

SWT Projekt: Text-Uhr App

Tim Lehr, Matrikelnr. 980353

Inhaltsverzeichnis

Inhaltsverzeichnis	2
1. Problembeschreibung	3
2. Analyse	4
2.1 Zeitformatierung	4
2.2 Zeitanpassung	4
2.3 Tageszeit	4
3. Design	5
3.1 GUI - Storyboard	5
3.2 Klassen	5
3.2.1 Die Klasse AppDelegate	5
3.2.2 Die Klasse ViewController	6
3.2.3 Third Party Klassen (CocoaPods)	6
3.3 Klassendiagramm	7
4. Implementierung	8
5. Installation & Betrieb	8

1. Problembeschreibung

Für das MIB Modul „Softwaretechnik“ soll eine Anwendung als Kleinprojekt entwickelt werden. Dabei handelt es sich um eine spezielle Art von Uhr, die dem Nutzer die aktuelle Uhrzeit in Textform präsentiert. Beispiel: „Fünf vor Neun Uhr“, „Fünf nach halb Drei“.

Die Anzeige verzichtet dabei auf die Darstellung von Sekunden oder Minuten und löst die aktuelle Uhrzeit nur in Fünf-Minuten Intervallen auf. Die Anwendung soll als iPhone App in den Programmiersprachen Objective-C realisiert werden. Der Umfang der App beschränkt sich lediglich auf die Anzeige der Uhrzeit, soll dem Nutzer diese jedoch optisch ansprechend präsentieren. Außerdem sollte die App verhindern können, dass sich das Gerät in den Standby Modus versetzt oder den Bildschirm automatisch dimmt.



Bild 1: Konzept-Skizze der App

2. Analyse

2.1 Zeitformatierung

Die Uhrzeit wird nur „grob“ in Fünf-Minuten Intervallen dargestellt. Dazu muss die Grammatik der gewählten App-Sprache analysiert und in Segmente zerlegt werden. Für das Projekt beschränke ich mich daher vorerst auf Deutsch als Sprache:

Uhrzeit (Zahlen)	(Punkt)	Minute	Prefix	(Halb)	Stunde
13:40		ZWANZIG	VOR		ZWEI
12:00	PUNKT				ZWÖLF
09:45		VIERTEL	VOR		ZEHN
05:35		FÜNF	NACH	HALB	SECHS
20:05		FÜNF	NACH		ACHT
22:15		VIERTEL	NACH		ZEHN

Es ergeben sich die folgenden Segmente der Zeitangabe:
(Punkt), Minute, Prefix, (Halb) und Stunde

2.2 Zeitanpassung

Damit die Uhrzeit korrekt in der geplanten Textform dargestellt werden kann, muss die Systemzeit etwas aufgearbeitet werden. So werden die Sekunden gänzlich vernachlässigt und die Minuten auf das nächste Fünf-Minuten Intervall auf- / abgerundet.

XX:X1 und XX:X2 Uhr wird **abgerundet**. Beispiel: 12:22 wird zu 12:20 Uhr.
XX:X3 und XX:X4 Uhr wird **aufgerundet**. Beispiel: 17:53 wird zu 17:55 Uhr.

2.3 Tageszeit

Die App informiert den Nutzer weiterhin über die aktuelle Tageszeit und muss diese daher anhand der aktuellen Uhrzeit bestimmen. Je nach Tageszeit soll zudem das Wettericon angepasst werden.

