



Interne Entwicklerschulung

"Results, Maybes und
die Fluent-Schreibweise"

Agenda

Result

- Was sind "Results"?
- Welche Vorteile bieten sie?
- Welche Nachteile bringen sie mit sich?
- Library vs. Eigenimplementierung
- Typische Einstiegshürden

Maybe

- Was sind "Maybes"?
- Welche Vorteile bieten sie?
- Welche Nachteile bringen sie mit sich?
- Library vs. Eigenimplementierung
- Typische Einstiegshürden

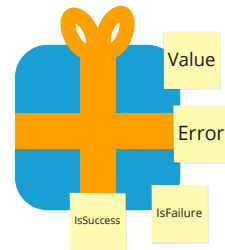
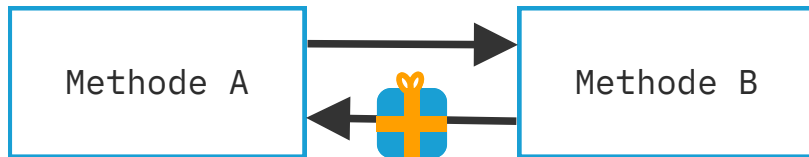
Fluent-Schreibweise

- Was versteht man unter "Fluent"?
- Woher kennt man diese Schreibweise?
- Wie erweitere ich sie korrekt?




Result - Was sind "Results"?


➡ Kontainer für (Rückgabe-)Objekte mit Status über Erfolgs- oder Fehlerfall.




Result - Welche Vorteile bieten sie?



Ersatz für
bekannte
Exceptions




Bekannte
Fehlerfälle
klar definiert




Verpflicht-
endes Result
Handling

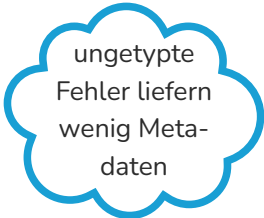
Result - Welche Nachteile bringen sie mit sich?




Verpflicht-
endes Result
Handling



Zusätzlicher
Ressourcen-
Overhead

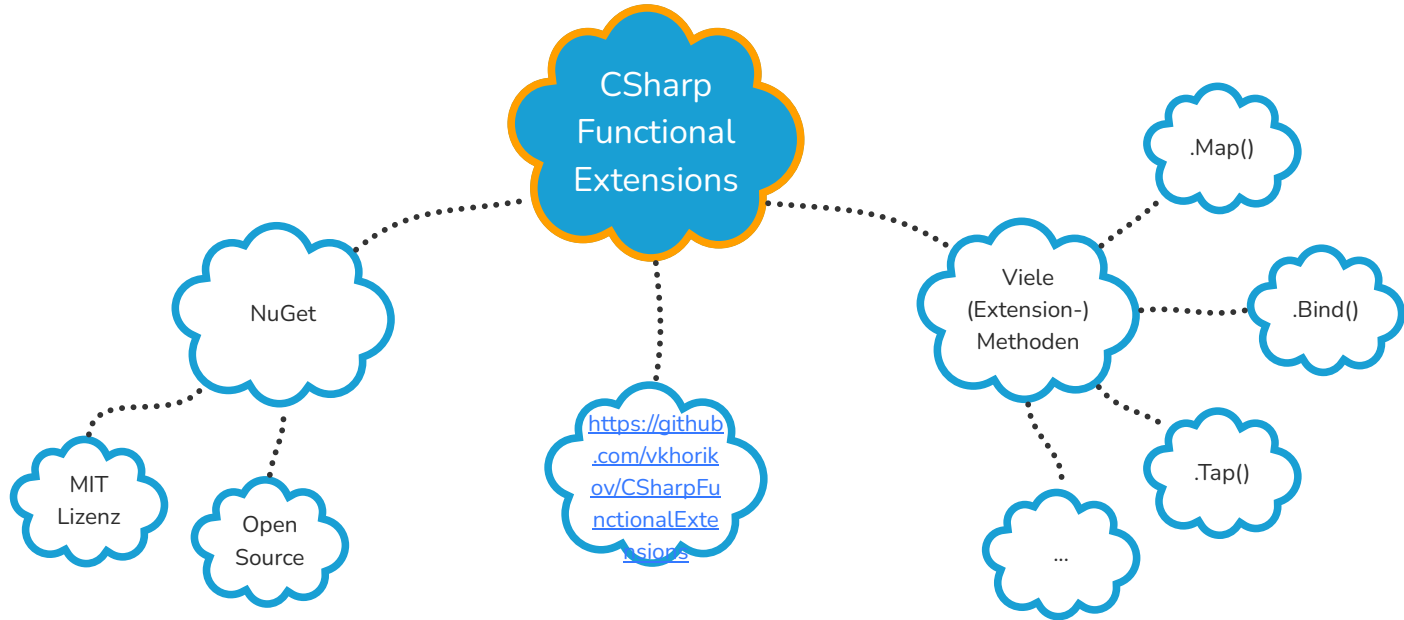


ungetypte
Fehler liefern
wenig Meta-
daten



Kein
"Fail Fast" in
Kombination
mit Fluent

Result - Library vs. Eigenimplementierung



Result - Typische Einstiegshürden

Wann soll ich ein
"Result.Failure" und wann
"throw Exception"
verwenden?

