

# Wpf Tadeot-Administration

Zur Administration der Besucher-Anmeldung während des Tags und Abends der Offenen Tür an der HTL Leonding soll ein Wpf-Client entwickelt werden.

- Ein Prototyp für das Registrierungsformular sowie die Statistik-Auswertung kann auf <https://vm64.htl-leonding.ac.at> eingesehen werden.


## Lehrziele

- Wpf XAML Layouts, Styles
- Wpf **DataGrid**
- Wpf Mvvm: **BaseViewModel**, **NotifyPropertyChanged**, **RelayCommand**, **ObservableCollection**, **WindowController**
- Wpf Datenbankankbindung:
  - **UnitOfWork** Integration
  - **Bogus** Testdatengenerator
  - **ExecuteRawSql**, EF Change Tracking
  - **Dependency Injection**, Anwendung in WPF
  - **Unittest**, Testen des ViewModels

## Aufgabenstellung

Die Hauptseite der Anwendung listet alle **Visitors** in einem **DataGrid** und zeigt in der Überschrift die Anzahl der registrierten Besucher/innen an:

Tadeot Admin Panel



Tadeot Administration

Besucherregistrierungen: 300

Id	Zeit	Begl.	Schultyp	Schulstufe	Besuchsgrund	Geschlecht	Bezirk	Int. HIF	Int. HITM	Int. HEL	Int. HBG	Int. FEL	Kommentar
1	16.10 09:17	0	HAK	9	Anderes	w	Sankt Johann im Pongau	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	sit unde recusandae facilis liber
2	16.10 01:07	0	HTL	8	Messe Wels	m	Vöcklabruck	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	neque vel voluptas et accusam
3	16.10 04:39	0	HTL	9	Plakat	w	Bregenz	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	neque quibusdam autem ut on
4	16.10 09:50	0	HTL	7	Werbung Straßenbahn	w	Oberwart	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	id possimus excepturi hic volup
5	16.10 10:30	1	HBLA	9	Freunde	m	Zell am See	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	sed iste ut laborum et
6	16.10 01:24	1	HAK	7	Schule	w	Horn	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	quaerat itaque sint autem esse
7	16.10 10:21	3	Lehre	9	Plakat	w	Salzburg-Umgebung	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	veniam tempora qui rerum plac
8	16.10 11:26	2	Fachschule	9	Schule	m	Spittal an der Drau	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	commodi consequuntur nam te
9	16.10 08:06	2	HAK	9	Guter Ruf der Schule	m	Innsbruck-Land	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	quia repellendus veritatis vel qu
10	16.10 06:35	4	HTL	8	Freunde	m	Hallein	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	quam laborum velit dolorum se
11	16.10 06:15	2	HTL	8	Schule	m	Schärding	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	nisi consequatur dicta architect



Download CSV-Datei

Alle Registrierungen Löschen

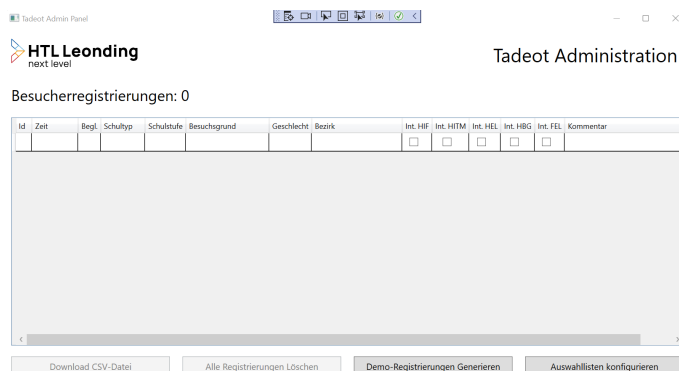
Demo-Registrierungen Generieren

Auswahllisten konfigurieren

Die Buttons sind mit folgenden Funktionen belegt:

