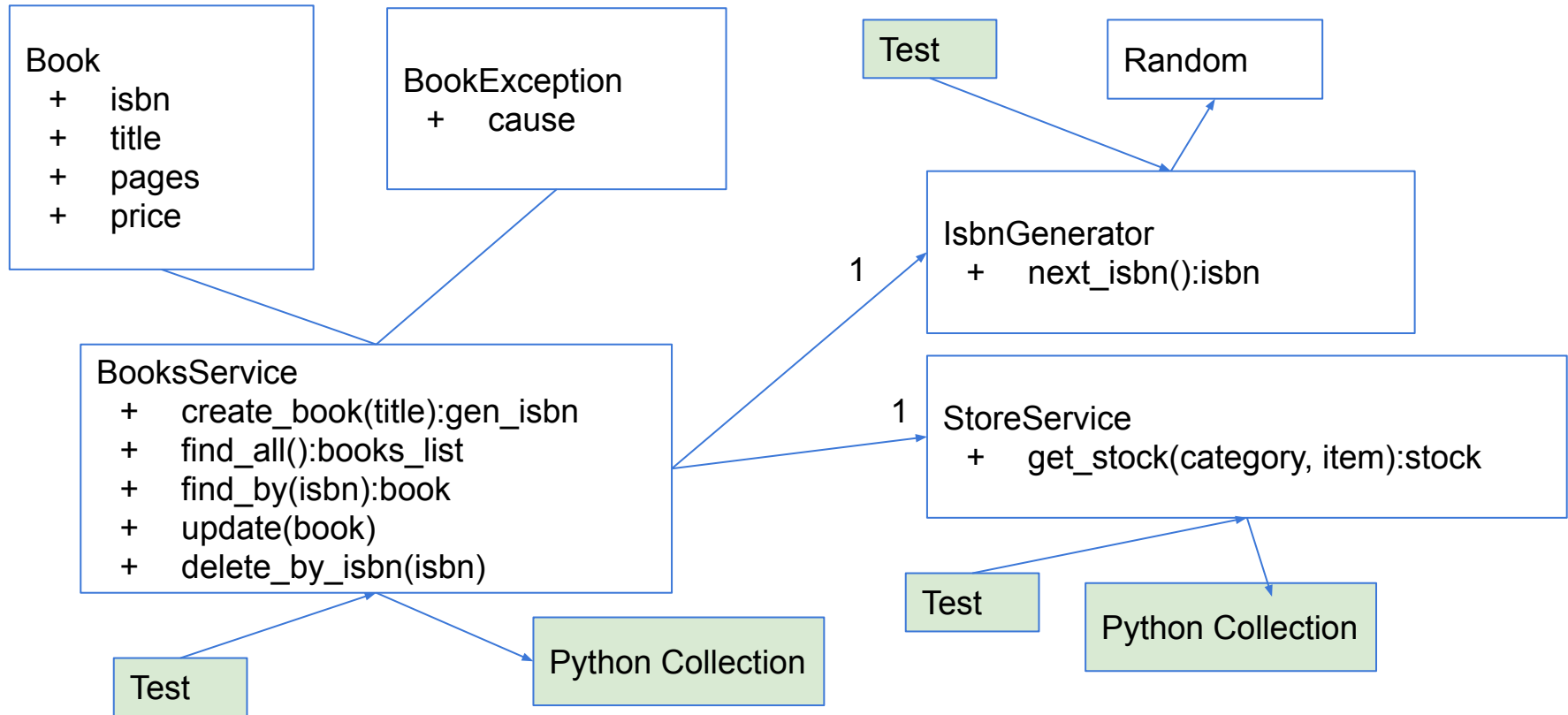
- Wie konsequent ist die Objekt-Orientierung umgesetzt?
 - Arbeiten mit Referenzen ist umgesetzt
 - Verfolgen der Referenzen erfolgt über den '.'-Operator
 - Python-Runtime implementiert einen Garbage Collector
 - (Polymorphes Verhalten ist umgesetzt)
 - "Alles" ist ein Objekt
 - Built-In-Functions von Python haben keinen Bezug zu Objekten
 - z.B. zur Bestimmung der Länge einer Liste gibt es die Built-In function len(list)
 - oop: list.length
 - Übersicht
 - https://www.w3schools.com/python/python_ref_functions.asp
 - Python stellt eine umfangreiche Klassenbibliothek zur Verfügung
 - Typische Funktionalitäten / Algorithmen werden als Methoden von Klassen bereitgestellt
 - z.B. String https://www.w3schools.com/python/ref_string_capitalize.asp

