

Uživatelské rozhraní v iOS

Martin Hrubý
FIT VUT v Brně

Úvod

- Historie mobilních zařízení firmy Apple.
 - Apple začínal jako výrobce počítačů (1976).
 - Dnes spotřební elektronika: *iPhone*, iPad (Pod), Apple TV, Apple Watch. *Jsou to však stále programovatelné počítače!*
 - Hlavním problémem bylo vždy *vymyslet koncept uživatelského rozhraní*.
- Koncept UI ovládaného gesty (dotyk, pohyb).
 - Hlasové ovládání — asistentka Siri. Diktování, ovládání, AI.
 - iPhone X — rozpoznání obličeje. Další rozvoj?





Počátky mobilních počítačů

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



Počátky multi-touch

- iPod, 2001. Projekt Newton.
- Tablet — ovládání tužkou nebo multi-touch.
- Pokusy v Microsoftu. Spolupráce s Motorola (ROKR)

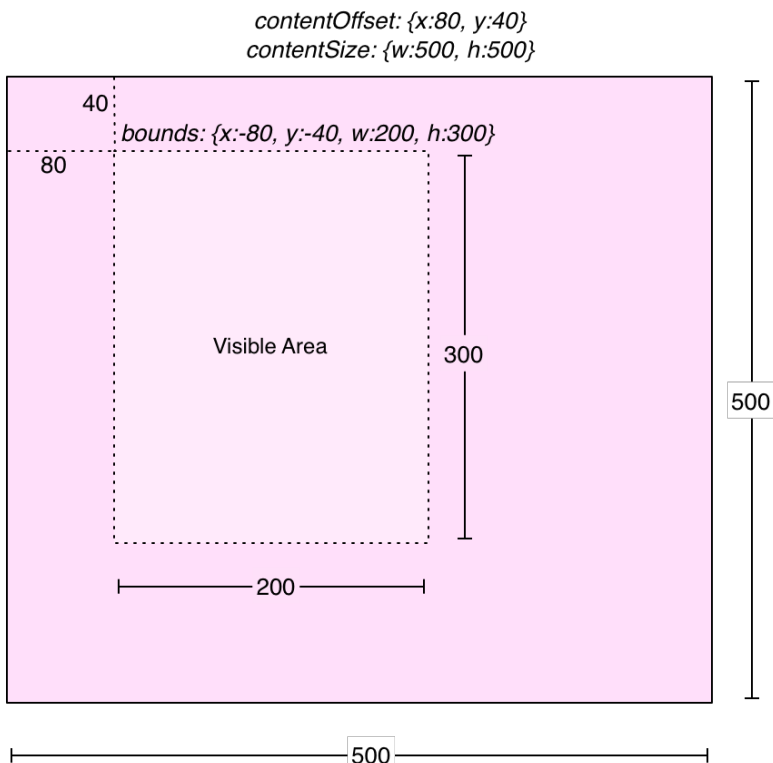


iPod a iTunes

- Zařízení se konfiguruje na desktop počítači (Mac, Windows).
- Uživatel nesmí do zařízení nic neautorizovaně nahrávat!
- Uživatel není kompetentní zařízení sám konfigurovat. Velký odklon od doby koncepce počítačů Apple I. a II.
- Proto v iOS není uživatelský home adresář.
- Data a aplikace patří k sobě. Sandbox aplikace.
 - Dropbox. iCloud Drive.

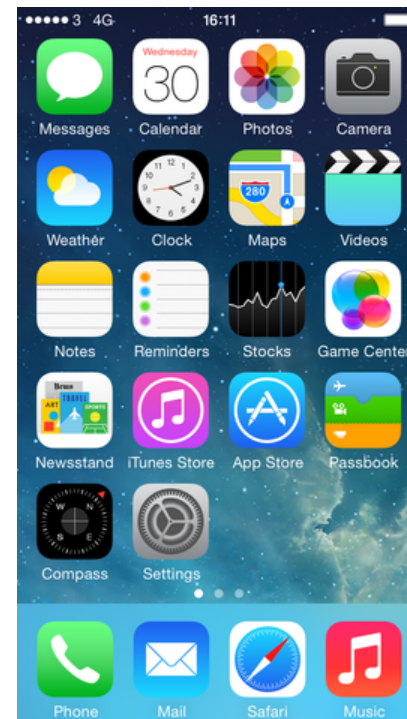
Vývoj v počátcích

- Posuvný obsah obrazovky (UIScrollView).
- Ovládání gesty (rozpoznávače gest).



Projekt vývoje iPhone

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



iPhone OS, iOS

- březen 2008 — iOS připouští cizí aplikace. iOS SDK.
- léto 2008 — Apple AppStore.
 - Registrování vývojáři, schvalovací proces.
- Konference WWDC.

Uživatelské rozhraní v iOS

- Uživatelské rozhraní OS a základní koncepty.
 - Spotlight, spouštění / přepínání aplikací, notifikační centrum.
 - Činnost aplikace na pozadí.
- UI pro aplikace. Human Interface Guidelines.
 - Schvalovací proces pro aplikace.
- Knihovny — Foundation, Cocoa Touch a další.
 - Knihovny Apple mají platnost desetiletí.
 - Objective-C a nástup Swiftu.

