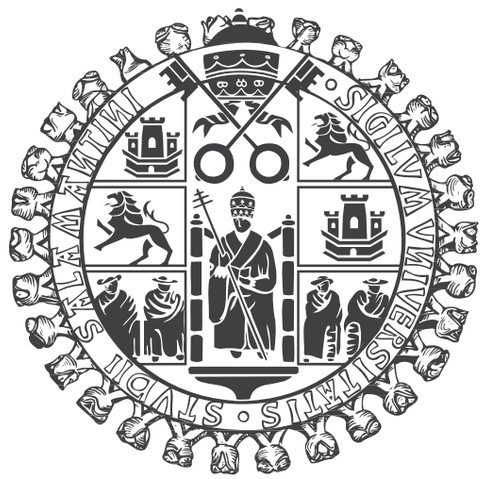
Hochschule Ulm



VNiVERSiDAD
Ð SALAMANCA

There is no magic!

Alain Andrieux (Apple Computer) ca. 1995

Dr. J.R.García-Bermejo Giner
coti@usal.es

Andere Steuerelemente

April-Mai 2015

Dr. J.R.García-Bermejo Giner
coti@usal.es

`NSSlider`

Schieberegler sind Steuerelemente, die um Gleitpunktzahlen einzugeben dienen. Die Klasse **`NSSlider`** hat Eigenschaften um Maximum und Minimum Werte, sowieso um laufend oder aussetzend Aktivität, einzustellen.

Es gibt drei Sorten Schieberegler: horizontal, vertikal und kreisförmig. In iOS (aber nicht in Mac OS) sind horizontalen Schieberegler häufigsten.

NSNumberFormatter

Da Schieberegler Gleitpunktzahlen ergeben, braucht man eine Prozedur um Nachkommastellen zu kontrollieren. Die Klasse **`NSNumberFormatter`** ist für dies perfekt geeignet.

CarManager

Die Anwendung **CarManager** kalkuliert den Kraftstoffverbrauch eines Autos mithilfe zwei Schiebereglern. Der durchschnittliche Verbrauch in Liter pro 100 km folgt die Formel:

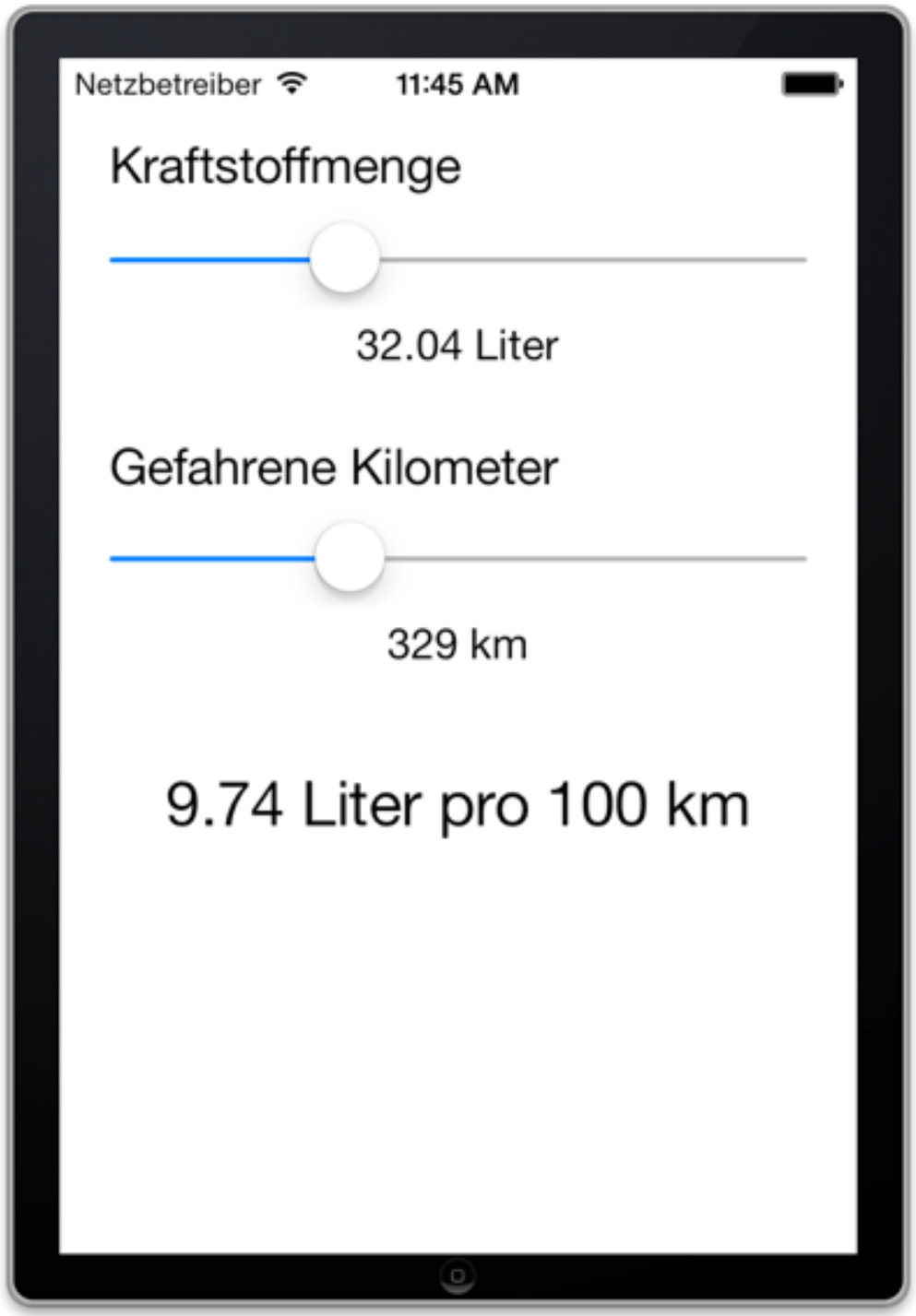
dv = Kraftstoffmenge (L) / Fahrstrecke (km) * 100;