Wann soll ich ein "Result"
und wann "Result<boolen>"
verwenden?

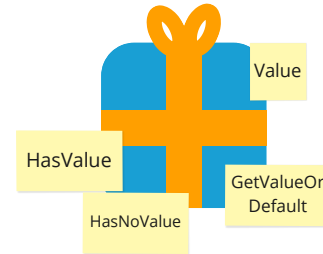
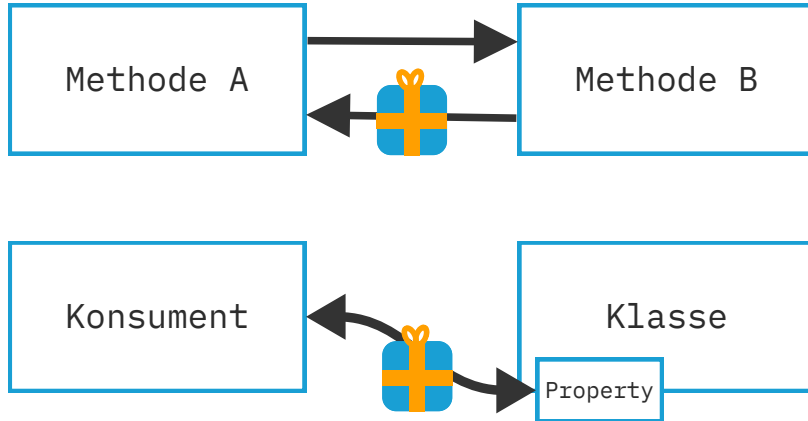
Wie kann ich mehrere
Objekte "durchschleifen"?
(Fluent)

".Map()", ".Bind()", ".Tap()" ...
WHAT???

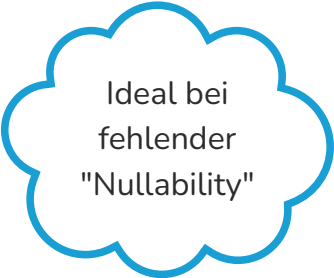


Maybe - Was sind "Maybes"?


➡ Kontainer für (Rückgabe-)Objekte mit Status über Existenz- oder Abwesenheit.



Maybe - Welche Vorteile bieten sie?




Ideal bei
fehlender
"Nullability"




Explizite
Optionalität


Maybe - Welche Nachteile bringen sie mit sich?



