

# NuGet-Pakete

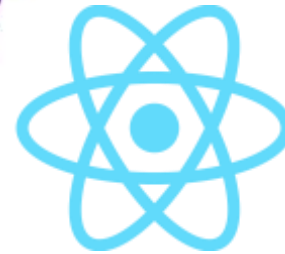
Wann, welche und warum? Oder doch nicht?



Thorsten Kansy (tkansy@dotnetconsulting.eu)

# Meine Person- Thorsten Kansy

Freier Consultant, Software Architekt,  
Entwickler, Trainer & Fachautor



Azure Cosmos DB

# Mein Service- Ihr Benefit

- Individuelle Inhouse Trainings
- (Online on-demand) Projektbegleitung
- Beratung
  - Problemanalyse und Lösungen
  - Technologieentscheidungen







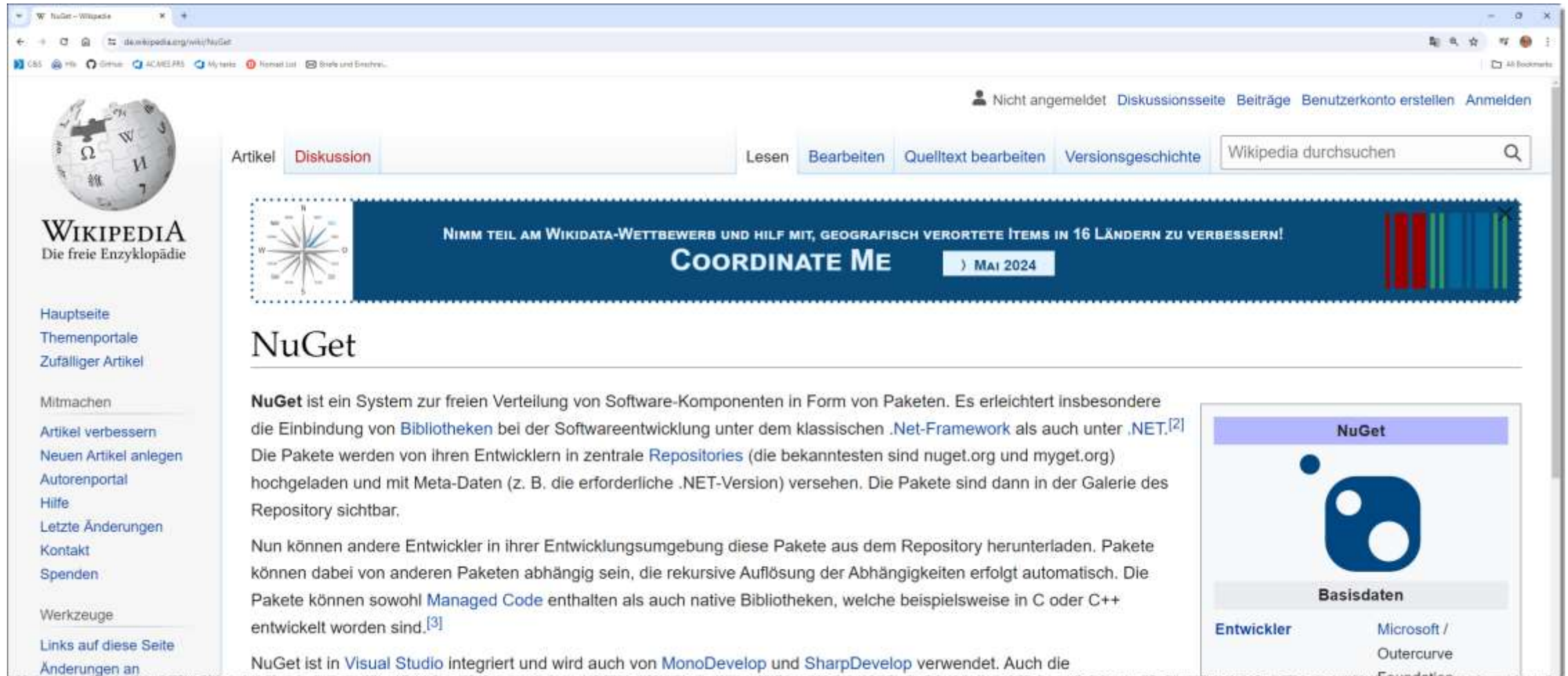
# Agenda

# Agenda

- Warum Nugets? Vor- und Nachteile
- Ein paar Empfehlungen



# NuGet? What?



The screenshot shows the Wikipedia article for NuGet. At the top, there's a navigation bar with 'Artikel' and 'Diskussion' tabs. Below that is a search bar and a banner for 'COORDINATE ME' with a date of 'MAI 2024'. The main content area starts with the title 'NuGet' and a paragraph explaining it as a system for distributing software components. To the right, there's a table with the NuGet logo and basic information.