Steuerelemente

Um beide Zahlen einzugeben werden zwei Instanzen von **`NSSLider`** angewendet. Diese Zahlen und auch das Ergebnis werden mithilfe Instanzen von **`UILabel`** angezeigt.

Die App

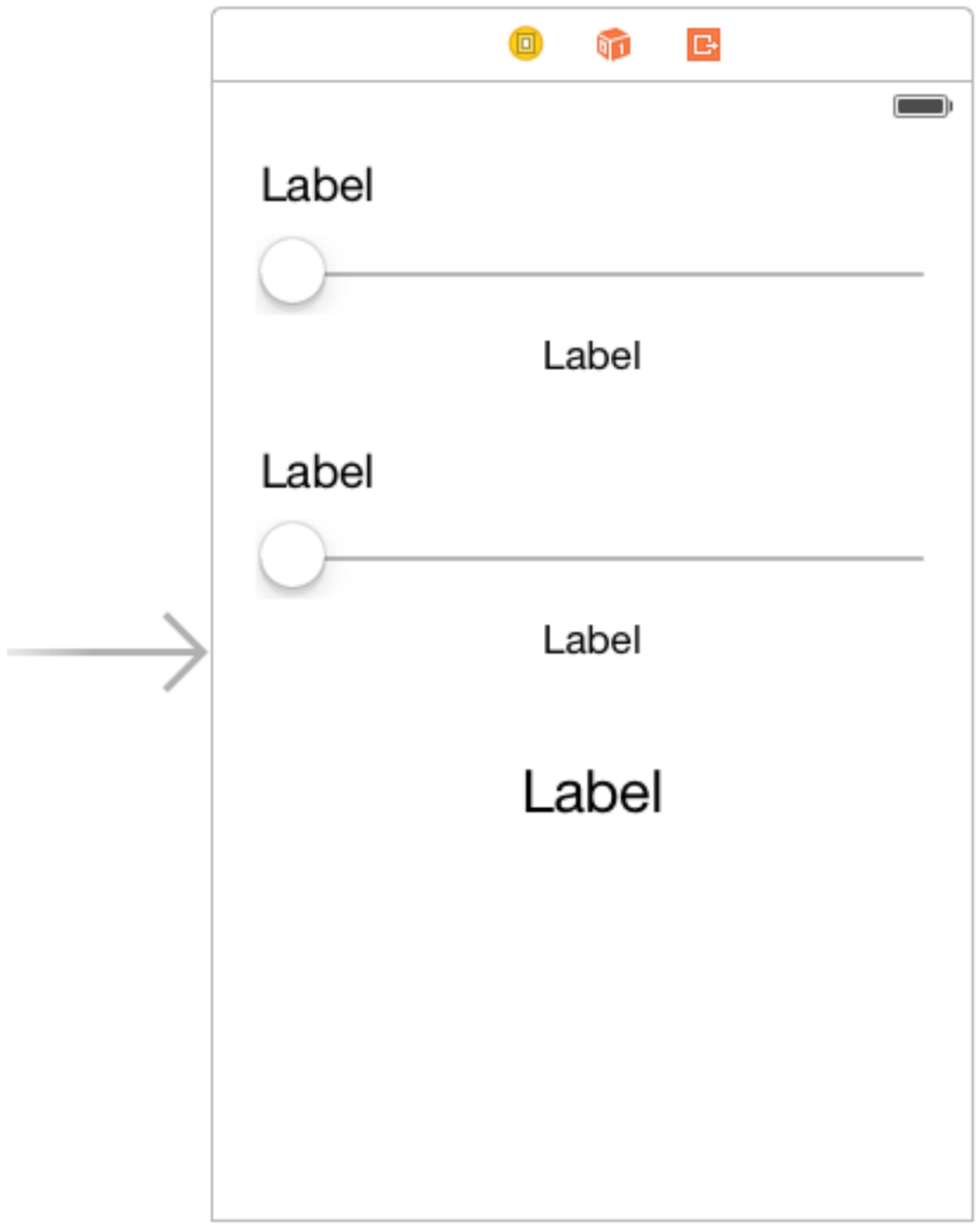
Die erledigte App sieht so aus:



Steuerelemente

Um die App zu erzeugen werden wir wie üblich die Single View Application Vorlage. Dazu fügen wir zwei Schieberegler und fünf Bezeichnungen:

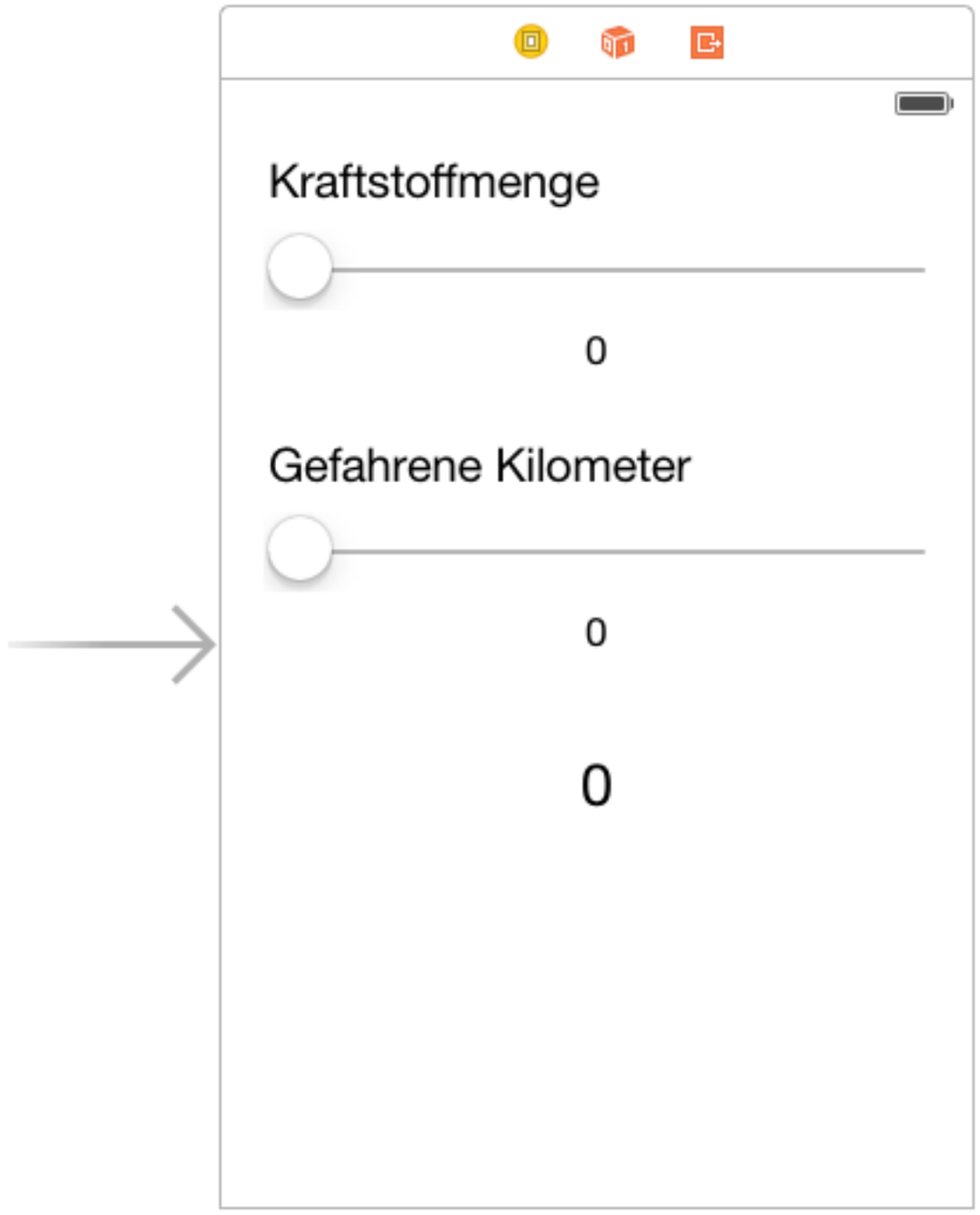
Steuerelemente



Steuerelemente

Danach füllen wir die Inhalten der Beschreibungen aus:

