

Uživatelské rozhraní v iOS

Přednáška pro předmět ITU

Martin Hrubý

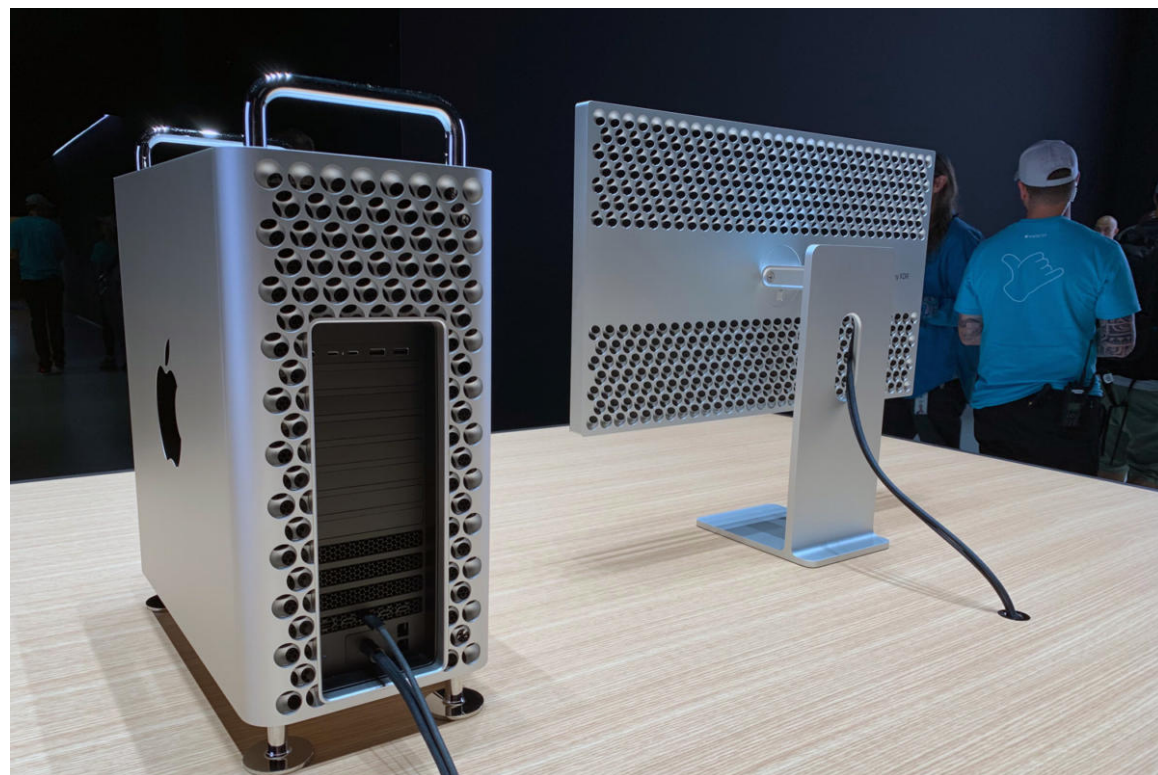
FIT VUT v Brně

2021 / 22

Historie Apple

- Steve Jobs, Steve Wozniak — 1976.
- Počítače Apple. Apple II. MacIntosh (1984).
 - Jobsův exil v NEXT Computers: NextSTEP.
 - Jobsův návrat do Apple (1995): Mac OS X.
- Mobilní zařízení (iPhone, iPad).
 - iOS/iPadOS.
 - watchOS — aplikace tvoří pár: iPhone app + watch extension.





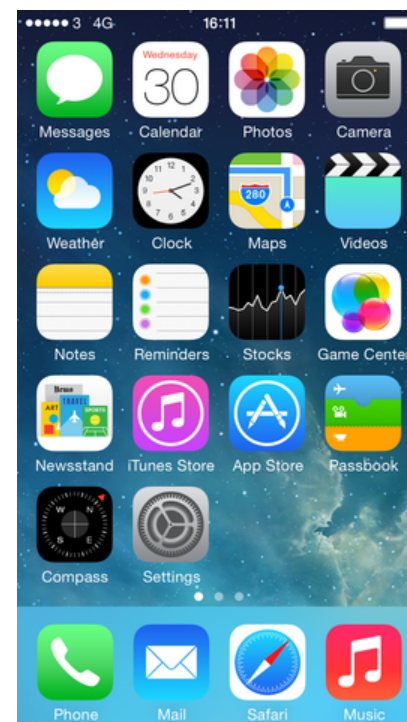
Počátky mobilních počítačů

- PDA — Personal Digital Assistant.
- 1993 — Apple, Newton.
- 1996 — Palm, Palm Pilot.



Projekt vývoje iPhone

- 2005 — Projekt extrémní významnosti pro Apple.
- Jaká forma? iPod nebo dotyková obrazovka.
- iPhone 2G — leden 2007. V prodeji červen 2007.



Operační systémy Apple

- System 1-9.
- (Mach, BSD) -> NextSTEP -> Darwin ->
 - mac OS X
 - -> iOS
- iOS ->
 - tvOS, watchOS
 - iPadOS.
- Značné znovupoužití kódu: Foundation.

Vývoj aplikací pro iOS

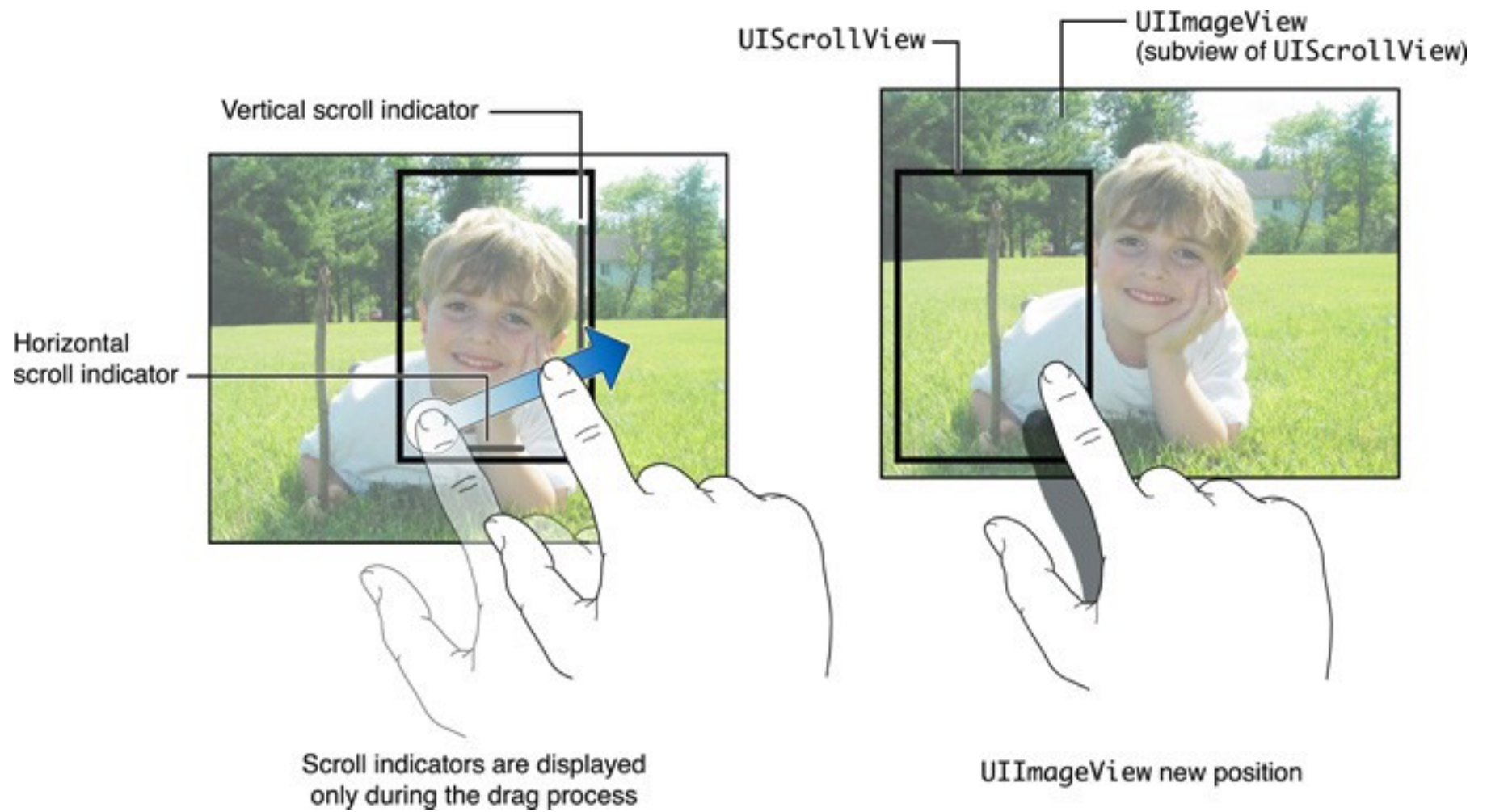
- Vývojové prostředí XCode, Mac.
 - Musí tam být obrázek jablka :)
- Vývojářský účet.
- Emulátor iOS — virtualizace iOS zařízení.
- Podepisování aplikací.
- Návaznost na iCloud — dokumenty, CloudKit.
- **Univerzitní vývojářský účet pro iOS aplikace.**

Klíčová slova

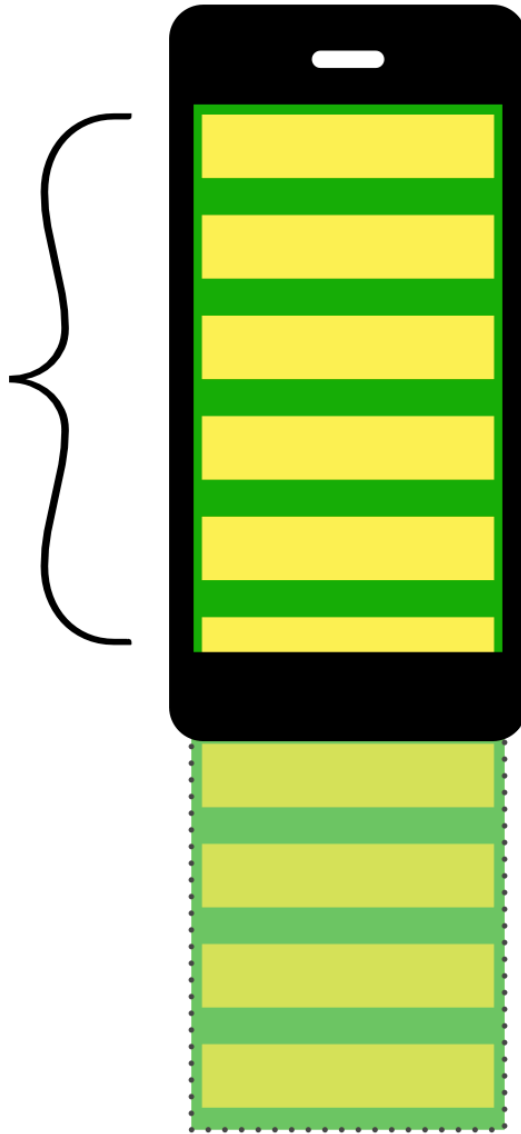
- Cocoa (Foundation, AppKit).
- Cocoa Touch (UIKit).
- Objective-C. Swift.
- MVC (Smalltalk / Xerox). Storyboard.
- MVVM — SwiftUI.

