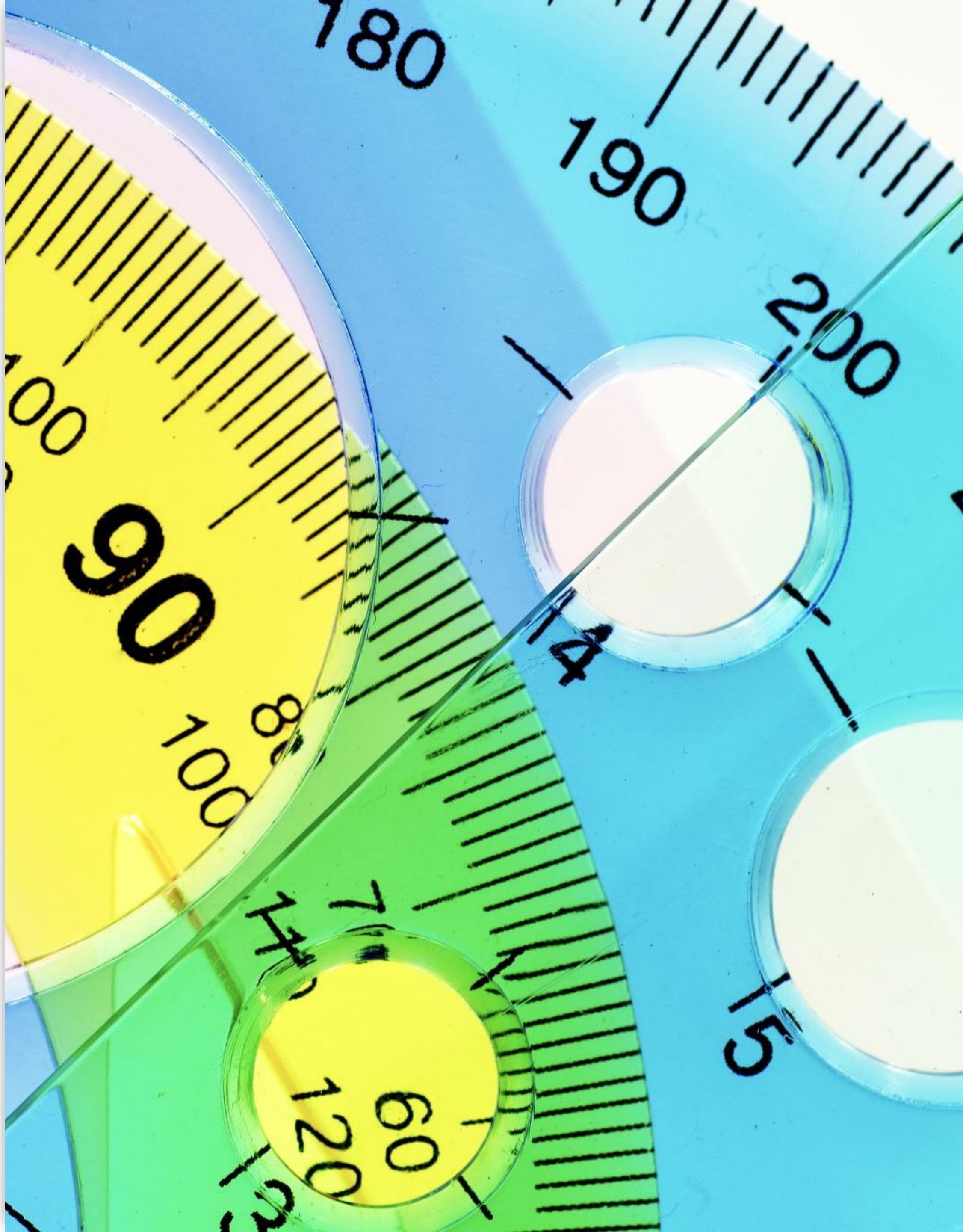
Abstract Factory

Ayesha, Linus, Daniel und Arved

Inhalt

- Creational pattern
- Factory Pattern
- Abstract Factory
- Besonderheiten
- Beispiel Code
- Workshop