Steuerelemente



Steuerelemente

Steuerelemente fertig, jetzt müssen wir Outlets und IBActions in AppDelegate kreieren. Wie üblich, man zieht aus das Steuerelement auf die bezügliche Datei. Um die Datei sichtbar zu machen muss man einfach das Steuerelement markieren und die Assistant Editor Schaltfläche anklicken.

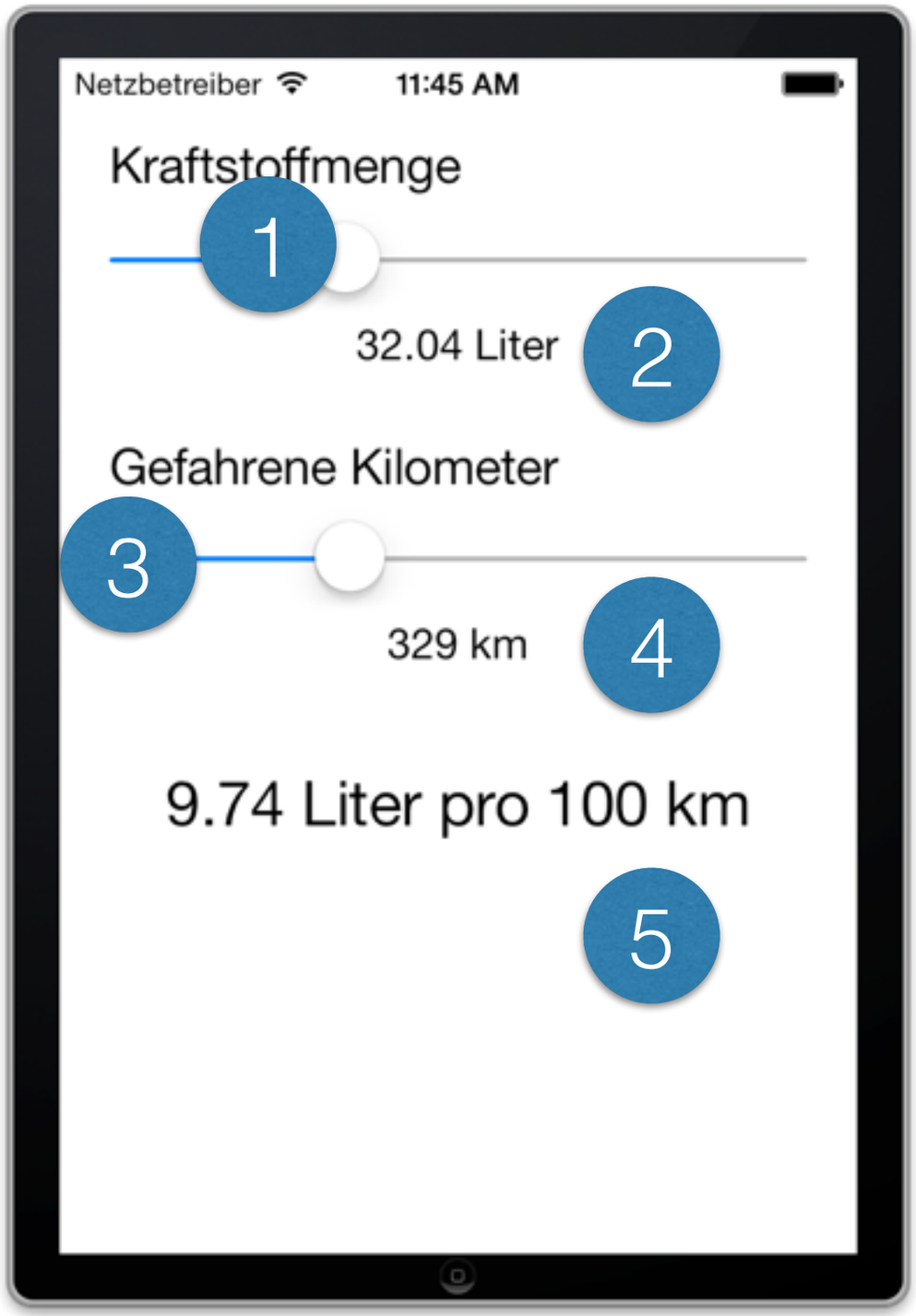
Die Datei ist natürlich ViewController.

