dealing with dependencies

taking your architecture to rehab

me

I Am:

@jon_fuller

practicing apprentice

SEP

I Am Not:

expert

all-knowing

simplicity

flexible

maintainable

testable

agile

low coupling

DIP

IoC

OMG!

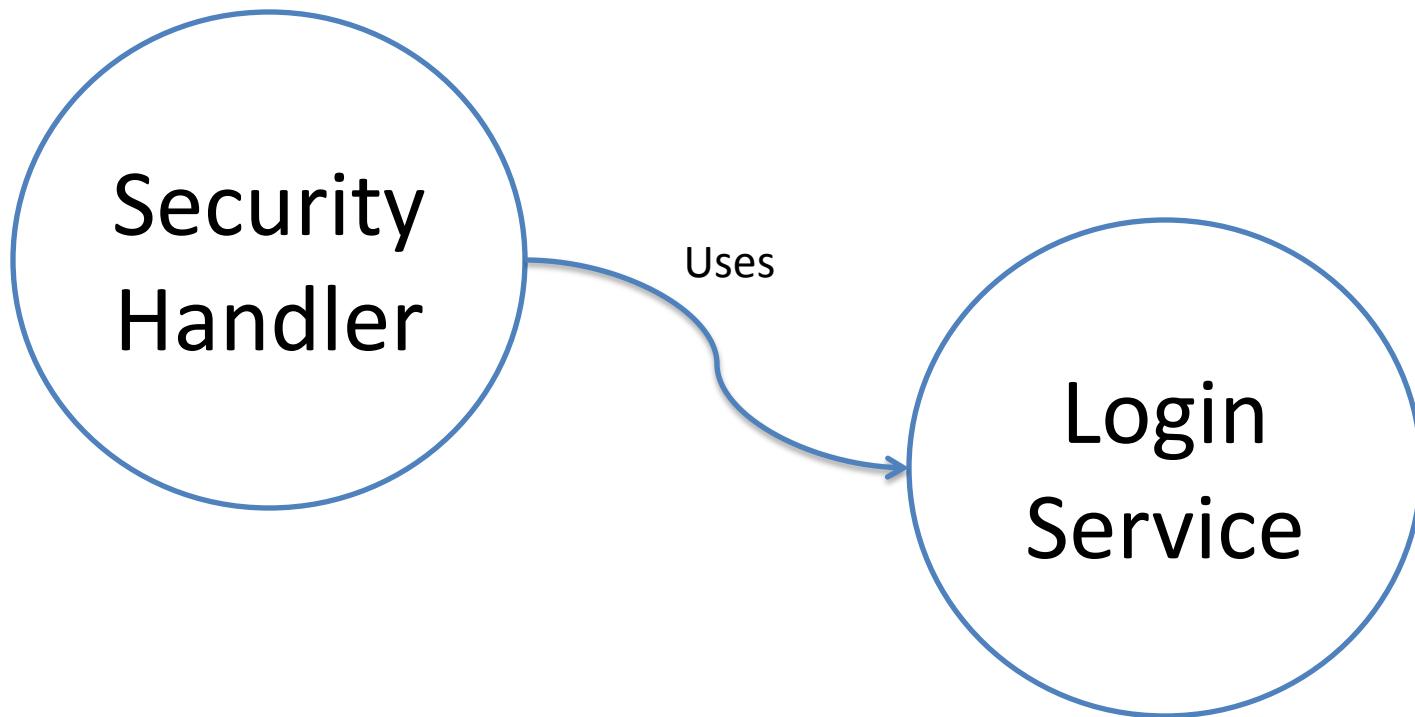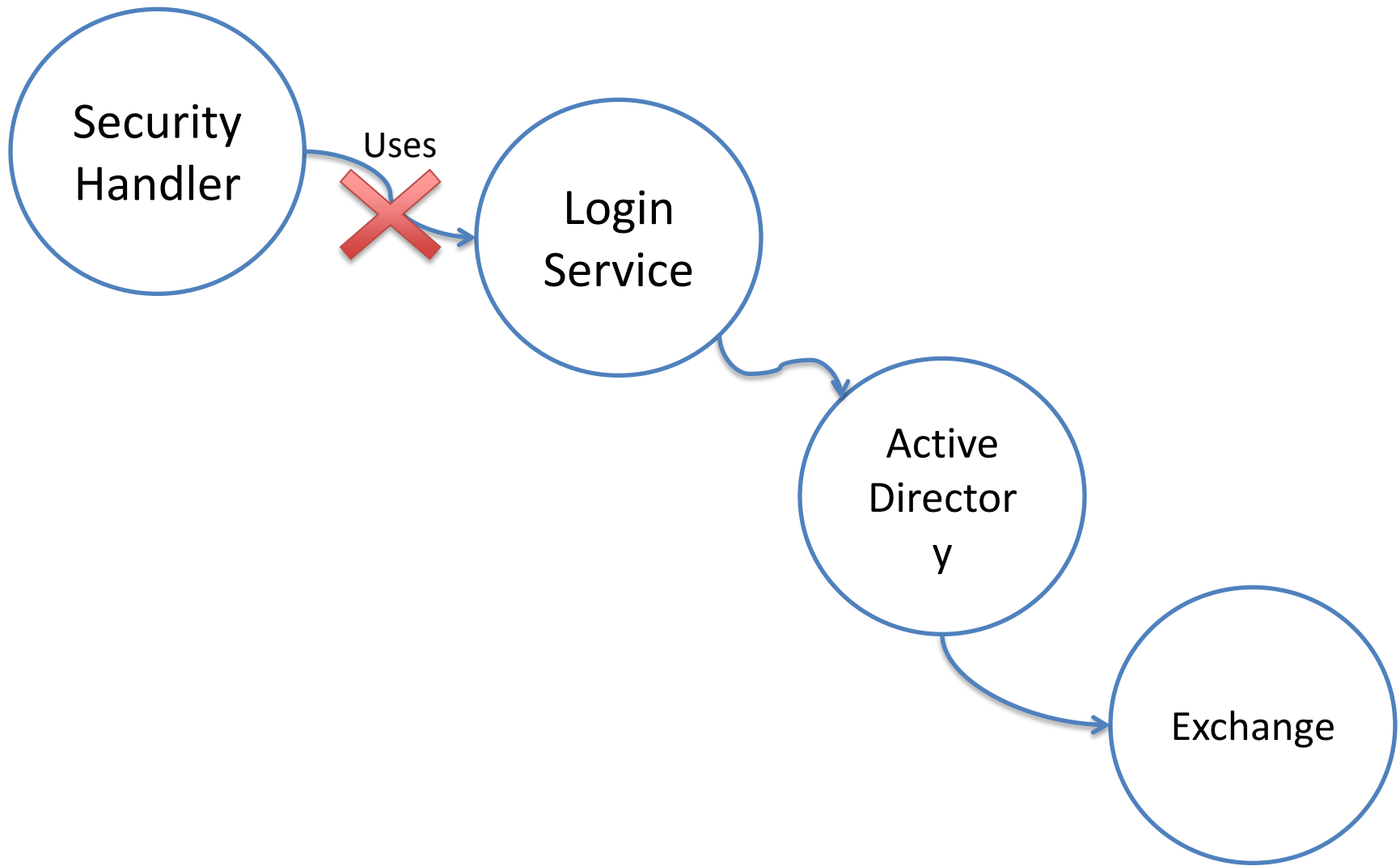acronym overload