Morgen	06:00 bis 11:00 Uhr
Mittag	11:00 bis 13:00 Uhr
Nachmittag	13:00 bis 18:00 Uhr
Abend	18:00 bis 22:00 Uhr
Nacht	22:00 bis 06:00 Uhr

3. Design

3.1 GUI - Storyboard

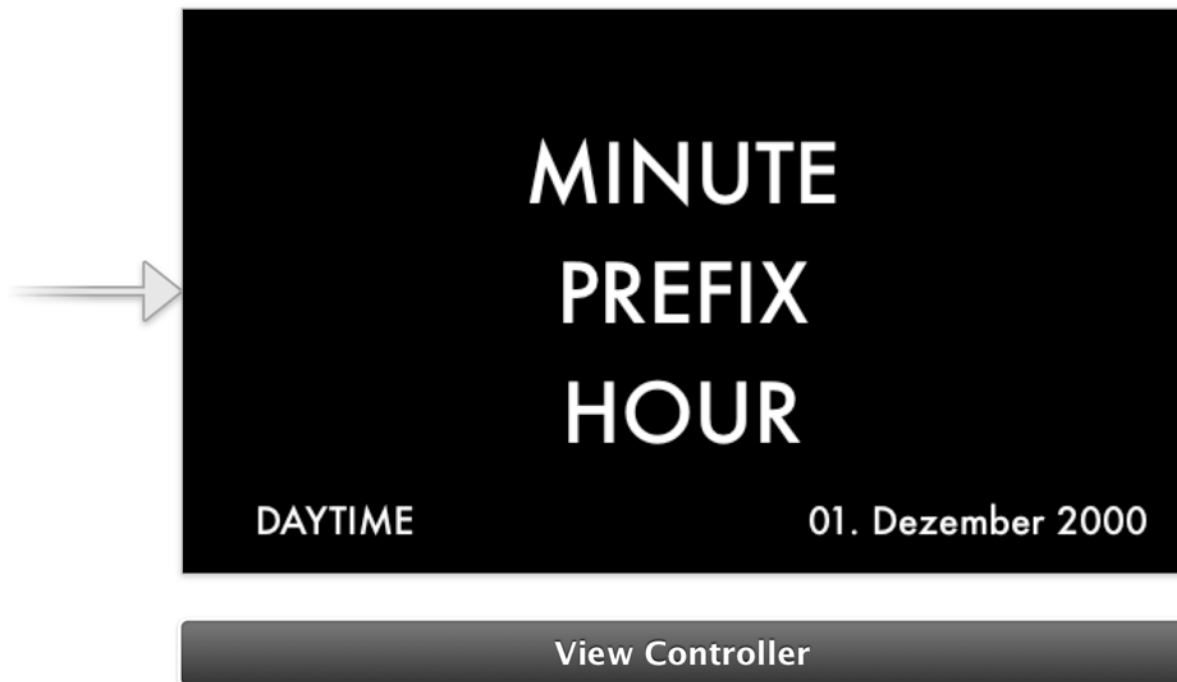


Bild 2: GUI Prototyp, erstellt in XCodes Storyboard Editor

Die grundlegende Oberfläche der App wird über ein sogenanntes Storyboard definiert. Diesem per Drag & Drop erstellten Interface lassen sich erste Parameter (Schriftart, Farbe ...) zuweisen, welche in der View Controller Implementierung weiter angepasst werden. Über sogenannte Outlets werden die Elemente im Storyboard mit dem eigentlichen Objective-C Code verknüpft.

Das Storyboard der App besteht lediglich aus einem View Controller, welcher direkt beim Start der App geladen wird. Dieser enthält auch die Elemente zur Anzeige der Uhrzeit. (siehe Bild 2) Bei der Erstellung des Storyboards muss darauf geachtet werden, dass die Positionierung des Textes an die Größe des Displays angepasst wird, damit der Bildschirm stets optimal ausgenutzt wird.

3.2 Klassen

Die App wird im Kern durch die iOS App spezifische Klasse *AppDelegate* gesteuert, während die Klasse *ViewController* für sämtliche App Funktionen wie die Zeitberechnung und deren Darstellung zuständig ist. Hinzu kommen einige importierte 3rd-Party Klassen, die für zusätzliche Features in die App eingebunden werden.

3.2.1 Die Klasse AppDelegate

AppDelegate wird von Apples iOS Software Development Kit bei Anlegung eines neuen Projektes automatisch generiert. Dort lassen sich spezifische Parameter direkt beim Start der App setzen und Methoden ausführen, bevor irgendeine Oberfläche geladen wird. Für dieses Projekt ist eine solche Funktionalität jedoch nicht geplant.