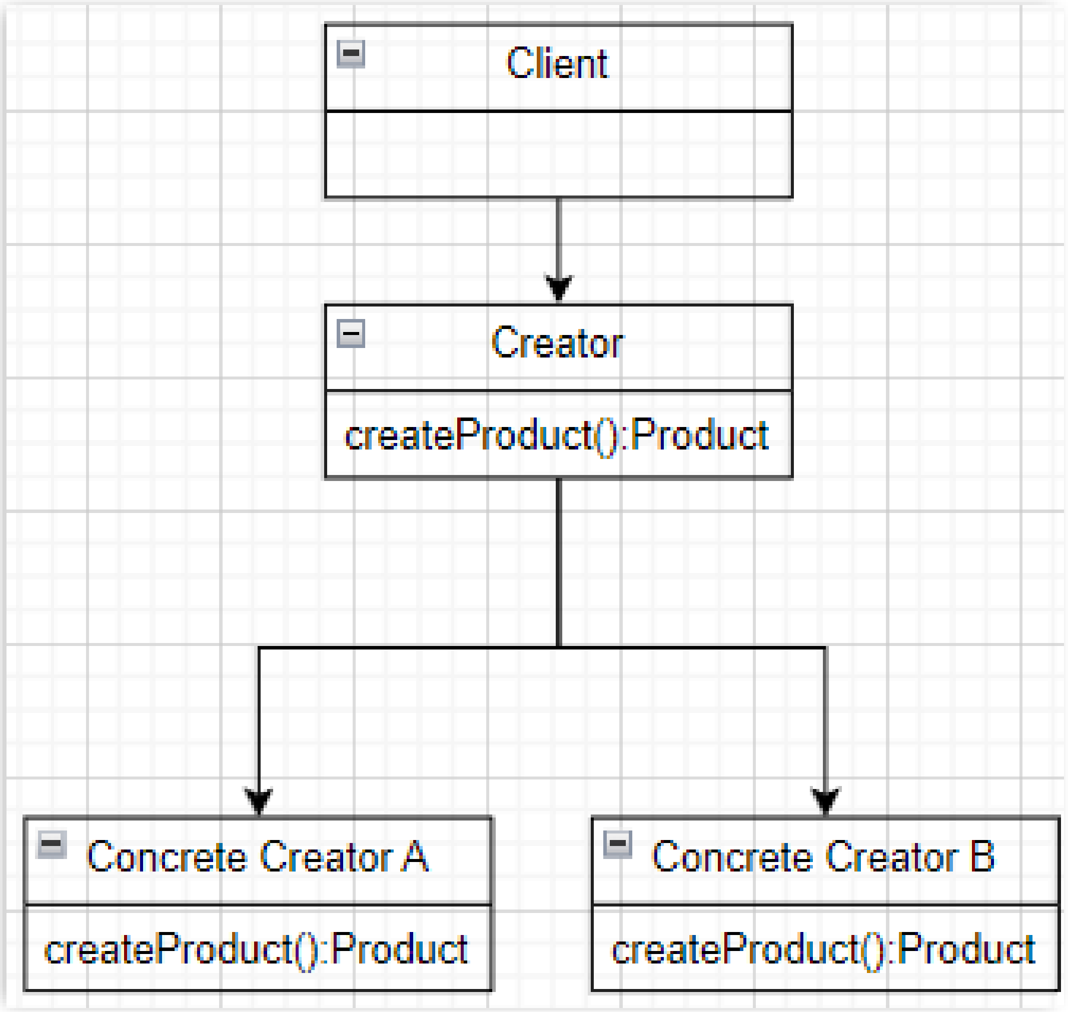


Creational Pattern

- Erstellen von Objekten
- Abstrahierung
- Wiederverwendung von Code
- Flexibel

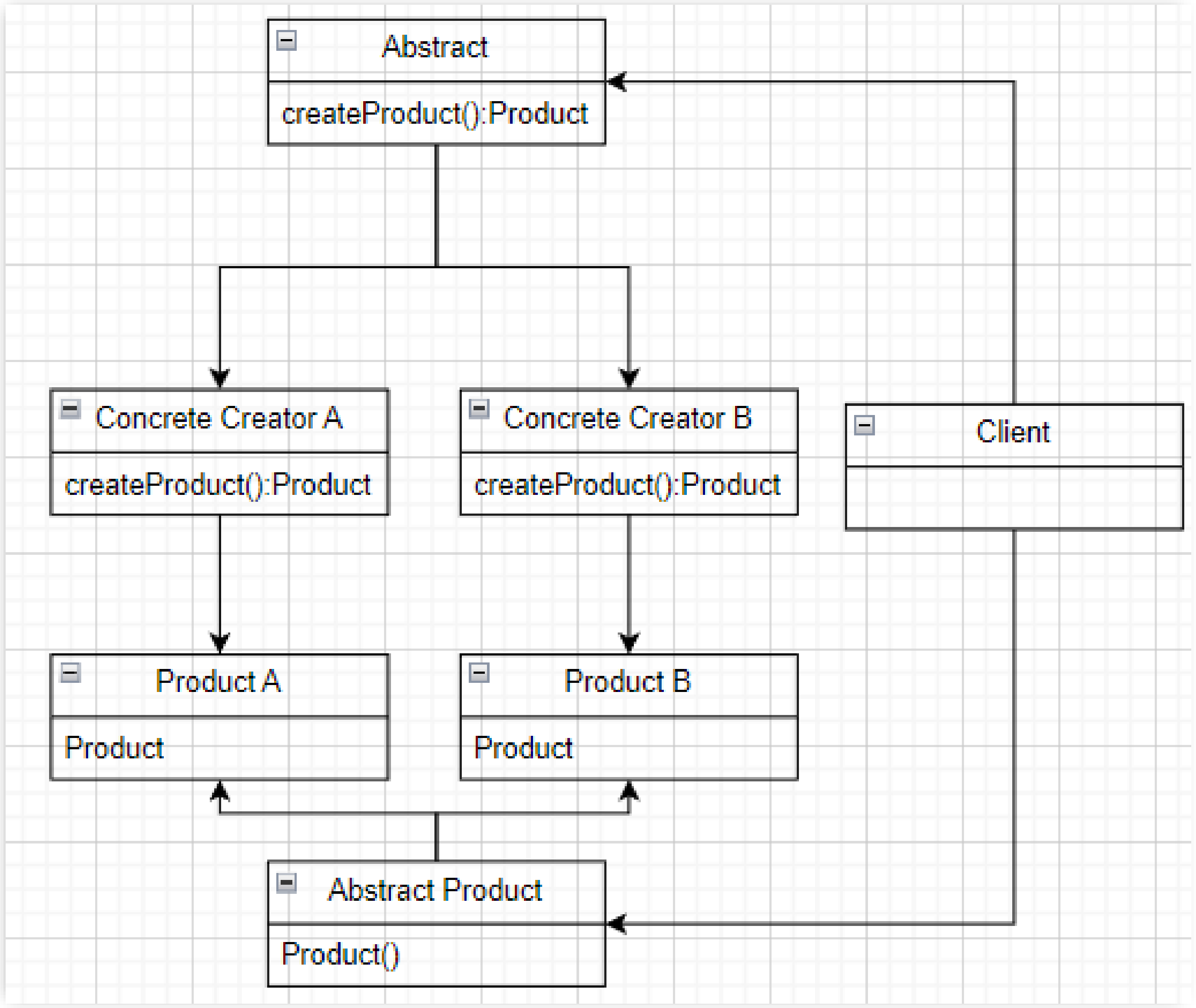
Factory Pattern