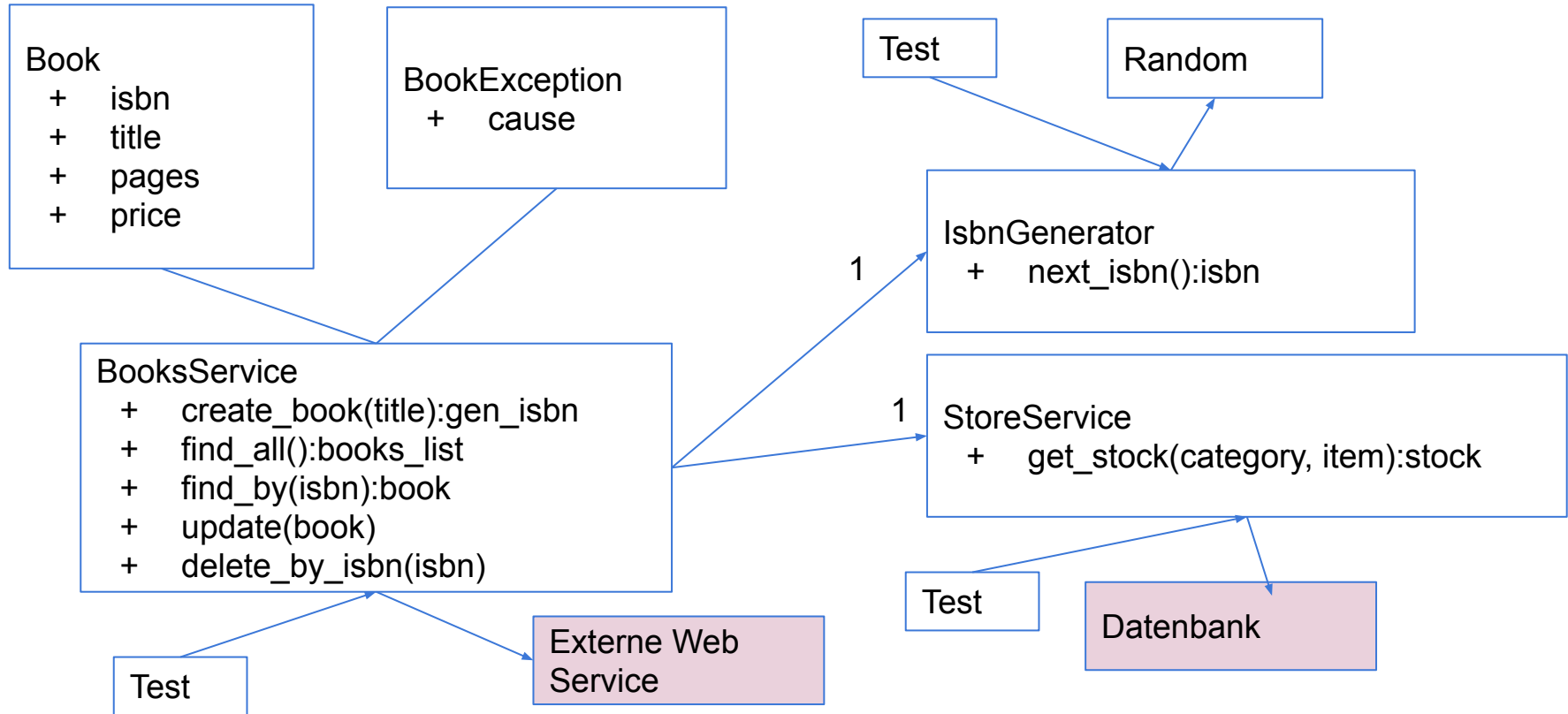
- Stellen Sie die marry / divorce auf einen dynamische Stil um
 - hasattr, delattr
- Verschaffen Sie sich einen Überblick
 - built-ins
 - String-Methoden
- range()-Funktion zum Erzeugen eines Range-Objektes z.B. zum Iterieren über einen Wertebereich
 - for (i= 0; i < 10; i = i + 1) -> Formulierung mit range
-



Anwendungsprogrammierung

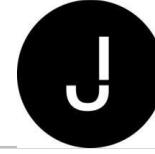






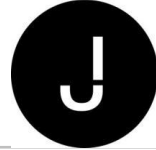
- Übernehmen Sie das books-Verzeichnis von GitHub
- Kopieren Sie books in Ihr Projekt
- Passen Sie den Modul-Namen an
 - javacream -> Name Ihrer Organisation
- Verifizieren Sie
 - application.py ist lauffähig
 - Erzeugen Sie darin einen IsbnGenerator und generieren eine Isbn

