

.NET-Anwendungen konfigurieren

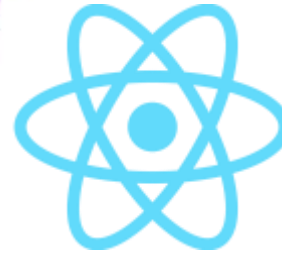
Wege und Möglichkeiten



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

- Provider
- Quellen
- IOptions-Pattern
- Validierung
- User Secrets
- Custom Provider



Provider

Konfigurationsprovider

Provider	NuGet-Paket
JSON-Datei	<code>Microsoft.Extensions.Configuration.Json</code>
XML-Datei	<code>Microsoft.Extensions.Configuration.Xml</code>
INI-Datei	<code>Microsoft.Extensions.Configuration.Ini</code>
Umgebungsvariablen	<code>Microsoft.Extensions.Configuration.EnvironmentVariables</code>
Programmargumente	<code>Microsoft.Extensions.Configuration.CommandLine</code>
In-Memory-Collection	–
Key-per-File	<code>Microsoft.Extensions.Configuration.KeyPerFile</code>
Azure Key Vault	<code>Azure.Extensions.AspNetCore.Configuration.Secrets</code>
User Secrets (nur ASP.NET Core)	<code>Microsoft.Extensions.Configuration.UserSecrets</code>
Custom	–

```
Install-Package Microsoft.Extensions.Configuration.Json (s.o.)
```



Quellen & Werte

Quellen & Werte

- Provider können mehrfach verwendet werden
- Auf die Reihenfolge kommt es an!
- Alle Quellen werden der Reihe nach verarbeitet
 - Jeder Wert wird unter einem Schlüssel gespeichert
 - `Section1:SubSection1:SubSubSection1:Value1`
 - Bereits vorhandene Werte mit gleichen Schlüssel werden überschrieben

Unabhängig von der Quelle!

```
},  
"Section1": {  
  "SubSection1": {  
    "SubSubSection1": {  
      "Value1": "Value1",  
      "Value2": "Value2"  
    }  
  }  
}
```


Quellen – ASP.NET Core

- Environment variables (Prefix “ASPNETCORE_”)
- Environment variables (Prefix “DOTNET_”)
- appsettings.json
- appsettings.{Environment}.json
- Secret Manager (Development environment)
- Environment variables (kein Prefix)
- Command-line arguments



<https://learn.microsoft.com/en-us/aspnet/core/fundamentals/configuration/>

Quellen – GenericHost

- Environment variables (Prefix “DOTNET_”)
- appsettings.json
- appsettings.{Environment}.json
- Secret Manager (Development environment)
- Environment variables (kein Prefix)
- Command-line arguments



<https://learn.microsoft.com/en-us/dotnet/core/extensions/generic-host?tabs=appbuilder>



Demo





IOptions-Pattern

IOptions, IOptionsSnapshot & IOptionsMonitor

`IOptions<TOptions>`

- Konfigurationsdaten lesen, nachdem die App gestartet wurde

`IOptionsSnapshot<TOptions>`

- Ist in Szenarios nützlich, in denen Optionen bei jeder Anforderung neu berechnet werden sollten..

`IOptionsMonitor<TOptions>`

- Wird verwendet, um Optionen abzurufen und Benachrichtigungen über Optionen für `TOptions`-Instanzen zu verwalten
- Unterstützt:
 - Änderungsbenachrichtigungen
 - Benannte Optionen
 - Erneut ladbare Konfiguration
 - Selektive Optionsvalidierung (`IOptionsMonitorCache<TOptions>`)

Konfiguration via DI zur Verfügung stellen

IOptions<>-Pattern

```
private static void ConfigureServices(...)
{
    ...
    IConfigurationSection doSomethingJobConfigurationSection =
        hostBuilderContext.Configuration.GetSection("Jobs:DoSomethingJob");
    services.Configure<DoSomethingJobSettings>(doSomethingJobConfigurationSection);
    ...
}

public DoSomethingJob(IOptions<DoSomethingJobSettings> doSomethingJobSettings) // Konstruktor
{
    _doSomethingJobSettings = doSomethingJobSettings.Value;
}
```



Demo



A wide-angle photograph of a traditional Korean gate (Gwanak) with a prominent blue tiled roof and white pillars. A paved path leads through the gate towards a person standing in the distance. The scene is set against a clear blue sky and green trees.