- Konstruktor
- Spezialisierte Methode
- Subklasse Ändern

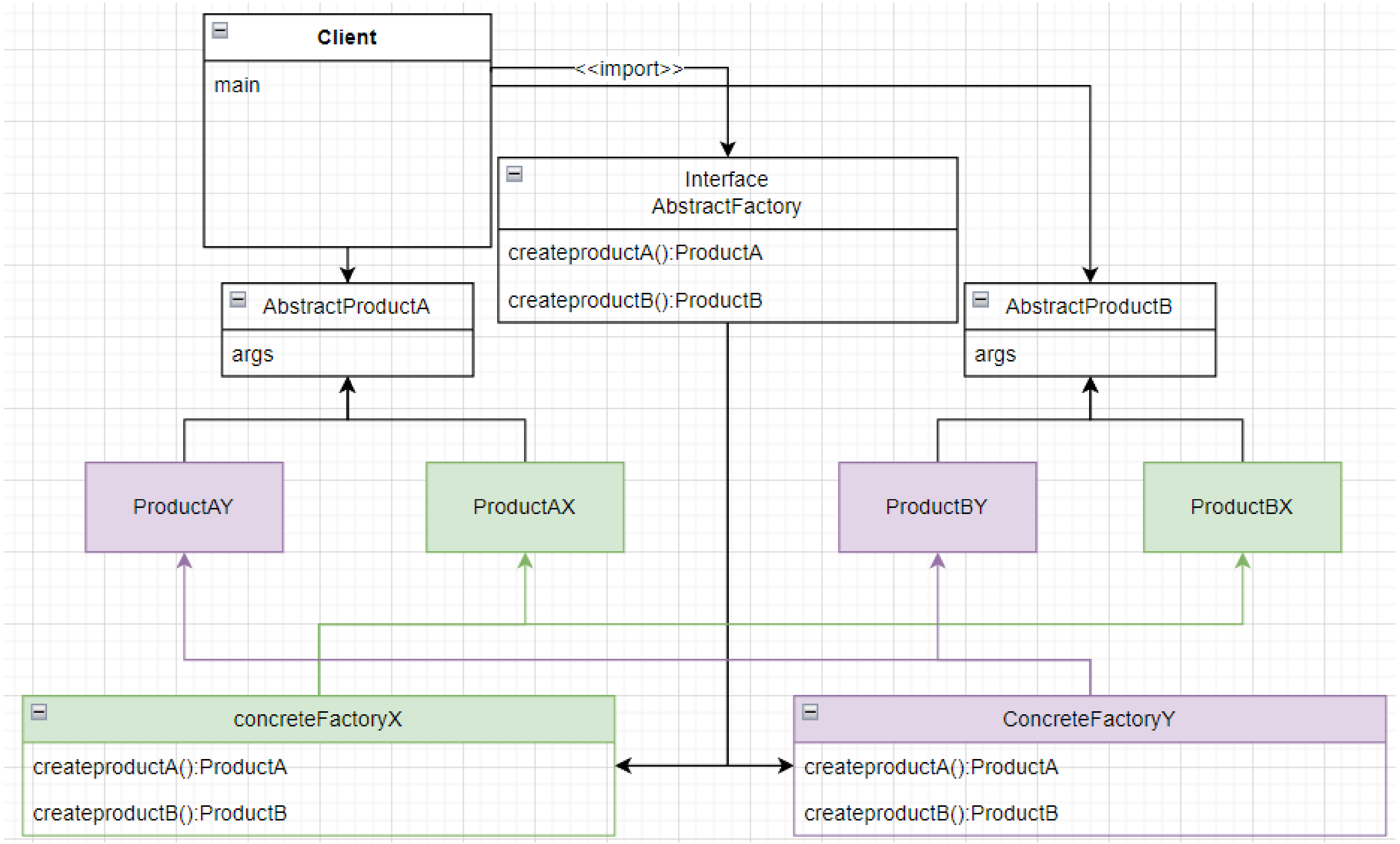


Abstract Factory

- Abstract Erweiterung
- Trennung von Objektfamilien Erzeugung und Implementation

