



.NET 6 Neuerungen

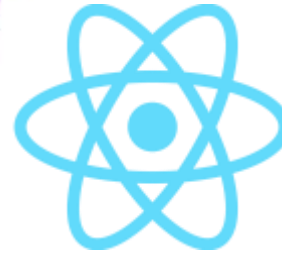
.NET 6, ASP.NET 6.0, Entity Framework Core 6.0 & Co.



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

- .NET 6
- ASP.NET Core 6.0
- Entity Framework Core 6.0
- Breaking Changes
- Sonstiges

A black and white bird, possibly a nighthawk, is perched on a branch with small, rounded green leaves. The bird is facing right, with its head slightly tilted upwards. The background is a soft-focus green, suggesting a natural outdoor setting. A dark blue horizontal band is overlaid across the middle of the image, containing the text ".NET 6".

.NET 6

Koexistenz .NET Core, .NET 5.0 & 6

Festlegen der Version via CLI

```
dotnet new tool-manifest -force  
  
dotnet tool install --local --version 3.1.21 dotnet-ef  
dotnet tool install --local --version 5.0.12 dotnet-ef  
dotnet tool install --local --version 6.0.0 dotnet-ef  
  
dotnet tool update --global --prerelease dotnet-ef
```

dotnet-tools.json

global.json

Neue Projektvorlagen

- Console und Web-App in minimaler Form
 - Top-Level-Statement (C# 9)
 - Implicit Namespace Imports (C# 10)
- Nullable reference Types aktiviert



Demo



PreCompiler switches

`#IF...#ELIF...#ENDIF`

Target Frameworks	Symbols
.NET Framework	NETFRAMEWORK, NET48, NET472, NET471, NET47, NET462, NET461, NET46, NET452, NET451, NET45, NET40, NET35, NET20, NET48_OR_GREATER, NET472_OR_GREATER, NET471_OR_GREATER, NET47_OR_GREATER, NET462_OR_GREATER, NET461_OR_GREATER, NET46_OR_GREATER, NET452_OR_GREATER, NET451_OR_GREATER, NET45_OR_GREATER, NET40_OR_GREATER, NET35_OR_GREATER, NET20_OR_GREATER
.NET Standard	NETSTANDARD, NETSTANDARD2_1, NETSTANDARD2_0, NETSTANDARD1_6, NETSTANDARD1_5, NETSTANDARD1_4, NETSTANDARD1_3, NETSTANDARD1_2, NETSTANDARD1_1, NETSTANDARD1_0, NETSTANDARD2_1_OR_GREATER, NETSTANDARD2_0_OR_GREATER, NETSTANDARD1_6_OR_GREATER, NETSTANDARD1_5_OR_GREATER, NETSTANDARD1_4_OR_GREATER, NETSTANDARD1_3_OR_GREATER, NETSTANDARD1_2_OR_GREATER, NETSTANDARD1_1_OR_GREATER, NETSTANDARD1_0_OR_GREATER
.NET 5+ (and .NET Core)	NET, NET6_0, NET6_0_ANDROID, NET6_0_IOS, NET6_0_MACOS, NET6_0_MACCATALYST, NET6_0_TVOS, NET6_0_WINDOWS, NET5_0, NETCOREAPP, NETCOREAPP3_1, NETCOREAPP3_0, NETCOREAPP2_2, NETCOREAPP2_1, NETCOREAPP2_0, NETCOREAPP1_1, NETCOREAPP1_0, NET6_0_OR_GREATER, NET6_0_ANDROID_OR_GREATER, NET6_0_IOS_OR_GREATER, NET6_0_MACOS_OR_GREATER, NET6_0_MACCATALYST_OR_GREATER, NET6_0_TVOS_OR_GREATER, NET6_0_WINDOWS_OR_GREATER, NET5_0_OR_GREATER, NETCOREAPP_OR_GREATER, NETCOREAPP3_1_OR_GREATER, NETCOREAPP3_0_OR_GREATER, NETCOREAPP2_2_OR_GREATER, NETCOREAPP2_1_OR_GREATER, NETCOREAPP2_0_OR_GREATER, NETCOREAPP1_1_OR_GREATER, NETCOREAPP1_0_OR_GREATER