Východiska UI na iPhone

- Dotyková obrazovka. Multi-touch.
- Malé rozměry obrazovky.
- Vede na posuvný obsah — *UIScrollView*.
- *Vertikálně organizovaný obsah*, tj. seznam buněk. *UITableView*.
- *Většina aplikací pro iOS jsou různě organizované tabulky.*



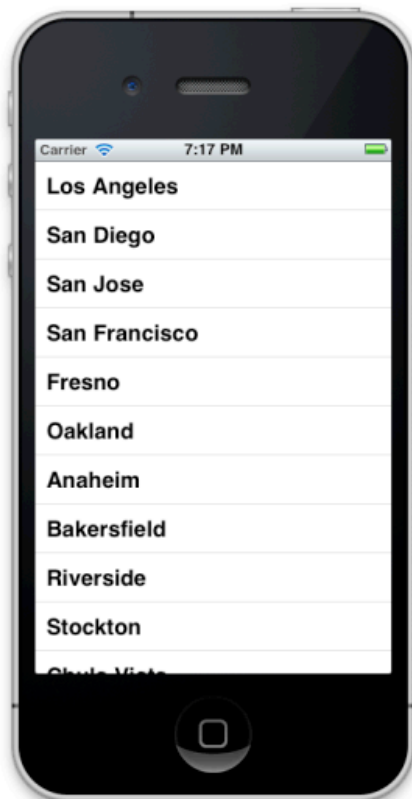
Visible
Content



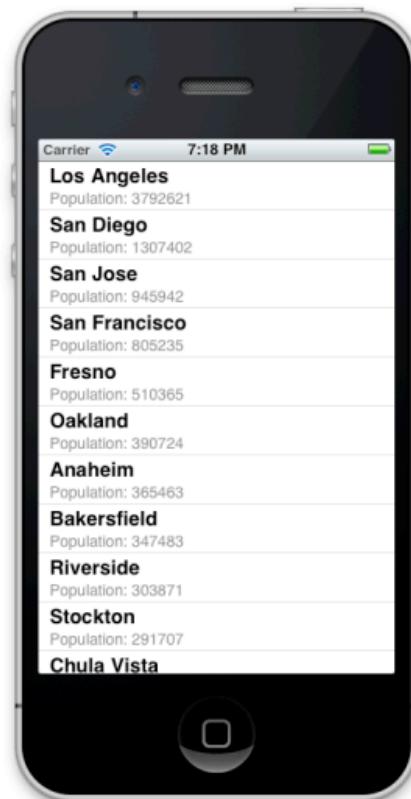
Clipped
Content

Table View Controller

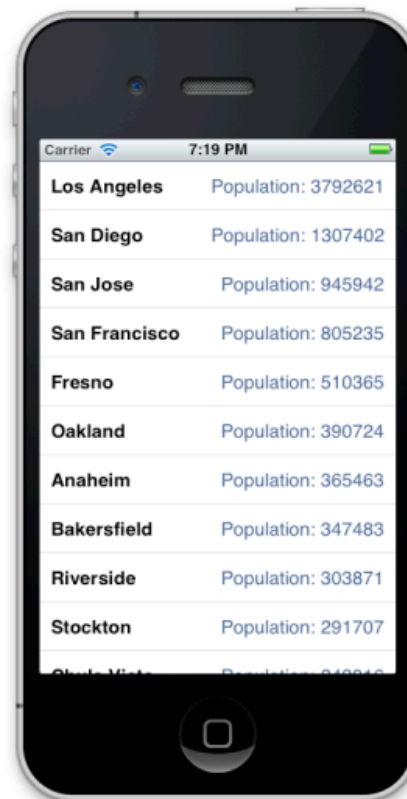
- Hlavní prvek UI iOS.
- Seznam buněk (TableViewCell).



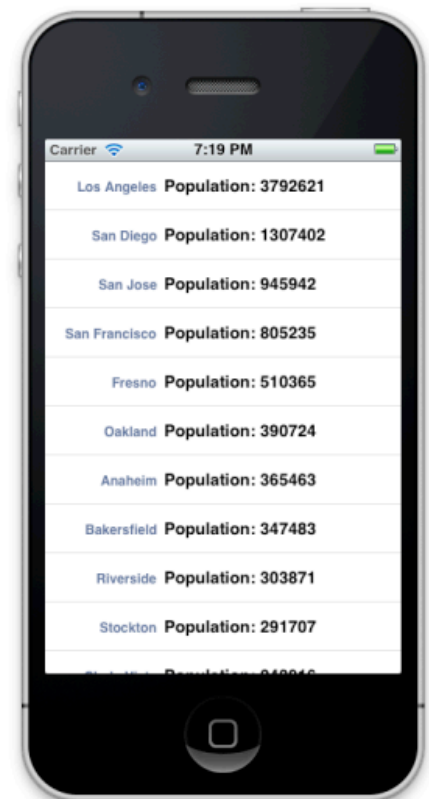
UITableViewCellStyleDefault



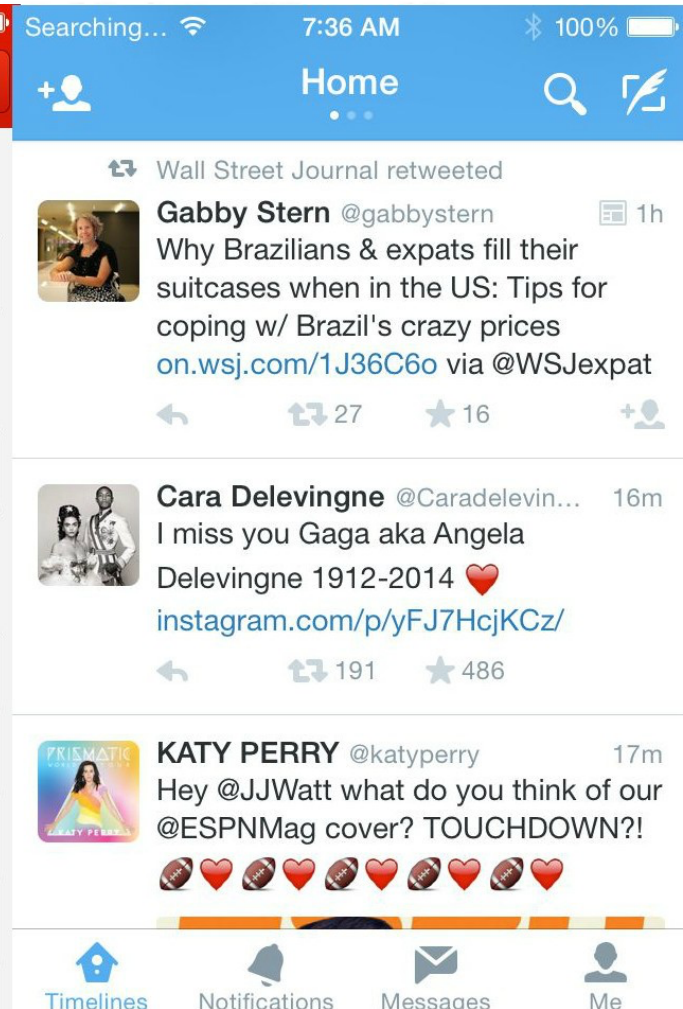
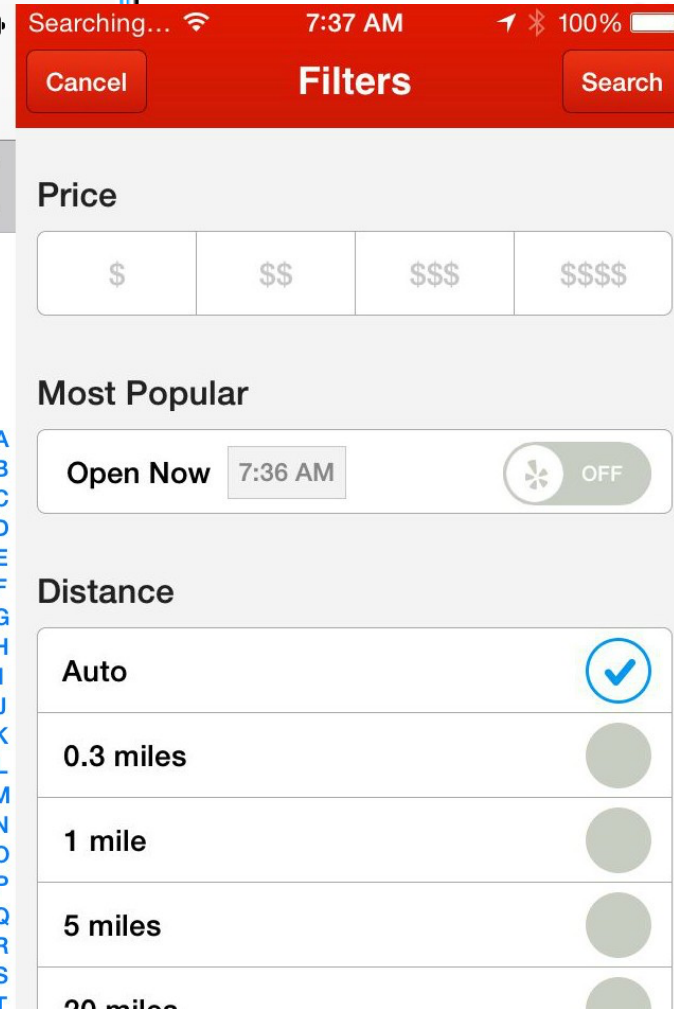
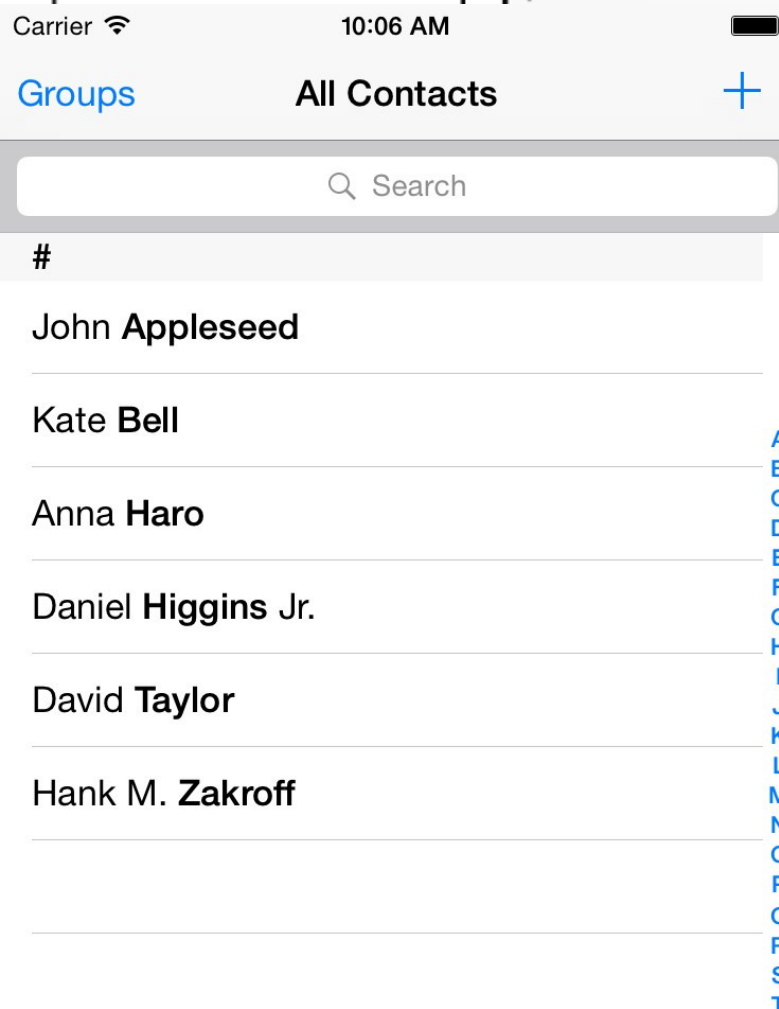
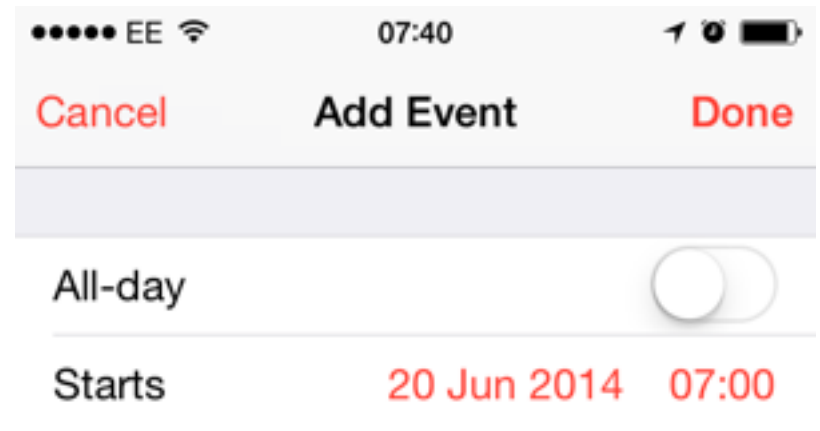
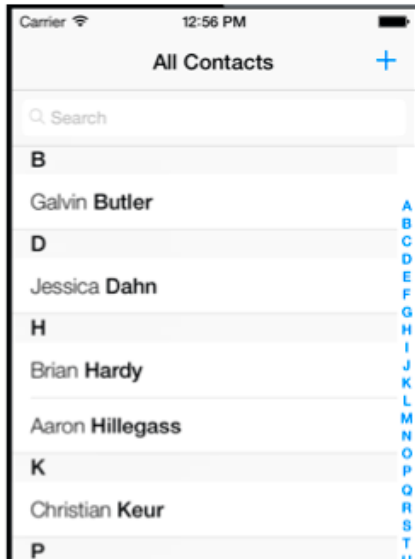
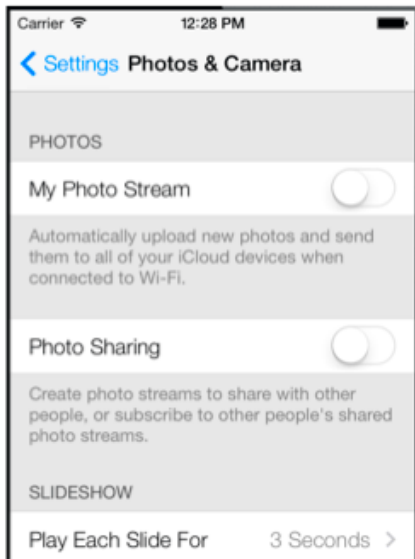
UITableViewCellStyleSubtitle

