Grundarchitektur DiversityMobile für WindowsPhone

DiversityMobile für WindowsPhone ist der mobile Client für die Arbeit mit DiversityCollection aus der Diversity Workbench ([www.diversityworkbench.net](http://www.diversityworkbench.net) ). DiversityMobile erlaubt als Teil der Diversity Workbench die mobile Felddatenerfassung mit Georeferenzierung und Aufnahme von Multimediadaten wie Fotos, Videos und Audiokommentaren. Bei der Felddatenerfassung wird ein verschlagwortetes Vokabular aus der Diversity Workbench verwendet. Die Felddaten werden aber lokal zwischengespeichert, so dass im Feld keine Onlineverbindung benötigt wird. Dementsprechend ist eine regelmäßige Kommunikation des mobilen Clients mit der zentralen Datenbank am SNSB notwendig.

# Kommunikation mit dem Repository

Die Architektur der Kommunikation gliedert sich in drei Komponenten:

* Datenbank am Repository
* Datenbank des mobilen Clients
* Webservice zum Austausch der Daten

Das Repository am SNSB ist eine MSSQL-Server Datenbank, welche die Daten zur Bearbeitung der Desktopanwendung DiversityCollection enthält. Für den mobilen Client wird die Teilmenge der Datenbank verwendet, die für die mobile Felddatenerfassung benötigt wird und an die Umgebung für die Verwendung in einem Handy angepasst. Da DiversityCollection weitere Aufgaben wie die Verwaltung von Sammlungen hat, die im Feld nicht benötigt werden, verfügt die Datenbank im mobilen Client nur über einen Teil der Tabellen aus dem Repository. Darüber hinaus wird die Datenstruktur zur Verwendung im mobilen Gerät optimiert, so dass verwendete Tabellen anders aufgebaut sein können. Einen Überblick über diese Anpassungen findet sich im Dokument: -> *DatenModell DiversityPhone.*

Die Übertragung von Feldaten aan das Repository sowie die Übertragung von Definitionen an das Mobilgerät wird über WCF-Webservices realisiert, die am SNSB gehostet werden. Diese sind im Folgenden:

* DiversityService: Übertragung von Felddaten an das SNSB, Download von Definitionen, Download von taxonomischen Listen. Verantwortlich für die Synchronisation (siehe Dokument -> *Synchronisation*)
* MapMediaService und MediaService4: Annahme von MultimediaObjekten und Versorgung mit Webadressen zum Download von Kartenmaterial. Diese Webservices werden am SNSB programmiert und sind somit nicht Teil dieser Dokumentation. Die Anwendung der Webservices findet sich in den Dokumenten :-> *Upload Multimedia* und *Download Karten.*

Die Webservices sind als *ServiceReferences* in DiversityPhone eingebunden und werden so aus dem SourceCode von DiversityPhone angesprochen. Für näheres hierzu siehe: <http://msdn.microsoft.com/en-us/library/bb628652.aspx> . Der Webservice DiversityService greift dabei auf die Datenbank des Repositories zu und wandelt die Daten des mobilen Clients in passende Objekte mit der Struktur des Repositories um. Da mit verscheidenen Datenbanken im Repository gearbeitet werden kann, die zwar über eine ähnlich aber nicht dieselbe Struktur verfügen, muss ein sehr flexibles ORM in diesem WebService verwendet werden. Deshalb wird im DiversityService PetaPoco verwendet (-><http://www.toptensoftware.com/petapoco/>), da dieses ORM eine sehr flexible Anpassung der Datenstrukturen erlaubt. Näheres hierzu findet sich im Dokument *Synchronisation.*

Die Daten werden auf dem mobilen Client in einer MSSQLServerCE-Datenbank gehalten, die nativ in WindowsPhone ab Version 7.5(Mango) enthalten ist. Dementsprechend setzt DiversityMobile auch die Betriebssystem-Version 7.5 voraus. Der Zugriff auf diese Datenbank ist in verschiedenen internen Services gekapselt. Diese sprechen über linq2sql (siehe : <http://msdn.microsoft.com/de-de/library/vstudio/bb386976.aspx>) die Datenbank auf dem Mobilgerät an. Dazu wird für diese Daten ein sogenannter DataContext angelegt, welcher das logische Datenbankmodell in C# enthält. Genauere Informationen finden sich im Dokument -> *OfflineStrorage*.

