Задание к модулю Reflection

Общая часть

Наша задача: разработать IoC-контейнер (следуя принципу «Каждый программист должен разработать свой IoC/DI контейнер» © ☺).

В качестве примера мы возьмем Managed Extensibility Framework (MEF), в котором основная настройка контейнера происходит за счет расстановки атрибутов.

Но (!) весь код, включая объявление атрибутов у нас будет свой.

Задание 1.

Используя механизмы Reflection, создайте простейший IoC-контейнер, который позволяет следующее:

1. Разметить классы, требующие внедрения зависимостей одним из следующих способов:
- Через конструктор (тогда класс размечается атрибутом [ImportConstructor])

```
[ImportConstructor]
public class CustomerBLL
{
    public CustomerBLL(ICustomerDAL dal, Logger logger)
    { }
}
```

- Через публичные свойства (тогда каждое свойство, требующее инициализации, размечается атрибутом [Import])

```
public class CustomerBLL
{
    [Import]
    public ICustomerDAL CustomerDAL { get; set; }
    [Import]
    public Logger logger { get; set; }
}
```

При этом, конкретный класс, понятное дело, размечается только одним способом!

2. Разметить зависимые классы:
- Когда класс используется непосредственно

```
[Export]
public class Logger
{ }
```

- Когда в классах, требующих реализации зависимости используется интерфейс или базовый класс

```
[Export(typeof(ICustomerDAL))]
public class CustomerDAL : ICustomerDAL
{ }
```

3. Явно указать классы, которые зависят от других или требуют внедрения зависимостей

```
var container = new Container();
container.AddType(typeof(CustomerBLL));
container.AddType(typeof(Logger));
container.AddType(typeof(CustomerDAL), typeof(ICustomerDAL));
```

4. Добавить в контейнер все размеченные атрибутами [ImportConstructor], [Import] и [Export], указав сборку

```
var container = new Container();
container.AddAssembly(Assembly.GetExecutingAssembly());
```

5. Получить экземпляр ранее зарегистрированного класса со всеми зависимостями

```
var customerBLL = (CustomerBLL)container.CreateInstance(
                                typeof(CustomerBLL));
var customerBLL = container.CreateInstance<CustomerBLL>();
```

Задание 2. (необязательное)

Доработайте контейнер из Задания 1, так чтобы для создания экземпляра использовался сгенерированный кода на основе механизм System.Reflection.Emit;