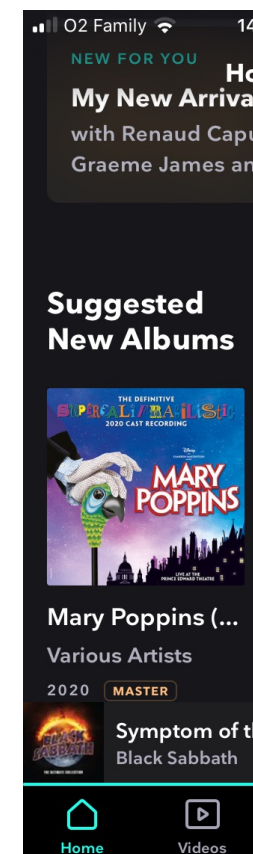
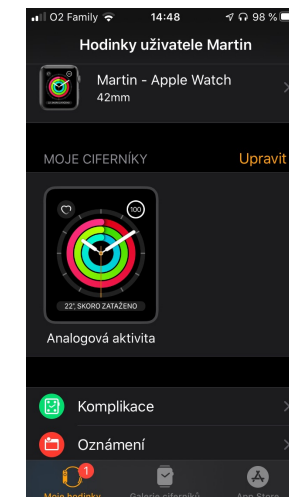
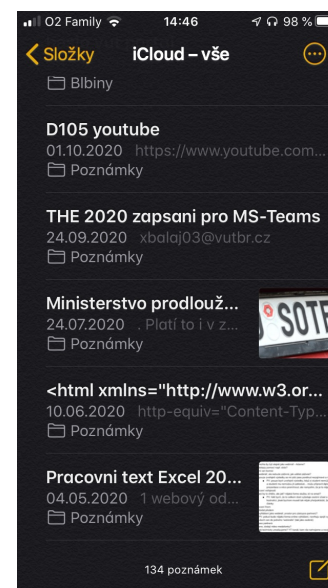
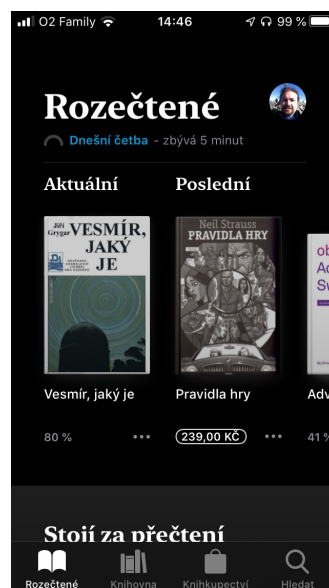
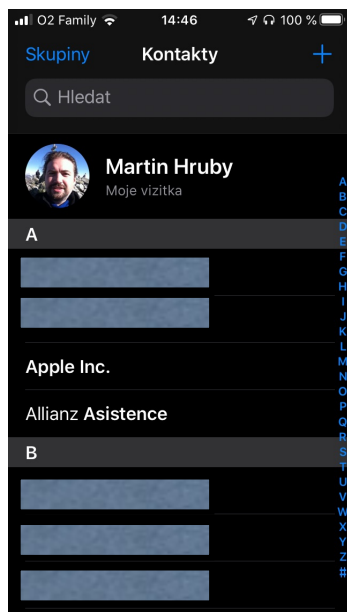
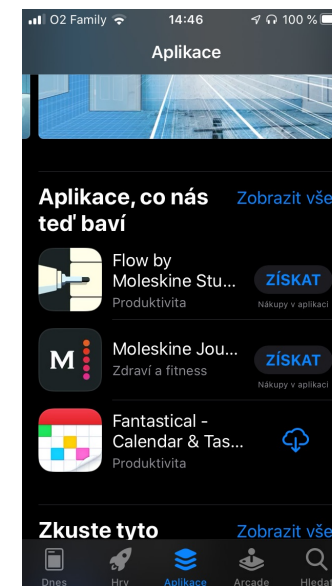
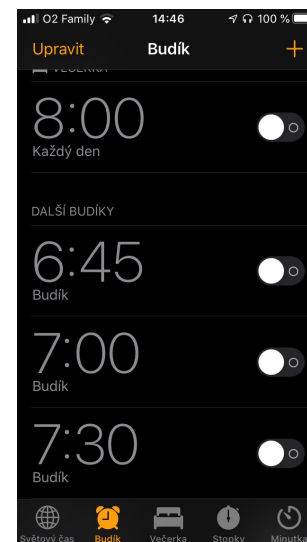
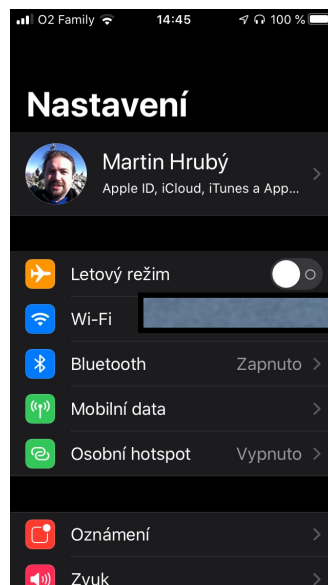
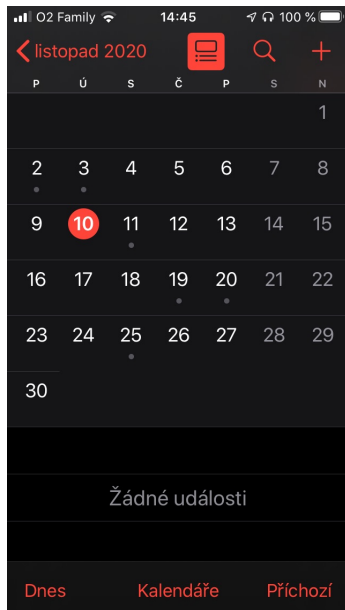
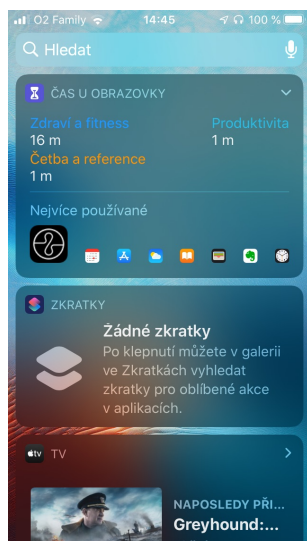


UITableViewCellStyleValue1



UITableViewCellStyleValue2

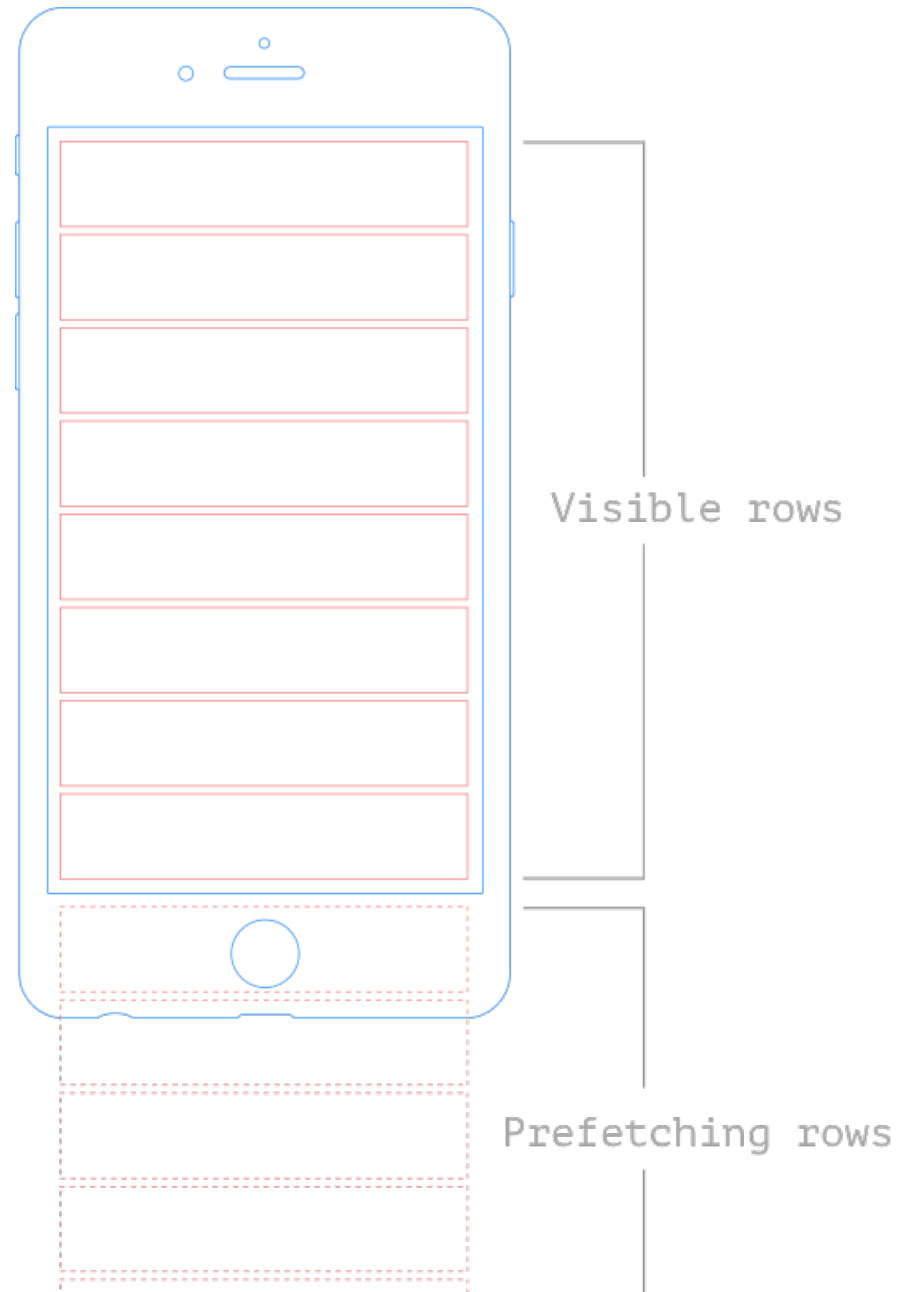




Tabulka, UITableView

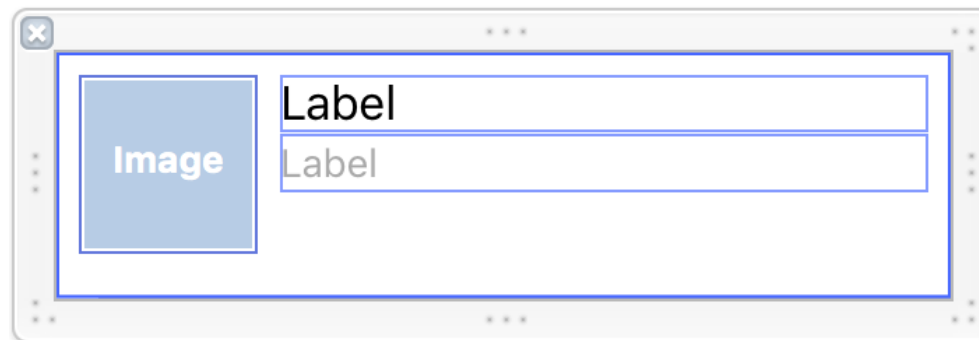
- Seznam buněk. *UITableViewCell*.
- Sekce. Řádky.
- Každá buňka tabulky smí být unikátní.
- Vertikálně-horizontální obsah:
 - UITableView, kde cell je *UICollectionView*.

Dynamika přístupu na dataSource



Implementace tabulky

- MVC.
- Model — UITableViewDataSource.
- View — UITableView.
- Controller — UITableViewController.