Vývoj aplikací pro iOS

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

Zařízení pod iOS

- Tablety iPad.
- Telefony iPhone. iPod Touch.
- Odvozeně tvOS a watchOS.
- Rozdíly programování pro iPad a iPhone/iPod:
 - Velikost obrazovky. Konfigurace "view controllers".
 - Univerzální aplikace pro iPad/iPhone.

Systémové základy iOS

- macOS a iOS jsou **unixové** systémy.
 - Srovnáme Linux (pro "ty ajťáky") a macOS (dost cool).
- FreeBSD, Mach. POSIX.
 - Apple dále jádro vyvíjí (řízení procesů, spotřeba, komprimace RAMky, souborové systémy).
- Historické vlivy NextSTEPu.
 - (Mach, BSD) -> NextSTEP -> Darwin -> mac OS X -> iOS

Programování pro iOS

- Jazyk — Objective-C a Swift.
- Základní koncepty **MVC**, tzv. **ViewControllers**.
- Více-vláknovost a asynchronní řízení. GCD.
- Vnitřní komunikace mezi objekty (KVC, KVO).
- Komunikaci v rámci eko-systému (Continuity).
- Databáze (CoreData, CloudKit, Documents).
- Multimédia. Hry.

Objective-C

- C s konceptem Smalltalkového posílání zpráv.
- Původ B. Cox a T. Love, Stepstone.
- Jazyk pro NextSTEP (1988). Později OS X a iOS.
 - 1996 — NeXT přešel pod Apple.
- Knihovna Foundation. Cocoa, Cocoa Touch (UIKit).

Koncepty Objective-C

- Protokoly — vícenásobná dědičnost na úrovni rozhraní.
- Kategorie — rozšiřitelnost rozhraní třídy.
- Garbage collection.
- Properties — atributy s programovatelnými setter / getter. Základ KVC / KVO.
- Bloky.
- Zpráva pro NULL objekt.

Interface třídy

```
@interface MTrida : NSObject
{
    int cislo;
}

@property (nonatomic, strong) NSString *jmeno;

-(id) initWith: (int) v;
+(MTrida *) allocWith: (int) v;

-(int) cislo;
-(void) setCislo: (int) v;
@end
```

Implementace třídy

```
@implementation MTrida
@synthesize jmeno;

-(id) initWith:(int)v {
    self = [super init];
    self.cislo = v;
    return self;
}

+(MTrida *) allocWith: (int) v {
    MTrida *p = [[MTrida alloc] init];
    p.cislo = v;
    return p;
}

-(int) cislo{
    return cislo;
}

-(void) setCislo:(int)v {
    [self willChangeValueForKey: @"cislo"];
    cislo = v;
    [self didChangeValueForKey: @"cislo"];
}

@end
```


Swift

- Představen na WWDC 2014.
- Syntaktická revize Obj-C a pár konceptů navíc.
- Koncept hodnota versus reference (a NULL).
 - Deklarace "var" a "let"
 - Konverze operátory ? a !
 - Wrapped value — String?, Int?
 - mujString?.count

Koncepty Swiftu

- struct / class — předávání hodnoty nebo ref.
- enum — výčet hodnot (strukturovaných)
- N-tice (nepojmenované struktury)
- Konstrukce s "@" vázané na architekturu OS.
- Licence pro volné využití i mimo Apple zařízení.

Ukázka enum

```
enum Person {  
    case unknown, dead  
    case called(String)  
    case aged(Int)  
    case registered(String, Int)  
}  
  
var p : Person  
  
let p1 = Person.unknown  
let p2 = Person.called("Honza")  
let p3 = Person.registered("Franta", 10)  
  
p = p2  
  
if case .called(let x) = p {  
    print("Jmeno \"(x)\"")  
}
```

Nosná část aplikace

- UIScreen — (ovladač) obrazovka zařízení.
- UIApplication Delegate — vazba app. na OS.
- Model-View-Controller — architektura aplikace.
- Paměťový model — dynamika.
 - Automatic ref. counting (ARC). Uvolňování paměti.
- Vlákna a asynchronní volání. GCD.
 - Aplikace je řízena událostmi.

UIScreen

- Objekt reprezentující obrazovku zařízení.
Podobně v tvOS.
- Aplikace může přistupovat k více UIScreen:
 - default, vzdálená obrazovka — Apple TV.
- Rotace (souřadný systém). Portrait, Landscape.
- Různé velikosti screens.
- Geometrie je v doublech, rastrizace do UIScreen.
- UIWindow — "okno" aplikace. Vrstvení oken.