Artikel **Diskussion** Lesen Bearbeiten Quelltext bearbeiten Versionsgeschichte

**WIKIPEDIA**  
Die freie Enzyklopädie

Hauptseite  
Themenportale  
Zufälliger Artikel

Mitmachen  
Artikel verbessern  
Neuen Artikel anlegen  
Autorenportal  
Hilfe  
Letzte Änderungen  
Kontakt  
Spenden

Werkzeuge  
Links auf diese Seite  
Änderungen an

Nicht angemeldet [Diskussionsseite](#) [Beiträge](#) [Benutzerkonto erstellen](#) [Anmelden](#)


**COORDINATE ME** MAI 2024

## NuGet

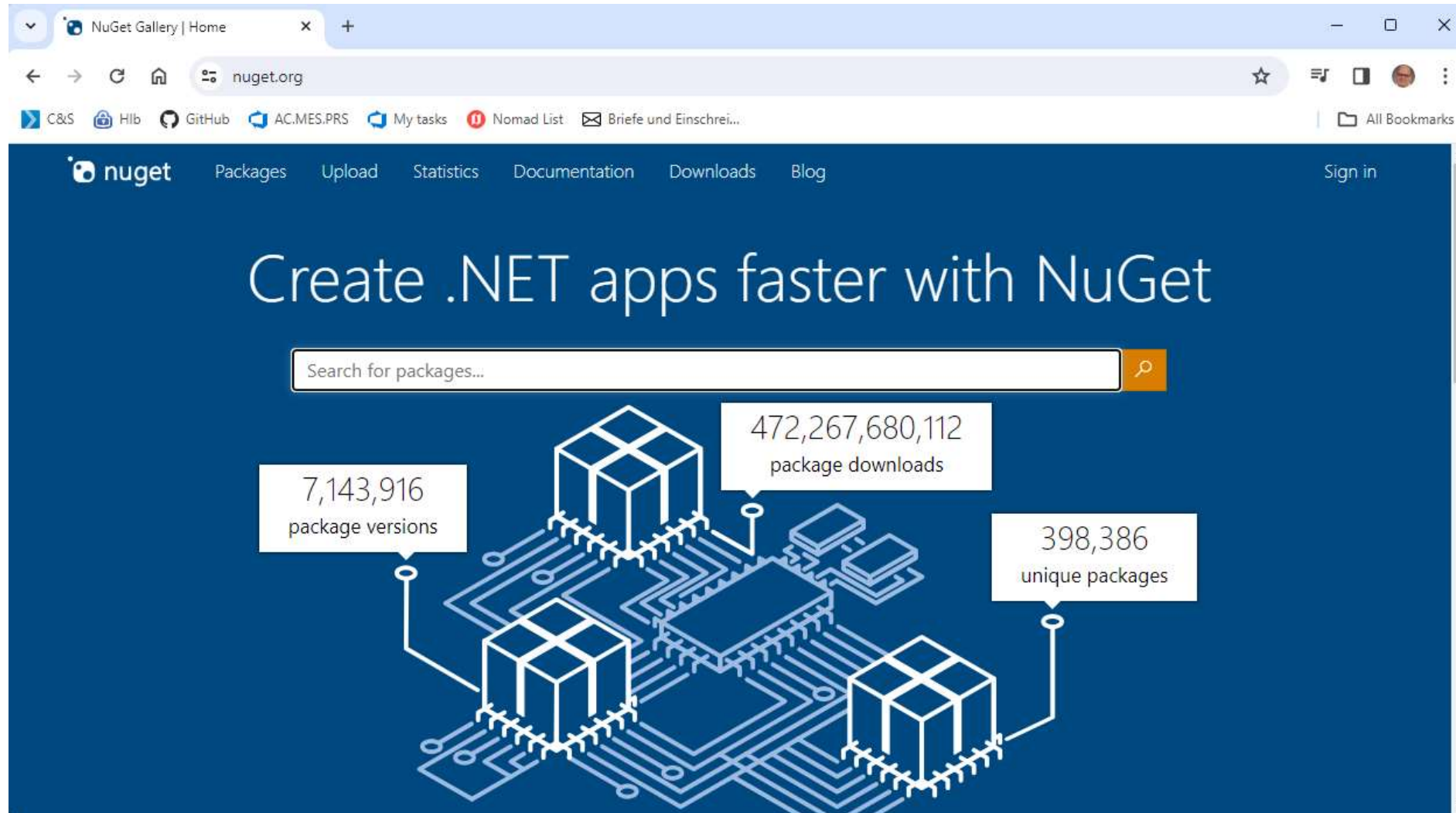
**NuGet** ist ein System zur freien Verteilung von Software-Komponenten in Form von Paketen. Es erleichtert insbesondere die Einbindung von [Bibliotheken](#) bei der Softwareentwicklung unter dem klassischen [.Net-Framework](#) als auch unter [.NET](#).<sup>[2]</sup> Die Pakete werden von ihren Entwicklern in zentrale [Repositories](#) (die bekanntesten sind [nuget.org](#) und [myget.org](#)) hochgeladen und mit Meta-Daten (z. B. die erforderliche [.NET-Version](#)) versehen. Die Pakete sind dann in der Galerie des Repository sichtbar.

