

# Building the Basic UI Layer

---



**Thomas Claudius Huber**

MICROSOFT MVP (WINDOWS DEVELOPMENT)

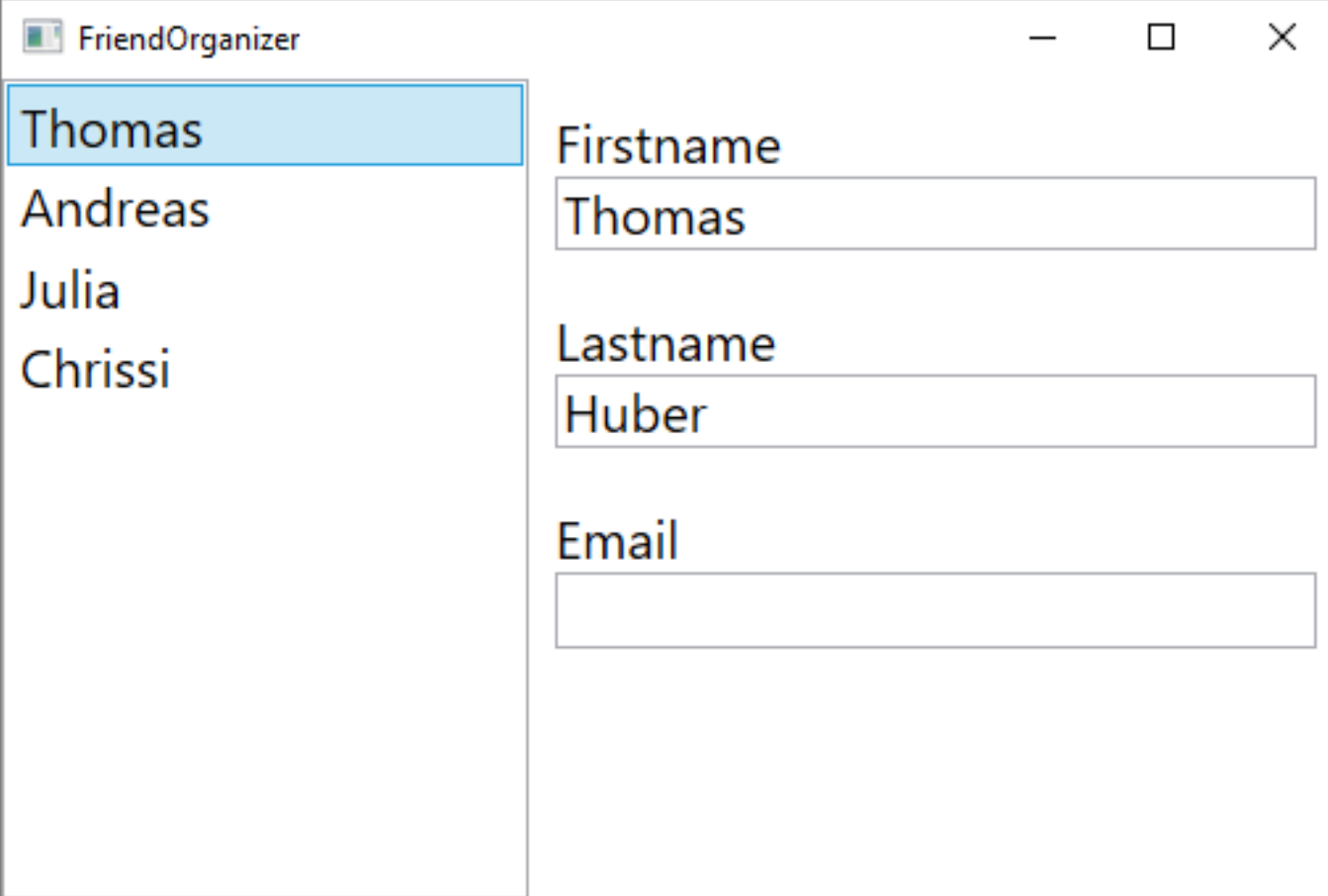
@thomasclaudiush [www.thomasclaudiushuber.com](http://www.thomasclaudiushuber.com)



Should you start  
without requirements?