Steuerelemente

Wann alle Outlets und Aktions hinzugefügt sind, wird ViewController so aussehen:

FuelConsumptionViewController

1. `@property (weak, nonatomic) IBOutlet UISlider *gasSlider;`
2. `@property (weak, nonatomic) IBOutlet UILabel *literOutlet;`
3. `@property (weak, nonatomic) IBOutlet UISlider *tripSlider;`
4. `@property (weak, nonatomic) IBOutlet UILabel *kilometerOutlet;`
5. `@property (weak, nonatomic) IBOutlet UILabel *meanGasConsumptionLabel;`



Steuerelemente

ViewController enthielt auch Methoden die als Actions der Schieberegler dienen. Diese Methoden werden auch aus die Schieberegler auf die Datei “hingezogen”. Sie heißen, zum Beispiel, **`fuelSliderDidMove`** and **`milageSliderDidMove`**. Daten werden in dieser Methoden nach **`ConsumptionCalculator`** übertragen.

Es gibt auch eine dritte Methode, **`calculateAndShowAverageConsumption`**, die sich um das Errechnen und das Anzeigen kümmert. Natürlich, sie ruft **`calculateConsumption`** (in **`ConsumptionCalculator`**) auf.

Steuerelemente

ViewController.h sieht wie folgt aus:

FuelConsumptionViewController.h

```
#import <UIKit/UIKit.h>

@interface FuelConsumptionViewController : UIViewController

@end
```

Steuerelemente

ViewController.m (die Umsetzung oder Implementierung) sieht wie folgt aus:

FuelConsumptionViewController.m

```
#import "FuelConsumptionViewController.h"
#import "ConsumptionCalculator.h"

@interface FuelConsumptionViewController ()

@property (strong, nonatomic)IBOutlet UILabel *fuelLabel;
@property (strong, nonatomic)IBOutlet UILabel *milageLabel;
@property (strong, nonatomic)IBOutlet UILabel *averageConsumptionLabel;

@end

@implementation FuelConsumptionViewController
{
    ConsumptionCalculator *consumptionCalc;
    NSNumberFormatter *formatter;
}
```

FuelConsumptionViewController.m

```
- (void)viewDidLoad
{
    [super viewDidLoad];
    consumptionCalc = [[ConsumptionCalculator alloc] init];

    self.fuelLabel.text = @"0,00 Liter";
    self.milageLabel.text = @"0 km";
    self.averageConsumptionLabel.text = @"";

    formatter = [[NSNumberFormatter alloc] init];
    // Die aktuellen Landeseinstellungen,
    // die currentLocale, zuweisen.
    formatter.locale = [NSLocale currentLocale];
    // Einen Style zur Formatierung festlegen.
    formatter.numberStyle = NSNumberFormatterDecimalStyle;
    // Zwei Nachkommastellen sollen es sein.
    formatter.maximumFractionDigits = 2;
    formatter.minimumFractionDigits = 2;
}
```

FuelConsumptionViewController.m

```
- (NSString *)formatDoubleToString:(double)value
{
    if (isnan(value) || value == INFINITY )
    {
        value = 0;
    }

    NSNumber *number = [NSNumber numberWithDouble:value];
    NSString *string = [formatter stringFromNumber:number];
    return string;
}

- (void)didReceiveMemoryWarning
{
    [super didReceiveMemoryWarning];
}
```

FuelConsumptionViewController.m

```
- (IBAction)fuelSliderDidMove:(id)sender
{
 // Den Typ der Absender überprüfen.
 if ([sender isKindOfClass: [UISlider class]])
 {
// Den Absender wieder in einen UISlider umwandeln.
    UISlider *slider = sender;

// Den Schiebereglerwert dem Calculator zuweisen, und den Wert anzeigen
  consumptionCalc.fuel = slider.value;
  self.fuelLabel.text = [NSString stringWithFormat:@"%.2f Liter”,
                         consumptionCalc.fuel];

// Den Wert in eine formatierte Zeichenkette umwandeln und anzeigen.
  NSString *fuelString =[self formatDoubleToString:consumptionCalc.fuel];
  self.fuelLabel.text =[NSString stringWithFormat:@"%.@ Liter”,
                        fuelString];

  // Rechnen und anzeigen
  [self calculateAndShowAverageConsumption];
 }
}
```

