EBook Gratis

APRENDIZAJE unity-container

Free unaffiliated eBook created from
Stack Overflow contributors.

#unity-container

Tabla de contenido

Acerca de

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.

The content is released under Creative Commons BY-SA, and the list of contributors to each chapter are provided in the credits section at the end of this book. Images may be copyright of their respective owners unless otherwise specified. All trademarks and registered trademarks are the property of their respective company owners.

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

Capítulo 1: Empezando con la unidad-contenedor

Observaciones

Unity Container (Unity) es un contenedor de inyección de dependencia ligero y extensible. Facilita la creación de aplicaciones de acoplamiento flexible y brinda a los desarrolladores las siguientes ventajas: Creación simplificada de objetos, especialmente para estructuras de objetos jerárquicos y dependencias. [https://msdn.microsoft.com/en-us/library/ff647202.aspx]

También debe mencionar cualquier tema importante dentro de unity-container y vincular a los temas relacionados. Dado que la Documentación para unity-container es nueva, es posible que deba crear versiones iniciales de los temas relacionados.

Versiones

Versión	Notas de lanzamiento	Fecha de lanzamiento
2.0.0	Unidad 2	2011-05-05
2.1.0	Unidad 2.1	2011-05-11
3.0.0	Unidad 3	2013-04-26
3.5.0	Unidad 3.5	2015-05-15
4.0.0	Unidad entregada a la comunidad OSS	2015-10-06

Examples

Instalación

Para comenzar, solo necesita instalar el paquete **Unity** nuget. Ejecute el siguiente comando desde la consola del administrador de paquetes:

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

Alternativamente, puede usar Visual Studio para instalar Unity en un proyecto en particular usando la *opción Administrar paquetes NuGet para la solución* en Herramientas -> Administrador de paquetes NuGet.

Hola Mundo

```
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();
    }
}
```

Inyección Constructor

```
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();
    }
}
```

Lea Empezando con la unidad-contenedor en línea: https://riptutorial.com/es/unity-container/topic/5292/empezando-con-la-unidad-contenedor

Capítulo 2: Unity WebAPI

Examples

Configurando Unity con API web.

1. Agrega Unity a tu proyecto.

Si usa NuGet puede usar el paquete Unity . Ejecute `Install-Package Unity` en la consola del Administrador de paquetes. Esto agregará la biblioteca de Unity (y sus dependencias) a su proyecto.

2. Crea una implementación de `IDependencyResolver` .

Por ejemplo:

```
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. Registre su `IDependencyResolver` en su `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);
    }
}
```

Lea Unity WebAPI en línea: https://riptutorial.com/es/unity-container/topic/6396/unity-webapi

Creditos

S. No	Capítulos	Contributors
1	Empezando con la unidad-contenedor	Clemens Tolboom, Community, Daniel J.G.
2	Unity WebAPI	smoksnes