# The Required User Interface



The screenshot shows a Windows application window titled "FriendOrganizer". On the left side, there is a list of names: "Thomas", "Andreas", "Julia", and "Chrissi". The name "Thomas" is selected and highlighted with a blue background. To the right of this list, there are three input fields. The first is labeled "Firstname" and contains the text "Thomas". The second is labeled "Lastname" and contains the text "Huber". The third is labeled "Email" and is currently empty.

Name	Firstname	Lastname	Email
Thomas	Thomas	Huber	
Andreas			
Julia			
Chrissi			

# Module Outline



**Plan the UI layer**

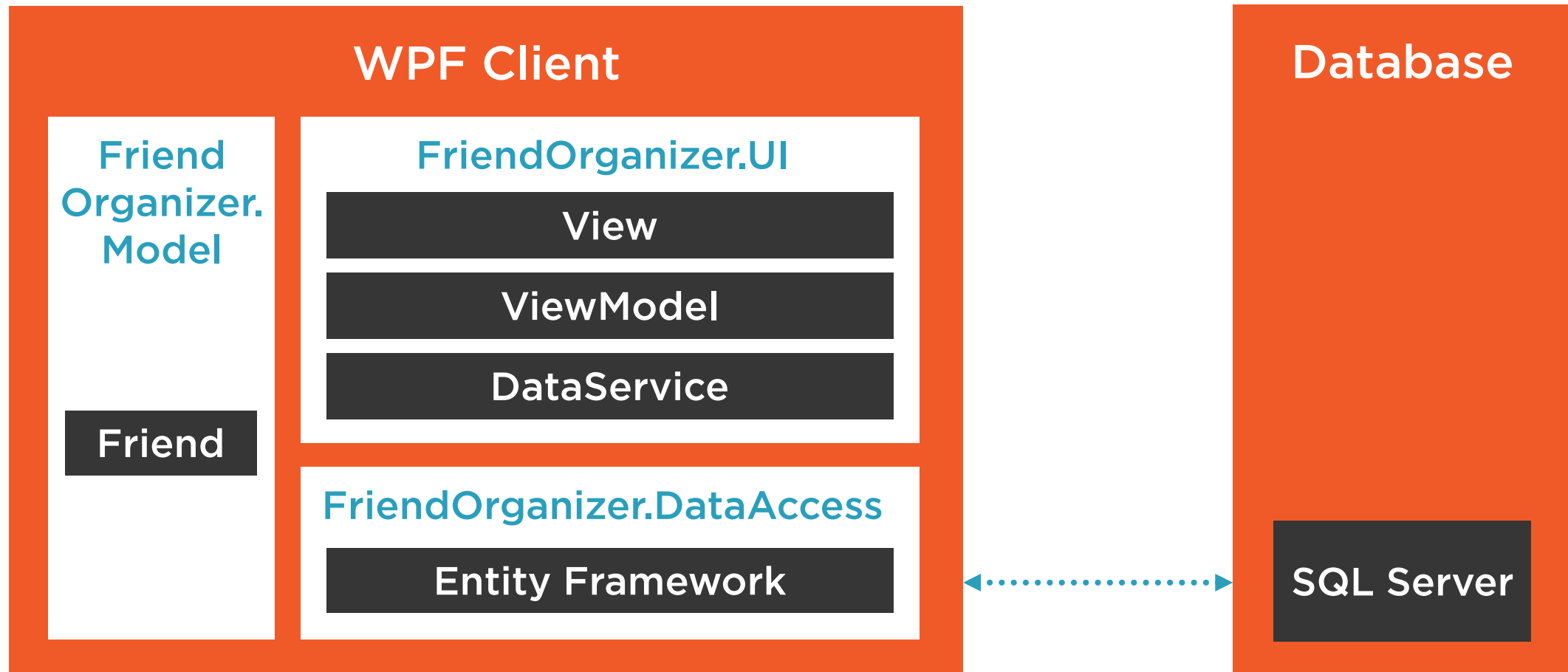