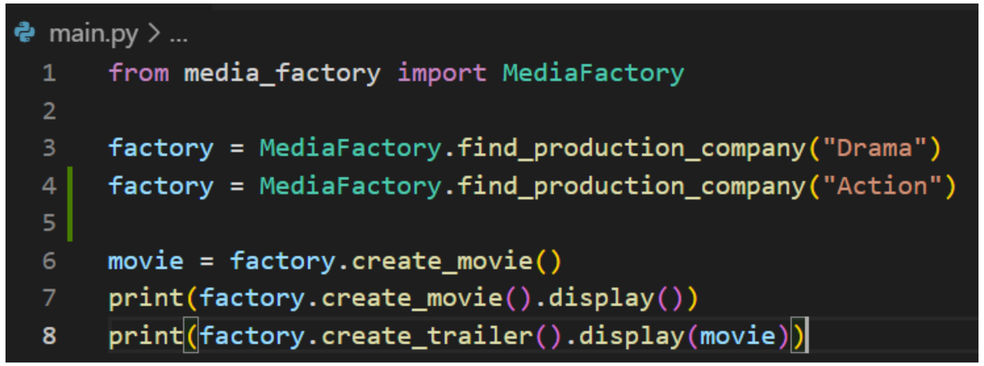


Klassendiagramm



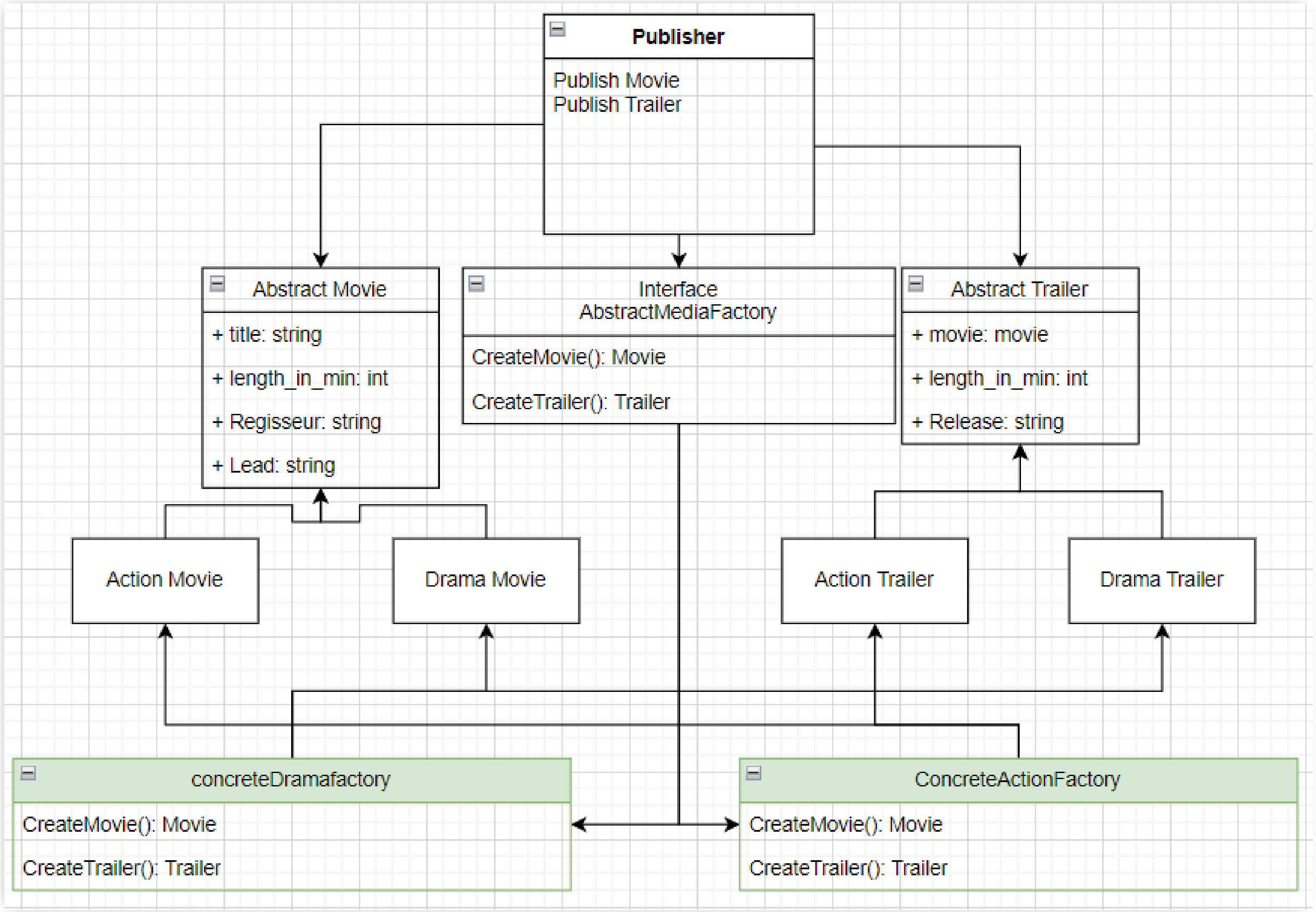
Besonderheiten

- Austauschbar
- Erweiterbar



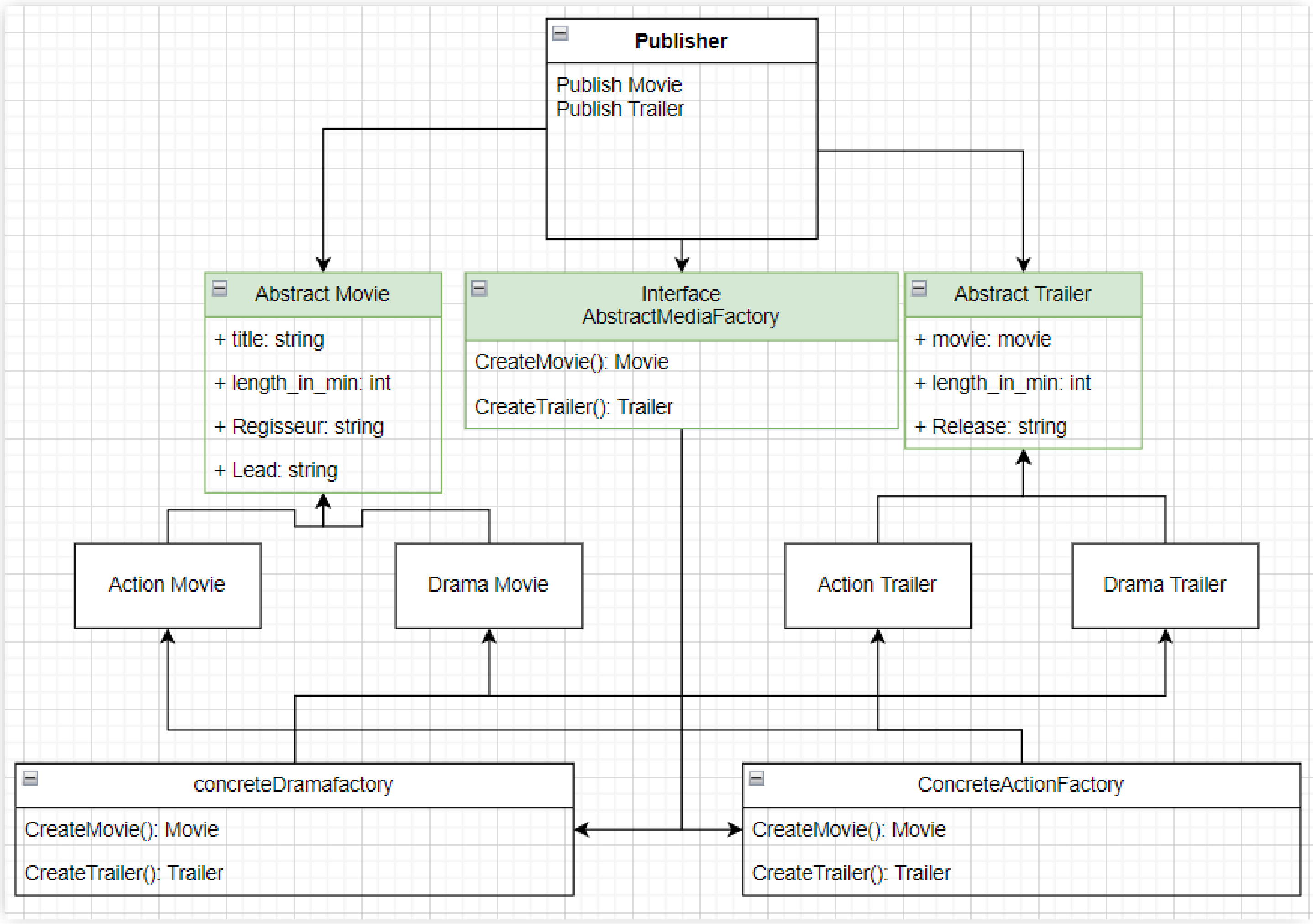
main.py > ...

```python
from media_factory import MediaFactory

factory = MediaFactory.find_production_company("Drama")
factory = MediaFactory.find_production_company("Action")

movie = factory.create_movie()
print(factory.create_movie().display())
print(factory.create_trailer().display(movie))
```