UIApplication Delegate

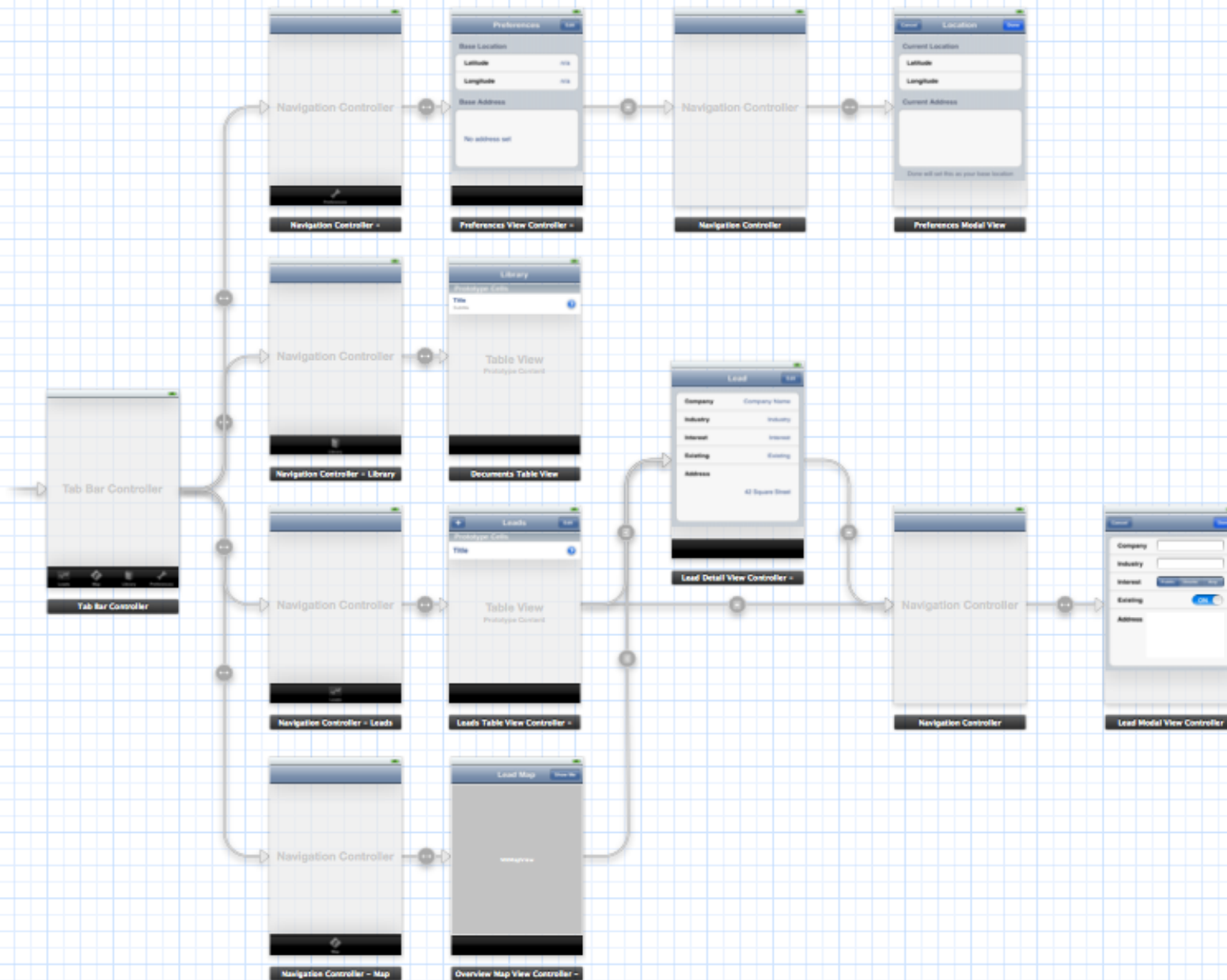
- Koncept *delegate* obecně.
- Hlavní objekt aplikace je delegátem systémového objektu UIApplication.
- Implementuje reakce na systémové události:
 - Start aplikace.
 - Přechod do pozadí/ popředí.
 - Činnost aplikace v pozadí (speciální případy).