Validierung

Validierung

- Attribute
- IValidateOptions<>-Pattern
- Validierung bei App-Start oder Zugriff auf Konfiguration



Demo





User Secrets



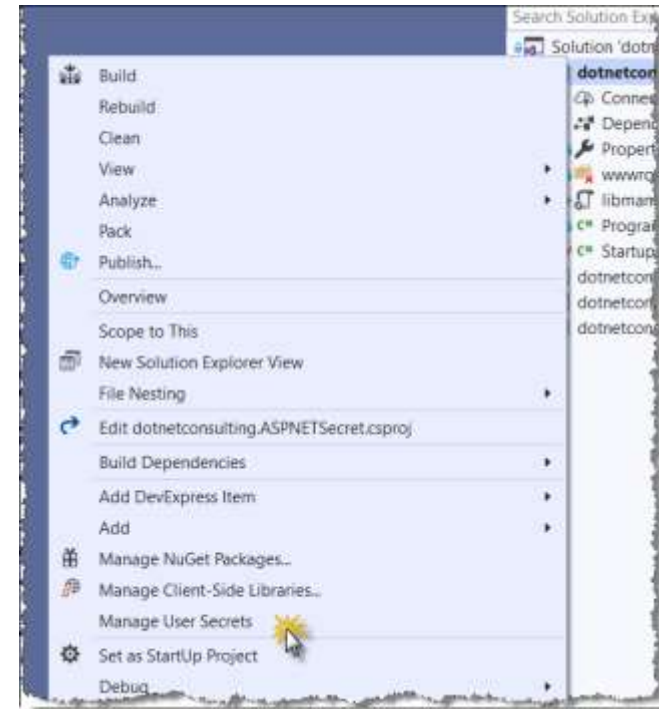
ASP.NET User Secrets

Sensible Einstellungen auslagern

```
<Project Sdk="Microsoft.NET.Sdk.Web">  
  <PropertyGroup>  
    <UserSecretsId>576504d1-3c17-4fc0-b0a0-23ee841be7e6</UserSecretsId>  
  </PropertyGroup>  
</Project>
```

Betriebssystem	Speicherort
Windows	%APPDATA%\microsoft\UserSecrets\ <usersecretsid>\secrets.json</usersecretsid>
Linux	~/.microsoft/usersecrets/<userSecretsId>/secrets.json
MacOS	~/.microsoft/usersecrets/<userSecretsId>/secrets.json

User Secrets verwalten



Aktion	.NET Core CLI
User Secret setzen	<code>dotnet user-secrets set Geheimnis "My Name is Bond"</code>
User Secrets auflisten	<code>dotnet user-secrets list</code>
User Secret löschen	<code>dotnet user-secrets remove Geheimnis</code>
Alle User Secrets löschen	<code>dotnet user-secrets clear</code>

 Demo 

Custom Provider

1 강남역 지하쇼핑센터
세요

M
1
출입구
강남
Gangnam
江南
カンナム
2호선
신분당



Custom Provider

- Eigener File Provider
- Eigener Custom Provider

Eigener File Provider

Code-Klasse	Aufgabe
<code>FileConfigurationProvider</code>	Konfigurationsprovider für Datei basierte Quellen
<code>FileConfigurationSource</code>	Konfigurationsquellen für Datei basierte Quellen



Demo



Eigenen Custom Provider schreiben

Code-Klasse	Aufgabe
<code>SqlConfigurationExtensions</code>	Erweiterung der Fluent API
<code>SqlDatabaseConfigurationSource</code>	Implementiert <code>IConfigurationSource</code> , um eine Konfiguration mittels Fluent API zu ermöglichen.
<code>SqlDatabaseConfigurationProvider</code>	Implementiert <code>IConfigurationProvider</code> und stellt die eigentliche Programmlogik dar, die die Konfigurationswerte bereitstellt.
<code>SqlDatabaseChangeToken</code>	Implementiert <code>IChangeToken</code> und bietet die Möglichkeit, Veränderungen bei den Konfigurationswerten zu signalisieren. Hier nicht weiter vertieft, da die Änderungen zwischen dem Lesen zwei zusammenhängender Werte auftreten können.




Demo



Fragen? Jetzt oder später!

Kontakt

 **E-Mail**
tkansy@dotnetconsulting.eu

 **Telefon**
[+49 \(0\) 6187 / 2009090](tel:+49(0)61872009090)

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

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

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

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



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.