DI

WTF?

Single Responsibility
Open-Closed
Liskov Substitution
Interface Segregation
Dependency Inversion

DIP

Thou Shall Invert Dependencies

DIP

Security Handler — Uses → Login Service
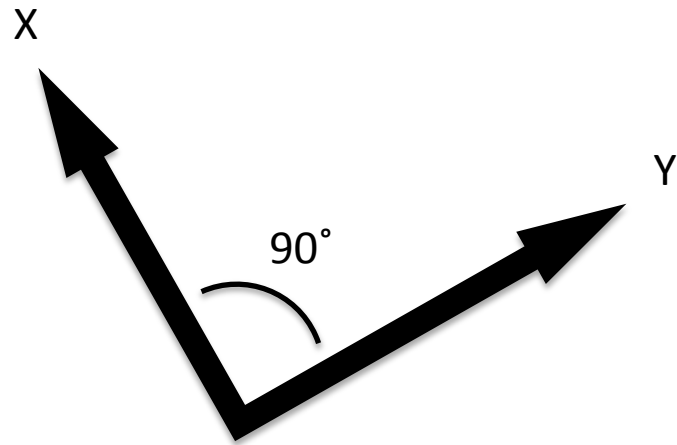
DIP

DIP
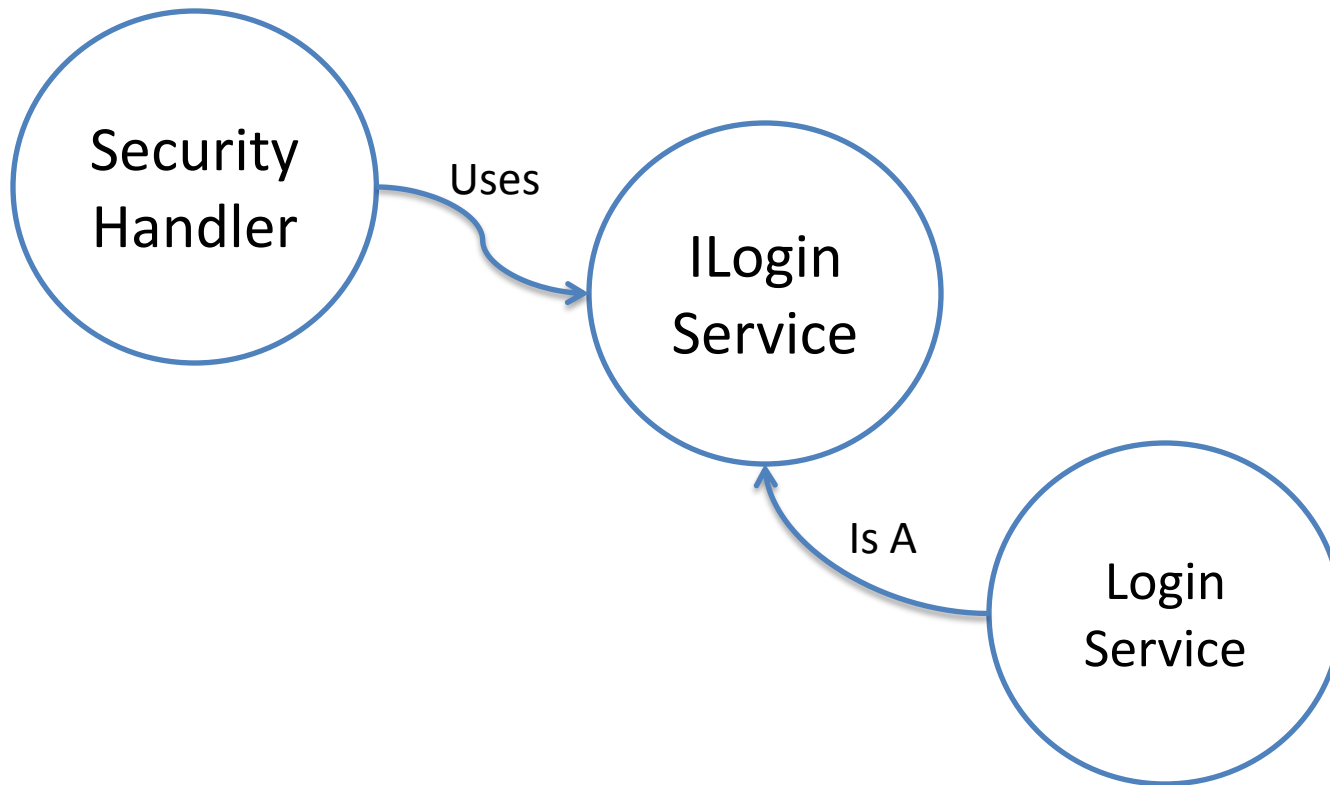
HIGH LEVEL MODULES SHOULD NOT DEPEND UPON LOW LEVEL MODULES. BOTH SHOULD DEPEND UPON ABSTRACTIONS.

DIP

X

Y

90°

DEPENDENCY INVERSION PRINCIPLE
Would You Solder A Lamp Directly To The Electrical Wiring In A Wall?