View a Controller

- Vše viditelné — **View** a jeho controller. (V, C).
- (Stromová) Hierarchie View:
 - subviews,
 - superview — geometrie — frame, bounds.
- Dynamika UI (Views) — vkládání/odebírání View z hierarchie (UIWindow).
 - `view1.addSubview(view2)`
 - `view2.removeFromSuperview()`

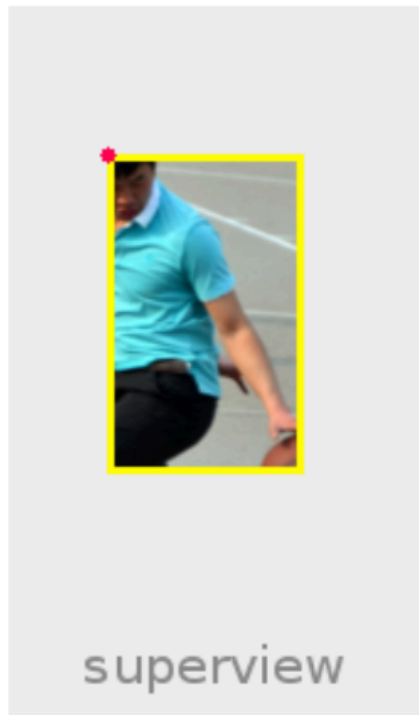
Frame

```
origin = (40, 60)  
width = 80  
height = 130
```

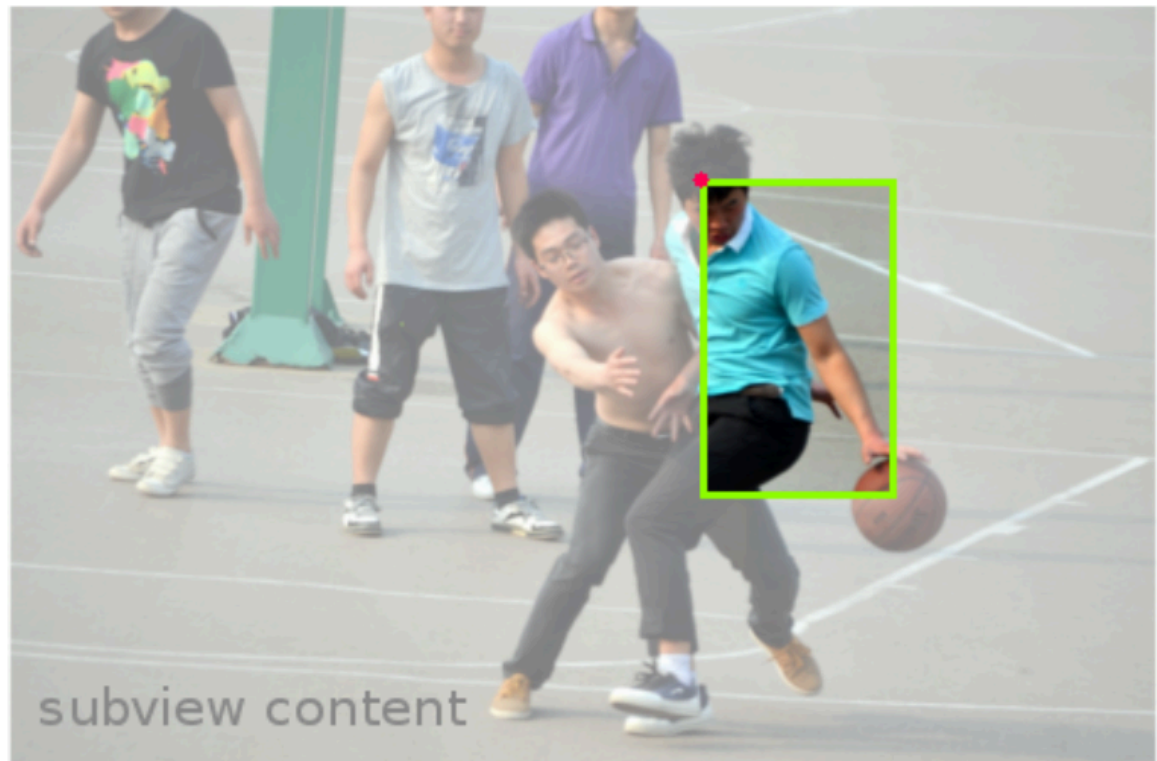
Bounds

```
origin = (280, 70)  
width = 80  
height = 130
```

Frame



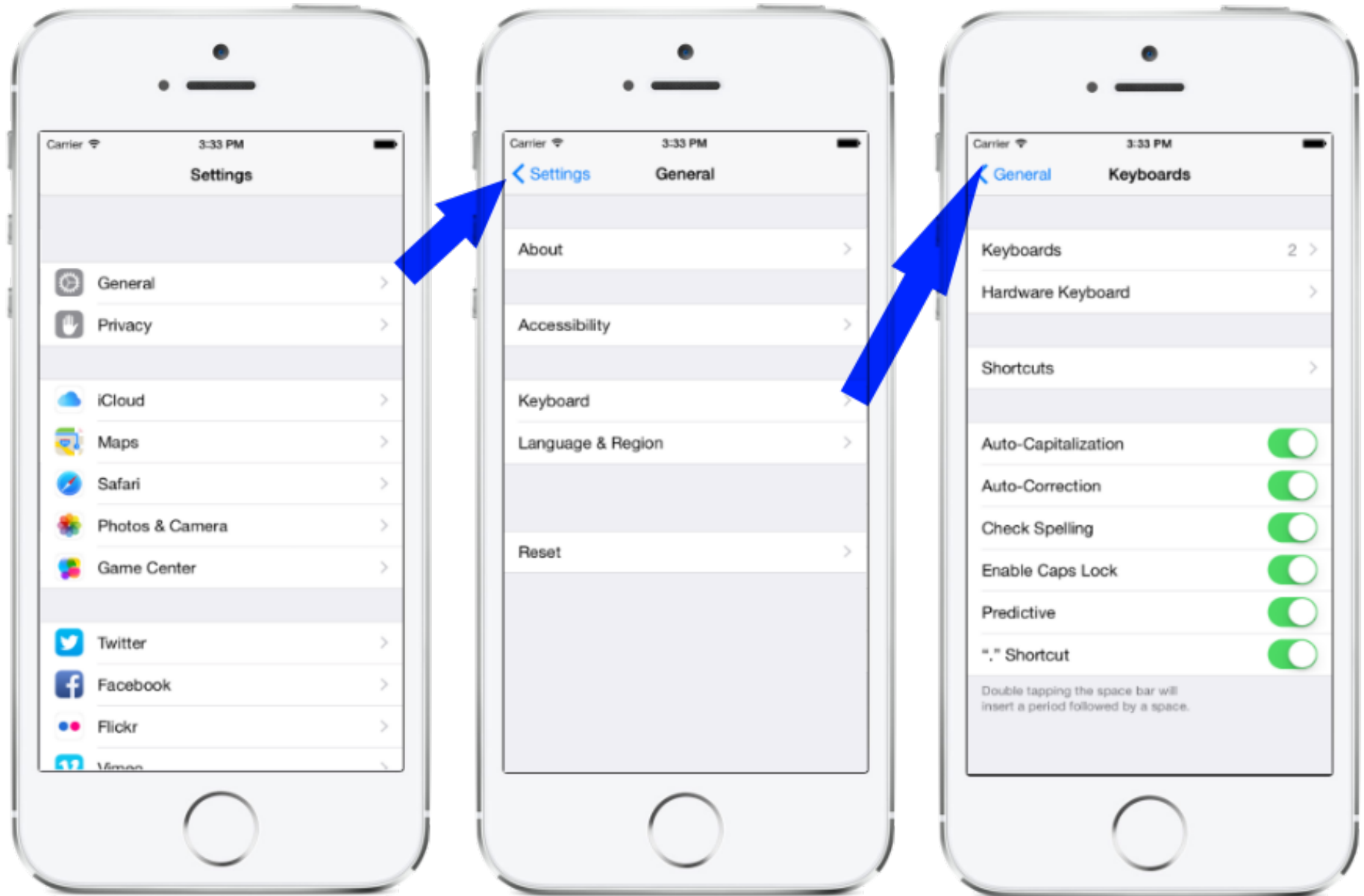
Bounds



Zasazení tabulky do rámce

- Abstraktní ViewController.
- UINavigationController.
 - zásobníků zanořených V-C, rootViewController, horní lišta.
- UITabBarController.
 - přepínání mezi V-C.
- Master-Detail (pouze pro iPad).

Navigation VC



2:45



ROOT

ahoj >

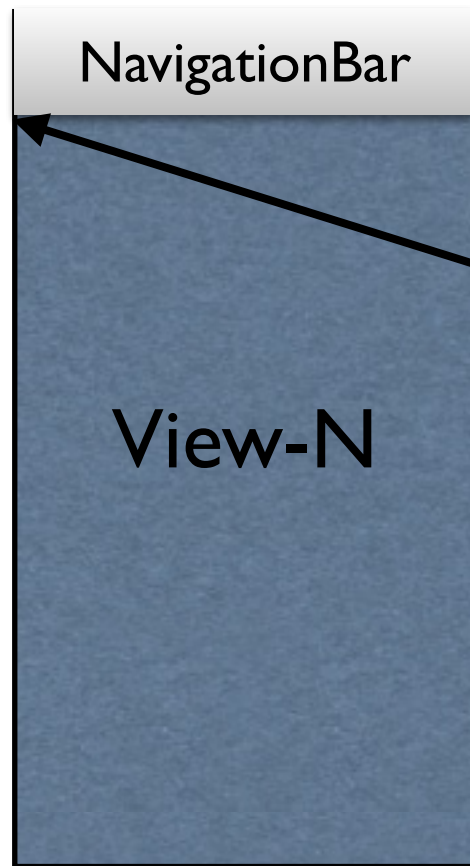
jak >

se >

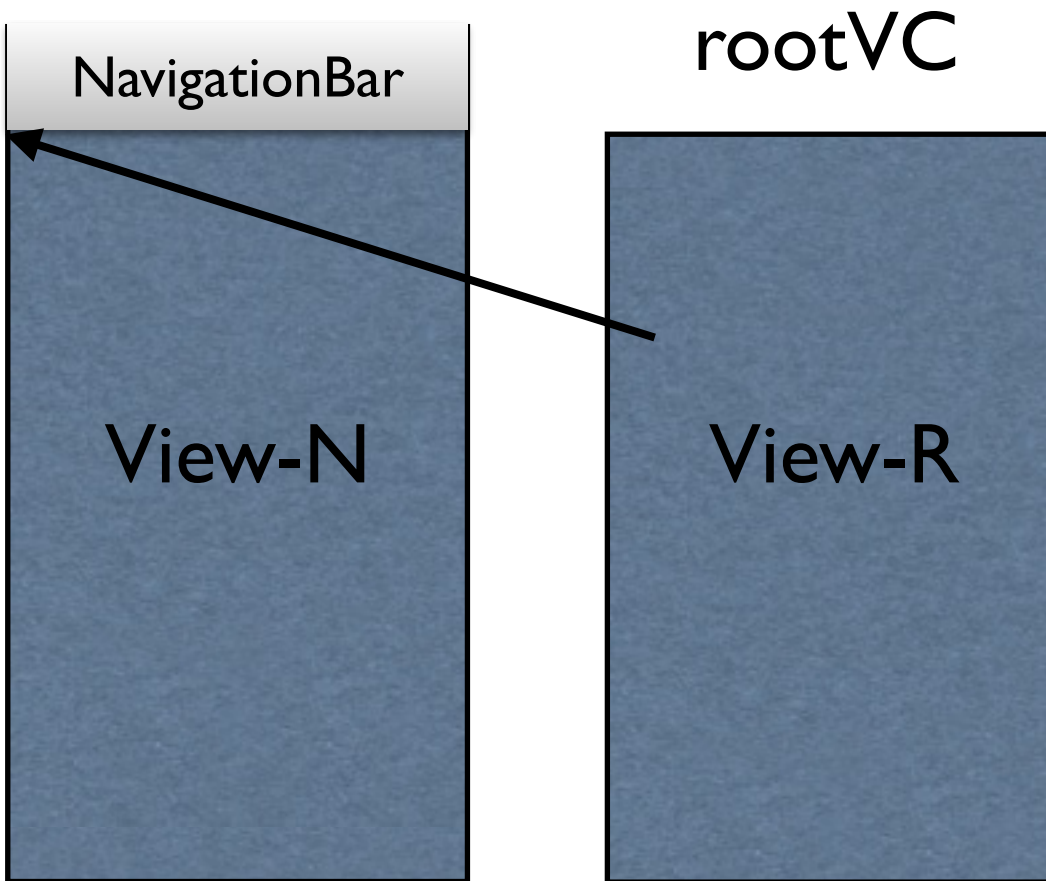
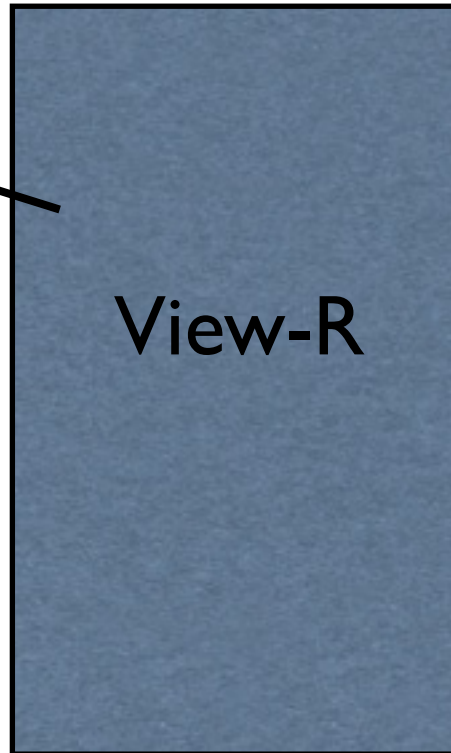
mas >