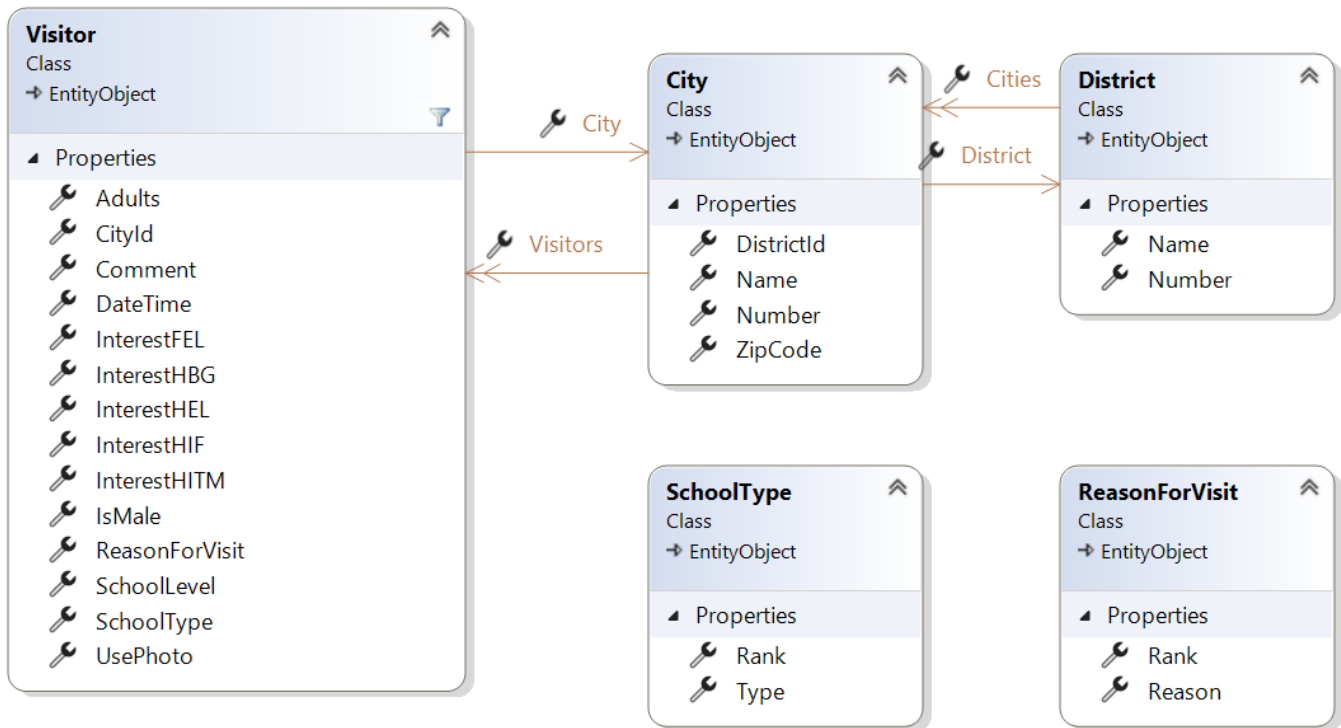
Button	Funktion	Ausführbar
Download CSV	Mithilfe eines <b>SaveDialog</b> werden alle <b>Visitor</b> -Daten in eine CSV-Datei exportiert.	Nur wenn Besucheranzahl > 0
Alle Registrierungen löschen	<p>Für den Echtbetrieb möchte man den Zählerstand auf 0 zurücksetzen und somit alle Testdaten löschen.</p> <p><b>Achtung:</b> Es reicht nicht, alle Daten aus der Tabelle zu löschen, sondern die Tabelle muss mit einem <b>rawSQL</b> Statement mithilfe des SQL <b>truncate</b> Befehls zurückgesetzt werden, damit auch die <i>auto-generated Ids</i> wieder bei 1 beginnen! Achten Sie dabei darauf, dass der DB-Context <b>ChangeTracker</b> zurückgesetzt wird.</p> <p>Vor dem Löschen soll eine Rückfrage erfolgen, ob man sich sicher ist, dass alle Daten gelöscht werden sollen:</p> <p>Registrierungen löschen </p> <p> Wollen Sie wirklich alle Registrierungen löschen?</p> <p><input type="button" value="Ja"/> <input type="button" value="Nein"/></p>	Nur wenn Besucherzahl > 0
Demo-Registrierungen generieren	<p>Mithilfe des <b>Bogus</b>-Testdatengenerators sollen <b>300</b> simulierte Anmeldungen erzeugt und in der DB gespeichert werden. Damit wird das Testen des Datenexports und der Statistik-Slideshow ermöglicht.</p> <p><b>Optional</b> können Sie die Anzahl der generierten Anmeldungen über eine Number-TextBox konfigurierbar machen.</p>	Nur wenn keine Besucher/innen registriert sind.
Auswahllisten konfigurieren	Öffnet das Fenster <b>RegistrationConfigWindow</b> modal ( <b>ShowDialog()</b> ).	immer

Hier die Anzeige, wenn keine Registrierungen vorhanden sind:



## Datenmodell

Die Vorlage enthält bereits das vollständige Datenmodell für die Anwendung.



