Taking Advantage of Loose Coupling

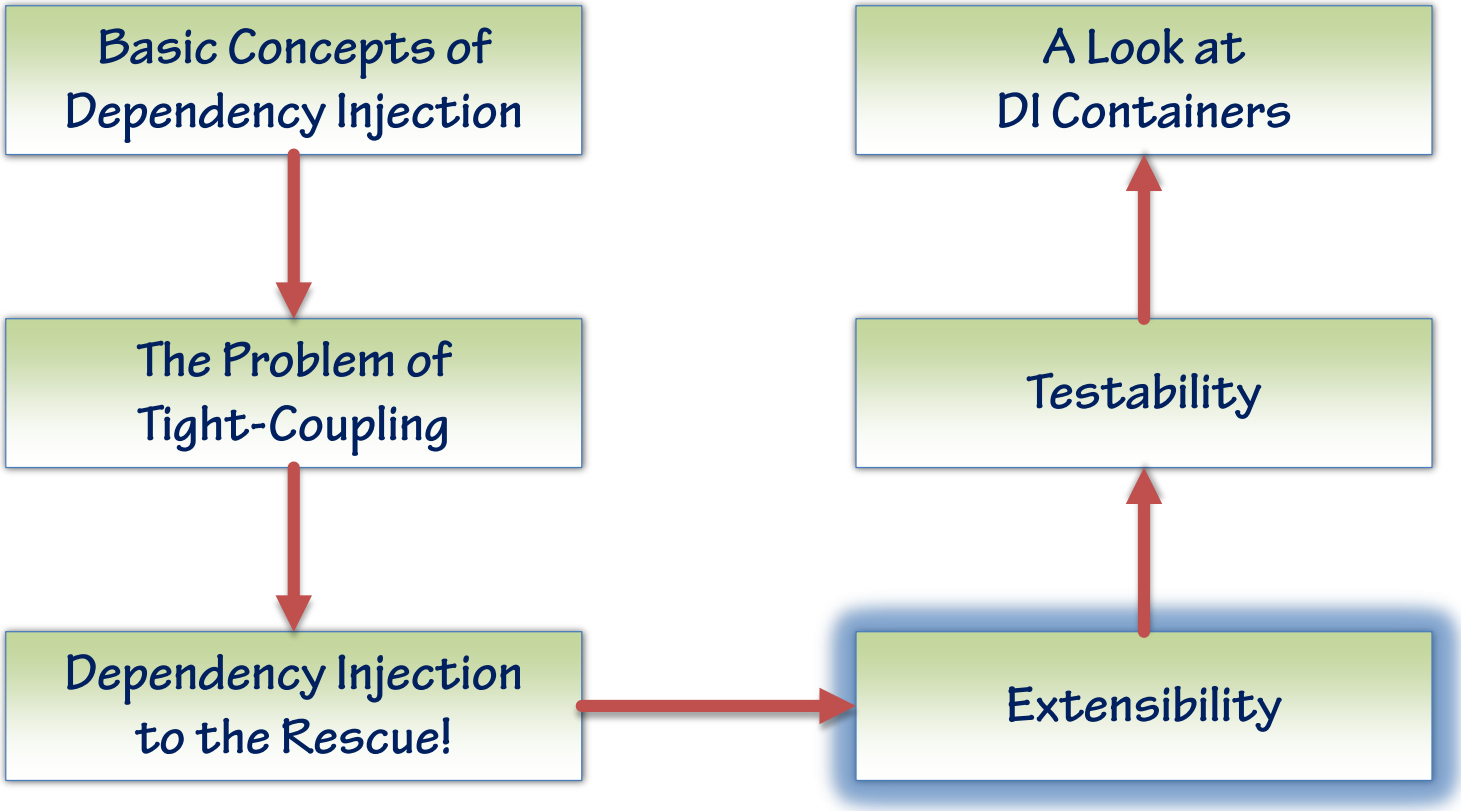
Easy Extensibility with Dependency Injection

