

The background is a light gray with abstract geometric patterns. On the left, there is a dense network of thin gray lines connecting small black dots, forming a complex web. Scattered across the rest of the page are various thin-lined triangles of different sizes and orientations. In the top right corner, there are small, faint circles or dots.

# PATTERN OF THE WEEK

---

Week 3: Factories & Co.

FACTORY METHOD

01

ABSTRACT FACTORY

02

BUILDER

03

## TABLE OF CONTENTS

04

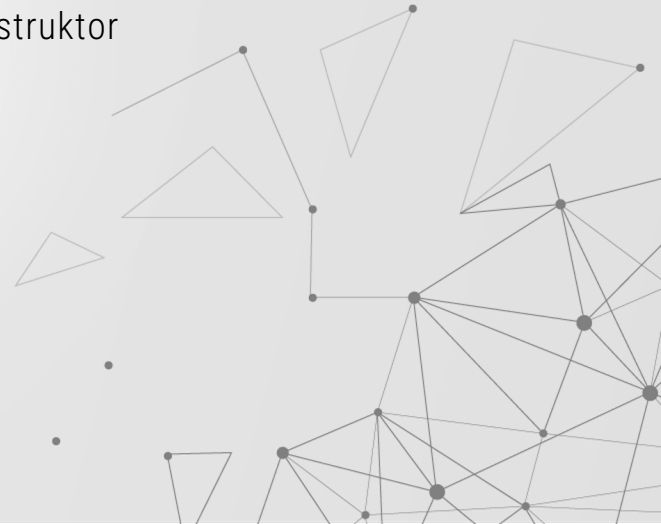
DIRECTOR

05

INTERPRETER PATTERN

# DIE "FALSCH" FACTORY METHOD

- Ein Objekt wird durch den Aufruf einer Methode erzeugt,  
anstatt durch den direkten Aufruf des Konstruktors
- Create() Methode in der Klasse statt des Konstruktors
- Validierung, Business Logik und lange Prozesse nicht im Konstruktor



# AUFGABE

## "FALSCHER" FACTORY METHOD IMPLEMENTIEREN

Projekt: "Person"

Konstruktor Privat  
Create() mit Daten  
ArgumentException wenn "null"  
oder Name weniger als 2  
Buchstaben



01

FACTORY METHOD

---



# FACTORY METHOD

- Ein Objekt wird durch den Aufruf einer Methode erzeugt,  
anstatt durch den direkten Aufruf des Konstruktors
  - Aufrufer kennt die Abhängigkeiten und den Instanziierungsprozess nicht
  - Trennung zwischen Erstellung und Verwendung
- Ich kenne jemanden, der das kann



# AUFGABE

## FACTORY METHOD IMPLEMENTIEREN

Projekt: "PdfTools"

IPdfArchiverFactory  
Create() gibt PdfArchiver zurück





02

## ABSTRACT FACTORY

---



# ABSTRACT FACTORY

- Erzeugt ein Gruppe von zusammengehörigen Instanzen
- Rückgabe als Interface
- Instanzen können in konkreten Typ gecastet werden
- Beispiel: IWindow, IButton, IContainerControl
- WpfWindow, WpfButton, WpfContainerControl

*New WpfButton( (WpfContainerControl) controlAsInterface)*



# AUFGABE

## ABSTRACT FACTORY IMPLEMENTIEREN

Projekt: "PdfTools"

Umbau der IPdfArchiverFactory

Interface IPdfArchiver



# VERWENDUNG

- Neue Instanz im aktuellen Code
- “DI” erzeugte Instanz reicht nicht aus
- Smell: State in der Instanz (nicht “static”)

→ Lambda-Factory



# LAMBDA-FACTORY METHOD

- Factory Method Pattern ohne Interface und Ableitung
- Konstruktorparameter `Func<T> factory`
- Funktion die beim Aufruf eine (neue) Instanz zurück gibt
- Aber: Ohne DI Container können sich die Implementierungen unterscheiden
- Wichtig: Bei OOTB Interfaces kann es zu Problemen kommen (`Func<String>`)



# LAMBDA-ABSTRACT FACTORY

- Abstract Factory mit generischer Implementierung
- Eine Klasse enthält mehrere Lambda-Properties
- `Public Func<T> TeeFactory { get; }`
- `T bar = FooAbstractFactory.TeeFactory()`