FuelConsumptionViewController.m

```
- (IBAction)milageSliderDidMove:(id)sender
{
    // Den Typ der Absender überprüfen.
    if ([sender isKindOfClass: [UISlider class]])
    {
        // Den Absender wieder in einen UISlider umwandeln.
        UISlider *slider = sender;

        // Den Schiebereglerwert dem Calculator zuweisen.
        // Durch die Umwandlung in eine ganze Zahl werden
        // die Nachkommastellen abgeschnitten.
        consumptionCalc.milage = (int) slider.value;

        // Den Wert anzeigen.
        self.milageLabel.text
        = [NSString stringWithFormat:@"%.0f km",
           consumptionCalc.milage];

        // Rechnen und anzeigen
        [self calculateAndShowAverageConsumption];
    }
}
```

FuelConsumptionViewController.m

```
- (void) calculateAndShowAverageConsumption
{
 // Das Ergebnis errechnen.
 double result = consumptionCalc.calculateConsumption;

 NSString *numberString = [self formatDoubleToString:result];

 NSString *resultText =
 [NSString stringWithFormat: @"%@ Liter pro 100 km”, numberString];
 // Und anzeigen
 self.averageConsumptionLabel.text = resultText;
}

@end
```

ConsumptionCalculator

Jetzt werden wir die Klasse ConsumptionCalculator erzeugen, um alles zum Verbrauchsberechnung im gleichen Ort zu haben.

Die neue Klasse ist aus NSObject abgeleitet. Ihre Attributen dienen als Speicher für die Werten der Schieberegler, und auch für das Ergebnis.

ConsumptionCalculator

ConsumptionCalculator sieht wie folgt aus. Diese Klasse ist sehr einfach,da sie nur zwei Methoden hat.

ConsumptionCalculator.h

```
#import <Foundation/Foundation.h>

@interface ConsumptionCalculator : NSObject

// Die Kraftstoffmenge.
@property (assign,nonatomic) double fuel;
// Gefahrene Kilometer
@property (assign,nonatomic) double milage;
// Der Durchschnittliche Verbrauch.
@property (readonly,nonatomic) double averageConsumption;

// Den durchschnittlichen Verbrauch ermitteln.
-(double) calculateConsumption;

@end
```

ConsumptionCalculator.m

```m
#import "ConsumptionCalculator.h"

@implementation ConsumptionCalculator

- (id)init
{
    self = [super init];
    if (self)
    {
        self.fuel = 0;
        self.milage = 0;
        _averageConsumption = 0;
    }
    return self;
}

- (double) calculateConsumption
{
    double consumption = self.fuel / self.milage * 100;
    _averageConsumption = consumption;
    return consumption;
}

@end
```

Steuerelemente

Noch etwas: können wir “Miles per Gallon” statt “Liters per 100Km” kalkulieren?

Hinweis: was geschieht, wenn wir eine neue calculateConsumption2 Methode hinzufügen?

Mehr Spaß: können wir einen Schalter hinzufügen, um L/100 Km oder MPG selektieren zu können?

Steuerelemente

Wichtige Fragen

- Wo wird die Instanz von ConsumptionCalculator deklariert?
- Wie werden Daten nach ConsumptionCalculator übertragen?

Steuerelemente

Wichtige Antworten

- consumptionCalc wird in ViewController deklariert

- Die Daten werden in die Aktionen der Schieberegler erhalten. Dort werden sie durch consumptionCalc nach ConsumptionCalculator übertragen.

CarManager

Noch etwas: können wir “Miles per Gallon” statt “Liters per 100Km” kalkulieren?

Hinweis: was geschieht, wenn wir eine neue calculateConsumption2 Methode hinzufügen?

Mehr Spaß: können wir einen Schalter hinzufügen, um L/100 Km oder MPG selektieren zu können?

Übung

Ein Lokfahrer weißt nicht, ob eine Lok den Zug ziehen kann. Das Gewicht des Zugs (in Tonnen) ist ihm bekannt, sowie das maximal Gefälle der Strecke, und der Zug muss mindestens Tempo 50 (in Kilometer pro Stunde) bergan fahren können. Wie viele PS braucht die Lok? Können Sie ihn mit einer App helfen?