NavigationVC

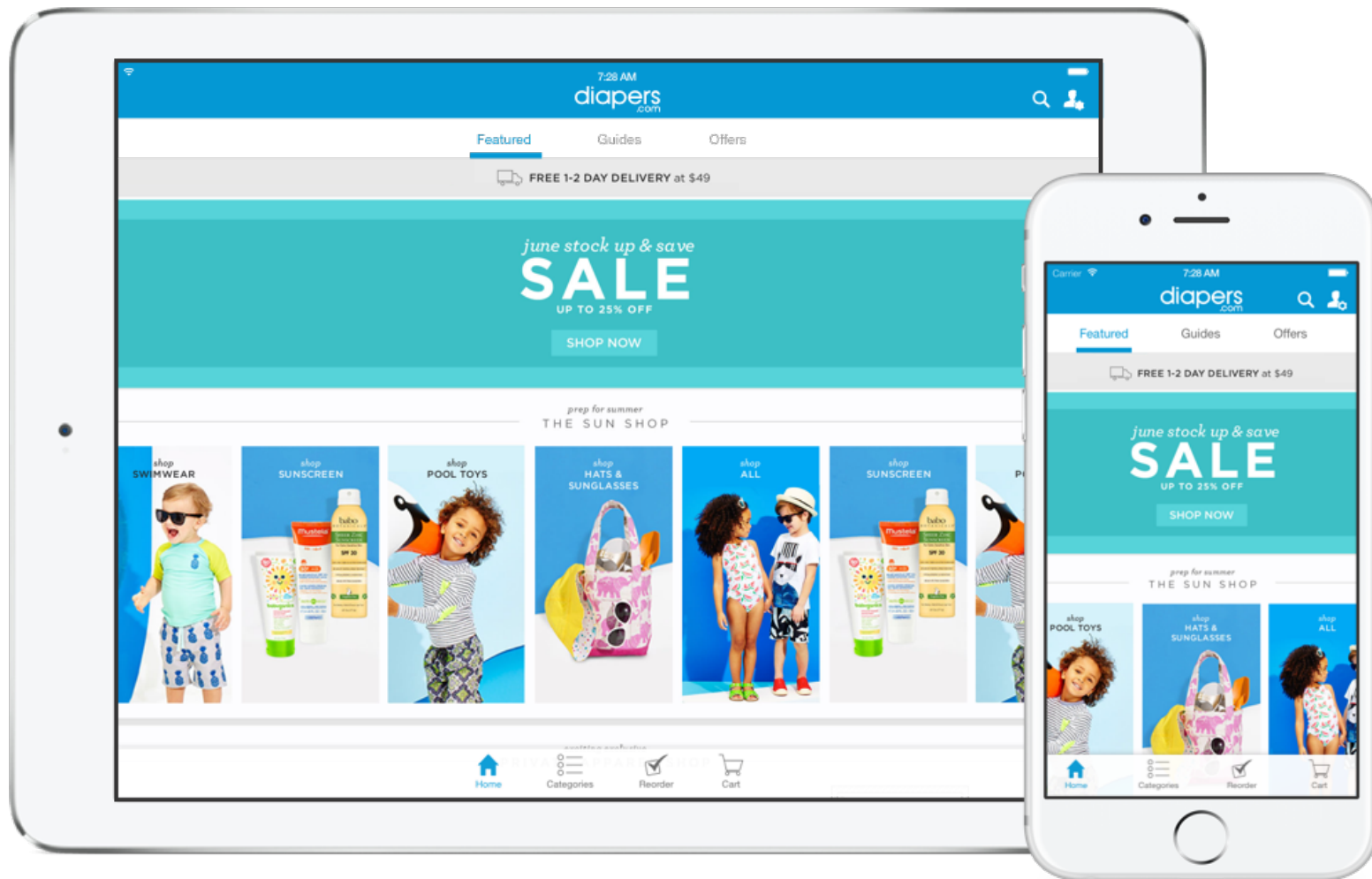
NavigationVC



rootVC



TabBar VC



3:02



ROOT

seznam >

cehosi >

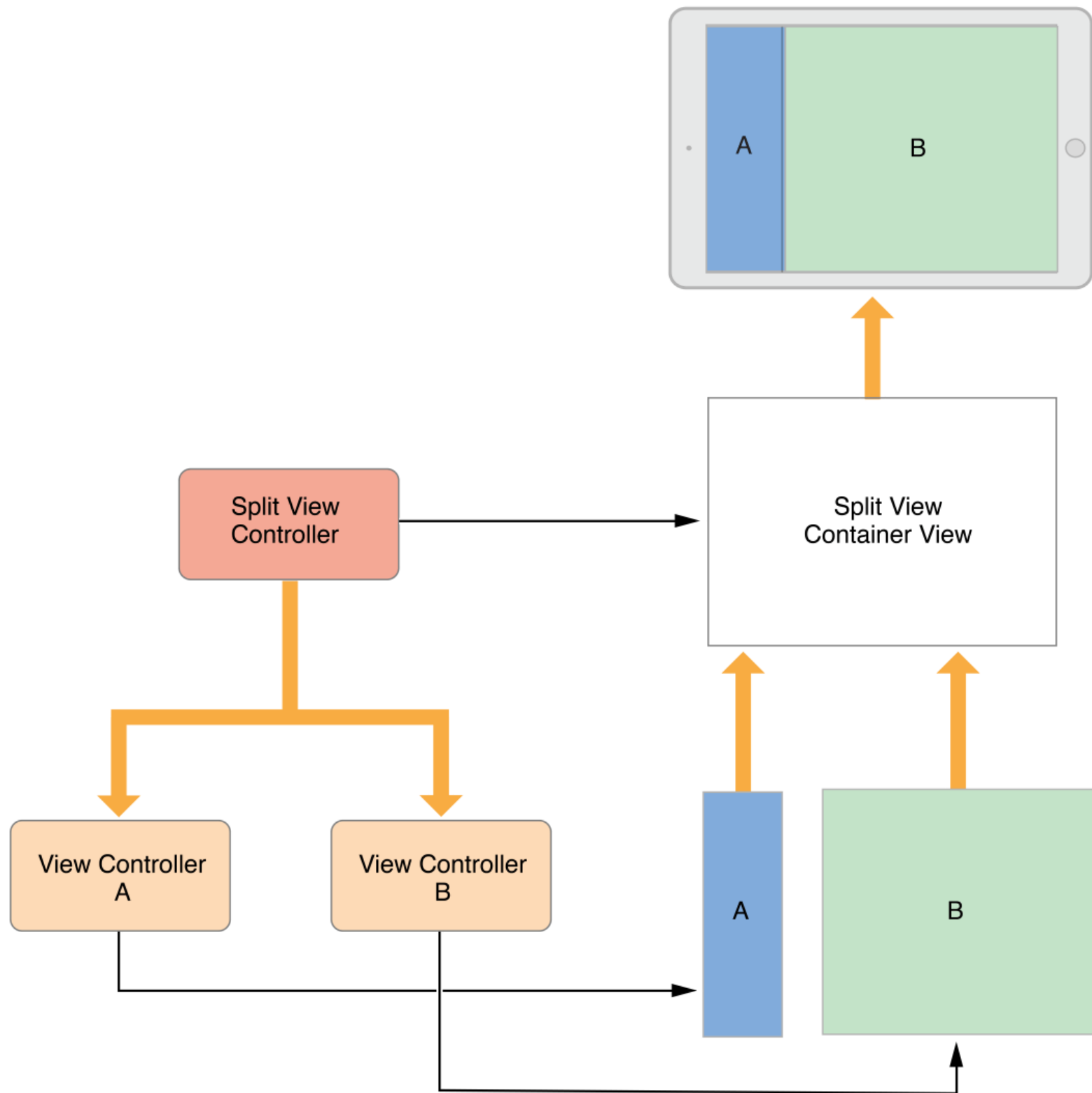
asi >

Prvni

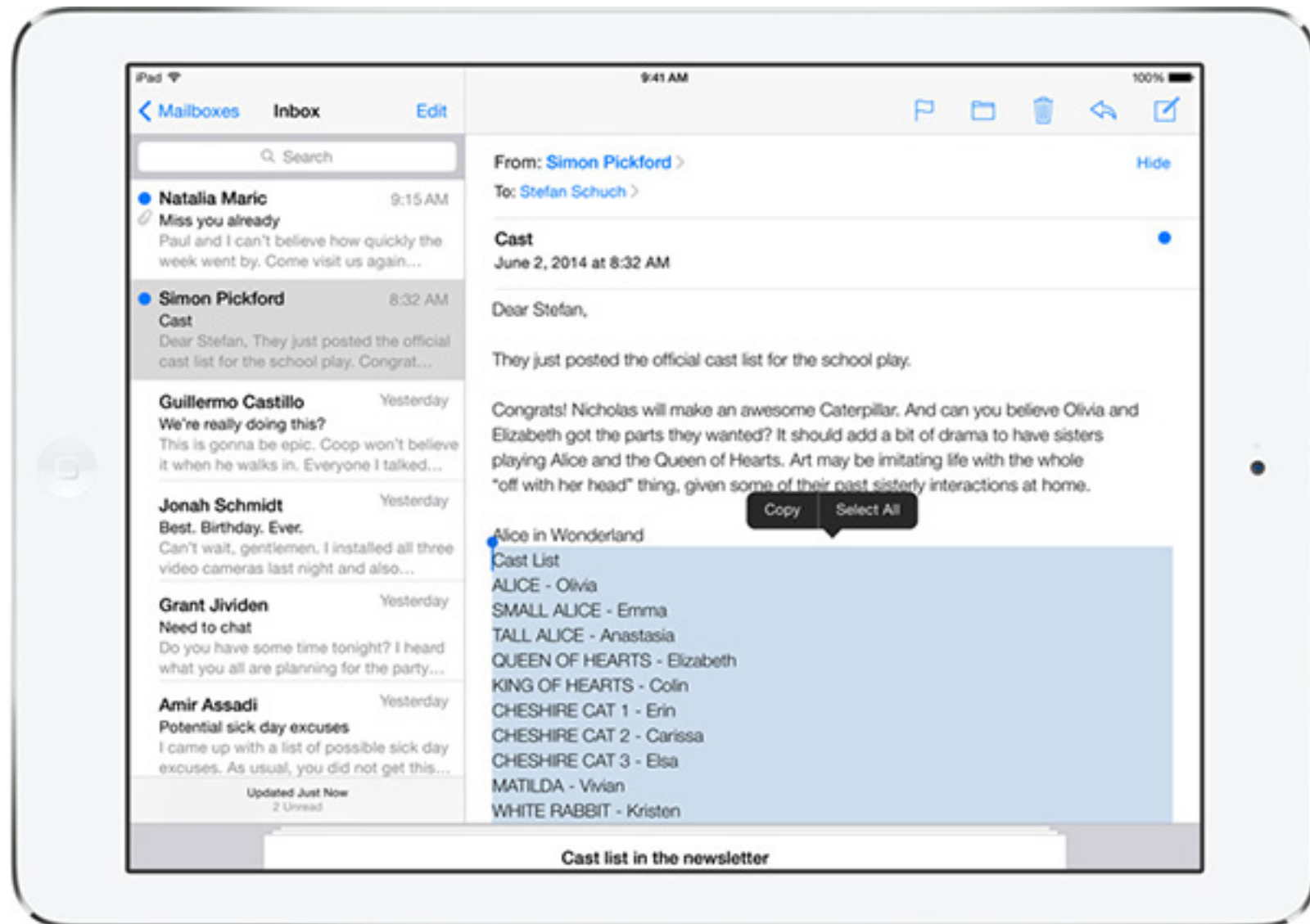
Druhy

Treti

Ctvrty

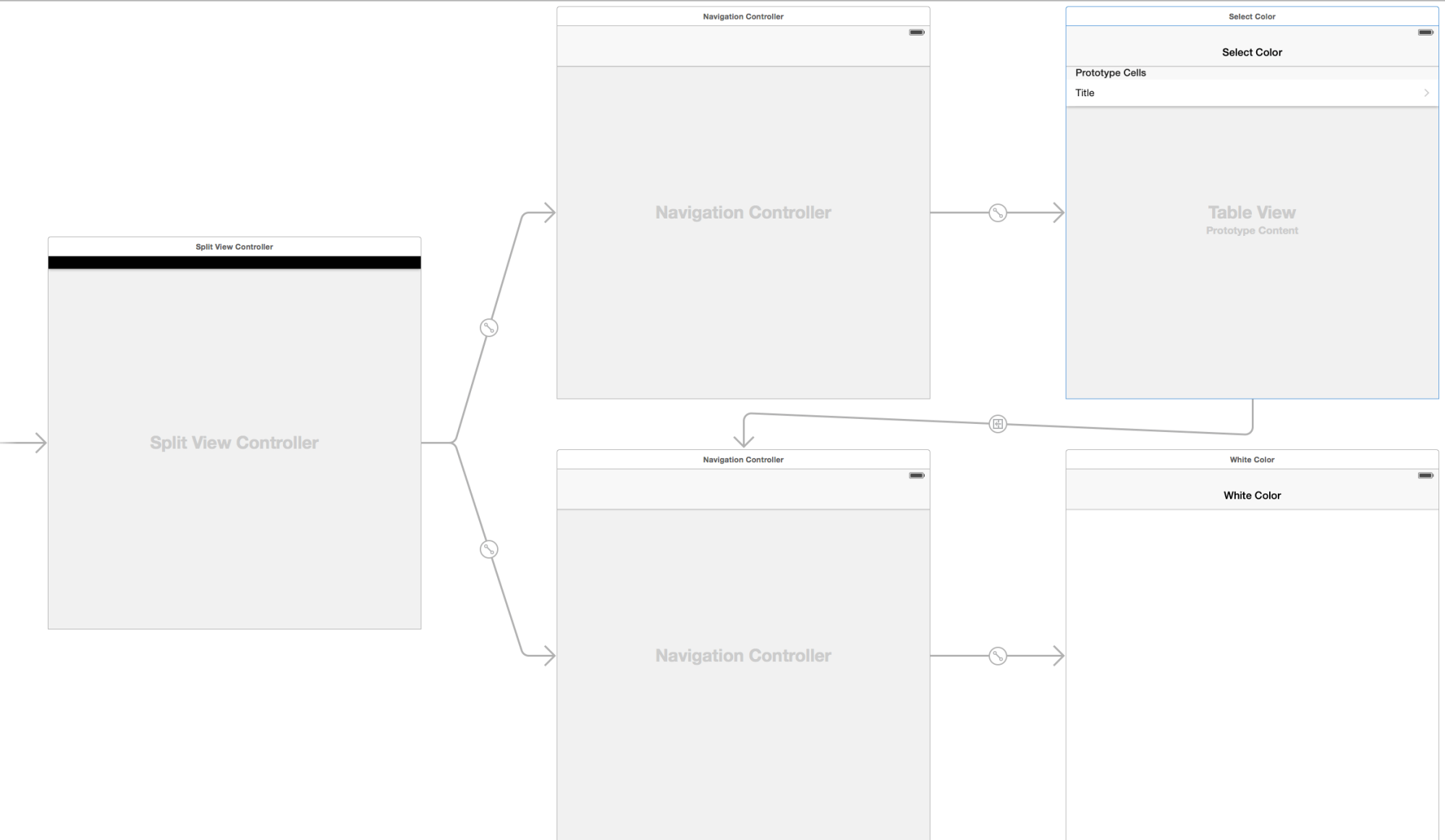


Mail app na iPadu



Architektura VC v M-D

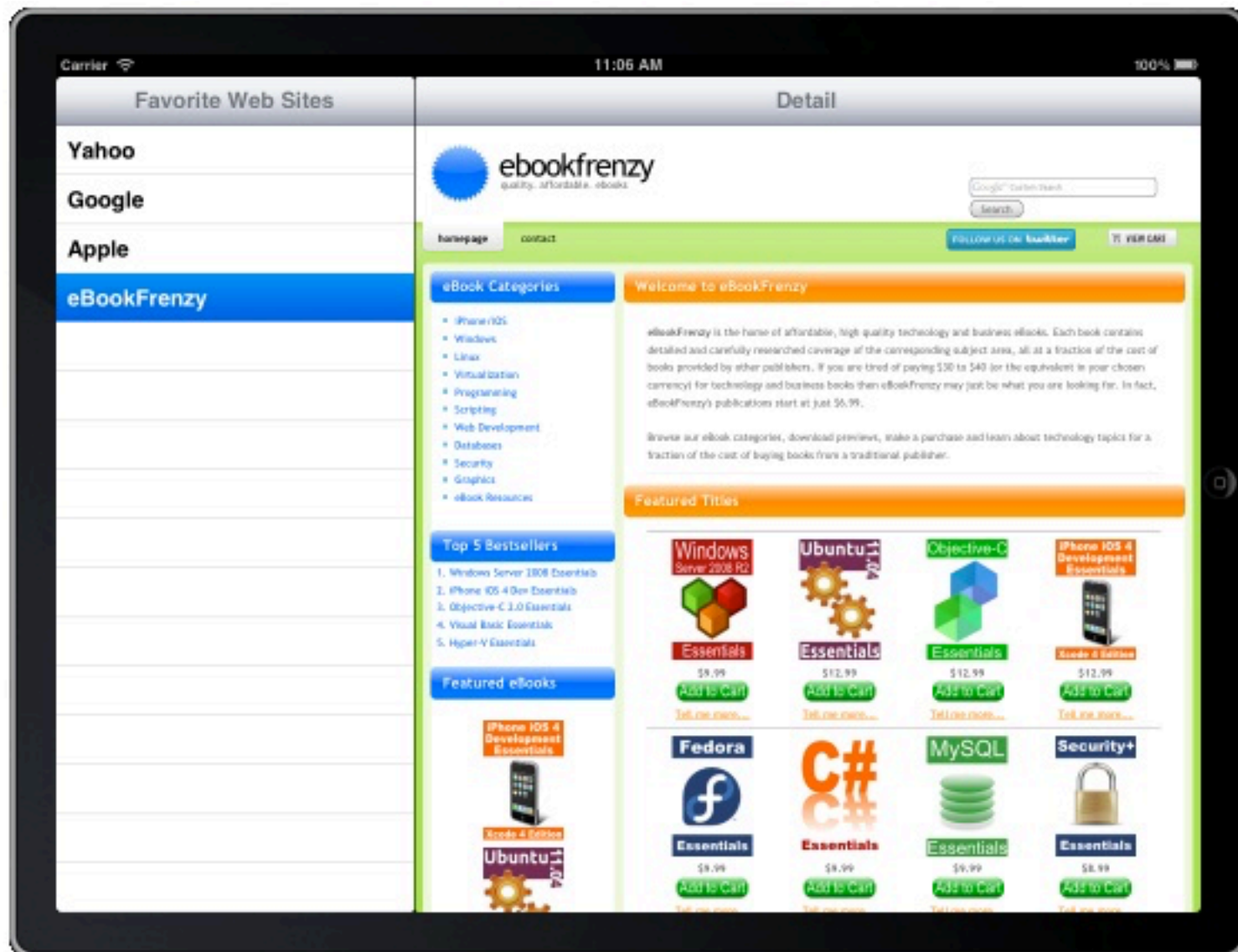