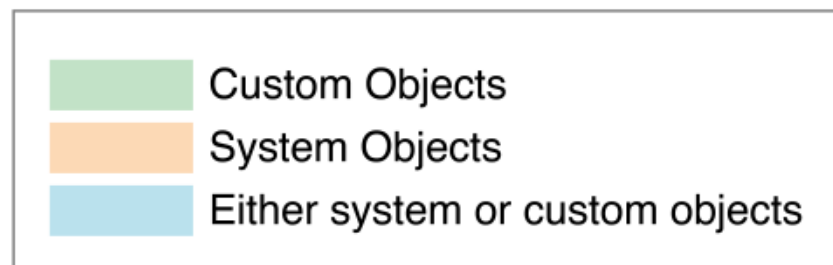
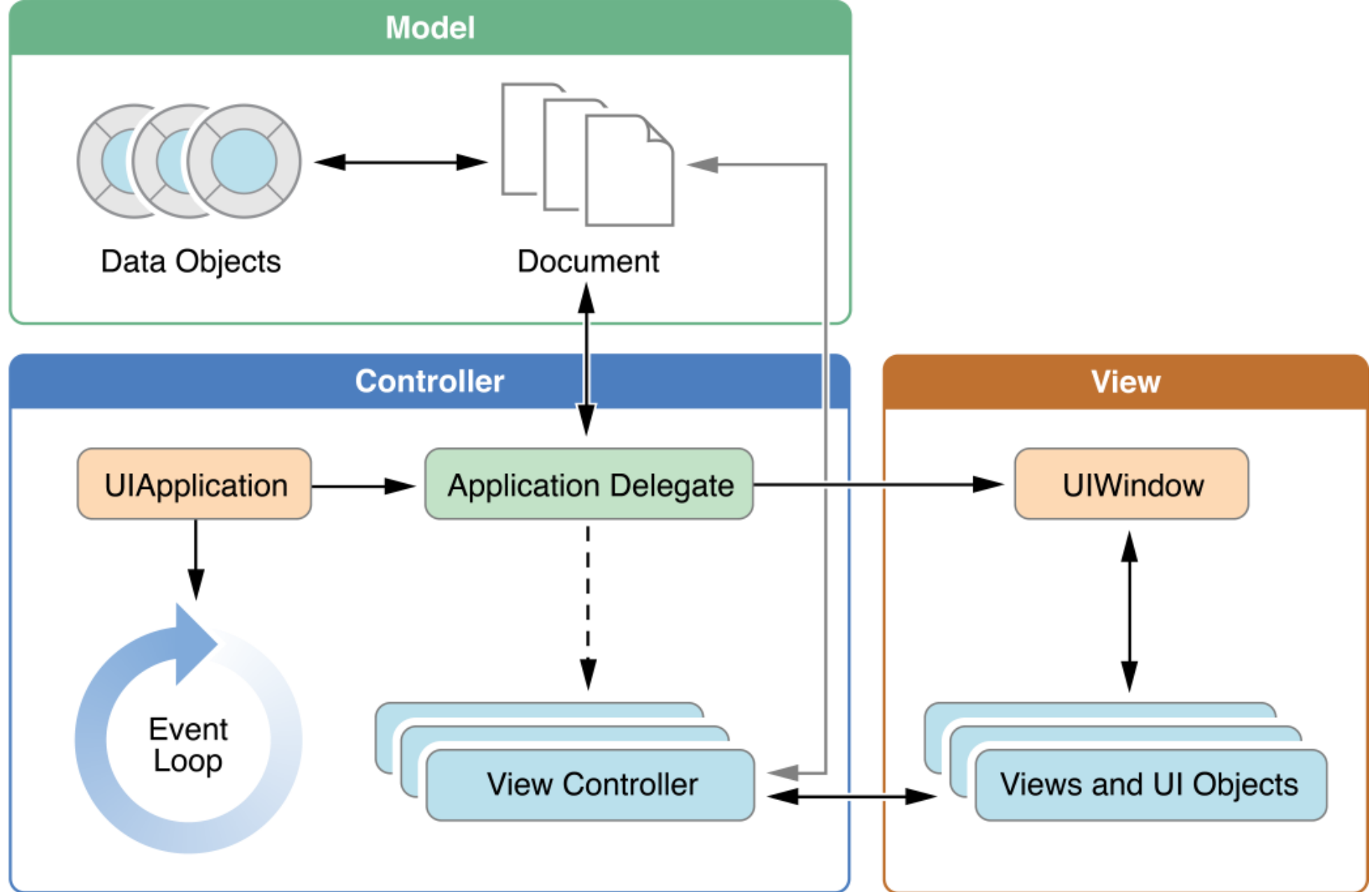
UIApplication

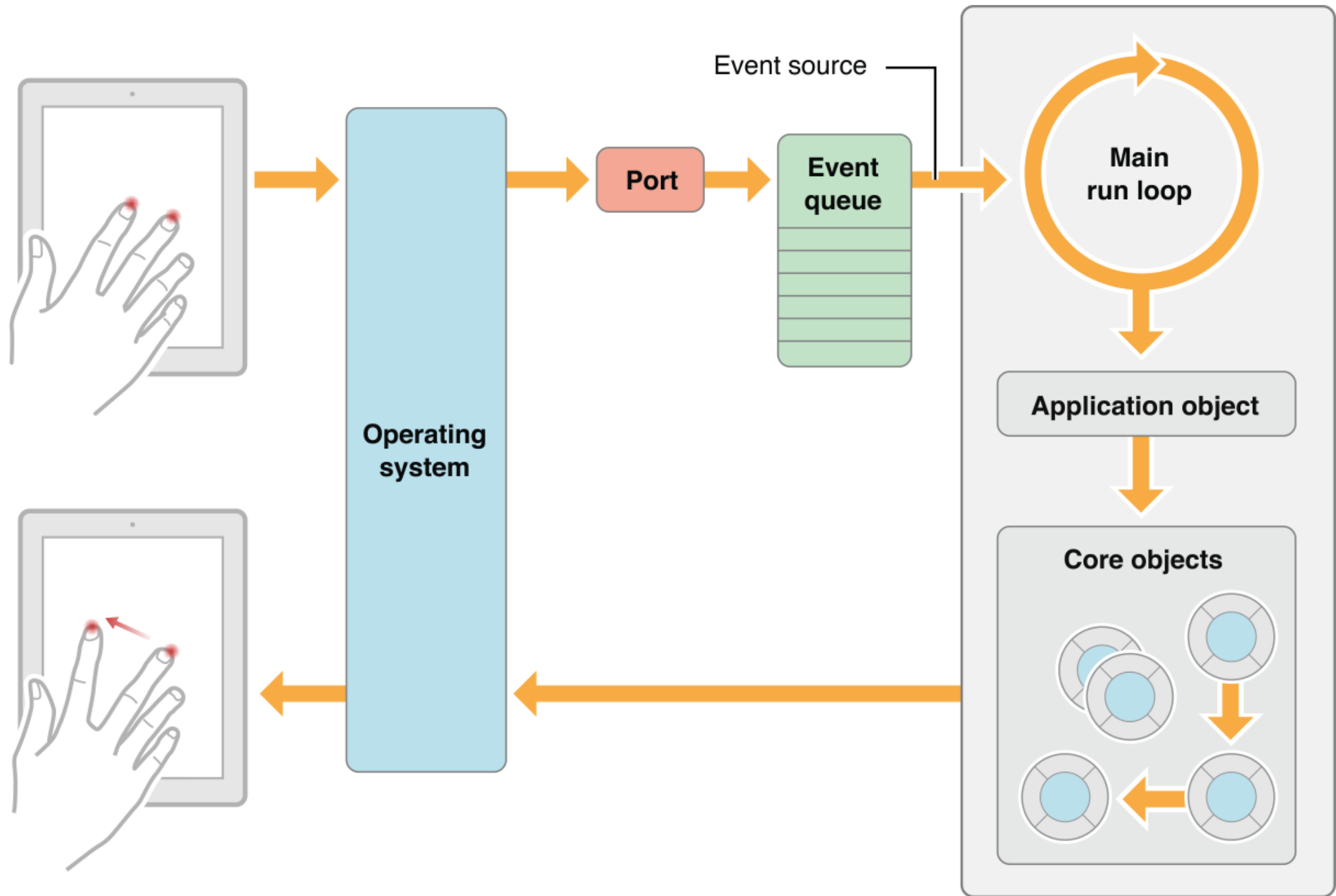
- `class UIApplication : UIResponder`
 - `.shared : UIApplication { get }` — je singleton.
 - `.delegate? : UIApplicationDelegate` — singleton.

