

12.12.19

MVVM Pattern

Nico Vogel und Lukas Sopora

Agenda

1. Welches Problem geht MVVM an?
2. Anwendungsbereiche
3. Was ist MVVM?
 - 3.1. Bestandteile
 - 3.2. Zusammenspiel
4. Vergleich MVC und MVVM
5. Demo
6. Kritische Würdigung

1. Welches Problem geht MVVM an?

Anwendungsbereich: **Entwicklung UI**

Angegangene Probleme:

- Starke Abhängigkeit zwischen UI und Logik
- Redesign problematisch
- Cross Platform
- schwer zu testen

2. Anwendungsbereiche

MVVM wird eingesetzt von:

- C# WPF (*Ursprung*)
- Delphi
- Silverlight
- AngularJS (*nicht Angular...*)

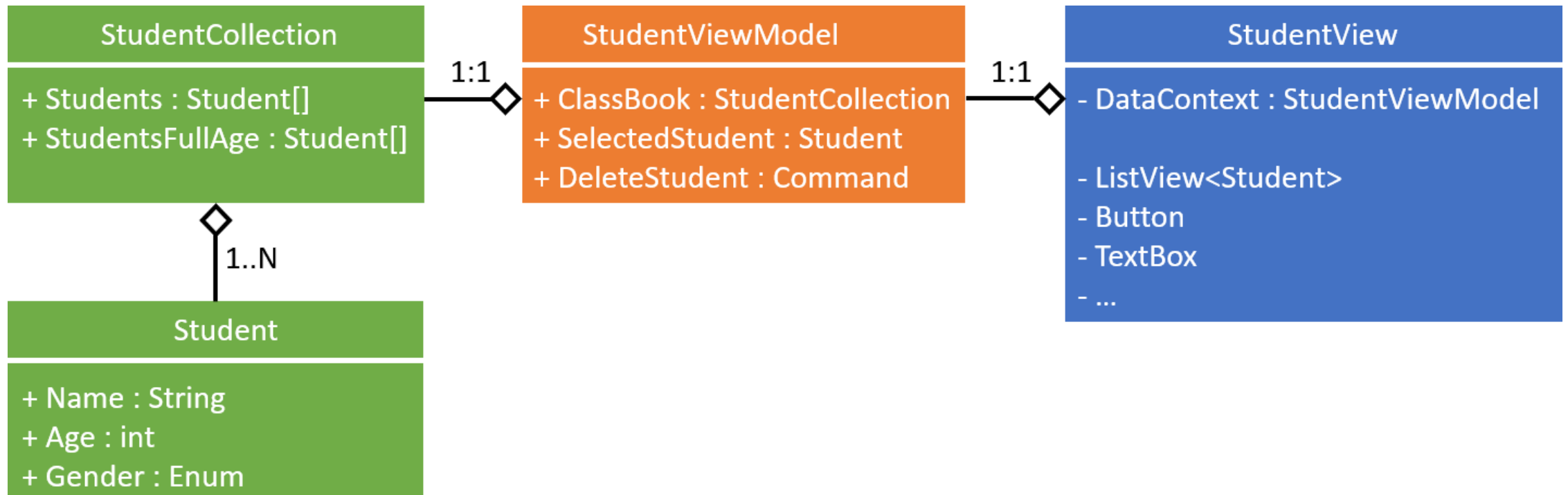
3.1. Was ist MVVM - Bestandteile?



