IOC Design Resources

Asked 16 years, 1 month ago Modified 15 years, 11 months ago

Viewed 593 times

5

I've done quite a bit of searching on this and haven't had much luck finding something cohesive. I'm a relatively new developer and have just started in my first professional development position. I know that I have a great deal to learn even in the realm of the basics. Based on listening to PodCasts, reading blogs, papers etc; I've come to understand that keeping separation of concerns, IOC, Dependency Injection in mind when designing and building software seems to be the right thing to do. I get the concepts at a VERY high level and want to approach what I do with this in mined as much as I can.

So, here's the rub. HOW the heck do I design stuff this way? I work on a team that has inherited a web based product that is very tightly coupled, very poorly documented and generally not an easy to maintain bit of sofware. Evryone seems to like the idea of removing some of this couple. They like the idea of developing automated tests (which from what I've read is easier to do with loosely coupled components). Nobody seems to know how to do it. I'm willing to take a stab at it, but I need guidance. Everything I've found always seems to talk about this stuff in a very high level way, or conversely, focuses on just a small piece of the whole. I'd like some guidance on a book, or series of tutorials, or videos, or SOMETHING that takes a somewhat real-world example and shows you how to apply these principles. Ideally, I'd LOVE to see something that says..."Take this order entry app, for example. THIS is how most people put it together today using standard ADO.NET DataSets, blah...blah...blah. NOW! If we apply the IOC principles to make this a loosely coupled project, here's what you do differently. Here's WHY you do it this way, and here's what you have to consider when you try to accomplish this."

I know this is a bit long winded, I'm just a bit frustrated that most of the comprehensive trainig material out there that I've found simply doesn't discuss this topic in a way that someone starting out can apply good practices from day one.

Thanks all for your time.

Steve

c# .net dependency-injection inversion-of-control alt.net

Share

Improve this question

Follow

asked Nov 19, 2008 at 16:11

Steve Brouillard

3,266 ● 5 ● 43 ● 60

6 Answers

Sorted by: Highest score (default)

4

I was in the same situation and i bought these two books

(The PDF version to print out)
http://www.manning.com/osherove/ and
http://www.manning.com/prasanna/

Share Improve this answer Follow

answered Nov 25, 2008 at 5:31

Andy
228 ● 1 ● 2 ● 11

Andy. I've seen the Osherove book before and have been trying to decide whether or not to get it. I hadn't seen the other, but it looks like I'm going to have to add these to the list. I find online reading tough, but I may need to buck-up.
– Steve Brouillard Nov 25, 2008 at 13:53

The Dependency injection book is quite good I +1 for the recommendation. – Thedric Walker Nov 25, 2008 at 17:34

FYI for those that see this later. There is a new DI book focused on .NET here: manning.com/seemann.
– Steve Brouillard Oct 12, 2009 at 12:43

4

You should definitely check out the IoC screencasts on dimecasts.net. They are very straight-forward and will help you to grasp some of the concepts.

Share Improve this answer

answered Nov 19, 2008 at 16:49

Follow

Aaron Palmer
8,982 ● 9 ● 50 ● 77

Thanks for the link. I will definitely check it out.
– Steve Brouillard Nov 19, 2008 at 17:19

3

I would suggest that you check out the book James Kovacs mentioned in [this blog post](). One is particularly poignant for your situation. That is "Working Effectively with Legacy Code." It has very good explanations of the concepts of refactoring. It also gives examples of these concept that, although in a C#, Java, and C++, are very easy to follow.

Share Improve this answer

Follow

answered Nov 23, 2008 at 16:02
Thedric Walker
1,857 ● 15 ● 23

Thedric. Thanks so much. You link to James Kovac's Blog was a fantastic resource! – Steve Brouillard Nov 24, 2008 at 13:15

No problem. I'm just share the wealth. I'm finding the book that I mentioned in my answer quite informative and enjoyable. – Thedric Walker Nov 24, 2008 at 13:40

2

I can only describe what we have come up with. We've borrowed usability syntax and such from various online libraries, but the code is all ours.

Basically, we have what we call a ServiceContainer, an object. There is always a global instance of it, a singleton copy if you wish, static, and thus in a web-application, shared between all users in the appdomain.

A ServiceContainer contains rules. The rules says things like *If someone asks for an object of type XYZ, here's how you go about providing them with it*.

For instance, a rule might be that in order for some code to get an object that implements IDbConnection, the container would construct, configure, and return a new SqlConnection object.

The code in question would thus not know, and not care, that it is using a SqlConnection object, as opposed to a OleDbConnection object.

Having written that, I realize this is not a very good example, because ultimately you end up asking the connection for command objects, and the SQL syntax you give to that object must be tailored to the type of connection. But if we can disregard that point right now, the code would not know that it is connecting to SQL Server, it just knows that it has a connection object.