Konstrukce App při startu

- AppDelegate dostane zprávu `didFinishLaunching...`
 - Konstrukce UIWindow.
 - Konstrukce počátečního "root" ViewControlleru.
 - Systémová inicializace aplikace — vedlejší vlákna.
 - UIWindow se stává "key and visible".
- Automatická instanciac (XIB, Storyboard).
- UI aplikace by mělo nastartovat okamžitě.

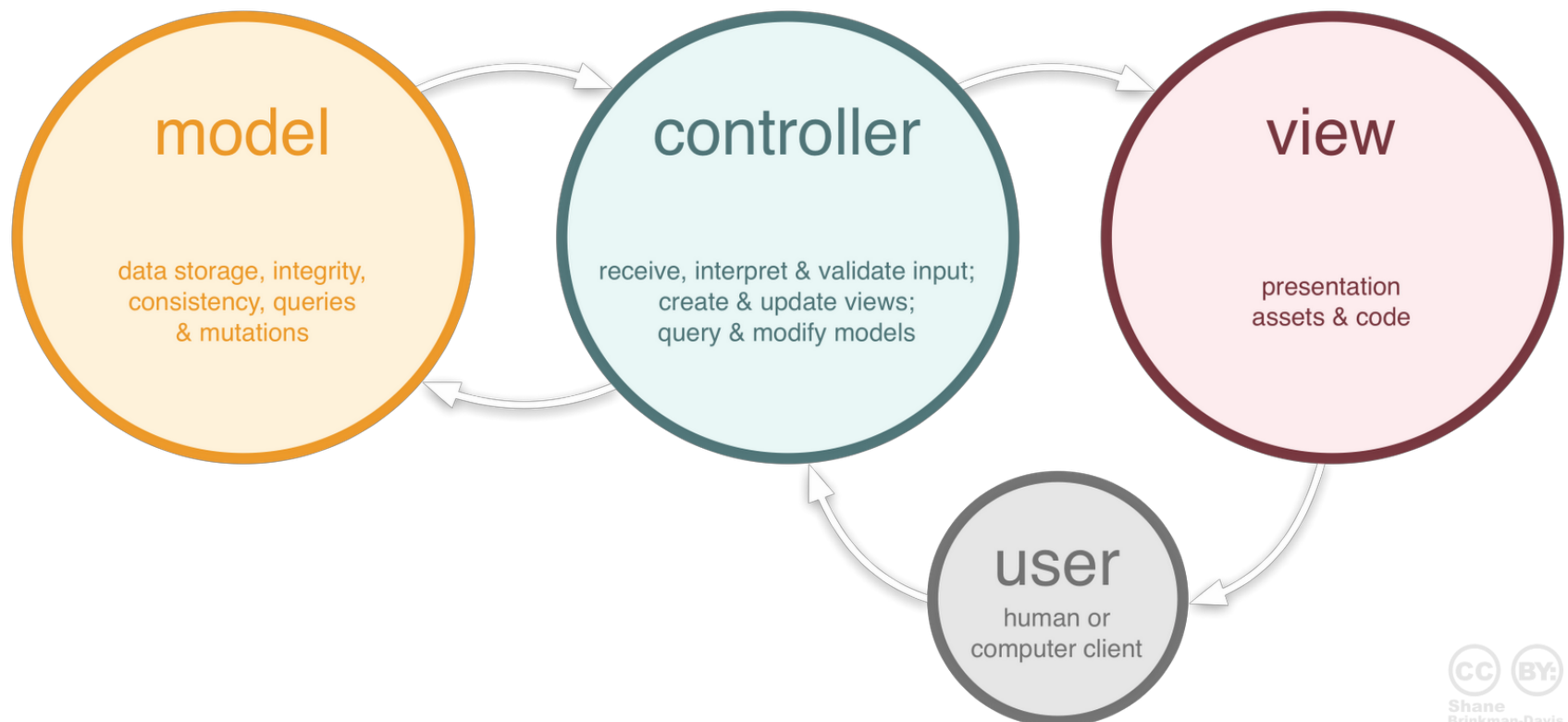






Model-View-Controller (MVC)

- Model — datová část aplikace.
- View — zobrazovací prvky do View / Window.
- Controller — řídicí objekty.