Nun können andere Entwickler in ihrer Entwicklungsumgebung diese Pakete aus dem Repository herunterladen. Pakete können dabei von anderen Paketen abhängig sein, die rekursive Auflösung der Abhängigkeiten erfolgt automatisch. Die Pakete können sowohl [Managed Code](#) enthalten als auch native Bibliotheken, welche beispielsweise in C oder C++ entwickelt worden sind.<sup>[3]</sup>

NuGet ist in [Visual Studio](#) integriert und wird auch von [MonoDevelop](#) und [SharpDevelop](#) verwendet. Auch die

NuGet	
	
Basisdaten	
<b>Entwickler</b>	Microsoft / Outercurve Foundation

# Warum NuGet-Pakete? Vor- und Nachteile



# Pro & Contras

## **PRO**

- Schnellere Entwicklung
- Features jenseits der eigenen Kenntnisse
- .NET

## **CONS**

- Upgrade-Risiko
- Support-Risiko
- Sicherheits-Risiko
- DLL-Hölle in neuem Gewand



# Worauf achte ich?

The screenshot shows the NuGet Package Manager interface for the project 'PGM.Efficiency.API'. The left pane displays a list of packages, and the right pane shows the details for the 'Polly' package.

**Package List:**

Package Name	Author	Downloads	Version
Newtonsoft.Json.Bson	James Newton-King	669M	1.0.2
Swashbuckle.AspNetCore.Swagger	Swashbuckle.AspNetCore.Swagger	614M	6.5.0
Swashbuckle.AspNetCore.SwaggerGen	Swashbuckle.AspNetCore.SwaggerGen	612M	6.5.0
AutoMapper	Jimmy Bogard	589M	13.0.1
Moq	Daniel Cazzulino, kzu	589M	4.20.70
System.Diagnostics.PerformanceCounter	Microsoft	583M	8.0.0
System.Diagnostics.EventLog	Microsoft	572M	8.0.0
System.Runtime.Caching	Microsoft	565M	8.0.0
Polly	Michael Wolfenden, App vNext	555M	8.3.1

**Polly Package Details:**

- Version:** Latest stable 8.3.1
- Package source mapping:** is off. [Configure](#)
- Options:** (collapse)
- Description:** Polly is a .NET resilience and transient-fault-handling library that allows developers to express resilience and transient fault handling policies such as Retry, Circuit Breaker, Timeout, Bulkhead Isolation, and Fallback in a fluent and thread-safe manner.
- Version:** 8.3.1
- Author(s):** Michael Wolfenden, App vNext
- License:** [BSD-3-Clause](#)
- Readme:** [View Readme](#)
- Downloads:** 555,026,751
- Date published:** Mittwoch, 6. März 2024 (06.03.2024)
- Project URL:** <https://github.com/App-vNext/Polly>
- Report Abuse:** <https://www.nuget.org/packages/Polly/8.3.1/ReportAbuse>
- Tags:** Polly, Exception, Handling, Resilience, Transient, Fault, Policy, Circuit, Breaker, CircuitBreaker, Retry, Wait, Cache, Cache-aside, Bulkhead, Fallback, Timeout, Throttle
- Dependencies:**
  - .NETFramework, Version=v4.8.2
    - Polly.Core (>= 8.3.1)
  - .NETFramework, Version=v4.7.2
    - Polly.Core (>= 8.3.1)
  - net6.0
    - Polly.Core (>= 8.3.1)
  - .NETStandard, Version=v2.0
    - Polly.Core (>= 8.3.1)

Each package is licensed to you by its owner. NuGet is not responsible for, nor does it grant any licenses to, third-party packages.