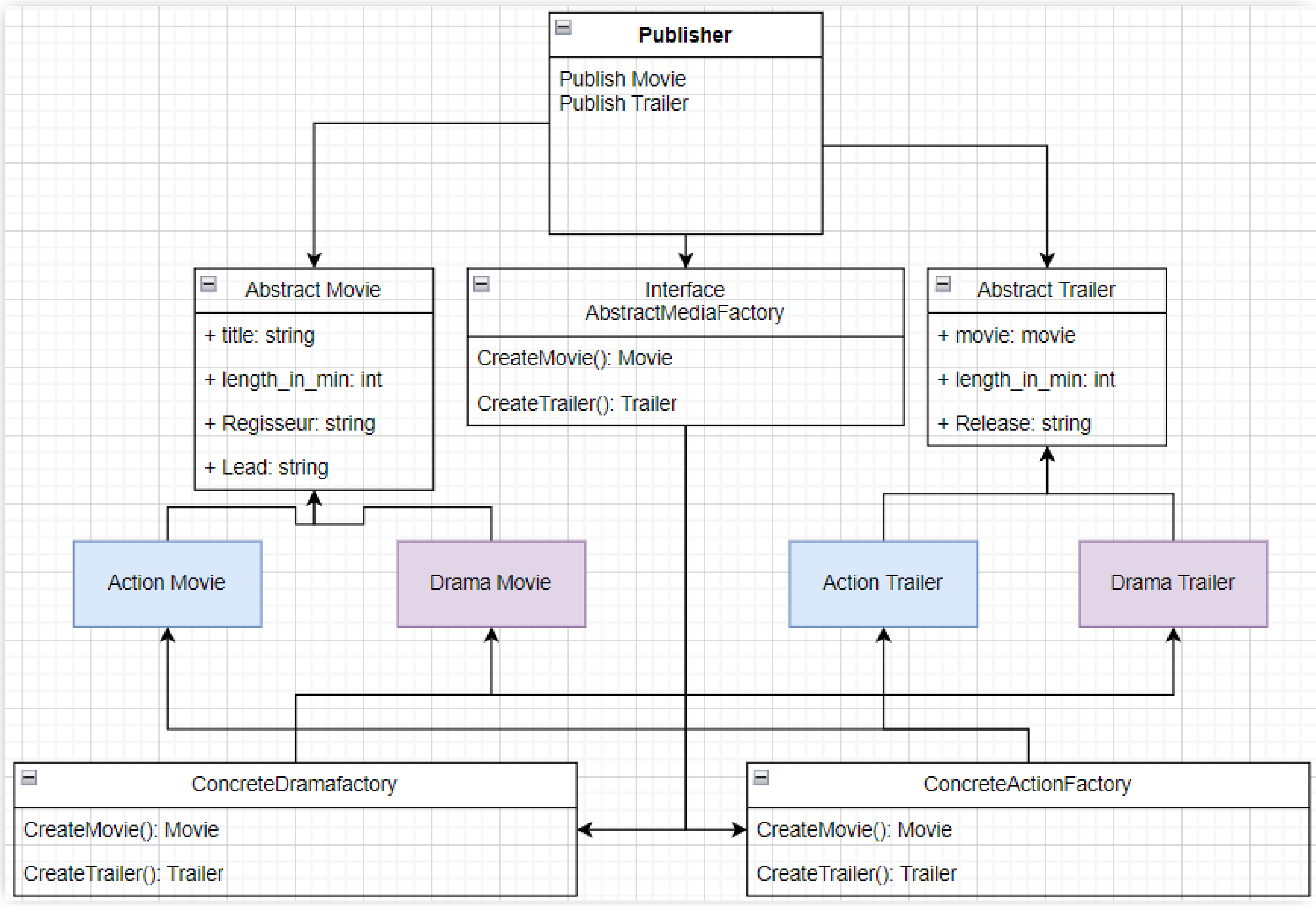
Besonderheiten

- Don‘t Repeat Yourself
- Single Responsibility Principle.