- Informationsaustausch
 - Binding
 - Command
 - Events

3.1 Was ist MVVM? - Anwendungsfall

- Verwaltung von Studentendaten im Kurs
 - "Klassenbuch"



3.1. Was ist MVVM? - Model

- POJO
- Nur Daten und Daten Logik (z.B. Validierung)

```
public class StudentCollection
{
    public IList<Student> Students { get; }
    public IList<Student> StudentsFullAge { get {
        return this.Students.where(x => x.Age >= 18).toList();
    }}
}
public class Student
{
    public String Name { get; set; }
    public int Age { get; set; }
    public GenderType Gender { get; set; }
}
```

3.1. Was ist MVVM? - ViewModel

- Schnittstelle zwischen UI und Logik
- Zusammenführung von Daten und Funktionen

```
public class StudentViewModel
{
    public StudentCollection ClassBook { get; }
    public Student SelectedStudent { get; set; }
    public ICommand DeleteCommand { get; private set; }

    public StudentViewModel()
    {
        this.ClassBook = StudentTestDataUtility.GetStudentTestData();
    }
}
```


3.1. Was ist MVVM? - View

- Keine Programmlogik, lediglich Rendering

```
<ListView>
  <ListViewItem>
    <DockPanel>
      <TextBlock Text="Andi Theke"/>
      <TextBlock Text="19"/>
      <TextBlock Text="Male"/>
    </DockPanel>
  </ListViewItem>
</ListView>
```

3.1. Was ist MVVM? - Binding

- Informationsaustausch zwischen View und ViewModel
- View "Sucht" sich die notwendigen Informationen aus dem ViewModel

```
<Window.DataContext>
    <viewModel:StudentViewModel/>
</Window.DataContext>

<ListView ItemsSource="{Binding ClassBook.Students}">
    <ListView.ItemTemplate> <DataTemplate>
        <DockPanel>
            <TextBlock Text="{Binding Name}"/>
            <TextBlock Text="{Binding Age}"/>
            <TextBlock Text="{Binding Gender}"/>
        </DockPanel>
    </DataTemplate> </ListView.ItemTemplate>
</ListView>
```

3.1. Was ist MVVM? - Binding

OneWay Binding

View \leftarrow ViewModel

oder

View \rightarrow ViewModel

(OneWayToSource)

TwoWay Binding

View \Leftrightarrow ViewModel

3.1. Was ist MVVM? - Events, Commands

Events

- Aktion triggert Methode in Code Behind der View
 - Bsp.: Clicked, OnHover, LostFocus

```
<Button  
    Click="btnClicked"/>
```