Don't show this again



A brown monkey is perched on a stone ledge, looking towards the right. The background features a lush green forest and a waterfall. A dark blue horizontal band is overlaid across the middle of the image, containing the text 'NLog'.

NLog



# NLog

## Gut konfigurierbares Logging-Framework

```
<?xml version="1.0" encoding="utf-8" ?>
<nlog autoReload="true" throwExceptions="true" internalLogFile="c:\temp\NLog.txt" internalLogLevel="T
  xmlns="http://www.nlog-project.org/schemas/NLog.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema
  <targets>
    <target name="LogToFile" xsi:type="File" fileName="{basedir}/Logs/PGMEfficiency.log" encoding
      maxArchiveFiles="10" archiveNumbering="Sequence" archiveAboveSize="1048576" archiveFileNa
  </targets>
  <rules>
    <logger name="*" writeTo="LogToFile" minlevel="Warning" />
  </rules>
</nlog>
```



<https://nlog-project.org/>

Install-Package NLog





Demo



Bogus



LORD MURUGA  
Donated by

# Bogus

(Test-)Daten per Fluent-API und mit Regeln generieren

```
var testOrder = new Faker<Order>()
    .StrictMode(true)
    .RuleFor(o => o.OrderId, f => orderIds++)
    .RuleFor(o => o.Item, f => f.PickRandom(fruits))
    .RuleFor(o => o.Quantity, f => f.Random.Number(1, 10))
    .RuleFor(o => o.LotNumber, f => f.Random.Int(0, 100).OrNull(f, .8f));

var orders = testOrder.Generate(3).ToList();
```