Jeremy Clark
www.jeremybytes.com
jeremy@jeremybytes.com

pluralsight
hardcore developer training

Goal

- Get Comfortable with Dependency Injection



Scenario 1: Different Repositories

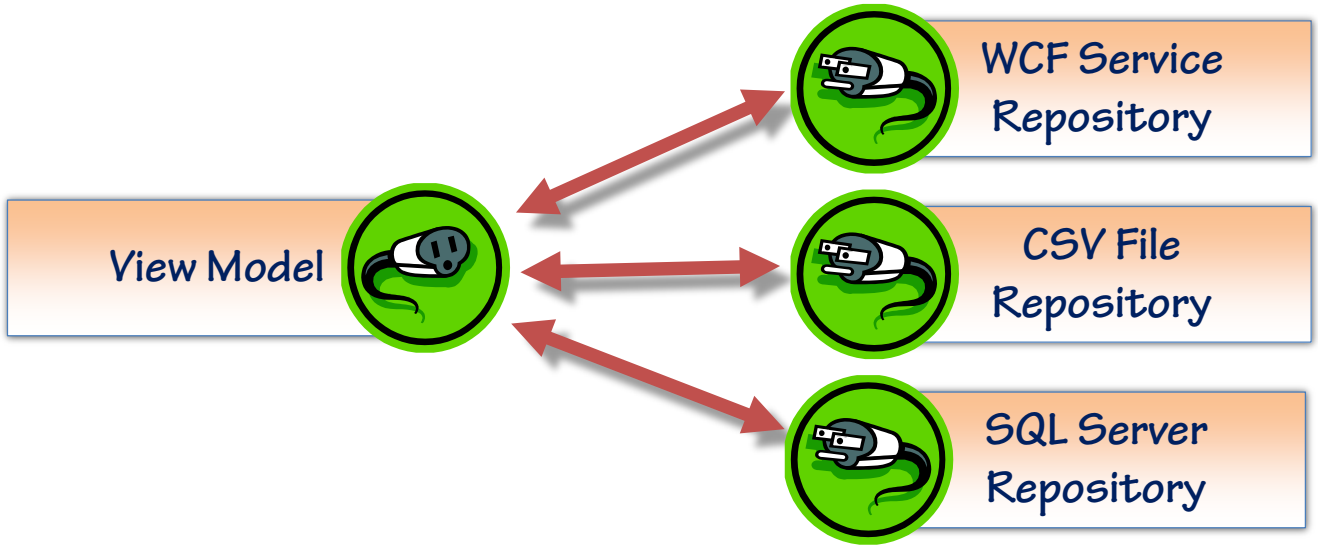
```
public class PeopleViewerViewModel : INotifyPropertyChanged
{
    protected IPersonRepository Repository;

    public PeopleViewerViewModel(IPersonRepository repository)
    {
        Repository = repository;
    }
    ...
}
```

```
private static void ComposeObjects()
{
  var repository = new ServiceRepository();
  var viewModel = new PeopleViewerViewModel(repository);
  Application.Current.MainWindow = new PeopleViewerWindow(viewModel);
}
```

- The View Model accepts any class implementing IPersonRepository
- We choose a Repository in our Composition Root

Pluggable Repositories

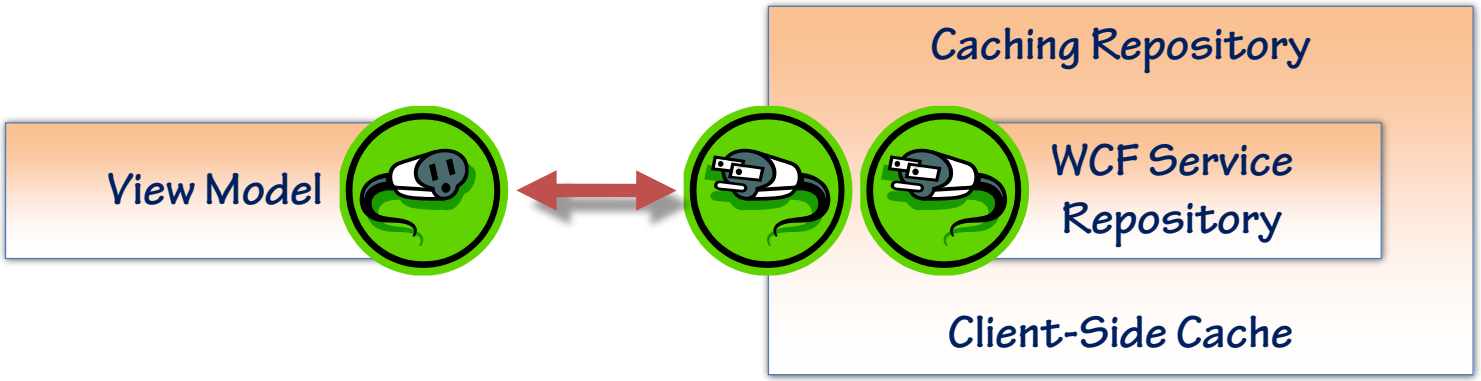


Scenario 2: Client-Side Caching

```
private static void ComposeObjects()
{
  var repository = new ServiceRepository();
  var viewModel = new PeopleViewerViewModel(repository);
  Application.Current.MainWindow = new PeopleViewerWindow(viewModel);
}
```

- **Add a Caching Repository**
- **Put the objects together a little differently in the Composition Root**

Creating a Caching Repository



Summary

- **Additional Repositories**
  - CSV Repository
  - SQL Repository
- **Client-Side Caching**
  - Caching Repository

- **Next Up: Unit Testing**
  Easy Testing with Dependency Injection