<https://docs.microsoft.com/en-us/ef/core/miscellaneous/cli/powershell>

 Demo 

CLI

Neue CLI Erweiterungen

```
dotnet sdk check
```

```
dotnet new ...
```

```
dotnet watch...
```

dotnet sdk check

Installierte SDKs überprüfen

```
dotnet sdk check
```

```
Command Prompt
Try out the newest .NET SDK features with .NET 6.0.100.

.NET Runtimes:
Name                               Version                               Status
-----
Microsoft.AspNetCore.All           2.1.30                               .NET 2.1 is out of support.
Microsoft.AspNetCore.App           2.1.30                               .NET 2.1 is out of support.
Microsoft.NETCore.App              2.1.30                               .NET 2.1 is out of support.
Microsoft.AspNetCore.App           3.1.20                               Patch 3.1.21 is available.
Microsoft.NETCore.App              3.1.20                               Patch 3.1.21 is available.
Microsoft.WindowsDesktop.App       3.1.20                               Patch 3.1.21 is available.
Microsoft.AspNetCore.App           5.0.9                                Patch 5.0.12 is available.
Microsoft.NETCore.App              5.0.9                                Patch 5.0.12 is available.
Microsoft.WindowsDesktop.App       5.0.9                                Patch 5.0.12 is available.
Microsoft.AspNetCore.App           5.0.11                               Patch 5.0.12 is available.
Microsoft.NETCore.App              5.0.11                               Patch 5.0.12 is available.
Microsoft.WindowsDesktop.App       5.0.11                               Patch 5.0.12 is available.
Microsoft.NETCore.App              6.0.0-preview.6.21352.12            Patch 6.0.0 is available.
Microsoft.WindowsDesktop.App       6.0.0-preview.6.21353.1            Patch 6.0.0 is available.
Microsoft.AspNetCore.App           6.0.0-preview.6.21355.2            Patch 6.0.0 is available.
Microsoft.AspNetCore.App           6.0.0-rc.2.21480.10                Patch 6.0.0 is available.
Microsoft.NETCore.App              6.0.0-rc.2.21480.5                 Patch 6.0.0 is available.
Microsoft.WindowsDesktop.App       6.0.0-rc.2.21501.6                 Patch 6.0.0 is available.

The latest versions of .NET can be installed from https://aka.ms/dotnet-core-download. For more information about .NET 1
lifecycles, see https://aka.ms/dotnet-core-support.

C:\Users\tkans>
```

Deployment

- Trimming (Tree Shaking)
 - Fraglich ob für WPF-Apps
- Single-File-Publishing
 - EnableCompressionInSingleFile: Optionale Komprimierung
 - Dateigröße vs. Startzeit

CLI Workloads

Modulare Erweiterungen durch optionale Workloads (Blazor Aot, MAUI, etc.)

```
dotnet workload install
```

```
dotnet workload update
```

```
dotnet workload repair
```



Demo



AOT (Ahead of time compilation)

Aktuell nur für Blazor WASM, verschoben auf .NET 7

```
dotnet workload install microsoft-net-sdk-blazorwebassembly-aot  
  
dotnet publish -c Release
```

```
Install-Package Microsoft.NET.Sdk.BlazorWebAssembly.AOT
```

Projects –Datei

```
<RunAOTCompilation>>true</RunAOTCompilation>
```


A stone sculpture of a seated figure, possibly a deity or a person of high status, with a prominent red mouth. The figure is set against a background of colorful, ornate patterns, including green and gold scrollwork. The sculpture is made of a light-colored stone, possibly limestone or marble, and shows signs of weathering and repair. The figure is seated in a meditative or contemplative pose, with hands resting on the lap. The background features intricate carvings and painted patterns in red, green, and gold, suggesting a traditional East Asian setting.