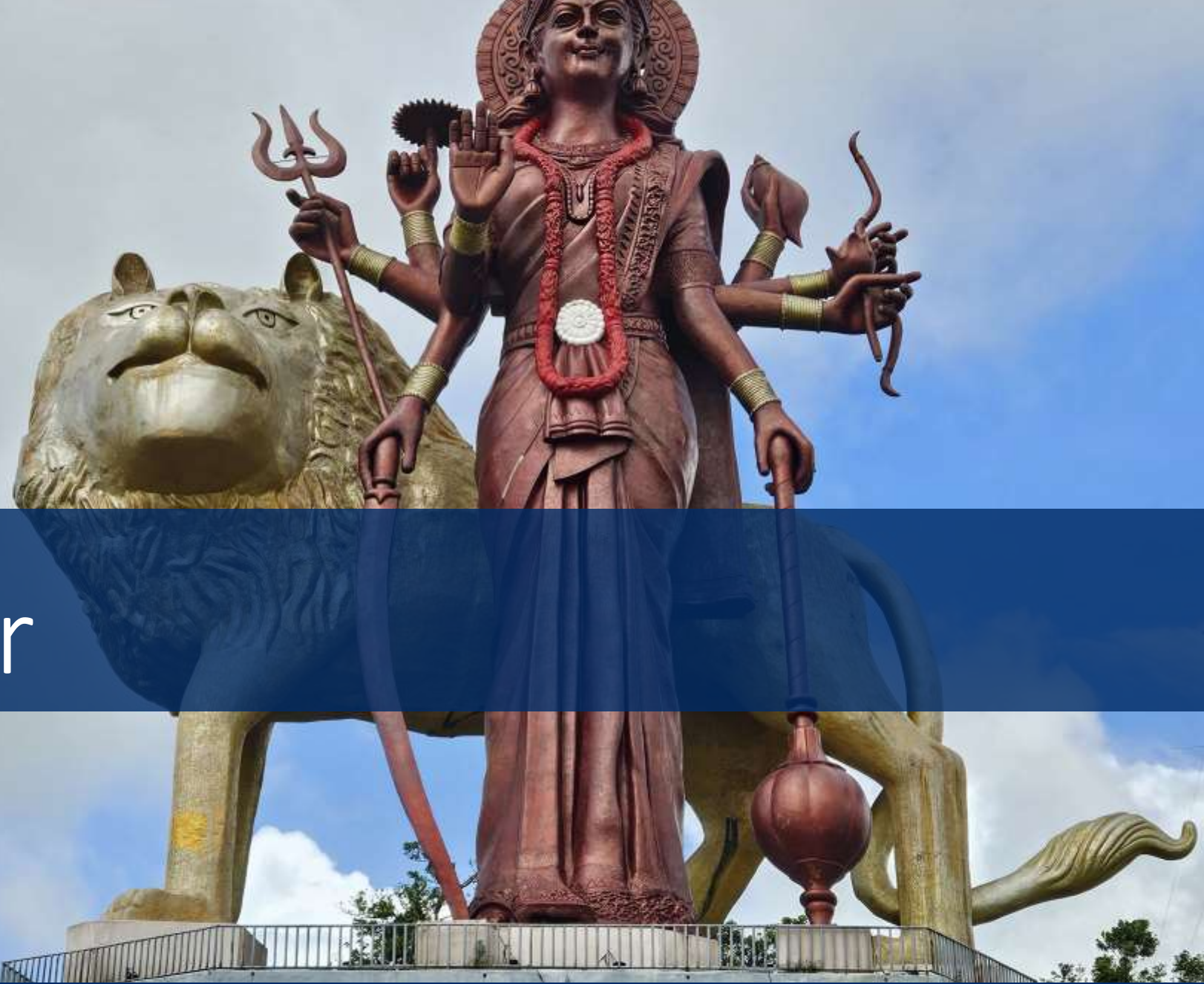
<https://github.com/bchavez/Bogus>

Install-Package Bogus



 Demo 

CsvHelper



# CsvHelper

Schnelles, gut konfigurierbares API zum Arbeiten mit CSV-Dateien

```
// Schreiben
using (var csv = new CsvWriter(writer, CultureInfo.InvariantCulture))
{
    csv.WriteRecords(testUsers);
}

// Lesen
using (var csv = new CsvReader(reader, CultureInfo.InvariantCulture))
{
    var readUser = csv.GetRecords<User>().ToList();
}
```



<https://joshclose.github.io/CsvHelper/>

```
Install-Package CsvHelper
```



 Demo 





# Hangfire





# Hangfire

(Hintergrund-)Job Automatisierung

- Fire-and-Forget Jobs
- Recurring Jobs
- Delayed Jobs
- Continuations
- Graphical Monitor
- Ideal mit IHostedServer in ASP.NET Core



<https://www.hangfire.io/>

Install-Package Hangfire



 ~~Demo~~ 

# Hashids (sqids)





# Hashids (sqids)

Vorhersagbare ID in öffentlichen Web APIs vermeiden

```
public class CustomerDto
{
    [JsonConverter(typeof(HashidsJsonConverter))]
    public int Id { get; set; }
}
```

```
[HttpGet]
[Route("{id:hashids}")]
public ActionResult<CustomerDto> Get([ModelBinder(typeof(HashidsModelBinder))] int id)
{
    return Ok(customers.SingleOrDefault(c => c.Id == id));
}
```



<https://sqids.org/>

Install-Package AspNetCore.Hashids



Demo





A dramatic landscape featuring a vast body of water in the foreground, reflecting the light from a cloudy sky. The sky is filled with dark, heavy clouds, with a bright patch of light breaking through near the top center. The water is calm, with subtle ripples and a shimmering reflection of the light. The overall mood is serene yet powerful. A dark blue horizontal band is overlaid across the middle of the image, containing the word "Humanizer" in white text.

Humanizer

# Humanizer

Zahlen, Größen, Uhrzeiten als leßbare Formate darstellen

```
WriteLine("=== Bytes & Co ===");  
WriteLine(7.Bits().ToString());           // 7 b  
WriteLine(8.Bits().ToString());           // 1 B  
WriteLine(.5.Kilobytes().Humanize());     // 512 B  
WriteLine(1000.Kilobytes().ToString());   // 1000 KB  
WriteLine(1024.Kilobytes().Humanize());   // 1 MB  
WriteLine(.5.Gigabytes().Humanize());     // 512 MB  
WriteLine(1024.Gigabytes().ToString());   // 1 TB
```



Install-Package Humanizer.Core



Demo





A scenic landscape featuring a bright blue sky with scattered white clouds. In the background, a large, dark rock formation or mountain peak rises above a body of water. The foreground shows a sandy beach with shallow, rippling water. A dark blue horizontal band is overlaid across the middle of the image, containing the text 'Automapper' in white.