**Know the MVVM pattern**

**Create the different parts**

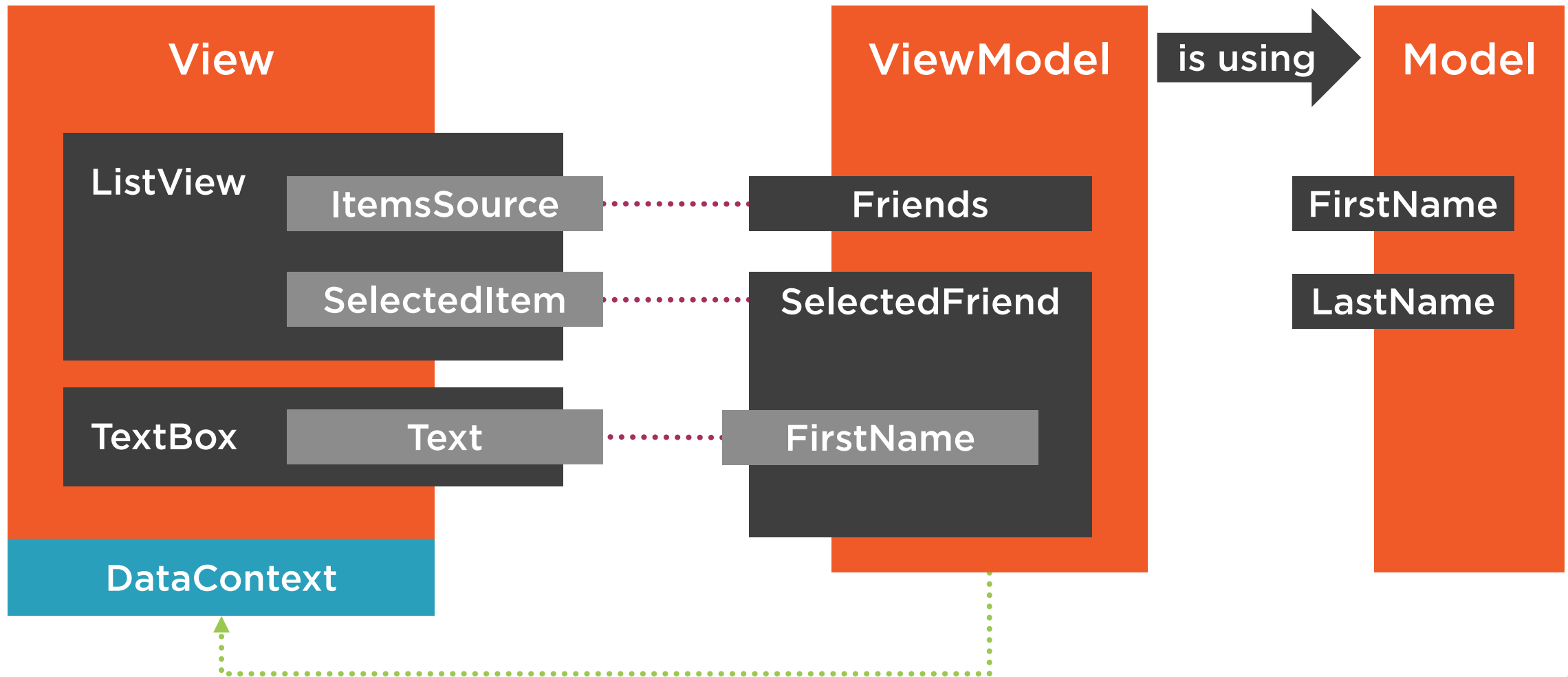
- Data Service
- ViewModel
- View

**Use Dependency Injection**

# Plan the User Interface Layer



# The MVVM Pattern



# Demo



Create the MainViewModel

Implement INotifyPropertyChanged

Set the DataContext

Define the User Interface



# Add Autofac for Dependency Injection

```
var mainWindow = new MainWindow(  
    new MainViewModel(  
        new FriendDataService())));
```





# Add Autofac for Dependency Injection

```
var mainWindow = container.Resolve<MainWindow>();
```



# Summary



## The MVVM-pattern

**You built a UI-layer that consists of**

- View
- ViewModel
- Data Service

**Autofac is used for Dependency Injection**