# Interne Kommunikation

DiversityMobile benötigt an verschiedenen Stellen der Programmausführung Daten aus anderen Teilen des Programms. Diese Kommunikation ist intern über eine Menge von Services gelöst, die jeweils eine spezifische Aufgabe haben. Dazu wird eine Schnittstelle(Interface) definiert, die von den ausführenden Klassen angesprochen wird. Die konkrete Implementierung der Services erfüllt diese Schnittstellen und Instanzen dieser Services sind zentral hinterlegt, so dass dies nicht bei jeder Verwendung neu instanziiert werden müssen. Die Services finden sich im Projekt DiversityPhone im Ordner Services.

Die internen Services sind im Folgenden:

* IFieldDataService: Zugriff auf die in der mobilen Datenbank gespeicherten Felddaten siehe -> Offlinestorage
* ITaxonService: Verwaltung von taxonomischen Listen und Zugriff auf Taxa siehe -> *Download von Taxa, Arbeiten mit Taxa*
* IVocabularyService: Verwaltung von Schlagworten aus DiversityCollection
* IBackgroundService: Organisation von Prozessen im Background, so dass während deren Ausführung weitergearbeitet werden kann -> *Backgroundservices*
* IConnectivityService: Service zum Überprüfen einer Onlineanbindung
* ILocationService: Service für die Georeferenzierung -> *Georeferenzierung*
* IMultiMediaClient: Kapselung des Webservices zum Upload von Multimediadaten -> *Upload Multimedia*
* ISettingsService: Nutzerverwaltung
* NavigationService: Bedienung des mobilen Clients. Steuerung der Navigation -> *Navigation*
* DialogService: Informationen an den Nutzer zur Programmlaufzeit z.B. Ausgabe von Fehlermeldungen.

Die Services werden zentral bei der Initialisierung der APP angelegt und sind die ganze Laufzeit verfügbar (siehe Klasse App.caml.cs –Methode Initialize()).

# Nutzeroberfläche

Die Nutzeroberfläche ist im Metro-Design von WindowsPhone gehalten. Die Oberflächengestalung von WindowsPhone basiert auf dem MVVM-Pattern, welches die Anzeige von der Ausführungslogik trennt (siehe -> <http://msdn.microsoft.com/de-de/magazine/dd419663.aspx> ). In DiversityPhone wurden dafür eine Reihe von Views im Ordner Views angelegt. Jedem dieser Views ist ein Viewmodel für die Ausführungslogik zugeordnet. Diese finden sich im Ordner Viewmodels.

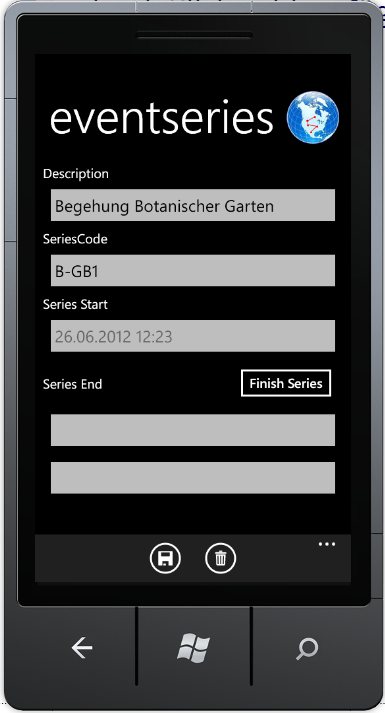
Darüber hinaus ist jede View über eine URI eindeutig gekennzeichnet, welche vom NavigationService verwendet wird um eine entsprechende Seite aufzurufen ->*Navigation*.

Eine View besteht aus 2 Komponenenten: Der Drag&Drop Oberflächengestaltung mit XAML-Code und dem Code-behind. XAML ist dabei eine besondere XML-Variante die für die Gestaltung von Oberflächen in Silverlight (-><http://de.wikipedia.org/wiki/Microsoft_Silverlight> ). In diesem werden die konkreten Elemente der Nutzeroberfläche angelegt.

In dider MVVM-nehmen die Views direkt die Nutzereingaben auf und leiten diese an die ViewModels weiter. Deswegen ist jeder View im Code-behind ein ViewModel zugeordnet, welches die Verarbeitungslogik enthält und somit auch für die Speicherung von verantwortlich ist. Um asynchron programmieren zu können wird auf das ReactiveUI zurückgegriffen: -> <http://www.reactiveui.net/>.