Automapper



# Automapper

(Rekursives) Mappen von Model/ DTOs/ Listen/ etc.

```
// (Rekusives) Mappen  
UserDto userDto = mapper.Map<UserDto>(user);
```



<https://automapper.org/>

Install-Package Automapper



Demo







# Dynamic Ling



# Dynamic Linq

## Dynamische LINQ-Abfragen (für EF Core)

```
IQueryable<T> source = ...
```

```
foreach (var filter in dataFilter.Filters.Filters)
{
    source = filter.Operator switch
    {
        DataFilterOperatorDto.Eq => source.Where($"{filter.Field} = @0", filter.Value),
        DataFilterOperatorDto.Neq => source.Where($"{filter.Field} <> @0", filter.Value),
        DataFilterOperatorDto.Contains => source.Where($"{filter.Field}.Contains(@0)", filter.Value),
        DataFilterOperatorDto.Doesnotcontain => source.Where($"NOT {filter.Field}.Contains(@0)", filter.Value),
        DataFilterOperatorDto.startswith => source.Where($"{filter.Field}.StartsWith(@0)", filter.Value),
        DataFilterOperatorDto.endswith => source.Where($"{filter.Field}.EndsWith(@0)", filter.Value),
        DataFilterOperatorDto.isnull => source.Where($"{filter.Field} == NULL"),
        DataFilterOperatorDto.Gte => source.Where($"{filter.Field} >= @0", filter.Value),
        DataFilterOperatorDto.Gt => source.Where($"{filter.Field} > @0", filter.Value),
        DataFilterOperatorDto.Lte => source.Where($"{filter.Field} <= @0", filter.Value),
        DataFilterOperatorDto.Lt => source.Where($"{filter.Field} < @0", filter.Value),
        DataFilterOperatorDto.Isnotnull => source.Where($"{filter.Field} != NULL"),
        _ => throw new InvalidOperationException(nameof(filter.Operator))
    };
}
```



<https://www.nuget.org/packages/System.Linq.Dynamic.Core/>

Install-Package System.Linq.Dynamic.Core



Demo



A street scene in a developing country. In the foreground, a dark mule is harnessed to a wooden cart. The cart is loaded with a large yellow cushion and a green metal container. A man in a light-colored shirt and cap is driving the cart. In the background, a motorized rickshaw is visible, along with several people walking on the street. The buildings are multi-story and have a warm, reddish-brown color. A sign for 'EXCHANGE' is visible on one of the buildings. The sky is clear and blue.

BenchmarkDotnet



# BenchmarkDotnet

## Performance Messungen für Codepassagen

```
_ = BenchmarkRunner.Run<ParseStringToInt>();
```

```
[MemoryDiagnoser]
0 references | Thorsten Kansy, 266 days ago | 1 author, 1 change
public class ParseStringToInt
{
    const string NUMBER = "123";

    [Benchmark]
    0 references | Thorsten Kansy, 266 days ago | 1 author, 1 change
    public int ParseByInternalBenchmark() => ParseByInternal(NUMBER);
    [Benchmark]
    0 references | Thorsten Kansy, 266 days ago | 1 author, 1 change
    public int ParseByInternalNoConstBenchmark() => ParseByInternal("123");

    2 references | Thorsten Kansy, 266 days ago | 1 author, 1 change
    public int ParseByInternal(string number) => int.Parse(number);
}
```



<https://github.com/dotnet/BenchmarkDotNet>

Install-Package BenchmarkDotNet



Demo



# Fragen? Jetzt oder später!



## Kontakt

---

 **E-Mail**  
[tkansy@dotnetconsulting.eu](mailto:tkansy@dotnetconsulting.eu)

 **Telefon**  
+49 (0) 6187 / 2009090

 **Microsoft Teams**  
[Meet now](#)

 **LinkedIn**  
[Link me](#)

 **XING**  
[Xing me](#)

 **X (Twitter)**  
[@tkansy](#)





# Bewertung der Session



# www.dotnetconsulting.eu

SQL Server meets .NET (Core)- professionally!



Ich berate, coache und trainiere im Bereich Entwicklung von .NET (Core) Anwendungen mit Microsoft SQL Server- mit Allem, was dazu gehört- und was man vielleicht weglassen sollte.