< > | SplitViewControllerDemo > SplitViewControllerDemo > Main.storyboard > Main.storyboard (Base) > Select Color Scene > Select Color



wAny hAny

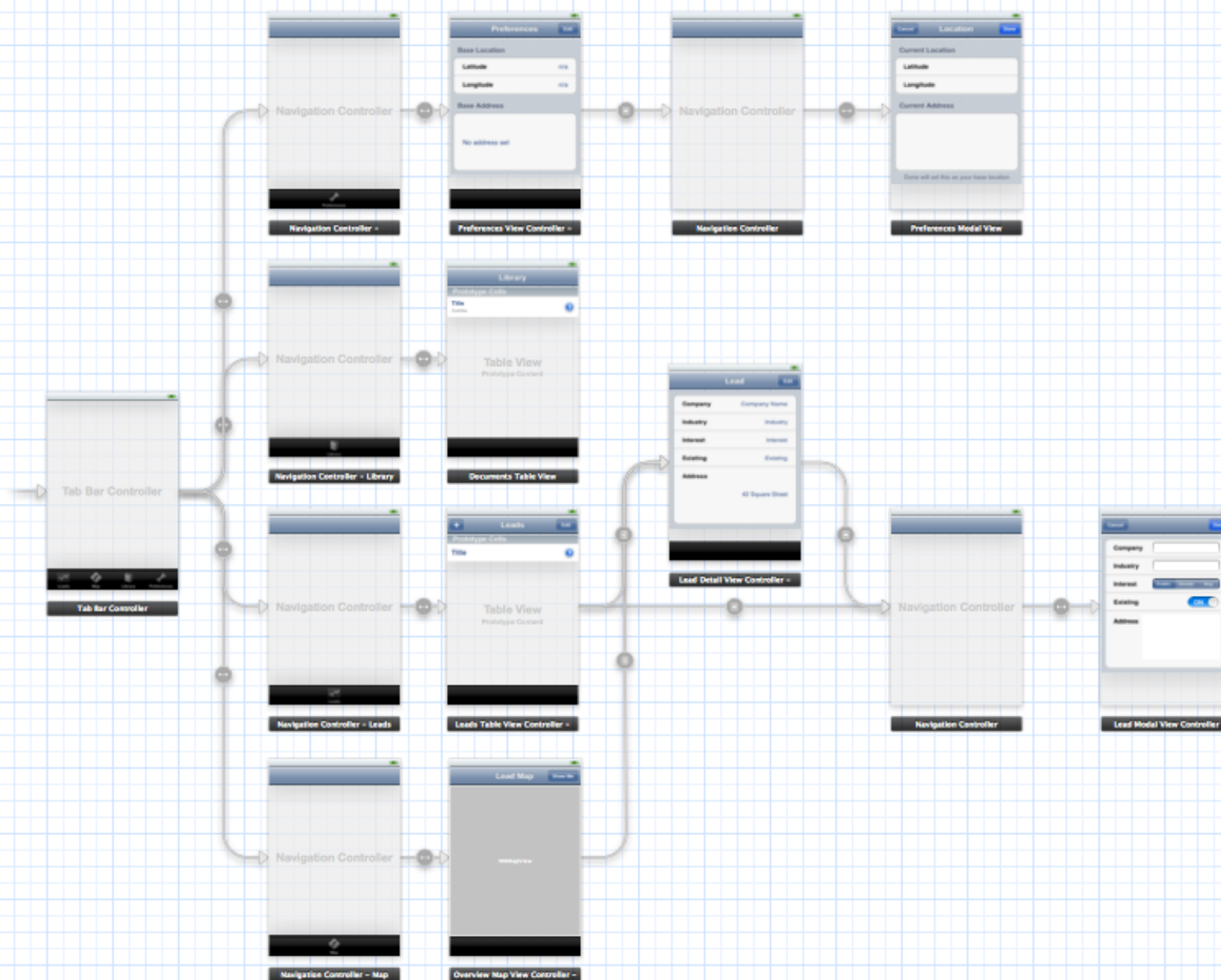
메서드

M-D v režimu Landscape



M-D v režimu Portrait

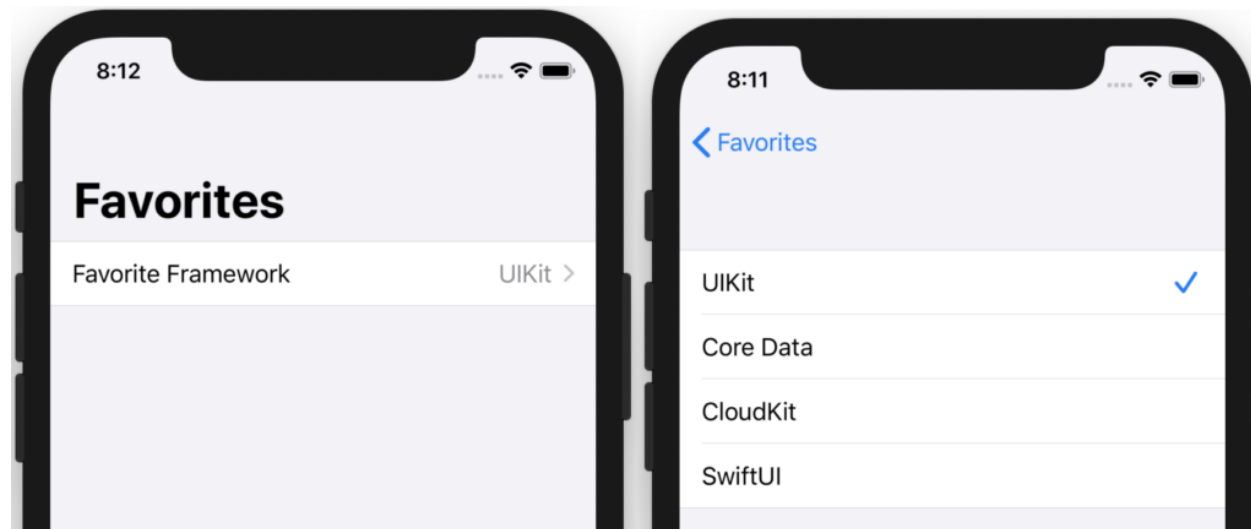




Elementární Views

- UIView.
- UILabel. UITextField. UISwitch. UIButton.
- UITableViewCell.
- Pickers — menu, date picker, ...