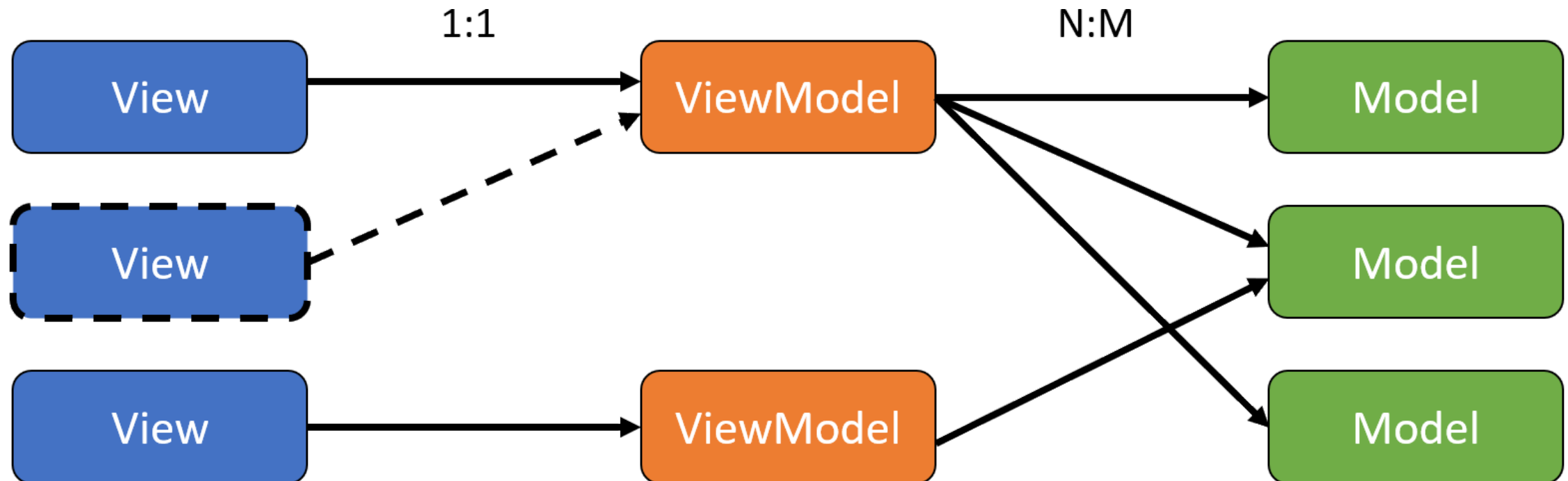
Commands

- Binding Aktion an Methode im ViewModel
 - meist Button Events

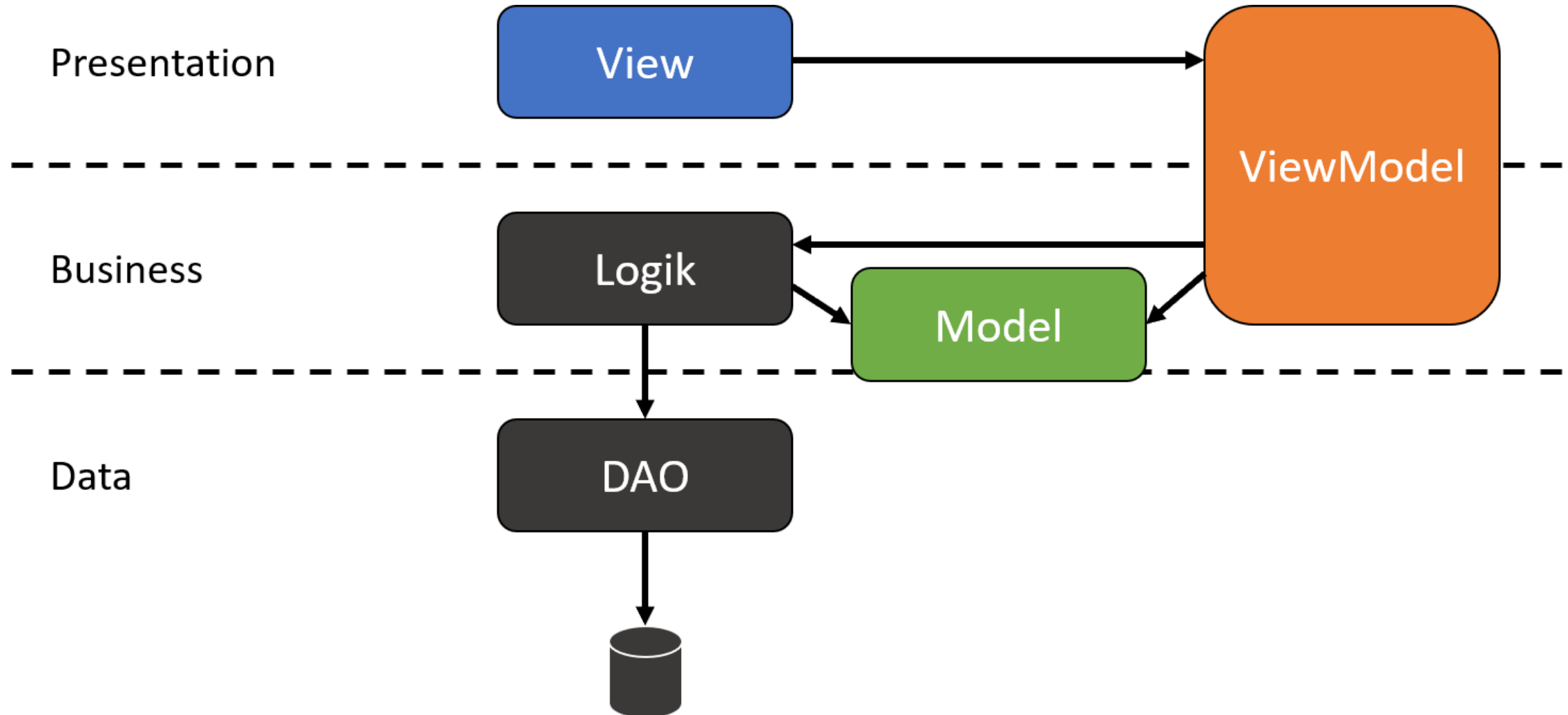
```
<Button  
    Command="{Binding ClickCommand}"/>
```