3.2.2 Die Klasse ViewController

Die Klasse *ViewController* enthält alle Funktionen der Text Uhr, neben den iOS-typischen View Controller Methoden wie *viewDidLoad* und zusätzlichen Methoden für die Steuerung des animierten Hintergrunds.




Interessant sind die Methoden unter den Marker „time display & calculations“. Hier wird die aktuelle Zeit abgefragt (*getTimeSegment*) und mithilfe der Funktionen *minuteToText* und *hoursToText* in Strings umgewandelt, welche mit der Funktion *updateLabel* auf die GUI überträgt. Der ganze Prozess wird vereinfacht über die Funktion *refreshTime* aufgerufen, welche mithilfe der NSTimer Funktion des iOS SDK im Sekundentakt aufgerufen wird.

Für die dynamischen Hintergründe der App muss zudem ein Delegate-Protokoll implementiert werden, welches dem Animationslayer seinen Farbraum zuweist. (*gradientColorsForGradientView*).


View methods

-  -viewDidLoad
-  -viewWillAppear:
-  -viewDidAppear:
-  -didReceiveMemoryWarning

animated background

-  -highColor
-  -lowColor
-  -colors
-  -backgroundSetup

time display & calculations

-  -getTimeSegment:
-  -getRoundedTime:
-  -minuteToText:
-  -hoursToText::
-  -getPrefix:
-  -getDaytime
-  -getDaytimeIcon
-  -getDateString
-  -getTimeArray
-  -updateLabel::
-  -refreshTime

gradient view delegate

-  -gradientColorsForGradientView:

Bild 3: Methoden der Klasse ViewController

3.2.3 Third Party Klassen (CocoaPods)

Um den Umfang des Projekts nicht zu sprengen und auf frei verfügbare Arbeit anderer Programmierer zurückzugreifen, kommt bei der Entwicklung das System CocoaPods zum Einsatz. Dieses ermöglicht ein bequemes Einbinden von Dritthersteller Bibliotheken über die Kommandozeile. Folgende „Pods“ arbeiten in der App:

FontAwesomeKit

Diese Dritthersteller Klasse ermöglicht das Einbinden ganzer Iconsets wie FontAwesome und IonIcons. Diese werden in der App zur Darstellung der Tageszeit Icons benötigt.