MVC

- MVC koncept byl poprvé použit ve Smalltalku (Xerox) — dále okno, myš, ... návaznost na Mac.
- NextSTEP (1989) — OS, NeXT Comp., S. Jobs
 - Základy pro moderní Mac OS.
 - Knihovna Cocoa.
 - Později Cocoa Touch.
 - Prefix "NS" v názvosloví.

MVC v aplikaci

- Programujeme (typicky) Model a Controller.
 - Znovupoužitelnost! Platformy macOS, iOS, tvOS, ...
- Je snaha automatizovat přesun dat z Modelu do Views (Bindings).
- Znovupoužitelnost ViewControllerů (VC).
 - konektory na views (IBOutlets),
 - konektory na model.
- Typizované VC. Styly aplikace.

Views

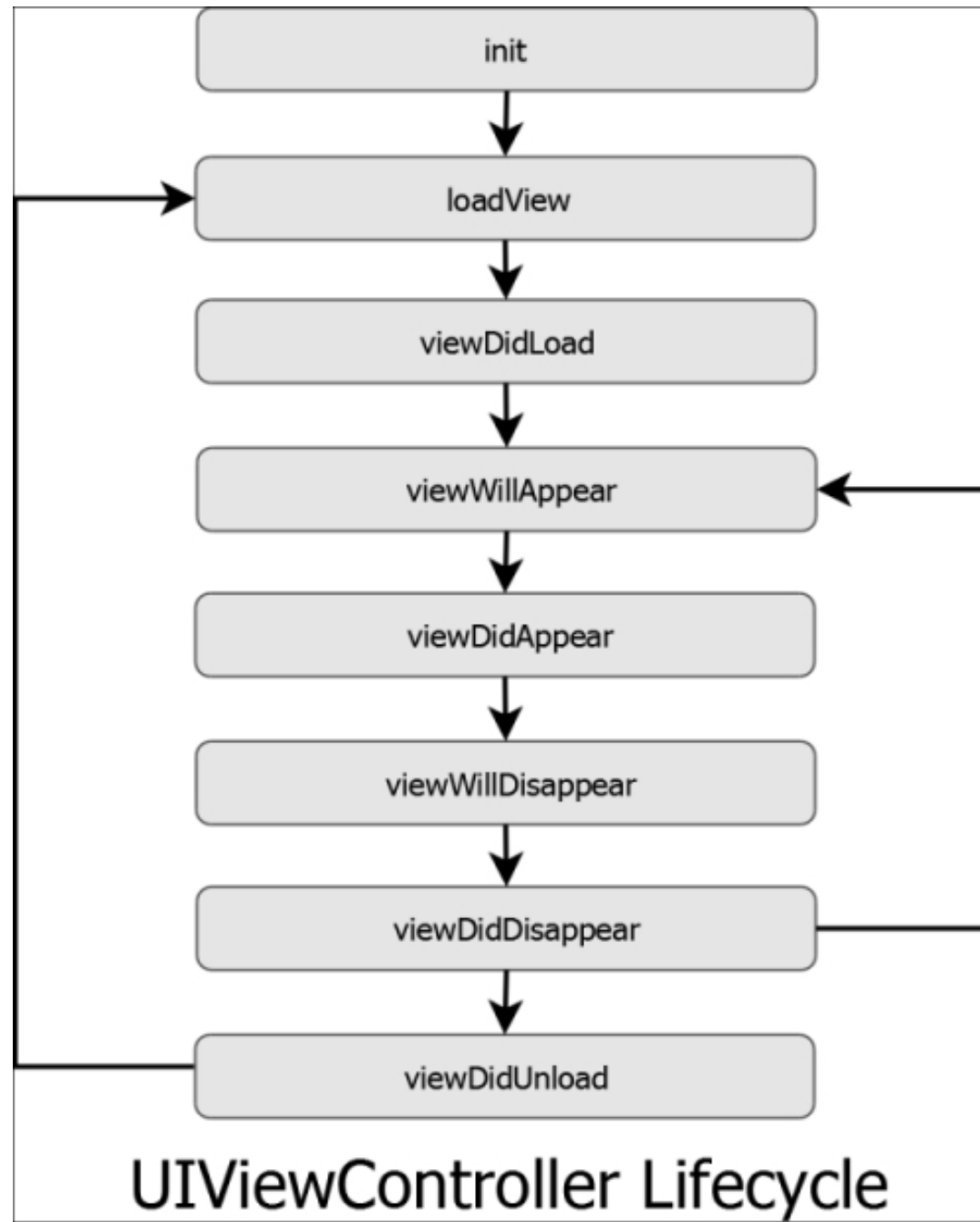
- UIView, UILabel, UITextField, UITableViewCell,
- Superview, subviews — hierarchie.
- Systém kompozice zobrazované bitmapy z elementárních views.
- Provádět změny do UI smí pouze hlavní vlákno.

```
class ViewController: UIViewController {  
    @IBOutlet var textik : UILabel?  
  
    func inicializuj() {  
        textik?.text = "Napis hello";  
    }  
}
```

ViewController (VC)

- Je vlastníkem objektu "view". Ten dále tvoří hierarchii dalších view.
- V aplikaci se manipuluje s VC.
- UIWindow referencuje svůj "root" VC.
 - Ten může organizovat další VC: navigation, tabbar, ...