3.2. Was ist MVVM? - Zusammenspiel

Interaktion der Komponenten:

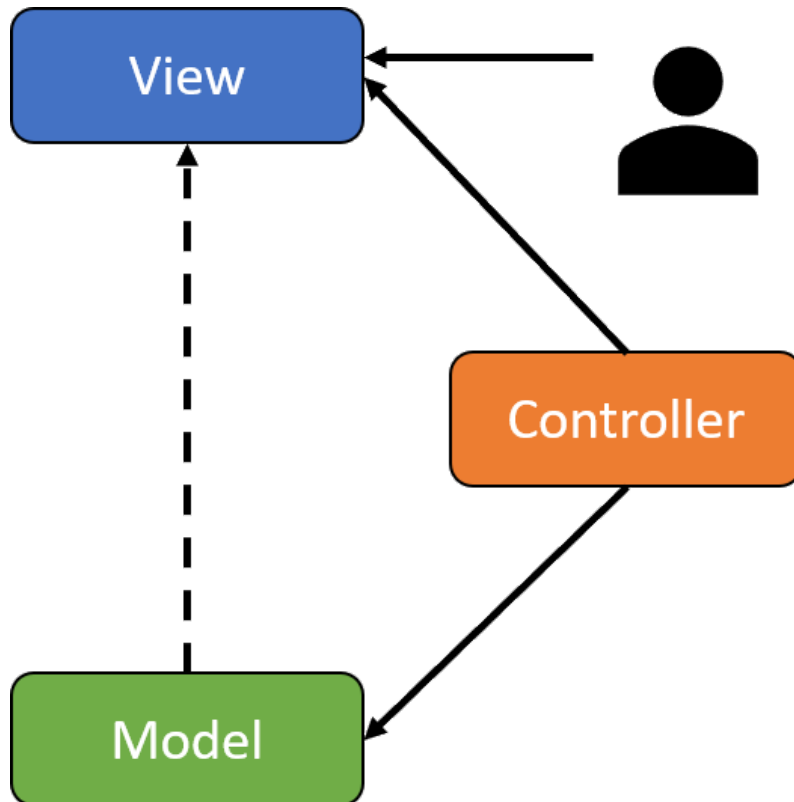


3.2 Was ist MVVM? Einordnung Application Layer

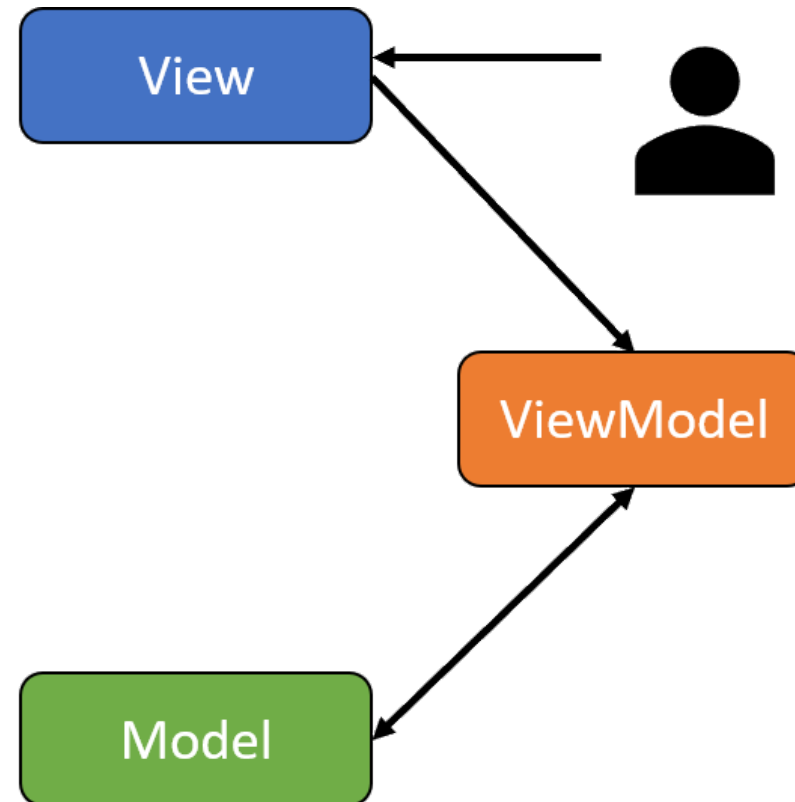


4. Vergleich MVC und MVVM

Model View Controller

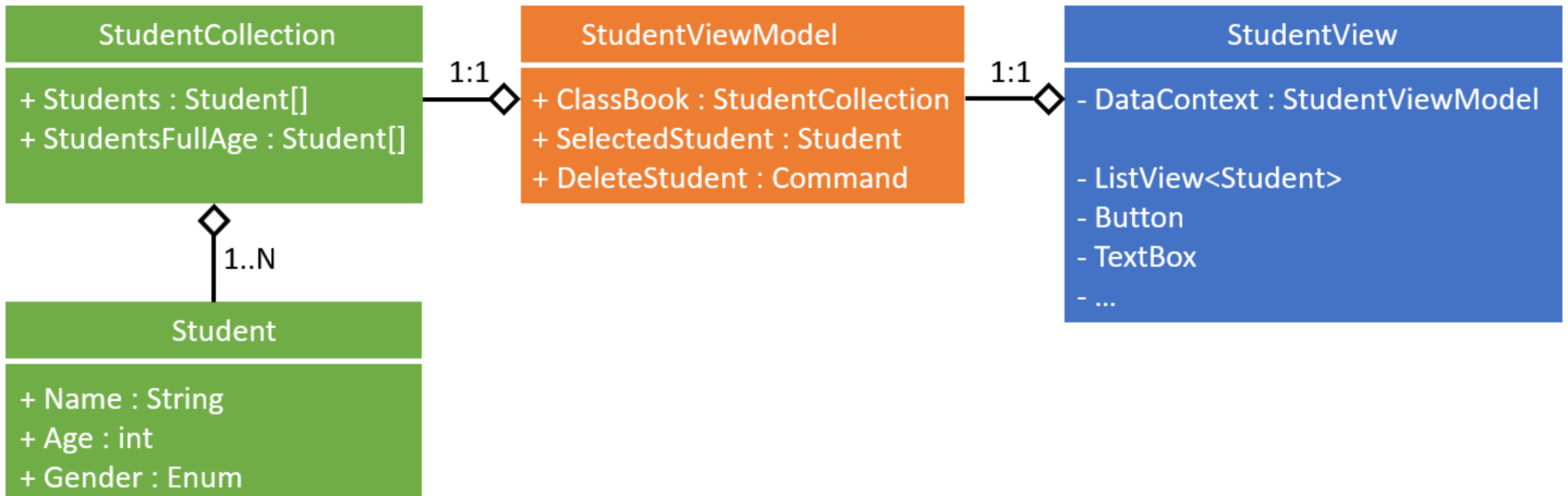


Mode View ViewModel



5. Demo

Aufbau der Demo Applikation:



6. Kritische Würdigung

Pro

- flexibel
- DataBinding
- steile Lernkurve (OOP, Testing)
- gut testbar

Con

- viel code für wenig resultat
- Binding eventuell laufzeitkritisch
- XAML Notation umfangreich
- OOP, Binding und Testing vorausgesetzt

**Danke für Eure
Aufmerksamkeit**