DEPENDENCY INJECTION

Clean Code:
Principles, Patterns, and Practices for Better Code

DEFINITION

- Dependency Injection is a software design pattern that allows a choice of component to be made at run-time rather than compile time.

  - Wikipedia 2012

©Jeremy Clark 2014

DEFINITION

- Dependency injection is a software design pattern that allows the removal of hard-coded dependencies and makes it possible to change them, whether at run-time or compile-time.

  - Wikipedia 2013

©Jeremy Clark 2014

DEFINITION

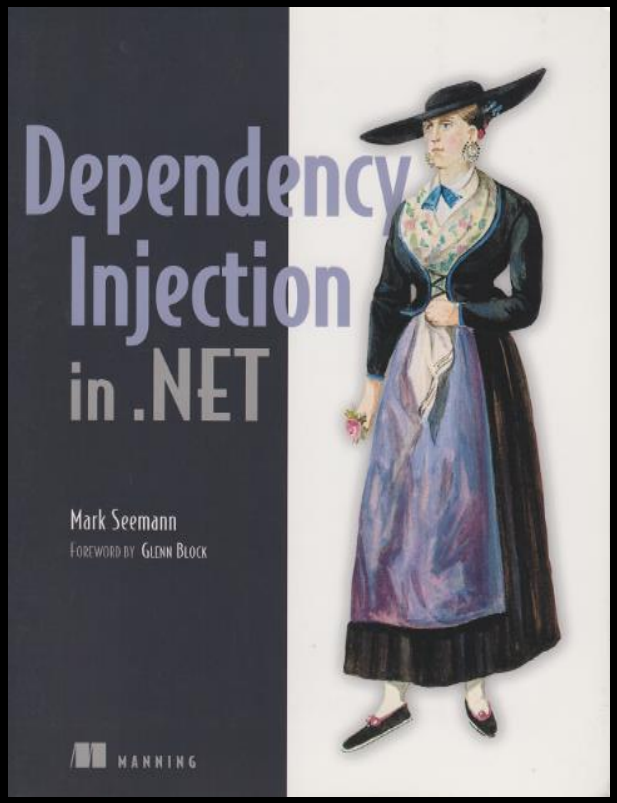
- Dependency Injection is a set of software design principles and patterns that enable us to develop loosely coupled code.

  - Mark Seemann

©Jeremy Clark 2014

DEPENDENCY INJECTION REFERENCE

- Dependency Injection in .NET
  - Mark Seemann



©Jeremy Clark 2014

PRIMARY BENEFITS

- Extensibility*
- Late Binding**
- Parallel Development
- Maintainability
- Testability*

- Adherence to S.O.L.I.D. Design Principles.**

*Topics we'll touch on today
**Covered in the available walkthrough document

©Jeremy Clark 2014

DEPENDENCY INJECTION CONCEPTS

- DI Design Patterns
  - Constructor Injection*
  - Property Injection**
  - Method Injection
  - Ambient Context
  - Service Locator
- Object Composition*
- DI Containers
  - Unity**, Castle Windsor, Ninject**, Autofac, StructureMap, Spring.NET and others

*Topics we'll touch on today
**Covered in the available walkthrough document

©Jeremy Clark 2014

APPLICATION LAYERS

View

- MainWindow

View Model

- MainWindowViewModel

Repository

- PersonServiceRepository

Service

- PersonService

©Jeremy Clark 2014

LOOK AT THE CODE

DependencyInjection - Microsoft Visual Studio

File Edit View Refactor Project Build Debug Team Data Tools Architecture Test Analyze Window Help

Windows Phone Emulator | Debug | buena park

Install Web Components | Publish:

App.xaml.cs | PersonServiceRepository.cs | ContainerPersonSer...eRepositoryTest.cs

DI.UI.App | OnStartup(StartupEventArgs e)

```
    /// <summary>
    /// Interaction logic for App.xaml
    /// </summary>
    public partial class App : Application
    {
        protected override void OnStartup(StartupEventArgs e)
        {
            base.OnStartup(e);
            ComposeObjects();
            Application.Current.MainWindow.Show();
        }

        private void ComposeObjects()
        {
            var wrappedRepository = new PersonServiceRepository();
            var repository = new CachingPersonRepository(wrappedRe
            var viewModel = new MainWindowViewModel(repository);
            Application.Current.MainWindow = new MainWindow(viewMo
        }
    }
}
```