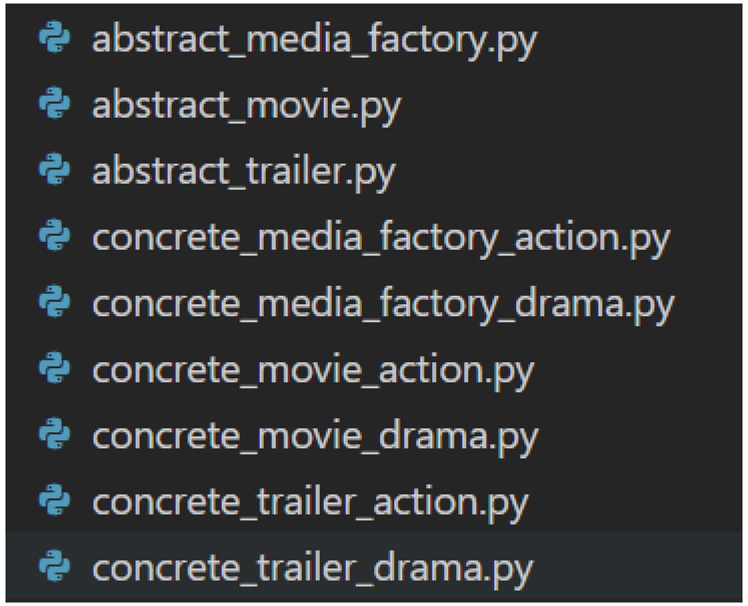
Besonderheiten

- Kompatibilität
  - Action Movie und Action Trailer

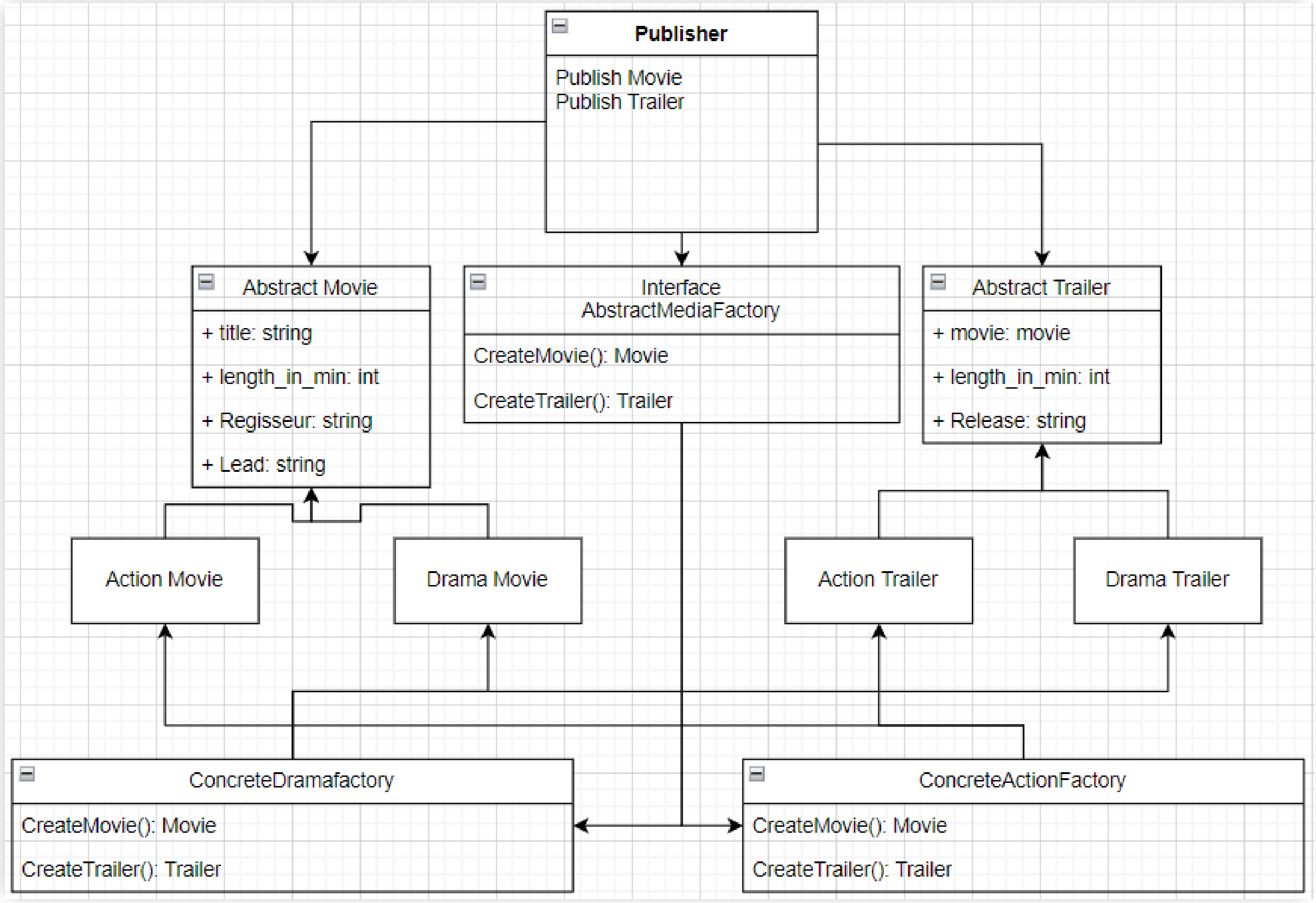


Besonderheiten

- Komplexität



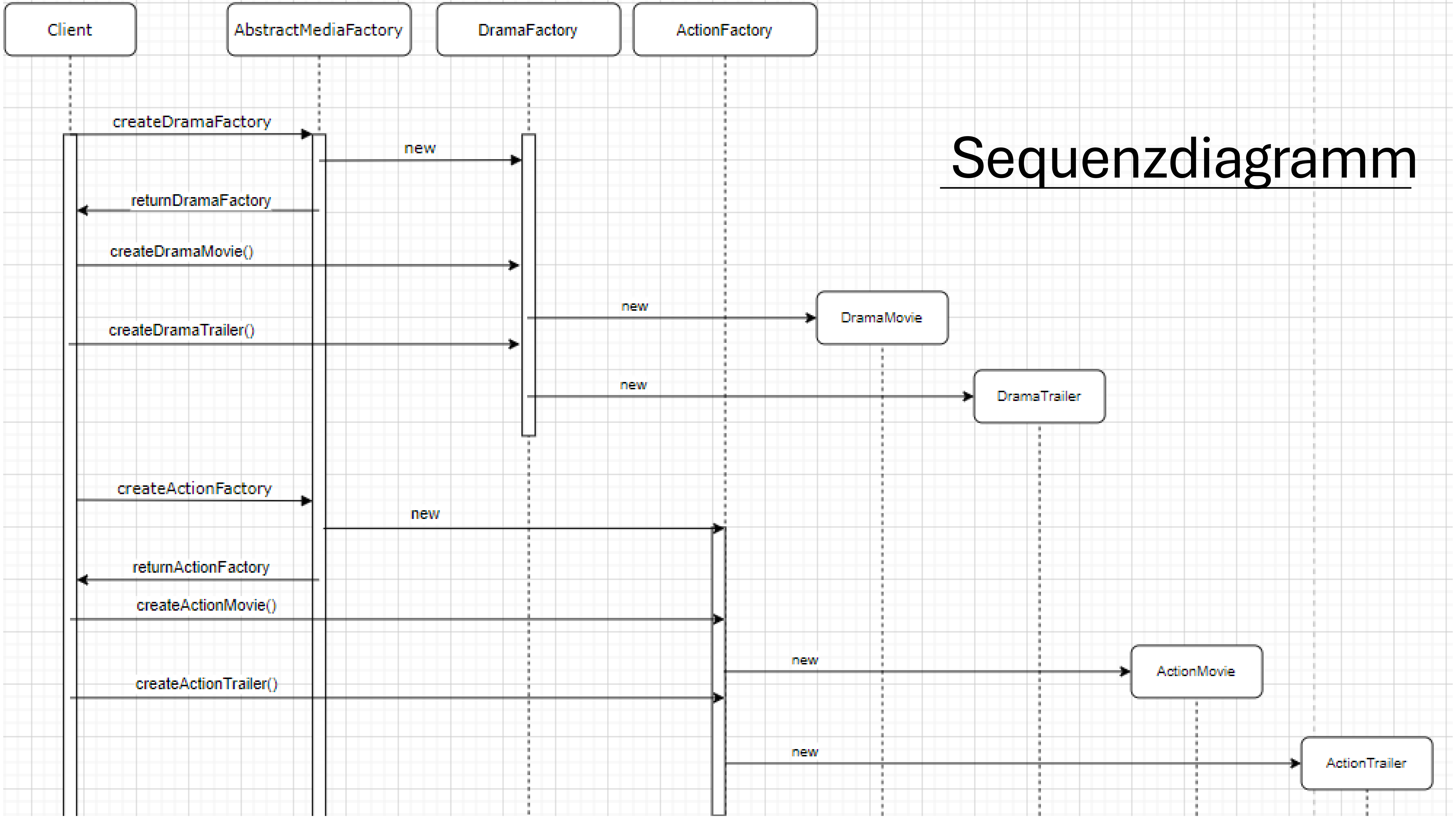
```
abstract_media_factory.py
abstract_movie.py
abstract_trailer.py