.NET 6 – API (Klassen)

DateOnly & TimeOnly

Nie wieder ein (Geburtstags-)Datum mit Uhrzeit :-)

```
DateOnly d1 = new(1971, 12, 18);  
DateOnly d2 = DateOnly.FromDateTime(DateTime.Today);  
Console.WriteLine($"{d1} / {d2}");
```

```
TimeOnly t1 = new(18, 30, 15);  
TimeOnly t2 = TimeOnly.FromDateTime(DateTime.Now);  
Console.WriteLine($"{t1} / {t2}");
```



Demo



LINQ-Operators

- `DistinctBy`, `UnionBy`, `IntersectBy`, `ExceptBy`
- `MaxBy` **und** `MinBy`
- `Chunk`
- `FirstOrDefault`, `LastOrDefault` **und** `SingleOrDefault`
 - Standardwert kann nun bestimmt werden (bei leeren Listen)
- **VERBESSERUNGEN**
 - `Zip`, `ElementAt`, `Take`
- `TryGetNonEnumeratedCount`



Demo



PriorityQueue

Priorisierte Queue

```
PriorityQueue<string, int> taskList = new();  
taskList.Enqueue("Task 1", 1);  
taskList.Enqueue("Task 2", 1);  
taskList.Enqueue("Task 3", 3);  
taskList.Enqueue("Task 4", 2);  
  
while (taskList.TryDequeue(out string? taskName, out int prio))  
{  
    Console.WriteLine($"prio: {prio}, taskName: {taskName}");  
}
```

 Demo 

EnsureCapacity

Kapazität einer Liste festlegen/ nachträglich erhöhen

```
List<int> list = new();  
// später im Code  
list.EnsureCapacity(CAPACITY);  
// wiederum später im Code  
list.EnsureCapacity(2 * CAPACITY);
```

```
// Alternative für (ohne nachträgliche Vergrößerung):  
List<int> list = new(CAPACITY);
```


 Demo 

Json

JsonNode	Json-Knoten
JsonObject	(Komplexes) Json-Objekt
JsonPropertyOrder-Attribute	Reihenfolge bei Serialisierung
IJsonOnSerializing, IJsonOnSerialized, IJsonOnDeserializing, IJsonOnDeserialized	Validierung, etc.

JSON

```
// Node aus JSON-Text erstellen (parsen)  
JsonNode jsonNode = JsonNode.Parse(@"{"Name": "Thorsten K
```

```
JsonObject jsonObject = new()  
{  
    ["Professional"] = new JsonObject()  
    {  
        ["Name"] = "Thorsten Kansy",  
        ["Topics"] = new JSONArray("Freier Consultant  
    }  
};  
  
// Zugriff auf Nodes  
Debug.Assert(jsonObject["Professional"]!["Topics"]![1]
```

```
int value = (int)jsonNode["Age"]!;
```

```
[JsonPropertyOrder(1)]
```

5 references | Thorsten Kansy, 2 days ago | 1 author, 1 change
`public string Id { get; set; }`

```
[JsonPropertyOrder(3)]
```

2 references | Thorsten Kansy, 2 days ago | 1 author, 1 change
`public string Givenname { get; set; }`

 Demo 

Direct IO

System.IO.RandomAccess

```
// Handle erzeugen
using Microsoft.Win32.SafeHandles.SafeFileHandle handle =
    File.OpenHandle(filename);

// Einlesen
Span<byte> buffer = new byte[255];
int bytesRead = RandomAccess.Read(handle, buffer, fileOffset: 7);
```

 Demo 

Open Telemetry

System.Diagnostics.Metrics.MeterListener („Empfänger“)

```
// Meter Listener konfigurieren
using var listener = new MeterListener();

listener.InstrumentPublished = (instrument, meterListener) =>
{
    if (instrument.Name == INSINSTRUMENTNAME && instrument.Meter.Name == COUNTERNAME)
        meterListener.EnableMeasurementEvents(instrument, null);
};

listener.SetMeasurementEventCallback<int>((instrument, measurement, tags, state) =>
{
    Console.WriteLine($"Instrument: {instrument.Name} has recorded the measurement: {measurement}
        Tagged: {tags[0].ToString()} State: {state} ");
});

// Starten
listener.Start();
```

Open Telemetry

System.Diagnostics.Metrics.Meter („Sender“)

```
// Meter erstellen
using var meter = new Meter(COUNTERNAME, "v1.0");
Counter<int> counter = meter.CreateCounter<int>(INSINSTRUMENTNAME);

// Werte erzeugen
for (int i = 0; i < 10; i++)
{
    counter.Add(Random.Shared.Next(100),
                KeyValuePair.Create<string, object?>("Time", DateTime.Now));
}
```