Es gibt verschiedene Arten von Views mit jeweils einem anderen Aufgabespektrum:

## Aufnahme/Editieren von Felddaten:



Über diese Views können Felddaten angelegt und gelöscht werden. Die View muss dementsprechend Felder für die Datenaufnahme der entsprechenden Klasse zu Verfügung stellen. Diese sind mit Properties der Viewmodels verknüpft und über DataBinding an diese gebunden. Damit diese stets den aktuellen Wert enthalten wird bei der Implementierung der Viewmodels auf das ReactiveUI zurückgegriffen: -> <http://www.reactiveui.net/>. Eine Umfassende erklärung zur Aufnahme von Daten mit allen Schritten der Aufnahme in das Datenmodell findet sich in dem Dokument -> *Datenklassen HowTo*.

### Anzeige von Listen einer Datenklasse mit Pivots

Diese Views zeigen eine der folgenden Hauptklassen aus DiversityCollection an:

* EventSeries
* Event
* Specimen
* IdentificationUnit

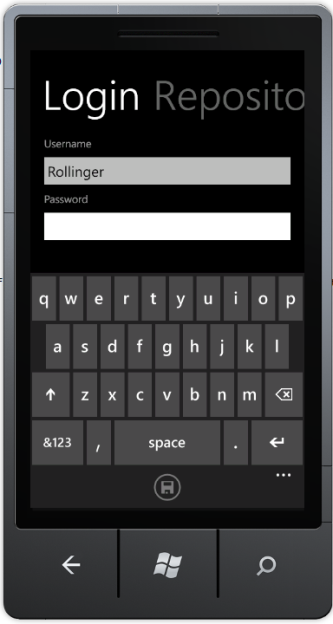
In den über die Pivots erreichbaren Listen sind jeweils die untergeordneten Objekte des angezeigten Objekts erreichbar. Dabei enthät das 1. Pivot die untergeordnete Hauptklasse. Das Pivot multimedia dem Objekt zugeordnete MultimediaObjekte und das Pivot descriptions alle anderen Objekte, die dem Objekt der Darstellung zugeordnet sind. Die Oberfläche kann auch während des Ladens der Listen bedient werden, was auf die Anwendung des ReaciveUI zurückzuführen ist: -> <http://www.reactiveui.net/>. Die Genau Technik ist im Dokument *Views der Hauptklassen* beschrieben.

## Settings



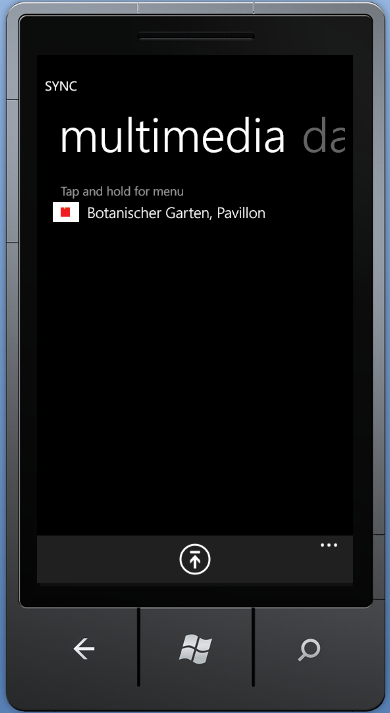
Die Settingspage zeigt die aktuellen Nutzereinstellungen an und ermöglicht die Navigation zur Taxonverwaltung und zur Synchronisation. Darüber hinaus können die Schlagworte aktualisiert werden und das Handy kann zur Neuinitialisierung zurückgesetzt werden.

## Initialisierung



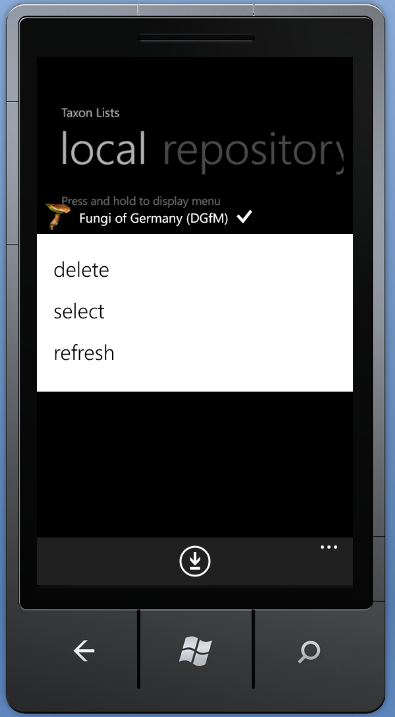
Setzt DiversityMobile für die erstmalige Benutzung auf. Hierbei wird eine Internetverbindung benötigt, da Daten aus dem Repository auf das Handy übertragen werden müssen. Im Rhamen der Initialisierung wird die Datenbank und das Projekt gewählt. Diese Verbindung kann ohne Löschen der Daten nicht aufgehoben werden siehe ->*Nutzerverwaltung*.

