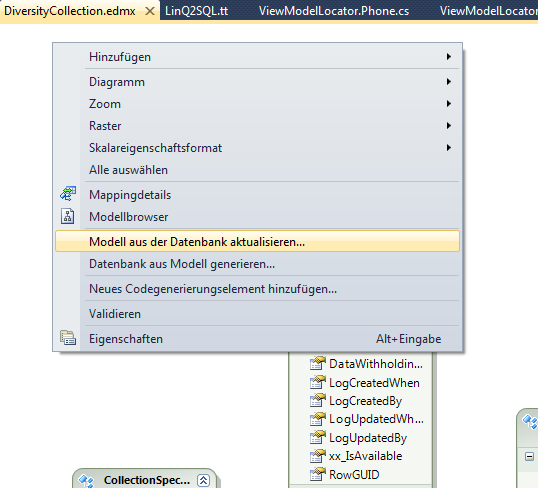
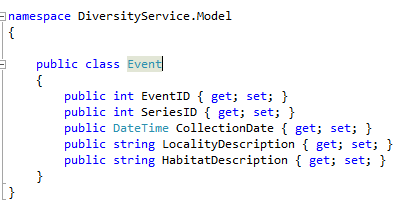
Einfügen einer neuen Datenklasse in DiversityPhone

# Import der Zugehörigen Tabelle

**Projekt DiversityService  
Im Entity Framework Designer**

# Anlegen der (reduzierten) Modellklasse

**Im DiversityService.Model Projekt**

# Einfügen der nötigen Zugriffsmethoden

**Im DiversityService Projekt**  
In das IDiversityService Interface  
In die DiversityService Implementierung  
🡪Bilden von der Enitity Framework Modelklasse auf das reduzierte Datenmodel ab.

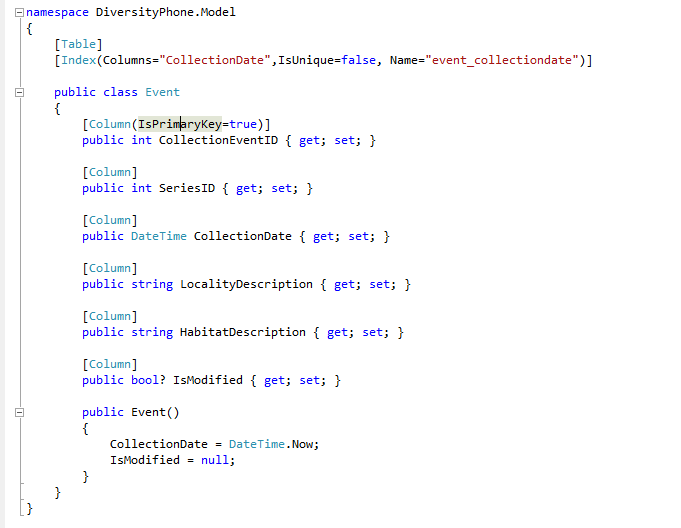
[OperationContract]

IEnumerable<Term> GetStandardVocabulary();

# Anlegen einer annotierten identischen Modellklasse

**Im DiversityPhone Projekt,   
Ordner Model**

Eventuell nur auf dem Mobilgerät verwendete Properties einfügen (IsModified z.B.)



# Anlegen einer Tabelle im DatenKontext

**Im DiversityPhone Projekt,   
DiversityDataContext**

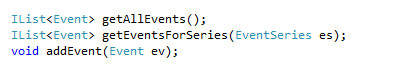
public Table<Event> Events;

# Einfügen der nötigen Zugriffsmethoden

**Im DiversityPhone Projekt,   
Ordner Services**

In das IOfflineStorage bzw. IOfflineFieldData Interface  
In die OfflineStorage Implementierung

Standard: Klassenweise Gruppierung



# ViewModels der gewünschten Seiten/Elemente anlegen

**Im DiversityPhone Projekt,   
Im Ordner ViewModels**

Es werden insgesamt 3 Klassen für jede Felddatenklasse des Typs EventSeries, Event, Specimen und IdentificationUnit angelegt, sowie 2 Klassen für alle weiteren editierbaren Klassen.

Allen Klassen ist gemein:

* Ableitung von ReactiveObject
* \_IMessageBus messenger zur Navigation
* IList<Disposable> \_subscriptions zur Steuerung der GarbageCollection

Grundklasse [Felddatenklasse]VM z.B.

public class EventVM : ReactiveObject

Enthält:

* Property Model mit Verweis auf das Objekt der zugrunde liegenden Klasse
* Property Description mit anzuzeigendem Text
* Property Icon für das Icon
* ReactiveCommands. Edit und Select

Klasse zum Editieren Edit[Felddatenklasse]VM z.B.:

public class EditEVVM : ReactiveObject

Enthält:

* Property Model mit Verweis auf das Objekt der zugrunde liegenden Klasse
* ReactiveCommands Edit (für Editiermodus), Save und Delete
* Editierbare Felder der zugrunde liegenden Klasse
* Regeln zur Freigabe des Speicherns (neu!) Verallgemeinerung von cansave
* Methode updateModel()
* Methode updateView([Feldatenklasse] f)
* Methode executeSave()

Klasse zur Auswahl von Kindklassen (nur ES, EV, CS , IU) View[Felddatenklasse]VM z.B.

public class ViewEVVM : ReactiveObject

Enthält:

* Property Model mit Verweis auf das Objekt der zugrunde liegenden Klasse
* Liste von ViewModelles der Kindklasse IList<KindklasseVM>
* Property Current mit ViewModel der aktuellen Klasse
* ReactiveCommand Add

# Seiten- ViewModels im ViewModelLocator registrieren

**Im DiversityPhone Projekt,   
Im Ordner ViewModels**

* IOC Constructor Lambda
* Zugriffs-Property:

public ViewEVVM ViewEV { get { return \_ioc.Resolve<ViewEVVM>(); } }

# View erstellen

**Im DiversityPhone Projekt,   
Im Ordner View**

PortraitPage Template

Template in der Templateklasse erstellen/wählen

🡪DataContext der Seite per Binding auf das ViewModel setzen:

DataContext="{Binding Source={StaticResource Locator}, Path=ViewEV}"

# Seite registrieren

**Im DiversityPhone Projekt,   
Im Ordner Services**

🡪Zum Page enum hinzufügen

🡪Im NavigationService eintragen (Navigate() und evtl. noch mehr Message Listener für Select oder Edit Nachrichten)