 Demo 

Direkter Speicherzugriff

System.Runtime.InteropServices.NativeMemory

```
// Speicher allokieren
int* number = (int*)NativeMemory.AllocZeroed(4);

// Wert via Pointer zuweisen
*number = 4;

// Quadrieren
SquarePtrParam(number);

// Ausgabe des Ergebnis
Console.WriteLine($"i: {*number}");

// Speicher freigeben
NativeMemory.Free(number);
```

 Demo 

Compile-time logging source generation

Logging per Attribut auf Interfaces oder partiellen Klassen

```
public static partial class MyClass
{
    [LoggerMessage( eventId = 0, Level = LogLevel.Critical, Message = "Could not open socket to `{hostName}`")]
    public static partial void CouldNotOpenSocket(ILogger logger, string hostName);
}

public static interface IMyInterface
{
    [LoggerMessage( eventId = 47, Level = LogLevel.Critical, Message = "Could not open socket to `{hostName}`")]
    static void CouldNotOpenSocket(ILogger logger, string hostName);
}
```

 Demo 



ASP.NET 6.0

Minimal API

APIs ohne Controller

```
app.MapGet(Urls.GETWEATHER,  
    (IWeatherService weatherService) => weatherService.GetWeather());  
  
app.MapGet(Urls.GETWEATHERLOCATION,  
    (IWeatherService weatherService, string location) =>  
        weatherService.GetWeather(location));  
  
// app.MapPost(...);  
// app.MapPut(...);  
// app.MapDelete(...);
```

 Demo 

Http Logging

- Loggen von Headers, Bodies, etc.

```
builder.Services.AddHttpLogging(logging =>
{
    // Customize HTTP logging here.
    logging.LoggingFields = HttpLoggingFields.All;
    logging.RequestHeaders.Add("My-Request-Header");
    logging.ResponseHeaders.Add("Sensitive");
    logging.ResponseHeaders.Add("Non-Sensitive");
    logging.MediaTypeOptions.AddText("text/html");
    logging.RequestBodyLogLimit = 4096;
    logging.ResponseBodyLogLimit = 4096;
});

app.UseHttpLogging();
```

appsettings.json

Einstellungen unter `Microsoft.AspNetCore.Kestrel.*`

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning",

      "Microsoft.AspNetCore.Kestrel.BadRequests": "Debug",
      "Microsoft.AspNetCore.Kestrel.Connections": "Debug",
      "Microsoft.AspNetCore.Kestrel.Http2": "Debug",
      "Microsoft.AspNetCore.Kestrel.Http3": "Debug"
    }
  },
  "AllowedHosts": "*"
}
```

 Demo 

HTTP/3 (QUIC)

Http3 Demo Server: <https://pgjones.dev/>

The screenshot shows a web browser displaying the 'Open Source' page of PG Jones. The page content includes the name 'P G Jones', the heading 'Open Source', and a description: 'These are the open source projects that I am involved with, either as the author or as a maintainer.' Below this is the 'Quart' logo and the text 'Quart is a Python ASGI web'. The browser's developer tools are open to the Network tab, showing a list of requests. The table below is a representation of the data shown in the Network tab:

Name	Status	Protocol	Type	Initiator
start-a8cd1609.css	200	h3	stylesheet	[index]
vendor-82544678.css	200	h3	stylesheet	[index]
start-df62298d.js	200	h3	script	[index]
vendor-2649d3b0.js	200	h3	script	[index]
__layout.svelte-115dbf93.js	200	h3	script	start-df62298d.js
error.svelte-c5cb32d.js	200	h3	script	start-df62298d.js
index.svelte-494825f.js	200	h3	script	start-df62298d.js
MeSudbury.webp	200	h3	text/plain	vendor-2649d3b0.js
Quart.webp	200	h3	text/plain	vendor-2649d3b0.js
Hypercom.webp	200	h3	text/plain	vendor-2649d3b0.js
data:image/svg+xml=...	200	data	svg+xml	theme.css
favicon.ico	200	h3	text/html	Other

Summary statistics at the bottom of the Network tab: 15 requests | 2.1 kB transferred | 475 kB resources | Finish: 130 ms | DOMContentLoaded: 68 ms | Load: 70 ms



<https://kinsta.com/blog/http3/>



Coming soon

HTTP/3 & Kestrel

Nur Windows 11/ Windows Server 2022 und neuer