DIP

service location
& factories

DI

```csharp
public class MovieLister
{
    private readonly IMovieFinder _finder;

    public MovieLister()
    {
        _finder = ServiceLocator.Locate<IMovieFinder>();
    }

    public IEnumerable<Movie> MoviesDirectedBy(string director)
    {
        return _finder
                .FindAll()
                .Where(movie => movie.Director == director);
    }
}
```

DI

dependency injection

```csharp
public class MovieLister
{
    private readonly IMovieFinder _finder;

    public MovieLister()
    {
        _finder = new ImdbMovieFinder();
    }

    public IEnumerable<Movie> MoviesDirectedBy(string director)
    {
        return _finder
                .FindAll()
                .Where(movie => movie.Director == director);
    }
}
```

DI

```csharp
public class MovieLister
{
    public MovieLister()
    {
    }

    public IMovieFinder Finder
    {
        get;
        set;
    }

    public IEnumerable<Movie> MoviesDirectedBy(string director)
    {
        return Finder
                .FindAll()
                .Where(movie => movie.Director == director);
    }
}
```

DI

```csharp
public class MovieLister
{
    private readonly IMovieFinder _finder;

    public MovieLister(IMovieFinder finder)
    {
        _finder = finder;
    }

    public IEnumerable<Movie> MoviesDirectedBy(string director)
    {
        return _finder
                .FindAll()
                .Where(movie => movie.Director == director);
    }
}
```

DI

poor man's DI

DI

```csharp
public class MovieLister
{
    private readonly IMovieFinder _finder;

    public MovieLister()
        : this(ServiceLocator.Locate<IMovieFinder>())
    {
    }

    public MovieLister(IMovieFinder finder)
    {
        _finder = finder;
    }

    public IEnumerable<Movie> MoviesDirectedBy(string director)
    {
        return _finder
                .FindAll()
                .Where(movie => movie.Director == director);
    }
}
```

DI

simplicity

WTF?

inversion of control

IoC

declarative

IoC

container

IoC

free your mind

IoC

```csharp
public class Program
{
    public static void Main()
    {
        var lister = new MovieLister(new ImdbMovieFinder());

        lister.MoviesDirectedBy("Kubrick").Each(
            movie => Console.WriteLine(movie.ToString()));
    }
}
```

IoC

```
ass Program

c static void Main()

ojectFactory.Configure(cfg =>

    cfg.For<IMovieFinder>().Use<ImbdMovieFinder>();
);

ar lister = new MovieLister(ObjectFactory.GetInstance<IMovieFinder>());

ister.MoviesDirectedBy("Kubrick").Each(
    movie => Console.WriteLine(movie.ToString()));
```

IoC

```csharp
public class Program
{
    public static void Main()
    {
        ObjectFactory.Configure(cfg =>
        {
            cfg.For<IMovieFinder>().Use<DatabaseMovieFinder>();
            cfg.For<IDbConnection>().Use<SqlConnection>()
                .Ctor<string>().Is("connection_string");
        });

        var lister = ObjectFactory.GetInstance<MovieLister>());

        lister.MoviesDirectedBy("Kubrick").Each(
            movie => Console.WriteLine(movie.ToString())));
    }
}
```

IoC

conventions

Java   .NET

Dynamic

IoC

lifecycle management

IoC

```csharp
lic class Program

 public static void Main()
 {
     ObjectFactory.Configure(cfg =>
     {
         cfg.For<IMovieFinder>().Use<DatabaseMovieFinder>();
         cfg.For<IDbConnection>().Singleton().Use<SqlConnection>()
             .Ctor<string>().Is("connection_string");
     });

     var lister = ObjectFactory.GetInstance<MovieLister>();

     lister.MoviesDirectedBy("Kubrick").Each(
         movie => Console.WriteLine(movie.ToString()));
 }
```

IoC

aop/interception

```csharp
lic class Program

 public static void Main()
 {
     ObjectFactory.Configure(cfg =>
     {
         cfg.For<IMovieFinder>()
             .Use<DatabaseMovieFinder>()
             .InterceptWith(new LoggingInterceptor());

         cfg.For<IDbConnection>().Singleton().Use<SqlConnection>()
             .Ctor<string>().Is("connection_string");
     });

     var lister = ObjectFactory.GetInstance<MovieLister>();

     lister.MoviesDirectedBy("Kubrick").Each(
         movie => Console.WriteLine(movie.ToString()));
 }
```

IoC

simplicity

?

@jon_fuller
fullerjc@gmail.com
github.com/jonfuller

Feedback please!