concrete_media_factory_action.py
concrete_media_factory_drama.py
concrete_movie_action.py
concrete_movie_drama.py
concrete_trailer_action.py
concrete_trailer_drama.py
```



Einsatzbereich

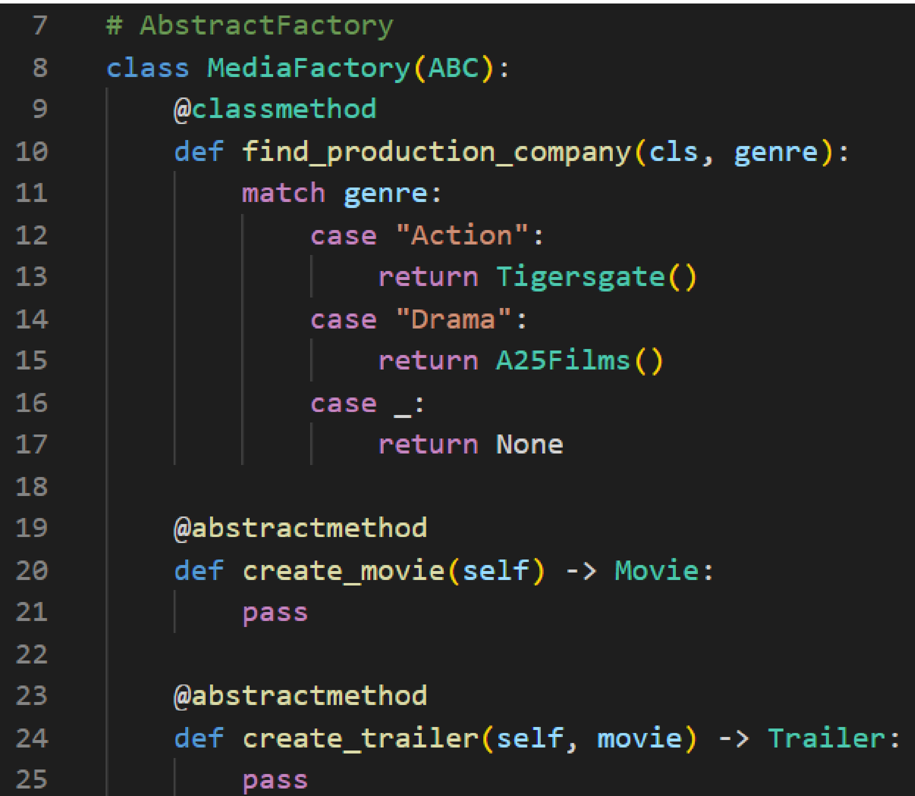
- Leichte Austauschbarkeit
- Großer Aufbau
- GUI-Elemente

