Aufgabe: Kundentabelle

Ziel

In dieser Aufgabe lernen wir, Daten von einer REST-Schnittstelle zu laden und in einer Tabelle darzustellen.

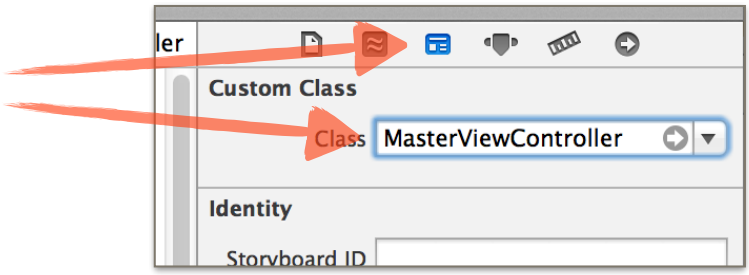
Aufgabe

1. Mega-CRM Skeleton Projekt kopieren und öffnen

2. Das momentan leere *MainStoryboard* öffnen und einen neuen *Navigation View Controller* von der Objektliste rechts unten auf die Designerfläche ziehen. Es wird gleichzeitig auch gerade ein *Table View Controller* erstellt, welchen wir brauchen, um die Kundenliste darzustellen.

3. Wir erstellen nun einen neue Klasse `MasterViewController`, welche vom Typ `UITableViewController` abgeleitet ist.

4. Damit dieser spezialisierte Controller nun mit unserer Tabelle verbunden wird, müssen wir in unserem Storyboard den Controller der Tabelle auswählen (gelbes Symbol unten an der Scene). Danach im *Identity Inspector* die neu angelegte Klasse `MasterViewController` setzen.



5. Wir importieren in der Implementation des `MasterViewControllers` die Headers von `Customer` und `CustomerService`. Anschliessend machen wir eine Interface Extension. Hinweis: Diese werden benötigt, um das Interface privat zu erweitern:

```
@interface MasterViewController () <CustomerServiceDelegate>

@property (nonatomic, strong) CustomerService* customerService;
@property (nonatomic, strong) NSMutableArray* customers;

@end
```

6. In der Methode `-viewDidLoad` können wir nun die Initialisierung des Controllers machen. Zuerst erstellen wir eine Instanz vom Customer-Service und speichern die Referenz darauf im zuvor erstellten Property.

7. Danach rufen wir `-loadCustomersWithDelegate` auf dem Customer-Service auf.

8. Jetzt müssen wir nur noch die Delegate Methode der Services implementieren und wir haben die Customers bei uns.

```
- (void)customersLoaded:(NSArray *)customers
{
    [self.customers removeAllObjects];
    self.customers = [NSMutableArray arrayWithArray:customers];
    [self.tableView reloadData];
}
```

9. Allerdings wird noch immer nichts in der Tabelle dargestellt. Nun müssen wir die Methode –`numberOfRowsInSection` implementieren. Diese soll die Anzahl Tweets zurückgeben.

10. Um nun in der Zelle wirklich etwas darzustellen, müssen wir die Methode –`cellForRowAtIndexPath` implementieren. Wichtig ist hier zu beachten, dass der *Identifier* mit jenem im *Interface Builder* zusammenpasst.
    Tipp: Der Zeilenindex ist im Object `indexPath` als Property `row` vorhanden.

11. Jetzt können wir die Applikation im Simulator starten und sollten, nach einer kurzen Wartezeit, eine Liste mit Customers sehen.

Detailansicht

12. Ziehen Sie nun im *Storyboard* für die Detailansicht einen leeren *View Controller* hinein. Erstellen Sie einen neue Klasse `DetailViewController`, welche von `UIViewController` abgeleitet ist. Diese Klasse setzen Sie wie zuvor bei dem `MasterDetailController` als *Custom Class* im *Identity Inspector* der neuen *Scene*.

13. Erstellen Sie einen “push” *Segue* zwischen der Zelle und unserer neuen Detail View. Setzen Sie als *Identifier* auf dem *Segue* “showDetail”.

14. Auf der Detailansicht brauchen wir nun zuerst einem zwei Elemente für den Vor- und Nachnamen. Idealerweise zwei *Text Field*s. Erstellen Sie im `DetailViewController` zwei *Outlets* für die Elemente und verbinden Sie diese im *Storyboard* entsprechend.

15. Zusätzlich brauchen wir noch ein öffentliches Property für den Customer. Dieses lesen wir in der Methode –`viewDidLoad` aus und setzen die beiden *Outlets* entsprechend. (Idealerweise diese Logik in eine eigene Methode auslagern)

16. Was nun noch fehlt, ist, dass beim Auswählen einer Zelle das entsprechende Objekt an die Detailansicht weitergereicht wird. Dies geschieht in der Klasse `MasterViewController`. Wir implementieren dazu die Methode –`prepareForSegue:sender:`. Die Implementation sollte in etwa so aussehen:

17. Wenn wir nun wieder die Applikation im Simulator starten, können wir nun eine Zeile auswählen und in

```
if ([[segue identifier] isEqualToString:@"showDetail"]) {
   NSIndexPath *indexPath = [self.tableView
                                indexPathForSelectedRow];
   Customer *customer = [self.customers
                                objectAtIndex:indexPath.row];

   [[segue destinationViewController] setCustomer:customer];
}
```

der Detailansicht die Daten einsehen.

Tipps:

- Um ein Element aus einem Array zu erhalten, verwenden Sie die Methode `–objectAtIndex:`

```
[self.arrayReferenz objectAtIndex:indexPath.row];
```

- Tabellen-Zellen mit einem Standard Style können über die Properties `detailTextLabel` und `textLabel` gefüllt werden.
- Bei Zellen mit einem Custom Style müssen die Elemente über die Methode `–viewWithTag:` gefunden werden. Beachten Sie, dass die Tags mit jenen im Interface Builder dieser Elemente übereinstimmen müssen.

Fazit

Folgende Punkte haben wir gelernt:

- Wir haben gesehen, wie mit einem Delegate Pattern die Daten aus dem Modell gelesen werden.
- Wir haben unser erstes komplettes Storyboard erzeugt mit einer *Scene*, die in einer Navigation eingebettet ist.