```
var builder = WebApplication.CreateBuilder(args);

builder.WebHost.ConfigureKestrel((context, options) =>
{
    options.Listen(IPAddress.Any, 5001, listenOptions =>
    {
        // Use HTTP/3
        listenOptions.Protocols = HttpProtocols.Http1AndHttp2AndHttp3;
        listenOptions.UseHttps();
    });
});
```

 Demo 



Blazor 6.0 WASM



PageTitle, HeadContent & HeadOutlet

```
var builder = WebAssemblyHostBuilder.CreateDefault(args);  
  
builder.RootComponents.Add<App>("app");  
builder.RootComponents.Add<HeadOutlet>("head::after");
```

Program.cs

```
...  
<PageTitle>TimeTracking: @DateOnly.FromDateTime(DateTime.Today)</PageTitle>  
...  
<HeadContent>  
    <meta name="description" content="Use this page to do things!" />  
    <meta name="author" content="Thorsten Kansy">  
    <link rel="icon" href="favicon.ico" type="image/x-icon">  
</HeadContent>  
...
```

*.razor

 Demo 

AoT

Leider nur bei Blazor-Anwendungen beim Deployment => .NET 7

Microsoft: 5 x schneller in der Ausführung, jedoch längere Ladezeiten

```
dotnet workload install wasm-tools
```

 Demo 

Blazor Query String Component Parameters

```
...  
@code{  
    [Parameter]  
    [SupplyParameterFromQuery]  
    public string Filter { get; set; }  
}  
...  
  
...  
@code{  
    [Parameter]  
    [SupplyParameterFromQuery(Name = "ParameterName")]  
    public string Filter { get; set; }  
}  
...
```

*.razor

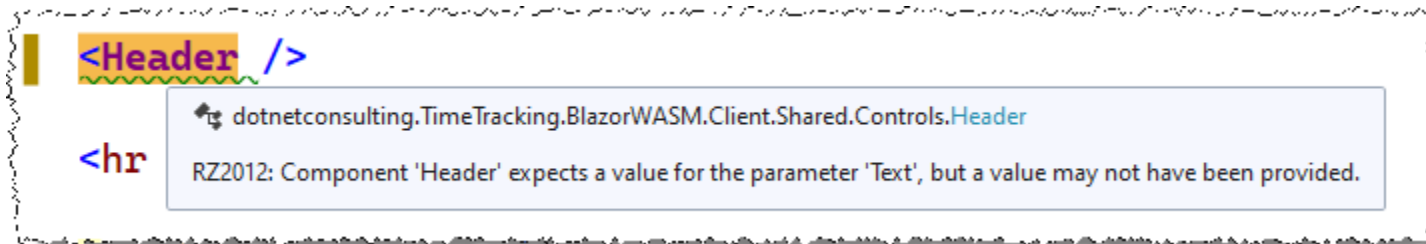
 Demo 

Required (Editor)Parameter

Pflichtparameter für den Editor, leider nur als Warnung

```
<h2>@Text</h2>  
@code  
{  
    [Parameter, EditorRequired]  
    public string Text { get; set; }  
}
```

Header.razor





Demo



Error Boundary

“Html-Try-Catch-Block”

```
...  
<ErrorBoundary>  
  <ChildContent>  
    <BadComponent /> <!-- Löst eine Exception aus -->  
  </ChildContent>  
  <ErrorContent Context="ex">  
    <p>  
      Exception: @ex.Message  
    </p>  
  </ErrorContent>  
</ErrorBoundary>  
...
```

*.razor

 Demo 

Accessibility Improvements

```
<Router AppAssembly="@typeof(Program).Assembly">
  <Found Context="routeData">
    <RouteView RouteData="@routeData" DefaultLayout="@typeof(MainLayout)" />
    <FocusOnNavigate RouteData="@routeData" Selector="h1" />
  </Found>
  <NotFound>
    <LayoutView Layout="@typeof(MainLayout)">
      <p>Sorry, there's nothing at this address.</p>
    </LayoutView>
  </NotFound>
</Router>
...
```



Demo





Entity Framework Core 6.0

IsTemporal (Fluent API)

- Temporal table (SQL Server)
- Standard: -