Verpflicht-
endes Maybe
Handling

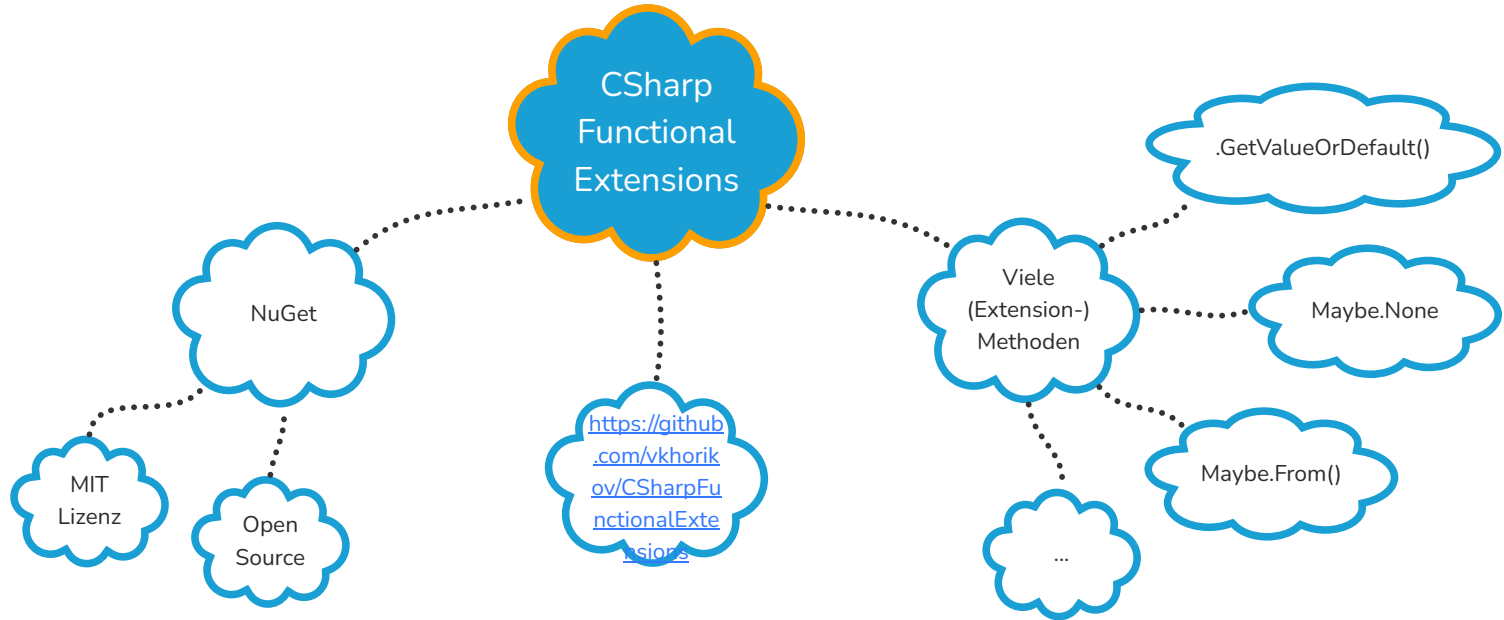


Oft unnötig
bei expliziter
"Nullability"



Zusätzlicher
Ressourcen-
Overhead

Maybe - Library vs. Eigenimplementierung



Maybe - Typische Einstiegshürden



Was liefert mir
".GetValueOrDefault()"
zurück?

Brauche ich Maybes, auch
wenn ich "Nullable"
aktiviert habe?



Fluent-Schreibweise - Was versteht man unter "Fluent"?

"Das Fluent Interface Design Pattern ermöglicht die Verkettung von Methodenaufrufen in einer Klasse, um den Code lesbarer und expressiver zu gestalten. Es erlaubt die aufeinanderfolgende Nutzung von Methoden, wodurch einfließende, anpassbare Konfigurationen intuitiv erstellt werden können. Typischerweise wird dies durch Rückgabe des Objekts selbst in jeder Methode ermöglicht.

Method Chaining ist ein allgemeines Konzept, bei dem aufeinanderfolgende Methodenaufrufe in einer Zeile verkettet werden, während das Fluent Interface Design Pattern speziell darauf abzielt, die Lesbarkeit und Expressivität des Codes zu verbessern, indem es Method Chaining verwendet, um eine fließende und anpassbare API zu schaffen."

ChatGPT

```
class Program
{
    static void Main()
    {
        FluentExample fluentObj = new FluentExample();
        int result = fluentObj.SetValue(5).Add(3).Add(2).GetValue();
        Console.WriteLine(result); // Output: 10
    }
}
```

Fluent-Schreibweise - Woher kennt man diese Schreibweise?

LINQ

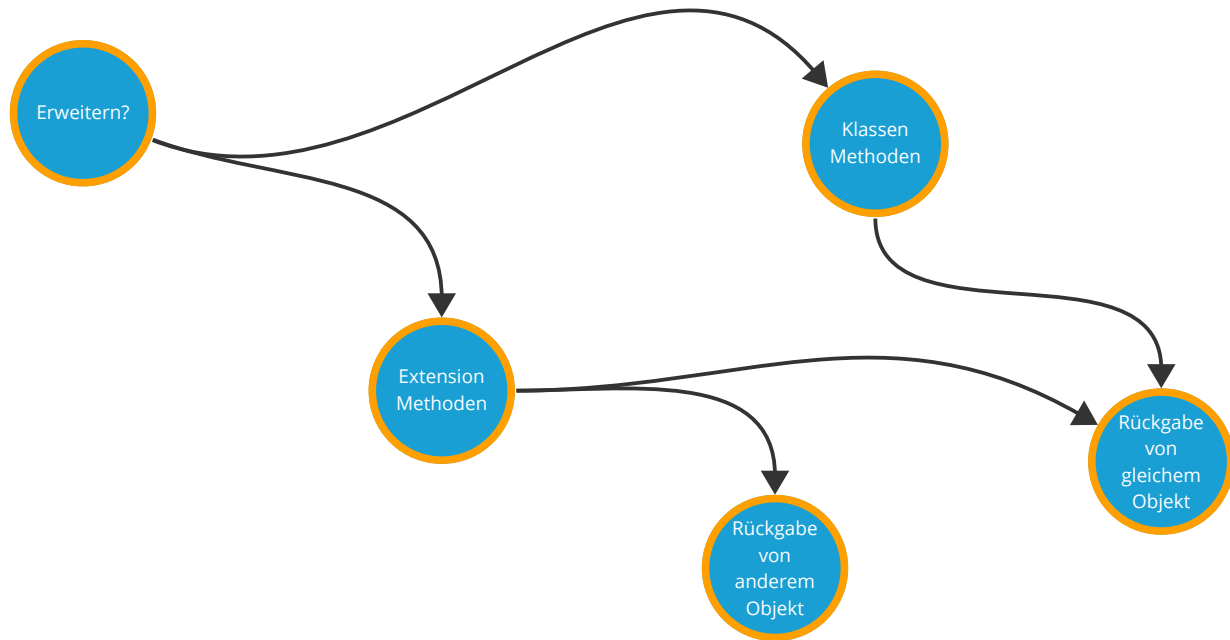
Fluent
Assertion

ASP.NET
Core

Entity
Framework

CSharp
Functional
Extensions

Fluent-Schreibweise - Wie erwidere ich sie korrekt?









Interne Entwicklerschulung

Danke