Životní cyklus ViewController

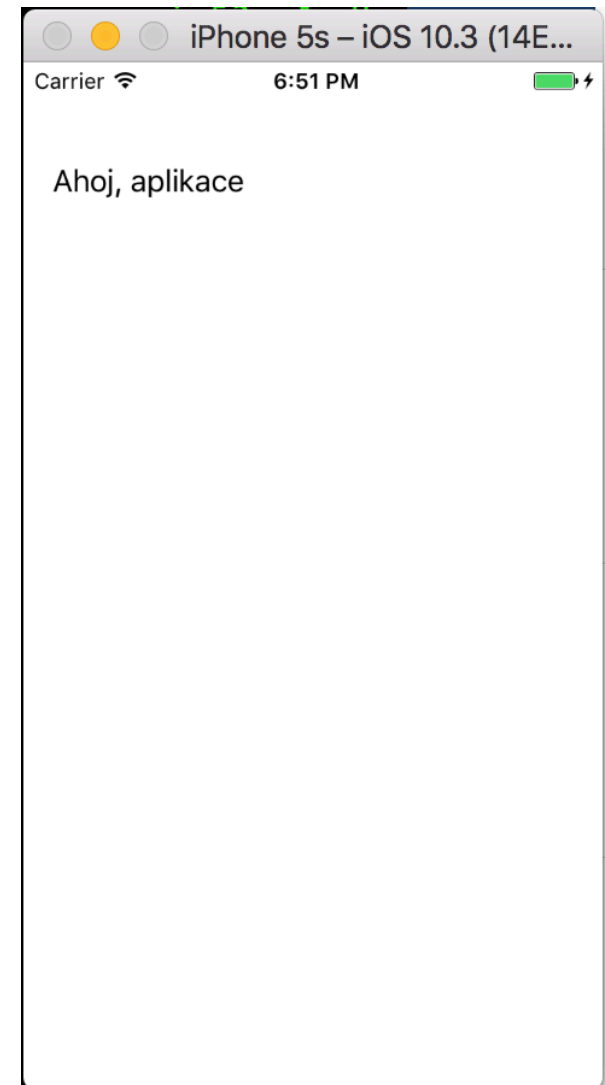


Základní úloha M-V-C

- UILabel má v okně zobrazit obsah zadané proměnné.
- Jak řešit dynamiku? Proměnná se může měnit:
 - V kódu volat aktualizaci UILabel na změnu proměnné.
 - Setter na proměnné volá aktualizaci UILabel (vhodné??).
 - Key-Value Observing.

M-V-C, Základní demo

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



Apple Bindings

- Koncept pracující v pozadí.
- Protokol Key-Value Observing.
 - Instanční proměnná třídy (tzv. property).
 - will / did-changeValueForKey
- Hodnota a její "observer".
- Setter na sledované property.
- CoreData a NSFetchedResultsController

M-V-C, KVO verze

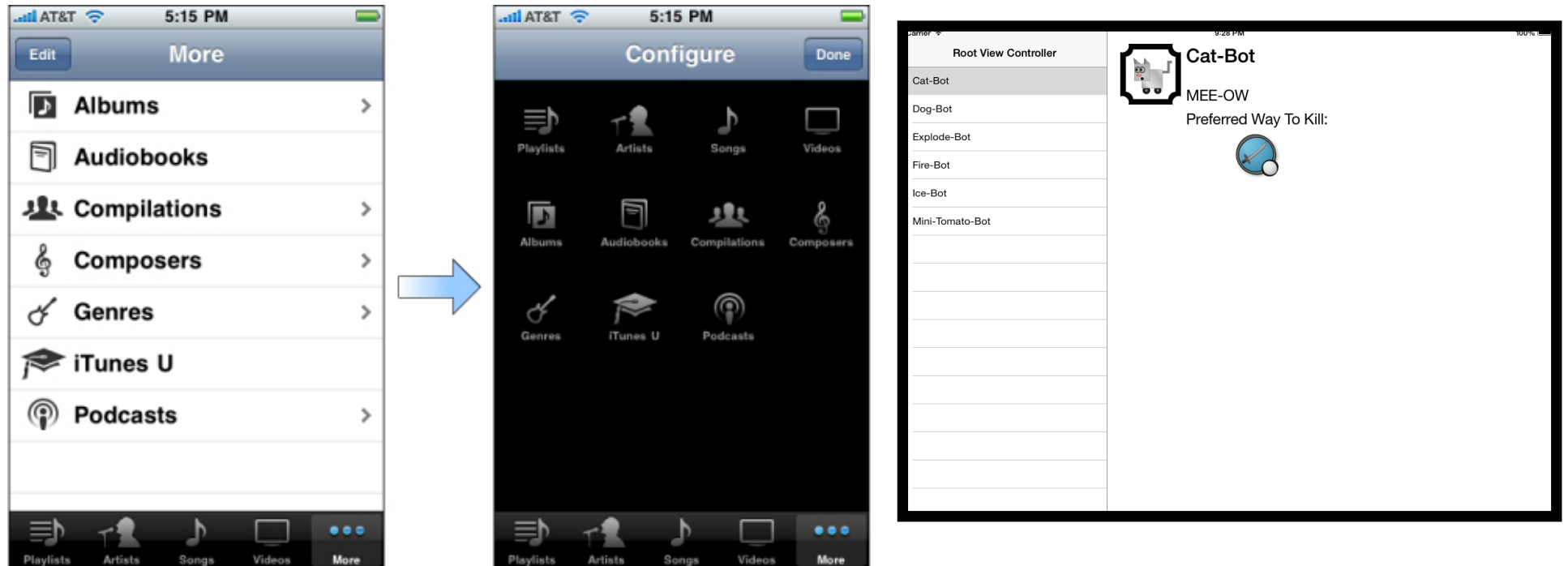
```
//
class MujModel : NSObject {
    dynamic var obsah : String = "Ahoj, aplikace"
}
//
class ViewController: UIViewController {
    @IBOutlet var lei : UILabel?
    let mujmodel = MujModel()
    //
    override func viewDidLoad() {
        super.viewDidLoad()

        lei?.text = mujmodel.obsah

        mujmodel.addObserver(self, forKeyPath: #keyPath(MujModel.obsah), options:
[.new, .old], context: nil);
    }
    //
    override func observeValue(forKeyPath keyPath: String?,
                                of object: Any?,
                                change: [NSKeyValueChangeKey : Any]?,
                                context: UnsafeMutableRawPointer?)
    {
        DispatchQueue.main.async {
            self.lei?.text = self.mujmodel.obsah
        }
    }
}
```