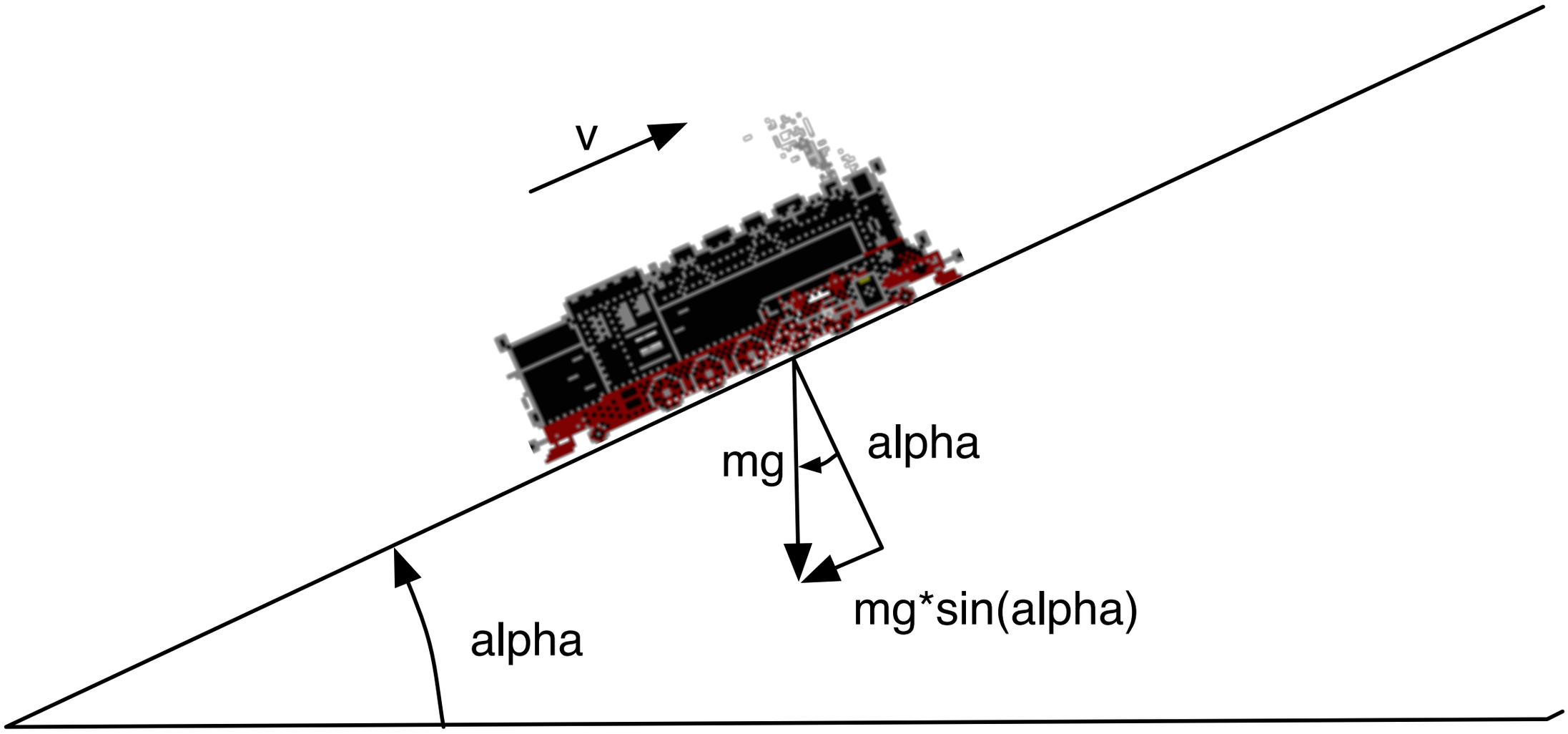
Unglücklicherweise, der Lokfahrer kann nicht die kleine Zeichen der Bildschirmtastatur berühren, da seine Finger zu groß sind. Er mag lieber Schieberegler benutzen…

Übung

Unglücklicherweise, der Lokfahrer kann nicht die kleine Zeichen der Bildschirmtastatur berühren, da seine Finger zu groß sind.

Er mag lieber Schieberegler benutzen…

Leistung = Kraft X Geschwindigkeit



mg*sin(alpha)*v = leistung

DieWilde13

Wie viele PS hat eine Lok?

Wie viele PS braucht ein 250-Tonnen-Zug, um Tempo 40 bergan fahren zu können? Sieht das Ergebnis zu viel, oder zu wenig aus? Gibt es ein Fehler?

Wie viele PS braucht ein 2-Tonnen-Auto, um Tempo 100 bergan fahren zu können? Sieht das Ergebnis zu viel, oder zu wenig aus? Gibt es ein Fehler?