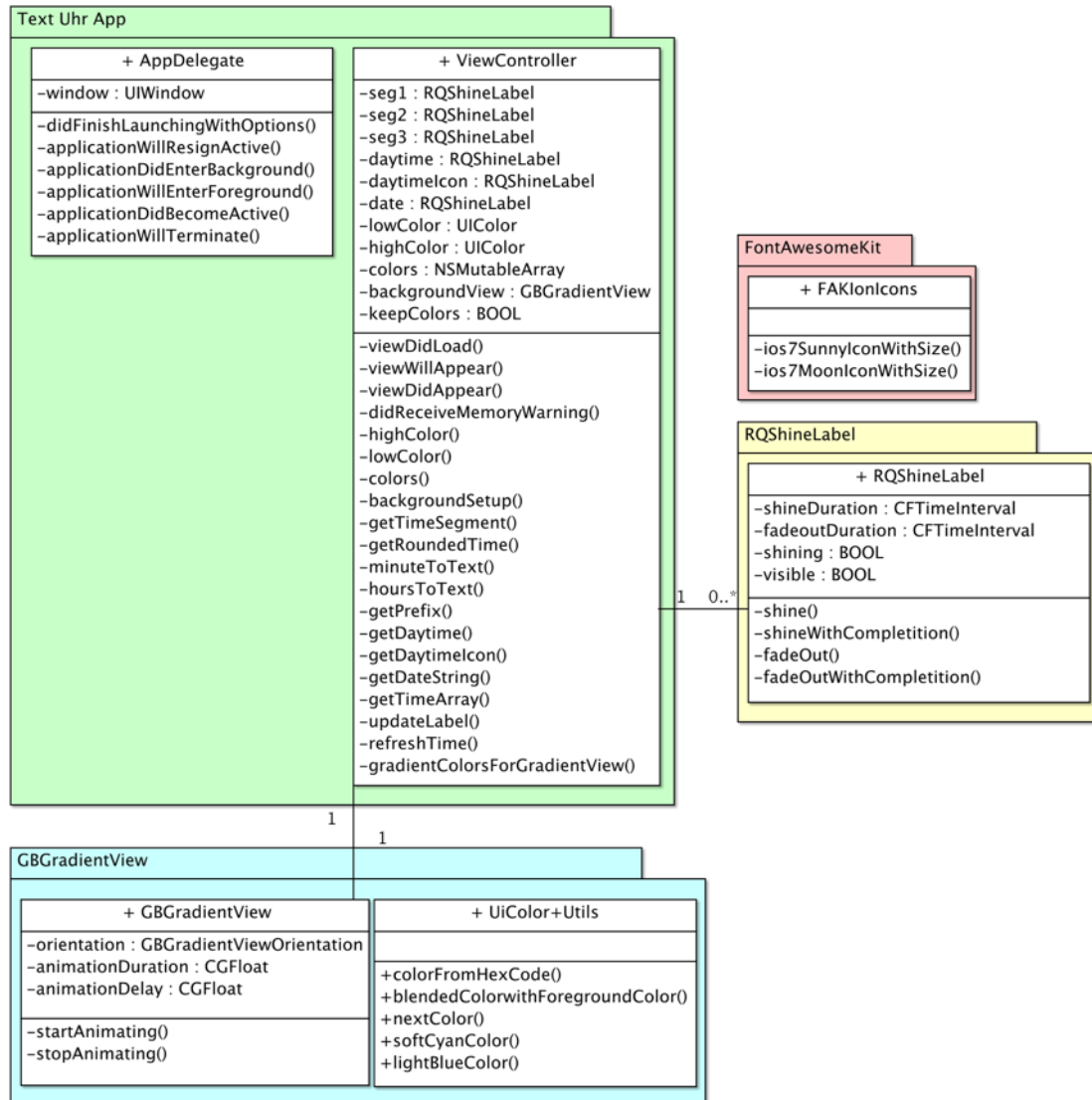
GBGradientView

Diese Subklasse der herkömmlichen UIView ermöglicht die Darstellung eines animierbaren Farbverlaufs im Hintergrund eines Views. In der App wird mithilfe dieser Klasse ein dynamischer Hintergrund erzeugt, welcher alle 60 Sekunden seine Farben wechselt. Die Farben werden mit der Klasse UIColor+Utils generiert, welche der Entwickler des Pods ebenfalls bereitstellt.

RQShineLabel

Eine Subklasse des UILabel welche die Animation eines Labeltextes vereinfacht. In der App wird mithilfe der Klasse der Textwechsel animiert, damit dieser auf den Benutzer flüssiger erscheint.

3.3 Klassendiagramm



4. Implementierung

Die Implementation der App kann aus den kommentierten Projektdateien entnommen werden.

5. Installation & Betrieb

Zur Installation muss die App zwangsläufig aus dem Apple App Store bezogen werden, da es unter iOS ohne Modifikationen keine andere Installationsmöglichkeit gibt.

Nach der Installation genügt ein Tap auf das App Icon, um die Text Uhr zu starten. Es ist zu beachten, dass die App lediglich für den Landscape Modus des iPhone ausgelegt ist und auch keine nativ iPad Unterstützung mit sich bringt. Aufgrund des erhöhten Stromverbrauchs durch des konstant aktiven Displays, sollte das iPhone zudem an eine externe Stromquelle angeschlossen sein.



Bild 4 & 5: Screenshots der fertigen App aus dem iOS Simulator von Xcode