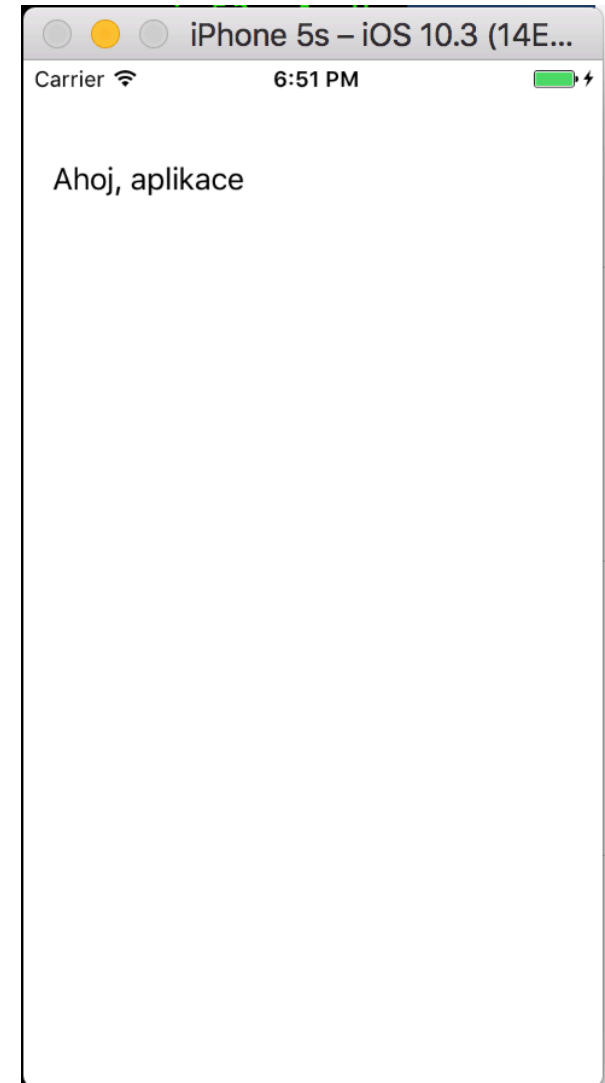
START TIME			END TIME		
1	50		4	50	
2	55	AM	5	55	AM
3	00	PM	6	00	PM
4	05		7	05	
5	10		8	10	
6	15		9	15	



SwiftUI	UIKit	AppKit	Introspect
List	UITableView	NSTableView	<code>.introspectTableView()</code>
ScrollView	UIScrollView	NSScrollView	<code>.introspectScrollView()</code>
NavigationView	UINavigationController	N/A	<code>.introspectNavigationController()</code>
<i>Any embedded view</i>	UIViewController	N/A	<code>.introspectViewController()</code>
TabView	UITabBarController	N/A	<code>.introspectTabBarController()</code>
TextField	UITextField	NSTextField	<code>.introspectTextField()</code>
Toggle	UISwitch	N/A	<code>.introspectSwitch()</code>
Slider	UISlider	NSSlider	<code>.introspectSlider()</code>
Stepper	UIStepper	NSStepper	<code>.introspectStepper()</code>
DatePicker	UIDatePicker	NSDatePicker	<code>.introspectDatePicker()</code>
Picker (SegmentedPickerStyle)	UISegmentedControl	NSSegmentedControl	<code>.introspectSegmentedControl()</code>

M-V-C, Základní demo

```
class MyModel {  
    var content : String = "Ahoj, aplikace"  
}  
  
class ViewController: UIViewController {  
    @IBOutlet var lei : UILabel?  
  
    let mymodel = MyModel()  
  
    override func viewDidLoad() {  
        super.viewDidLoad()  
  
        //  
        lei?.text = mymodel.content  
    }  
}
```



Shrnutí UIKit

- MVC koncepce provázela AppKit / UIKit několik dekád.
 - MVC macOS aplikací ještě rozsáhlejší.
- Ukecanost kódu, produktivita.
 - Přechod Objective-C -> Swift.
 - Přechod MVC / Storyboard -> SwiftUI.

SwiftUI

Fakta

- SwiftUI ohlášeno na WWDC 2019. Wow-effect.
- Určeno pro iOS 13, macOS Catalina.
- Stále "testing", tj. **SwiftUI není hotovo!**
- Aktuálně lze vývoj v XCode provádět:
 - ve Storyboard stylu,
 - ve SwiftUI stylu.
- Apple označuje **SwiftUI za budoucnost.**

Smysl SwiftUI

- Sjednocuje styl programování pro macOS, iOS...
- MVC -> MVVM.
- Deklarativní popis UI.
 - Model+procesy — knihovna Combine (reaktivní programování).
- Kód, který je ve Swiftu, ale moc tak nevypadá :)

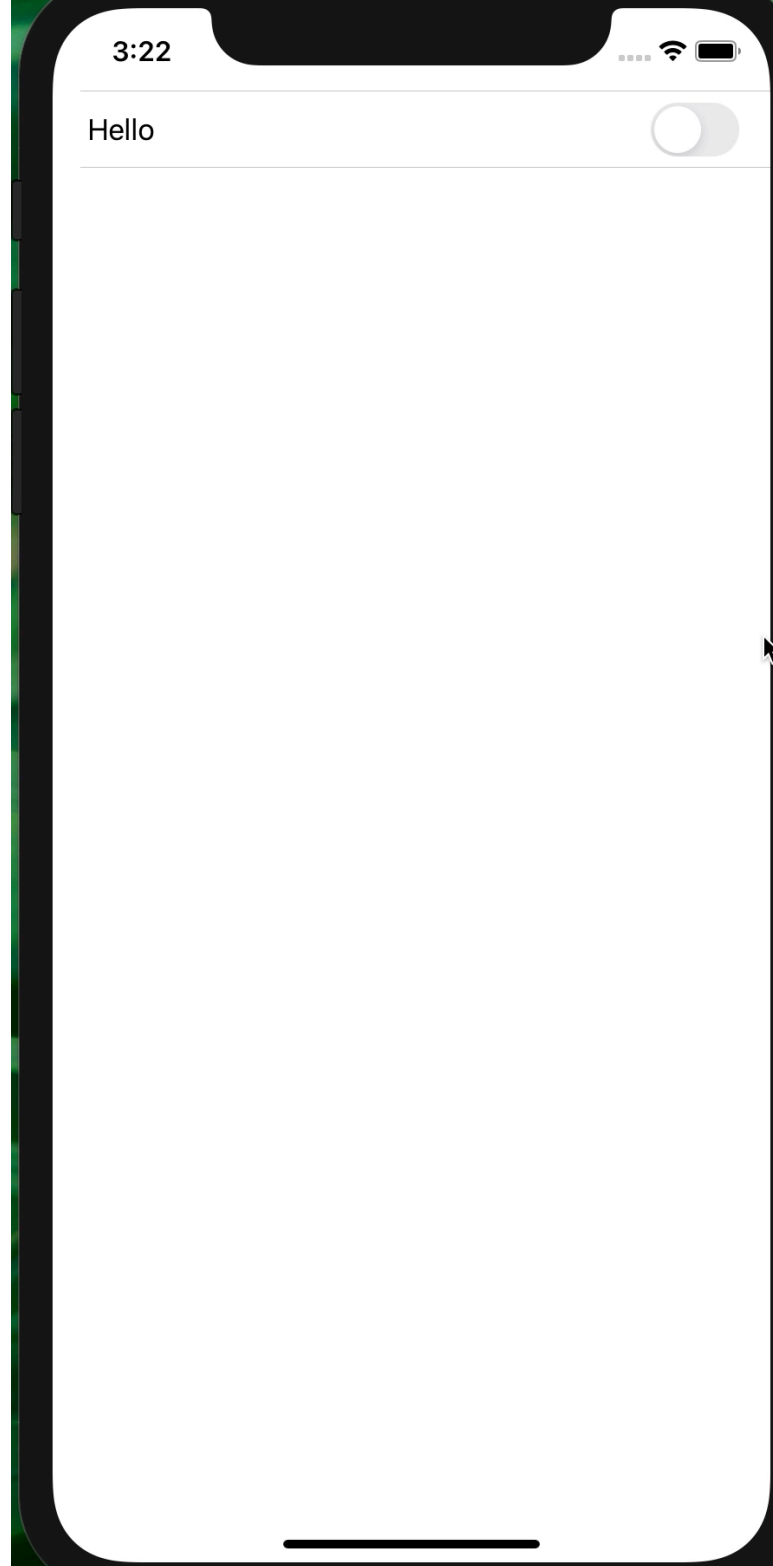
M-V-VM

- M — jako v MVC / UIKit.
- V — zvnějšku navázatelný vnitřní stav.
- VM — datově vyjádřený explicitní vnitřní stav UI (obrazovky).

Deklarativní popis UI

- @State, @Binding proměnné
 - property wrapper
- var body: some View {}
- Změní se VM, vyvolá se rekonstrukce V

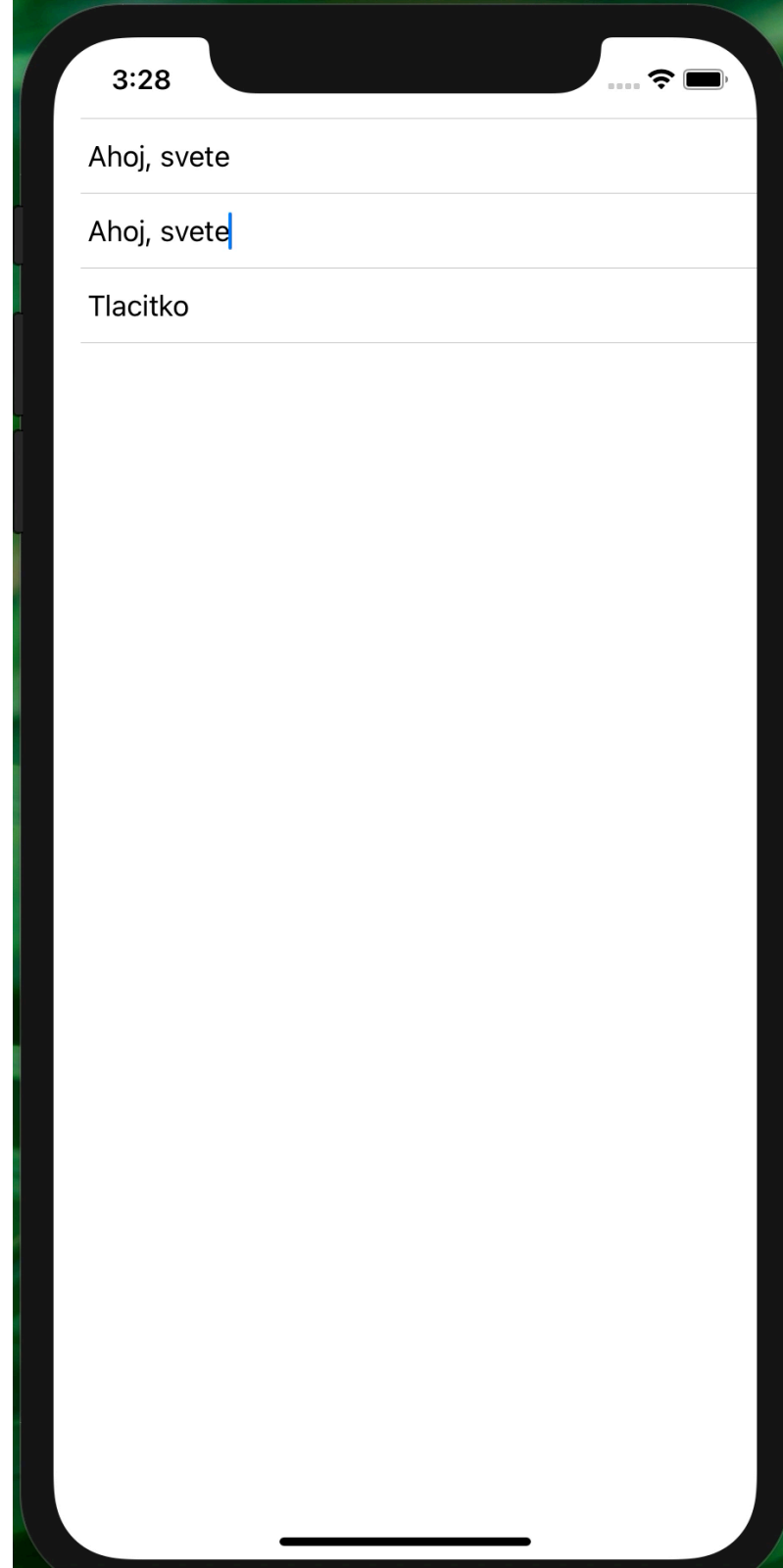
```
// View pro "obrazovku"
struct ContentView: View {
    // VM – vnitrni stav
    @State var isON = false
    // konstruktor pro View
    var body: some View {
        //
        List {
            // switcher – navazuje VM
            Toggle("Hello", isOn: $isON)
            // dynamika
            if isON == true {
                //
                Text("Je true...")
            }
        }
    }
}
```