```
modelBuilder.Entity<MyTemporalTable>()  
    .ToTable(t => t.IsTemporal(hist =>  
        {  
            hist.UseHistoryTable("SessionHistory");  
        }  
    );
```

```
dbContext.MyTemporalTable  
    .TemporalAsOf(<Date>).Where(w => ...);
```

Temporal AsOf & Co

Temporal table (SQL Server)

Linq Operator	Temporal Klausel
TemporalAll()	Alle Daten aus der Tabelle und der historischen Tabelle.
TemporalAsOf(...)	Daten zu einem bestimmten Zeitpunkt einschränken.
TemporalFromTo(...)	Daten aus der Tabelle plus der historischen Tabelle exk. Einschluss der oberen Grenze.
TemporalBetween(...)	Daten aus der Tabelle plus der historischen Tabelle inkl. Einschluss der oberen Grenze.
TemporalContainedIn(...)	Nur Daten aus den der historischen Tabelle aus dem Zeitraum.

 Demo 

Pre-convention model configuration

Standard Konvention

```
protected override void ConfigureConventions(ModelConfigurationBuilder configurationBuilder)
{
    // Standard Konvention festlegen
    configurationBuilder.Properties<string>()
        .HasMaxLength(100);
    configurationBuilder.Properties<decimal>()
        .HavePrecision(18, 3);

    // Auch Interfaces sind erlaubt
    configurationBuilder.IgnoreAny<INonPersisted>();

    base.ConfigureConventions(configurationBuilder);
}
```



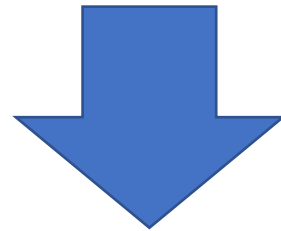

Demo



Migration Bundles

Migration via DevOps freundlicher EXE

```
dotnet ef migrations bundle
```



```
efbundle.exe
```

oder

```
efbundle.exe --connection {$ENVVARWITHCONNECTION}
```

 Demo 

Compiled Models

Geschwindigkeitsgewinn durch Vorkompilierung des Models zur Entwurfszeit

```
dotnet ef dbcontext optimize --output-dir  
MyCompiledModels --namespace MyCompiledModels
```

```
optimize-dbcontext -outputdir MyCompiledModels -  
namespace MyCompiledModels
```

- Keine Global Query Filter
- Kein LazyLoading (da kein Proxy Change-Tracking)

Compiled model bootstrapping

Model einbinden

```
...  
optionsBuilder.UseSqlServer("<connection string>");  
optionsBuilder.UseModel(MyCompiledModels.DemoDbContextModel.Instance);  
...
```



Demo



IsSparse (Fluent API)

- Gibt an, das es sich um eine Sparse Column handelt
- Standard: False

```
modelBuilder.Entity<Session>()  
    .Property(p => p.Abstract)  
    .IsSparse();
```

 Demo 

Misc I

- Scaffold-DbContext übernimmt Kommentare zu Tabellen und Spalten
- Neue Attribute
 - [Unicode]
 - [Precision]
 - [EntityTypeConfiguration]

Misc II

- `String.Concat()` mehr als zwei Parameter
- `EF.Functions.Random()`
- `EF.Functions.Contains()` & `EF.Functions.FreeText()`
funktionieren jetzt auch mit Value Convertern

Misc III



- SQL Server
 - `IsNullOrWhitespace()`
- SQLite
 - `ToString()`
 - Savepoints API



WPF

WPF

- Hot reload
- Blazor Desktop Integration (WebView2 mit Chromium engine)

 ~~Demo~~ 





WinForms



WinForms

- Hot reload
- Blazor Desktop Integration (WebView2 mit Chromium engine)
- Application-wide default font
- More runtime designers
- Accessibility improvements
- High DPI Support

 ~~Demo~~ 



Breaking Changes



Breaking Changes

Zum Glück recht überschaubar

- .NET 6
 - [Breaking changes in .NET 6 - .NET | Microsoft Docs](#)
- EF Core 6.0
 - [EF Core: Breaking Changes in EF Core 6.0 | Microsoft Docs](#)

Fragen? Jetzt oder später!



www.dotnetconsulting.eu



Links



@Tkansy



tkansy@dotnetconsulting.eu



www.dotnetconsulting.eu