- Implementieren Sie den StoreService auf Grundlage eines Dictionaries!
 - Nicht-vorhandene Kategorie oder nicht vorhandenen Item -> return 0
- Zusätzliche Methoden
 - `set_stock(category, item, stock)`
 - `get_categories():categorySet`
 - `get_items_for(category):itemSet`
- Erweitern Sie die `application.py`
 - Hinweis
 - Im Wesentlichen führen Sie hier Tests ein
 - Etwas später -> UnitTest-Framework
- https://www.w3schools.com/python/python_dictionaries.asp



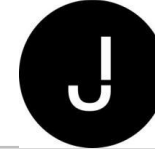
Python Module

- Python-Module sind Artefakte in einem Artefakt-Repository
 - PyPi
 - Identifikation erfolgt über eine Koordinate
 - Name + Versionsnummer
- pip
 - Package Manager zum Installieren von Modulen aus dem Repository
 - Ablage der Bibliotheken erfolgt zentral im Verzeichnisbaum des Entwickler-Rechners
 - Projekt-abhängige Ablage kann nur durch eine Sandbox erfolgen, die eine komplette Python-Umgebung bereitstellt



- Ein etabliertes Testing-Framework im Python-Umfeld
- `pip install unittest2`
- Testfälle üblicherweise gruppiert im `test / tests` -Directory
- Konventionen
 - name des Pythons: `testxyz.py`
 - Klasse erbt von `Testcase`
 - `set_up(self)`
 - `test_abc(self)`
 - Assertions
- Im `test`-Verzeichnis
 - `python -m unittest2 discover .`

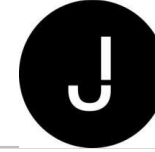
- Testfälle für
 - StoreService
 - Optional: IsbnGenerator
- Definitiv nicht zu testen
 - Book
 - eine reine Datenstruktur
 - BookException
 - ?
 - BooksService
 - noch gar nicht implementiert...



Exkurs: Context & Dependency Injection

- Alle relevanten Fachobjekte, also bei uns BooksService, IsbnGenerator, StoreService werden von einem Context instanziiert
- Der Context muss darüber hinaus die Dependencies setzen sowie die Objekte initialisieren
- CDI-Pattern ist damit eine Kombination, ein Meta-Pattern aus den GoF-Patterns Singleton, Factory und Strategy
-

- Es gibt kein komplett etabliertes CDI-Framework
 - Kein “Spring Python”
- Für unsere Anwendungen ist aber ein eigener Context total einfach zu implementieren



Zu unserem BooksService

- **CRUD-Operationen**
 - **create(title):isbn | BookException**
 - title muss mindestens aus einem Zeichen bestehen
 - price und pages können angegeben werden, falls nicht jeweils price=0, pages=1
 - isbn wird generiert
 - **find_by_isbn(isbn) : book | BookException**
 - isbn muss gesetzt sein
 - falls nicht oder kein Buch gefunden -> BookException
 - **update(book): void | BookException**
 - isbn muss vorhanden sein, title mindestens 1 zeichen, pages min 1, price >= 0
 - **delete_by_isbn(isbn) : void | BookException**
 - isbn muss gesetzt sein
 - falls nicht oder kein Buch gefunden -> BookException



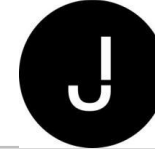
- Das Decorator-Pattern am Beispiel einer Testutility, trace
- Idee: NotNoneChecker für Parameter



- Zusätzliches Attribut im Book
 - available
- Bei einer Buch-Suche soll dieses Attribut durch eine store-Abfrage live bestimmt werden
-



- Umsetzung der Fachlichkeit
 - Im BooksService
 - In einem BooksServiceTest



Python Module, Part 2: Eigene Module

- Im Endeffekt ist jedes installierte Modul ein entpacktes Quellcode-Verzeichnis
- Root dieser Verzeichnisse ist im PYTHON_PATH zu finden
- Beim Laden = Importieren eines Moduls wird eine Initialisierung durchgeführt
 - `__init__.py`
 - Genauer: Durch eine `__init__.py` wird ein Verzeichnis zu einem Package
 - Ein Modul besteht aus einem oder mehreren Packages

- Initialisiert ein Package
- Stellt Meta-Informationen bereit
- `__all__ = ['name1', 'name2',]`
 - `__all__ = ['booksservice', 'context', ...]`
- Bei definiertem `__all__` funktioniert die Syntax
 - `from module import *`



- Erstellen der setup.py
- python setup.py sdist
 - Erzeugt ein tarball-Archiv
- push auf Nexus
 - .pypirc
 - pip.conf
 - twine /dist/archive
- pull
 - pip install
 -