100 %

Solution Explorer

- Solution 'DependencyInjection' (11 projects)
  - Application
    - DI.UI
      - Properties
      - References
      - App.config
      - App.xaml
        - App.xaml.cs
      - Converters.cs
      - MainWindow.xaml
    - DI.UI.Container
      - Properties
      - References
      - App.config
      - App.xaml
        - App.xaml.cs
      - Converters.cs
      - MainWindow.xaml
    - **DI.UI.LateBinding**
  - Repository
    - DI.CachingRepository
    - DI.Repository.CSV
    - DI.Repository.Service
      - Properties
      - References
      - Service References
      - app.config
      - PersonServiceRepository.cs
  - Service
    - People.Service

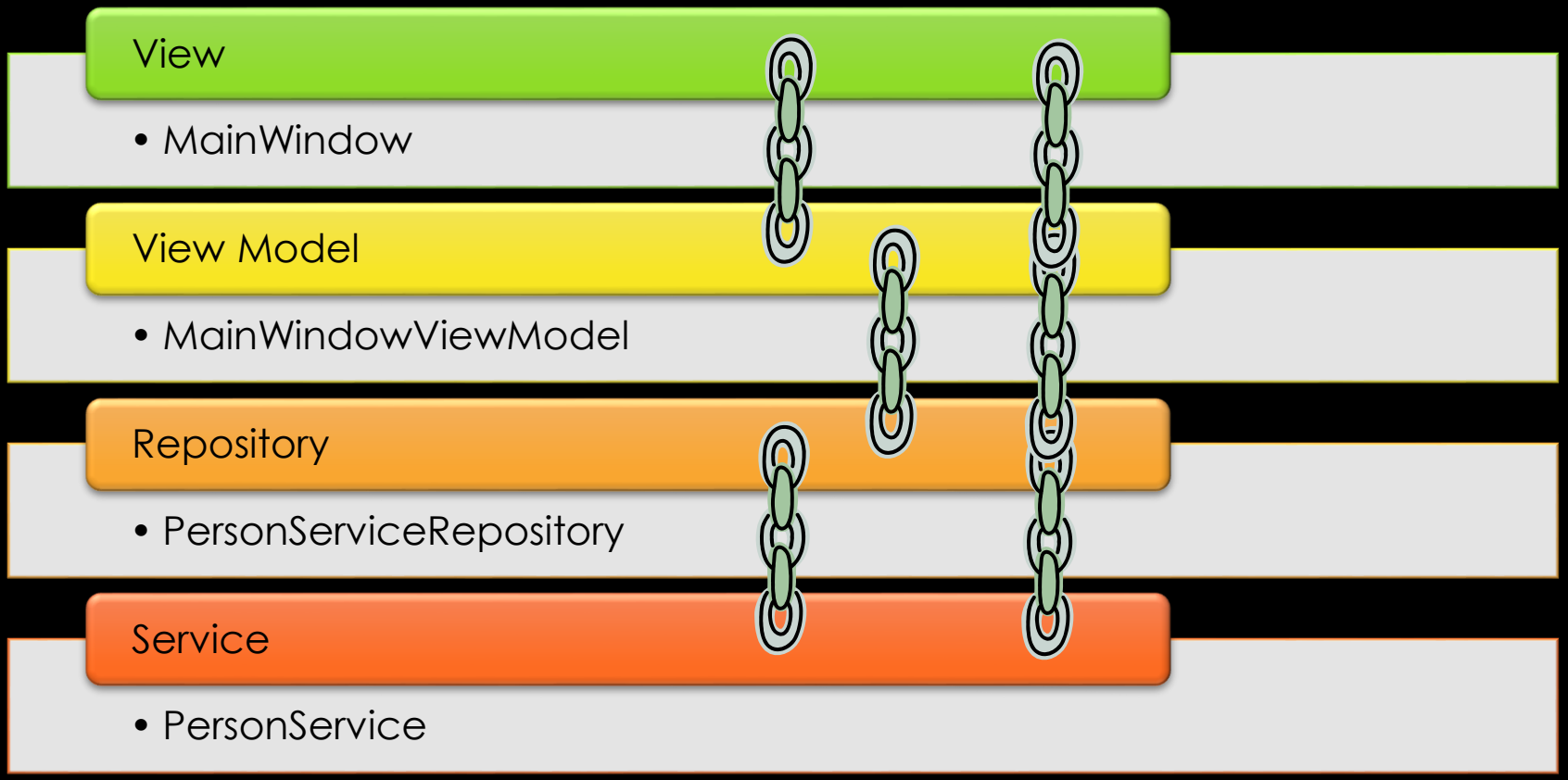
Properties | Solution Explorer | Team Explorer

