Kostenloses eBook

LERNEN unity-container

Free unaffiliated eBook created from
Stack Overflow contributors.

#unity-container

Inhaltsverzeichnis

Über

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: unity-container

It is an unofficial and free unity-container ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official unity-container.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Kapitel 1: Erste Schritte mit Unity-Container

Bemerkungen

Der Unity Container (Unity Container) ist ein einfacher, erweiterbarer Abhängigkeitsinjektionscontainer. Es erleichtert das Erstellen von lose verbundenen Anwendungen und bietet Entwicklern die folgenden Vorteile: Vereinfachte Objekterstellung, insbesondere für hierarchische Objektstrukturen und Abhängigkeiten. [https://msdn.microsoft.com/de-de/library/ff647202.aspx]

Es sollte auch alle großen Themen innerhalb des Unity-Containers erwähnen und auf die verwandten Themen verweisen. Da die Dokumentation für den Unity-Container neu ist, müssen Sie möglicherweise erste Versionen dieser verwandten Themen erstellen.

Versionen

Ausführung	Versionshinweise	Veröffentlichungsdatum
2.0.0	Einheit 2	2011-05-05
2.1.0	Einheit 2.1	2011-05-11
3.0.0	Einheit 3	2013-04-26
3.5.0	Einheit 3.5	2015-05-15
4.0.0	Die Einheit wurde der OSS-Gemeinschaft übergeben	2015-10-06

Examples

Installation

Um zu beginnen, müssen Sie nur das **Unity-** Nuget-Paket installieren. Führen Sie den folgenden Befehl in der Paket-Manager-Konsole aus:

```
PM> Install-Package Unity
```

Alternativ können Sie Visual Studio verwenden, um Unity in einem bestimmten Projekt mithilfe der Option *NuGet Packages for Solution verwalten* unter Tools -> NuGet Package Manager zu installieren.

Hallo Welt

```
interface IGreeter
{
    void Greet();
}

class Greeter : IGreeter
{
    public void Greet()
    {
        Console.WriteLine("Hello World");
    }
}

class SpanishGreeter : IGreeter
{
    public void Greet()
    {
        Console.WriteLine("Hola Mundo");
    }
}

class FrenchGreeter : IGreeter
{
    public void Greet()
    {
        Console.WriteLine("Bonjour le Monde");
    }
}

class Program
{
    static void Main(string[] args)
    {
        var container = new UnityContainer()
            .RegisterType<IGreeter, SpanishGreeter>("spanish")
            .RegisterType<IGreeter, FrenchGreeter>("french")
            .RegisterType<IGreeter, Greeter>();

        //Get default registration. Outputs "Hello World"
        var greeter = container.Resolve<IGreeter>();
        greeter.Greet();

        //Get specific named registration. Outputs "Hola Mundo"
        greeter = container.Resolve<IGreeter>("spanish");
        greeter.Greet();

        //Get all named registrations (excludes the default one)
        //Outputs "Hola Mundo" and "Bonjour le Monde"
        foreach (var g in container.ResolveAll<IGreeter>())
        {
            g.Greet();
        }

        Console.ReadLine();
    }
}
```

Konstruktorinjektion

```
interface IService
{
    void ProcessRequest();
}

interface IRepository
{
    IEnumerable<string> GetData();
}

class HelloWorldRepository : IRepository
{
    public IEnumerable<string> GetData()
    {
        return new[] { "Hello", "World" };
    }
}

class HelloWorldService : IService
{
    private readonly IRepository repo;
    public HelloWorldService(IRepository repo)
    {
        this.repo = repo;
    }
    public void ProcessRequest()
    {
        Console.WriteLine(String.Join(" ", this.repo.GetData()));
    }
}

class Program
{
    static void Main(string[] args)
    {
        var container = new UnityContainer()
            .RegisterType<IRepository, HelloWorldRepository>()
            .RegisterType<IService, HelloWorldService>();

        //Unity automatically resolves constructor parameters that knows about.
        //It will return a HelloWorldService with a HelloWorldRepository
        var greeter = container.Resolve<IService>();
        //Outputs "Hello World"
        greeter.ProcessRequest();

        Console.ReadLine();
    }
}
```

Erste Schritte mit Unity-Container online lesen: https://riptutorial.com/de/unity-container/topic/5292/erste-schritte-mit-unity-container

Kapitel 2: Unity WebAPI

Examples

Einrichten von Unity with Web API.

1. Fügen Sie Ihrem Projekt Unity hinzu.

Wenn Sie NuGet verwenden , können Sie das Unity-Paket verwenden . Führen `Install-Package Unity` in der Package Manager Console aus. Dadurch wird die Unity-Bibliothek (und ihre Abhängigkeiten) zu Ihrem Projekt hinzugefügt.

2. Erstellen Sie eine Implementierung von `IDependencyResolver` .

Zum Beispiel:

```
public class UnityResolver : IDependencyResolver
{
    protected IUnityContainer Container;

    public UnityResolver(IUnityContainer container)
    {
        if (container == null)
        {
            throw new ArgumentNullException("container");
        }
        this.Container = container;
    }

    public object GetService(Type serviceType)
    {
        try
        {
            return Container.Resolve(serviceType);
        }
        catch (ResolutionFailedException)
        {
            return null;
        }
    }

    public IEnumerable<object> GetServices(Type serviceType)
    {
        try
        {
            return Container.ResolveAll(serviceType);
        }
        catch (ResolutionFailedException)
        {
            return new List<object>();
        }
    }

    public IDependencyScope BeginScope()
    {
```

```
        var child = Container.CreateChildContainer();
        return new UnityResolver(child);
    }

    public void Dispose()
    {
        Container.Dispose();
    }
}
```

3. Registrieren Sie Ihren `IDependencyResolver` in Ihrer `WebApiConfig` .

```
public static class WebApiConfig
{
    public static void Register(HttpConfiguration config)
    {
        // Routes goes here..

        // Create your container.
        var container = new UnityContainer();

        // Do registrations here...

        // Assign your container.
        config.DependencyResolver = new UnityResolver(container);
    }
}
```

Unity WebAPI online lesen: https://riptutorial.com/de/unity-container/topic/6396/unity-webapi

Credits

S. No	Kapitel	Contributors
1	Erste Schritte mit Unity-Container	Clemens Tolboom, Community, Daniel J.G.
2	Unity WebAPI	smoksnes