DieWilde13

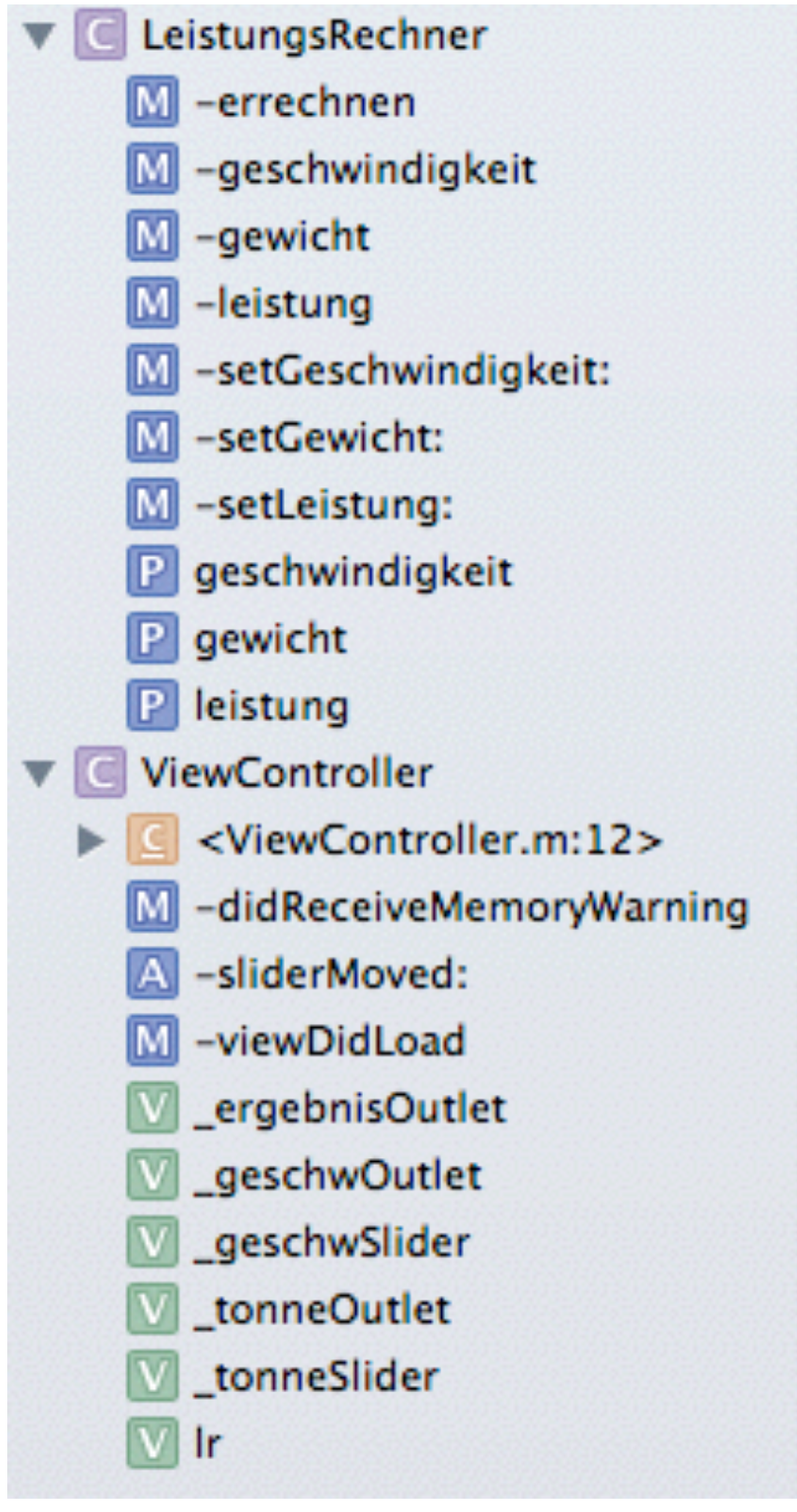
Die App kann so aussehen:

DieWilde13

Die Anwendung von Schieberegler statt Textfelder ist doch besser geeignet, da Bildschirmtastaturen nicht erforderlich sind. Deshalb gibt es mehr Platz im Bildschirm, um Steuerelement zu platzieren. DieWilde13 ist dem Lokführer besser!

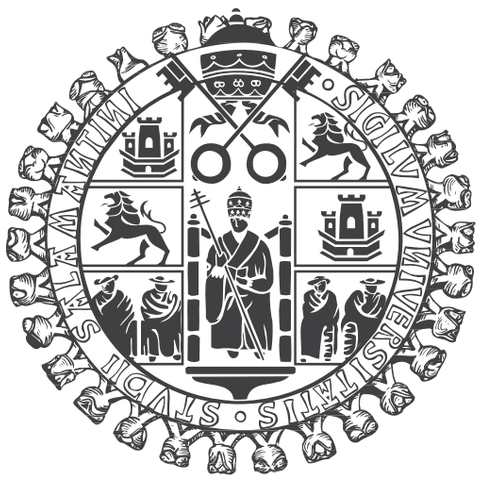
DieWilde13

Die Klassen, Attribute und Methoden können wie folgt sein:



Und jetzt…

`Storyboards` und **`Segue`** sind möglicherweise in unserer Zukunft



VNiVERSiDAD
Ð SALAMANCA

Hochschule Ulm