## Synchronisation



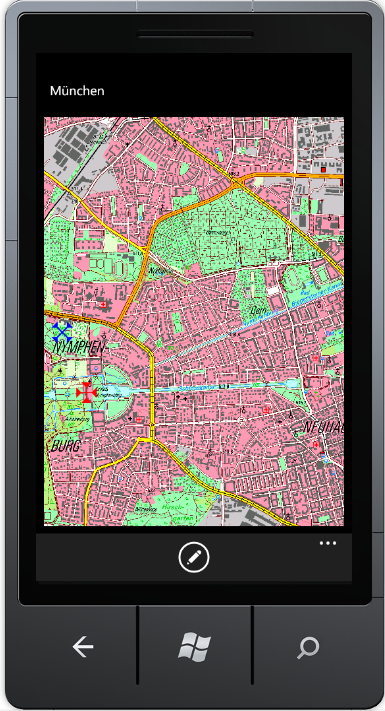
Im Synchronisationsmenü werden die Objekte, die aktuell synchronisiert werden können, als Liste angezeigt. Da die Synchronisation von MultimediaObjekten den Upload des verknüpften Objekts voraussetzt, werden diese erst nach dem entsprechenden Upload angezeigt siehe -> *Synchronisation*.

## Verwaltung von Taxa



Taxa werden über den DiversityService vom SNSB in die Datenbank des Mobilgeräts heruntergeladen. Dabei wird zwischen persönlichen und öffentlichen Listen unterschieden. Die persönlichen Listen sind mit dem Profil des Benutzers verbunden, die öffentlichen sind für alle Nutzer gleich. Da mit dem SNSB kommuniziert wird, wird eine Internetverbindung benötigt.

## Kartenanzeige



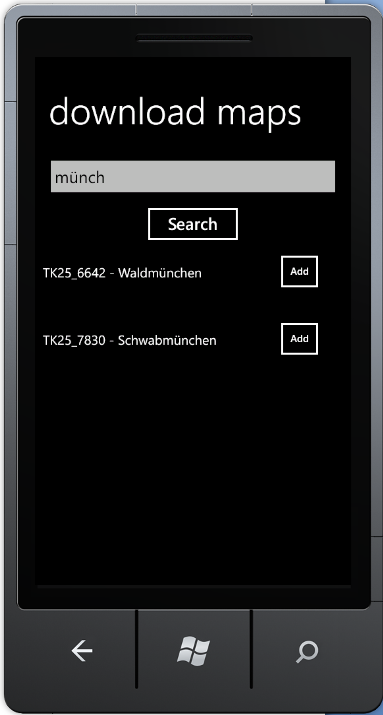
Anzeige von TK´s. Diese werden vom SNSB zu Verfügung gestellt. Die Karten sind Georeferenziert und erlauben die Anzeige des eigenen Standorts und von georeferenzierten Objekten. Mit Gesten kann die Karte gezoomt und gesteuert werden.

## Kartenauswahl



Auswahlmenü für die aktuelle anzuzeigende Karte. Diese wird anschließend im Speicher vorgehalten, so dass die Ladezeit verkürzt wird.

## Kartendownload



Kontakt des MapMediaservice zum Download der URL´s der Karten. Diese werden durch einen Klick auf Add von der entsprechenden URL passwortgeschützt heruntergeladen und georeferenziert.

## Multimedia

Für die aufnahme und Anzeige von Multimediadaten stehen verschiedene Views zu Verfügung, die nach Art des MultimediaObjekts spezifisch sind. Arten von Multimediaobjekten sind:

* Audio
* Video
* Foto

Dies Aufnahme und Anzeige von Multimediadaten leiten zwar von derselben Basisklasse ab, müssen aber in Abhängigkeit vom Medium stärker differenziert werden. Die Grundlagen für die Anbindung von Multimedia finden sich auf: <http://msdn.microsoft.com/en-us/library/windowsphone/develop/ff431744%28v=vs.92%29.aspx>

# Nutzerverwaltung

# Multimedia

# Georeferenzierung