Sequenzdiagramm

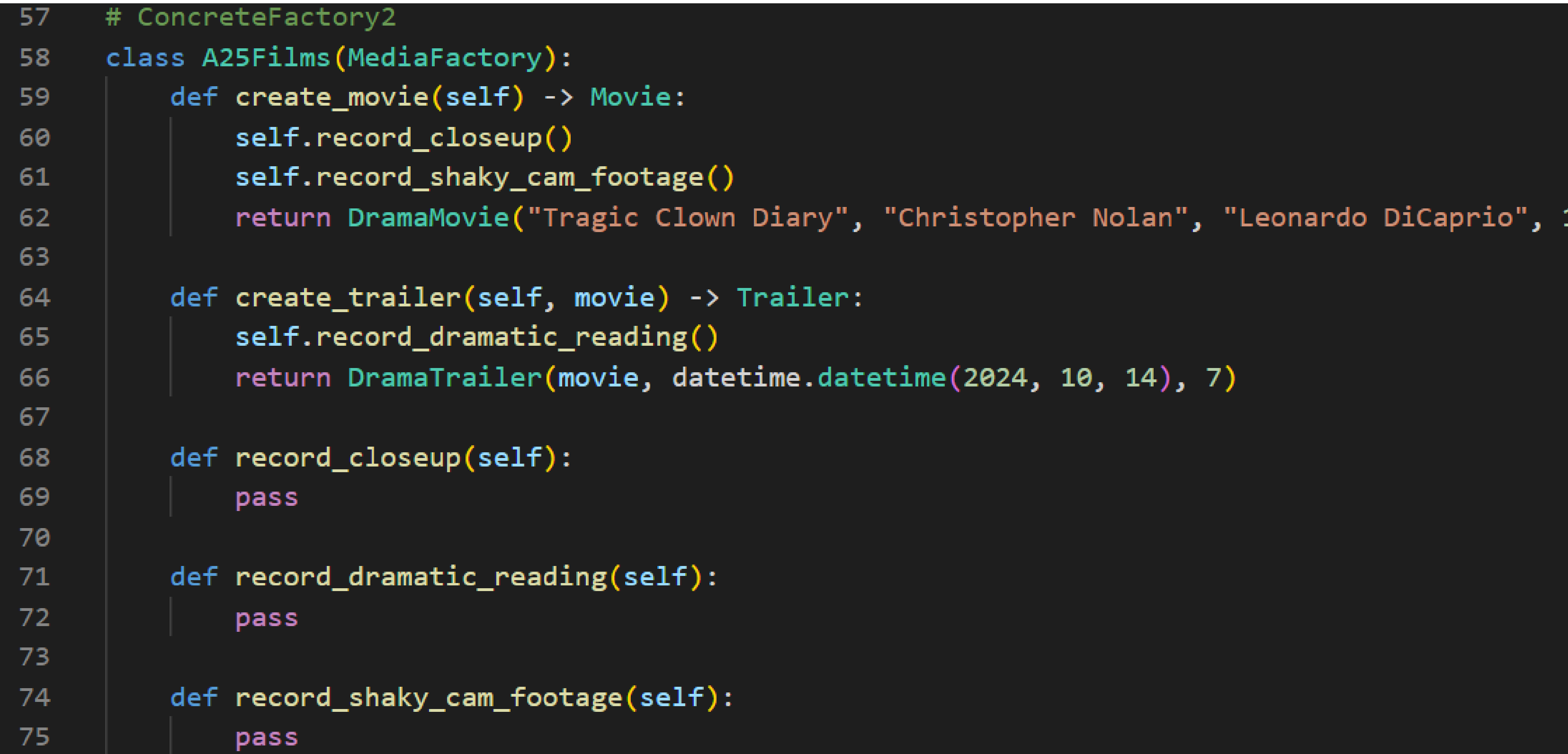


Abstract Factory in Python

- Abstract Base Classes (ABC)
- Decorators
  - @abstractmethod
  - @classmethod



```python
# AbstractFactory
class MediaFactory(ABC):
    @classmethod
    def find_production_company(cls, genre):
        match genre:
            case "Action":
                return Tigersgate()
            case "Drama":
                return A25Films()
            case _:
                return None

    @abstractmethod
    def create_movie(self) -> Movie:
        pass

    @abstractmethod
    def create_trailer(self, movie) -> Trailer:
        pass
```



```python
# ConcreteFactory2
class A25Films(MediaFactory):
    def create_movie(self) -> Movie:
        self.record_closeup()
        self.record_shaky_cam_footage()
        return DramaMovie("Tragic Clown Diary", "Christopher Nolan", "Leonardo DiCaprio", 

    def create_trailer(self, movie) -> Trailer:
        self.record_dramatic_reading()
        return DramaTrailer(movie, datetime.datetime(2024, 10, 14), 7)

    def record_closeup(self):
        pass

    def record_dramatic_reading(self):
        pass

    def record_shaky_cam_footage(self):
        pass
```

Workshop
Welcome to Hollywood!

Welcome to Hollywood! What's your dream? Everybody comes here; this is Hollywood, land of dreams. Some dreams come true, some don't; but keep on dreamin' - this is Hollywood. Always time to dream, so keep on dreamin'.

-(Pretty Woman, 1990)