Error List | Output | Find Results 1 | Test Results | Code Metrics Results

Ready | Ln 1 | Col 1 | Ch 1 | INS

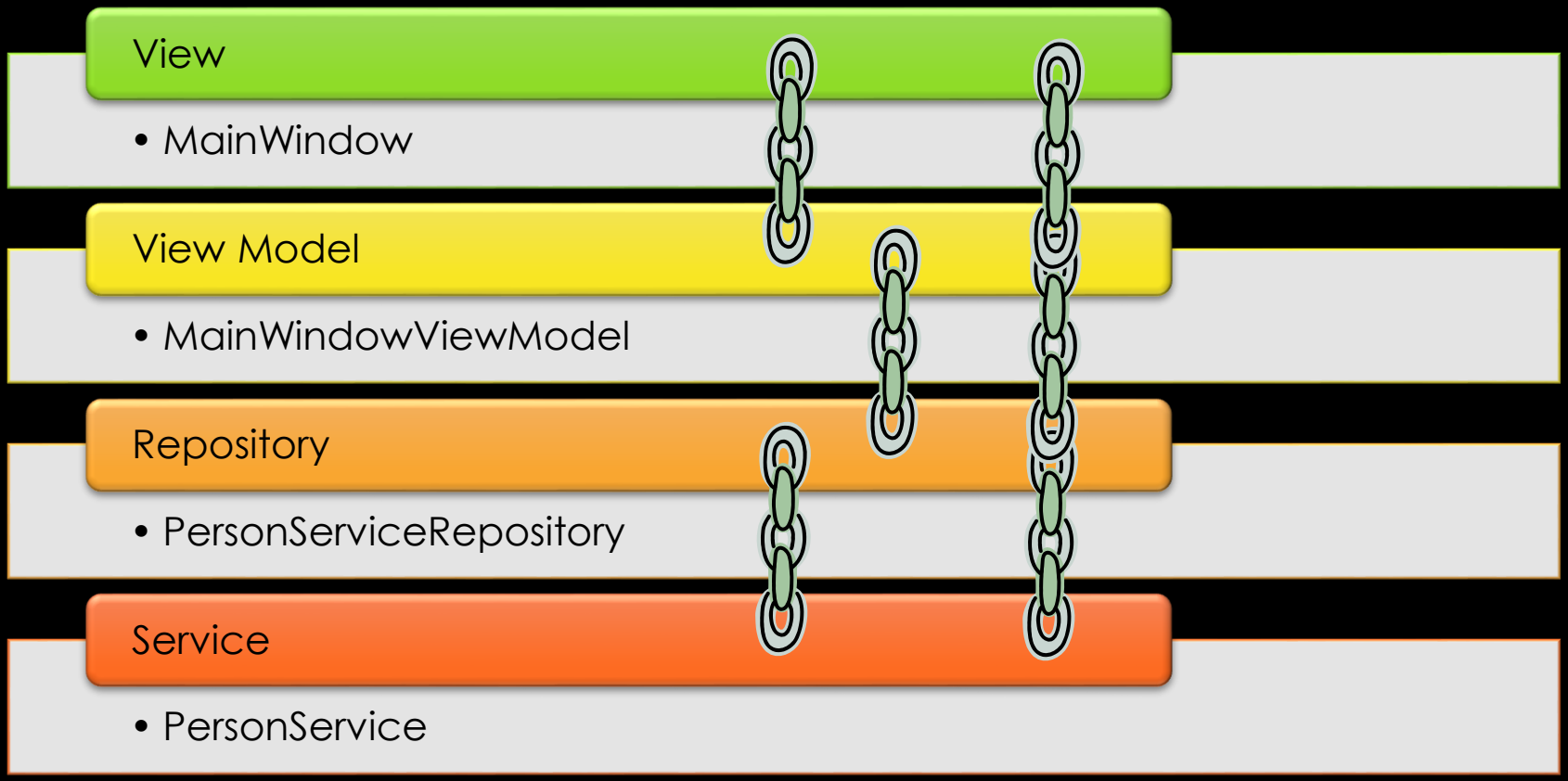
©Jeremy Clark 2014

TIGHT COUPLING



©Jeremy Clark 2014

LOOSE(R) COUPLING



©Jeremy Clark 2014

DEPENDENCY INJECTION CONCEPTS

- DI Design Patterns
  - Constructor Injection*
  - Property Injection**
  - Method Injection
  - Ambient Context
  - Service Locator
- Object Composition*
- DI Containers
  - Unity**, Castle Windsor, Ninject**, Autofac, StructureMap, Spring.NET and others

*Topics we'll touch on today
**Covered in the available walkthrough document

©Jeremy Clark 2014

PRIMARY BENEFITS

- Extensibility*
- Late Binding**
- Parallel Development
- Maintainability
- Testability*

- Adherence to S.O.L.I.D. Design Principles.**

*Topics we'll touch on today
**Covered in the available walkthrough document

©Jeremy Clark 2014

DEPENDENCY INJECTION

Next Up: Lab