Now, the *code* in question here would need to be given the container it should use, and thus the rules. This means that from a unit-testing perspective, I could create a new instance of ServiceContainer, write new rules into it for testing purposes, and ask the code to do its thing. Ultimately the code would want to execute some SQl (in this instance) and instead of talking to a real database, it would call my test implementation of IDbConnection and IDbCommand, and thus give me an opportunity to verify that things are working.

More importantly, it gives me a way to return back known dummy-data fitting the test, without having to mock up an entire database.

Now, for the *injection* part, in our case we can ask the container to provide us with objects, that has to be constructed, that relies on other objects.

For instance, let's say we have a IDataAccessLayer interface, and a MSSQLDataAccessLayer implementation.

While the interface doesn't give us any outwards sign that it does any logging, the actual implementation here needs to have somewhere to log all the SQL it executes. Thus, the constructor for the class might look like this:

```
public MSSQLDataAccessLayer(ILogger logger) { ... }
```

In the ServiceContainer object, we have registered the following rules (this is our syntax, you won't find that anywhere else, but it should be easy enough to follow):

```
ServiceContainer.Global.RegisterFactory<ILogger, FileL
    .FactoryScoped()
    .WithParameters(
        new Parameter("directory", @"C:\Temp")
    );
```

```
ServiceContainer.Global.RegisterFactory<IDataAccessLay
()
    .FactoryScoped();
```

FactoryScoped means that each time I ask the container for an object, I get a new one.

The rules, if I write them in english, are like this:

- If anyone needs an implementation if ILogger, construct a new FileLogger object, and take the constructor that needs a "directory" parameter, and use that constructor while passing in "C:\Temp" as the argument
- If anyone needs an implementation of IDataAccessLayer, construct a new MSSQLDataAccessLayer

Notice that I said before that the constructor to MSSQLDataAccessLayer takes an ILogger, yet I didn't specify any parameters here? This gives me the following code to get hold of the access layer object:

```
IDataAccessLayer dal = ServiceContainer.Global.Resolve
```

What happens now is that the container object figures out that there the object is MSSQLDataAccessLayer, and that it has a constructor. This constructor requires an ILogger object, but lo and behold, the container knows how to make one. The container will thus construct a new

FileLogger object, and pass this to the constructor of MSSQLDataAccessLayer object, silently.

Configuration of much of the application dependencies can thus be done once, somewhere central and executed during startup, while the rest of the code is blissfully unaware of all the magic happening here.

For unit-testing purposes I can rewrite the rules to provide my own dummy logger object that just stores the logged text in memory, which allows me to easily verify that what I expected the code to log was actually logged, without having to read in a file afterwards.

The rules gives us lots of power on how to actually provide object instances:

- From a delegate/method, which means we can do all the magic of constructing dependent objects ourselves if we want to
- Automatically from a constructor (either automatically figured out which one to use, or we can provide enough dummy-parameters by names/types to pick one)
- Or we can provide an existing instance to the container (this will then sort-of be like a singleton)

We looked at autofac before coming up with our own, basically we just looked at the wiki showing examples of call syntax and then sat down and wrote our own system that did what we needed.

Share Improve this answer Follow

edited Nov 19, 2008 at 16:38

answered Nov 19, 2008 at 16:33

Lasse V. Karlsen
391k ● 106 ● 646 ● 844

Thanks for this info. I'm actually looking resources that are very comprehensive in terms of teaching the concepts and showing implementation. There's plenty in your answer for me to pick through. – Steve Brouillard Nov 19, 2008 at 17:19

2

I have to direct you to the same open source project I answered for a person asking for example of good unit testing.

> [Looking for *small*, open source, c# project with extensive Unit Testing]()
>
> I recommend looking at CarTrackr it has a wide range of .Net technologies that a developer should be familiar with (Unity, MVC framework especially) and has extensive unit testing. The project is simple enough to digest in 1 sitting but complex enough to actually be more than a proof-of-concept. Their codeplex url is at http://www.codeplex.com/CarTrackr

Share Improve this answer

Follow

edited May 23, 2017 at 12:19

Community Bot
1 ● 1

answered Nov 19, 2008 at 19:49

Chris Marisic
33.1k ● 30 ● 168 ● 261

Thanks much for the info. Looks like I have a little bit of reading to do in between Silverlight 2 and Western Civ (The joy of returning to school). – Steve Brouillard Nov 19, 2008 at 20:43

1

This article by Ayende is best introduction to IoC I have ever seen.

Share Improve this answer

Follow

answered Jan 13, 2009 at 16:12

Mauricio Scheffer
99.7k ● 24 ● 195 ● 279