## Import

Die in der Vorlage bereits enthaltene **ImportConsoleApp** ist wie folgt zu implementieren:

- Löschen und Neuerstellen der Datenbank
- CSV-Import folgender Daten:
  - **districts.csv** enthält alle Bezirke, über die später Statistik-Auswertungen der Besucherzahlen erfolgen sollen.
  - **cities.csv** hingegen enthält alle Ortschaften und Städte, die im Registrierungsformular zur Auswahl angeboten werden. Jede **City** ist dabei mit seinem **District** über die erste CSV-Spalte verknüpft: Die ersten 3 Zeichen der Spalte **Number** entsprechen dem **Number**-Attribut eines **Districts**.
  - **reasonsforvisit.csv** und **schooltypes.csv** enthalten die Vorbelegungen für die DB-Initialisierung, die später im Wpf **RegistrationConfigWindow** noch adaptiert werden können.
- Bogus-Datengenerator
  - Während des Imports sollen exakt **400** simulierte Registrierungen erzeugt und in die Datenbank abgespeichert werden.
  - Da auch der Wpf-Client später diese Funktion benötigt, soll das Generieren und Speichern der Testdaten im **VisitorRepository** zentral implementiert werden.
  - Verwendet wird dazu das nuget-Paket <https://www.nuget.org/packages/Bogus>

## WpfTadeotAdmin - **MainWindow**

- **UnitOfWork** Integration  
Obwohl eine korrekte Datenbank-Anbindung aus Sicherheitsgründen über ein **RESTful Service** gehen müsste, verwenden wir zu Übungszwecken eine **UnitOfWork**-Instanz direkt in den Wpf-Viewmodel-Klassen.
- Das **ViewModel** ist von Windows getrennt. Daher Implementieren Sie die ViewModels in einer eigenen Assembly.

- Das Umschalten der Dialoge, der Win32 SaveFileDialog, ... wir in eine Klasse `WindowNavigator` ausgelagert. Im ViewModel ist nur ein Interface auf den WindowNavigator bekannt.

## WpfTadeotAdmin - `RegistrationConfigWindow`

### Konfiguration von Auswahllisten für das Anmeldeformular

Wie auf dem Anmeldungs-Prototyp ersichtlich, gibt es bei manchen Daten verschiedene Auswahlmöglichkeiten, die sich von Jahr zu Jahr ändern könnten. Daher sollen die folgenden beiden Auswahllisten im Admin-Client konfigurierbar sein:

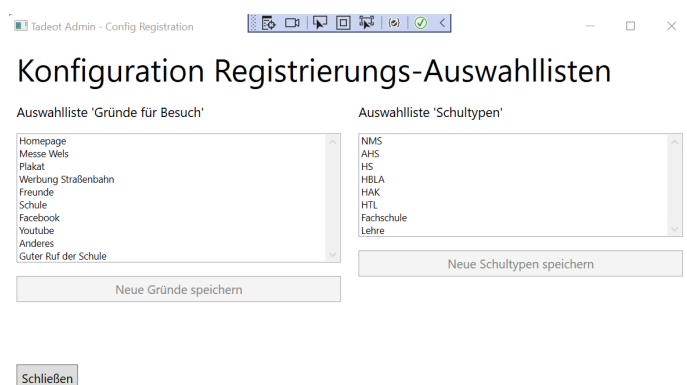
- Auswahlliste **Gründe für Besuch**
- Auswahlliste **Schultypen**

Bei beiden Auswahllisten werden - nur im Fall einer Änderung - jeweils alle Einträge in die Datenbank gespeichert, in der wie im UI abgebildeten Reihenfolge.

#### Hinweis:

- Da die jeweils gewählten `Visitor`-Attribute `ReasonForVisit` und `SchoolType` als `string` im Datenmodell abgebildet sind, hat eine Änderung einer Auswahlliste keinerlei Auswirkung auf bestehende Registrierungsdaten!
- Man kann also auch während der laufenden Veranstaltung noch Änderungen vornehmen, falls eine andere Reihung oder andere Einträge im Registrierungsformular gewünscht sind.

### GUI



Für die Darstellung der Auswahllisten wurden in diesem UI mehrzeilige TextBoxen verwendet, die alle Strings aus der DB mit `newline` zu einem einzigen String verkettet anzeigen. Sobald eine Änderung in der TextBox vorgenommen wird, wird der jeweilige Speichern-Button aktiviert.

Folgende Style-Properties sind dabei hilfreich:

```
<Window.Resources>
  <Style TargetType="TextBox" x:Key="TextList">
    <Setter Property="TextWrapping" Value="Wrap" />
    <Setter Property="AcceptsReturn" Value="True" />
    <Setter Property="VerticalScrollBarVisibility" Value="Visible"/>
  </Style>
</Window.Resources>
```

## Unit-Test

Die Unittests sollen die ViewModels überprüfen.

- Schreiben Sie die Unittests so, dass keine Datenbank benötigt wird.
- Ein Beispiel ist im Template enthalten.