Demo

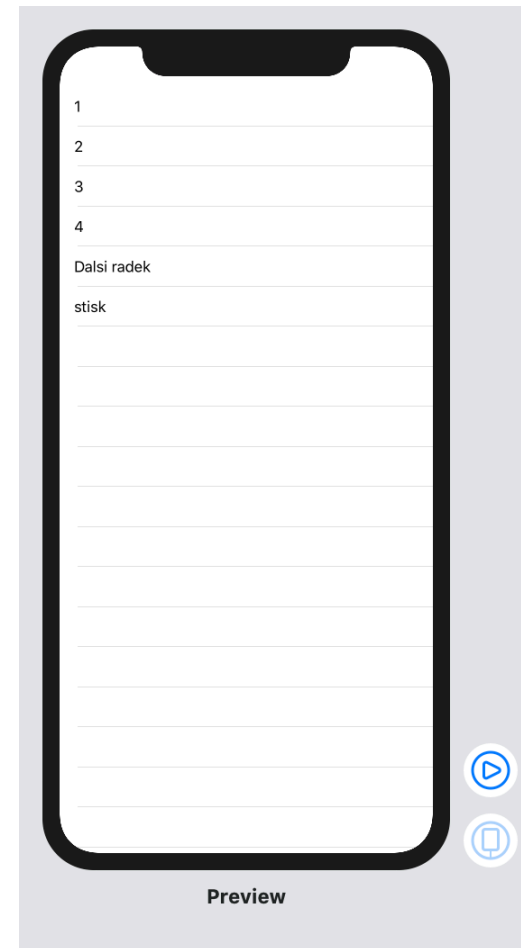
```
// je to struktura! Dusledky...
struct ContentView: View {
    // vnitrni stav view
    @State var obsah = "Ahoj, svete"
    // podoba view
    var body: some View {
        //
        List {
            Text(obsah)
            TextField("zadej",text: $obsah)

            Button(action: { self.akce() }) {
                Text("Tlacitko")
            }
        }
    }
    // neni "mutating"...proc
    func akce() { obsah = "... " }
}
```



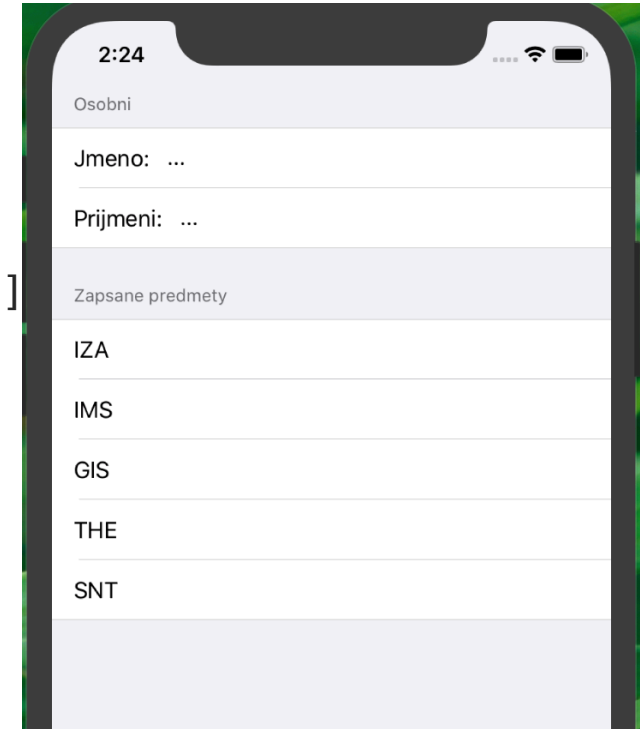
Demo: tabulka #1

```
// je to struktura! Dusledky...
struct ContentView: View {
    // podoba view
    var body: some View {
        // chci tabulku
        List {
            // 4 radky obsahu
            ForEach([1,2,3,4], id:\.self) { i in Text("\(i)") }
            // 5. radek
            Text("Dalsi radek")
            // 6. radek
            Button(action: {}, label: {Text("stisk")})
        }
    }
}
```



Demo: tabulka #2

```
struct ContentView: View {  
    //  
    @State var jmeno = "..."  
    @State var prijmeni = "..."  
    @State var predmety = ["IZA", "IMS", "GIS", "THE", "SNT"]  
    // podoba view  
    var body: some View {  
        // formular (UITableView, style Grouped)  
        Form {  
            //  
            Section(header: Text("Osobni")) {  
                //  
                List {  
                    //  
                    HStack { Text("Jmeno: "); TextField("jmeno", text: $jmeno)}  
                    HStack { Text("Prijmeni: "); TextField("jmeno", text: $prijmeni)}  
                }  
            }  
            //  
            Section(header: Text("Zapsane predmety")) {  
                //  
                List(predmety, id: \.self) { i in Text(i) }  
            }  
        }  
    }  
}
```



Komunikace mezi "obrazovkami"

- MVC — jeden VC referencuje druhý, komunikace.
- SwiftUI — nelze se navzájem referencovat.
- Komunikace probíhá přes zápis do VM, tj přes data.

Komplexní demo...

```
// Model v aplikaci. Je zapouzdreno do tridy
class MojeData: ObservableObject {
    // resp, do singletonu
    static let shared = MojeData()
    // datovy obsah, musi/nemusi byt @Published
    @Published var poznamky = [String]()
    // ...
    func add(_ s: String) { poznamky.append(s) }
}
```



```

struct ContentView: View {
    // stavam se observerem celeho objektu
    @ObservedObject var model = MojeData.shared
    // stav V rikajici, zda-li je "sheet" viditelny
    // !!!
    @State var sheetON: Bool = false
    //
    var body: some View {
        //
        List {
            // z pole generuju tabulku
            ForEach(model.poznamky, id:\.self) { poz in
                Text(poz)
            }
            // tlacitko, sheetON:false -> sheetON:true
            Button(action: { self.sheetON.toggle() }) { Text("volej sheet")}
        }
        // deklaruji (!!!), ze ma byt viditelny sheet, pokud
        .sheet(isPresented: $sheetON) {
            // vytvor sheet a propoj
            Detail(sheetON: self.$sheetON, mojeData: self.model)
        }
    }
}

```

```

// modálne prezentovaný View
struct Detail : View {
    // sem edituj nejaky nazev/text poznamky
    @State var textik: String = "neco zadej"
    // binding na ContentView::sheetON napr.
    @Binding var sheetON: Bool
    // vec vkusu: 1) bud dostanu ref, 2) MojeData.shared
    let mojeData: MojeData
    //
    var body: some View {
        //
        List {
            TextField("", text: $textik);
            // @escaping
            // modifikuju Model, modifikuju ViewModel
            Button(action: {
                self.mojeData.add(self.textik)
                self.sheetON.toggle() }
            ) { Text("Pridej") }
        }
    }
}

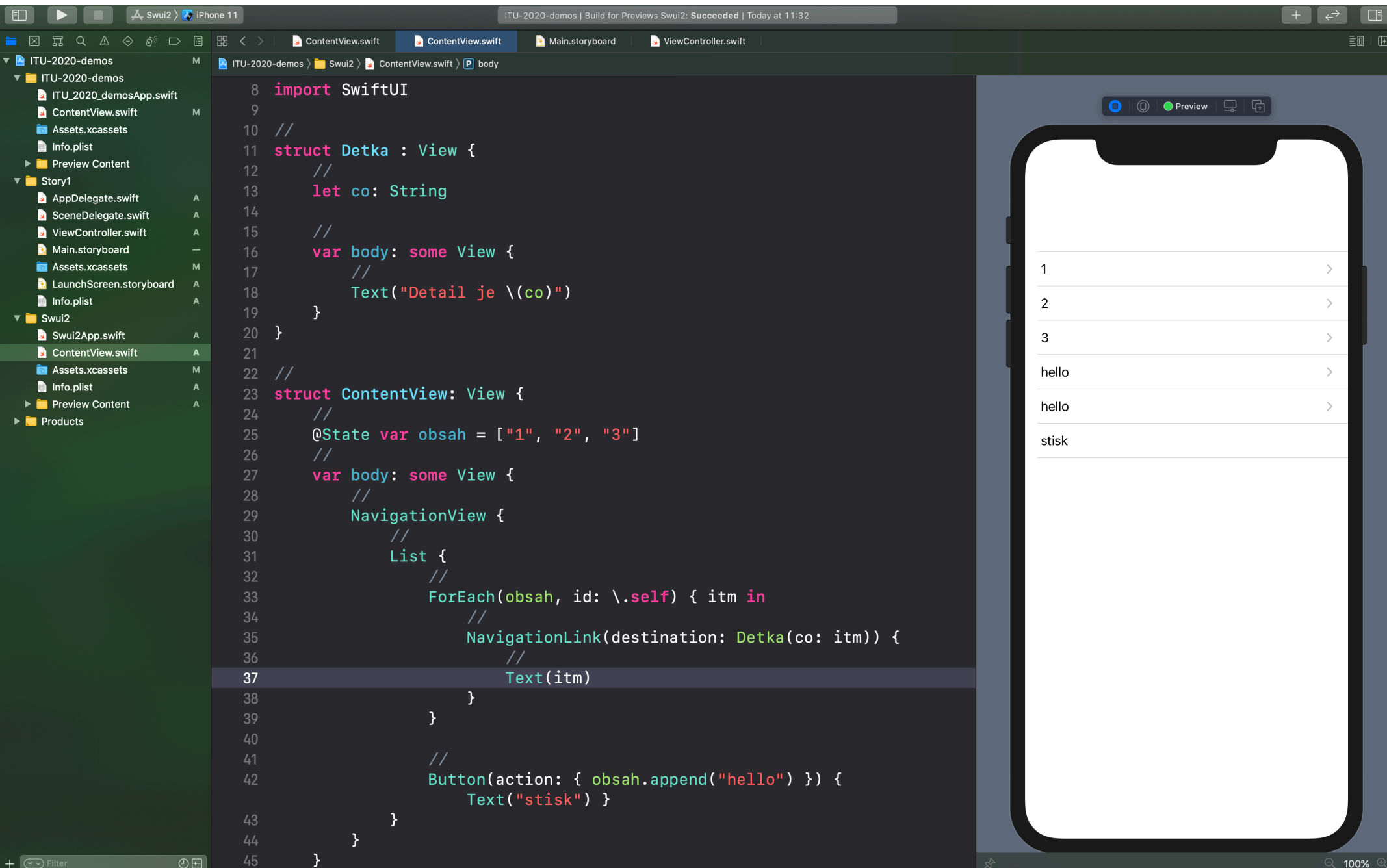
```

Výsledek

11:05

volej sheet

A mobile application interface for a volleyball sheet. The screen is white with a black status bar at the top showing the time 11:05, signal strength, and battery level. Below the status bar, the text 'volej sheet' is displayed. The main area of the screen is filled with horizontal lines, suggesting a list or table. A mouse cursor is visible on one of the lines. The app is framed by a black border with green accents at the corners.



Závěr

- Koncept programování iOS apps je snadný.
- Storyboard -> SwiftUI.
- Náročné může být jenom programování Modelu
 - Dokumenty, CoreData, CloudKit, synchronizace.