Typologie Stylů — VC

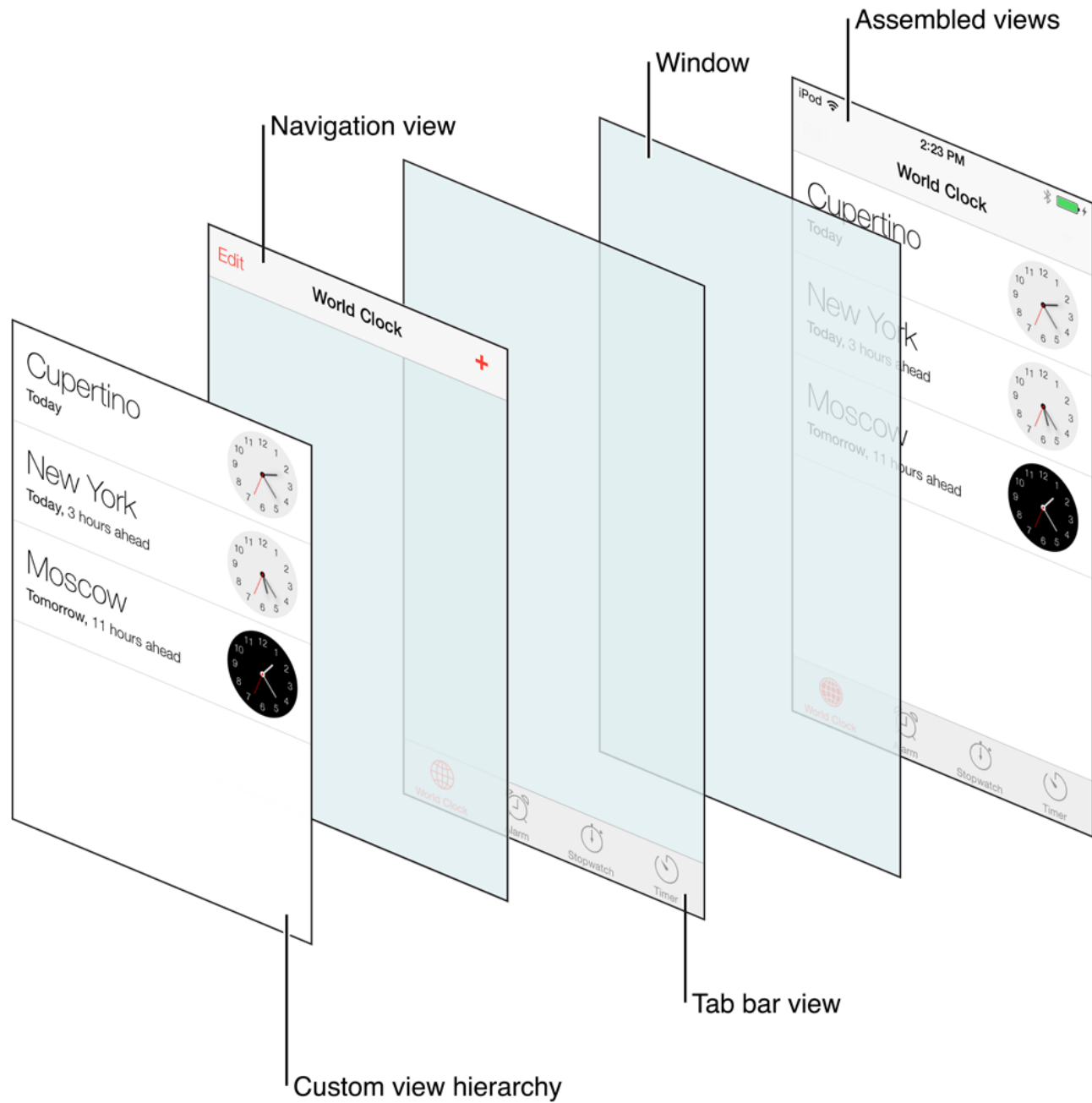
- Single View Controller. (Navigation VC)
- TabBar View Controller
- Master-Detail (speciálně pro iPad).



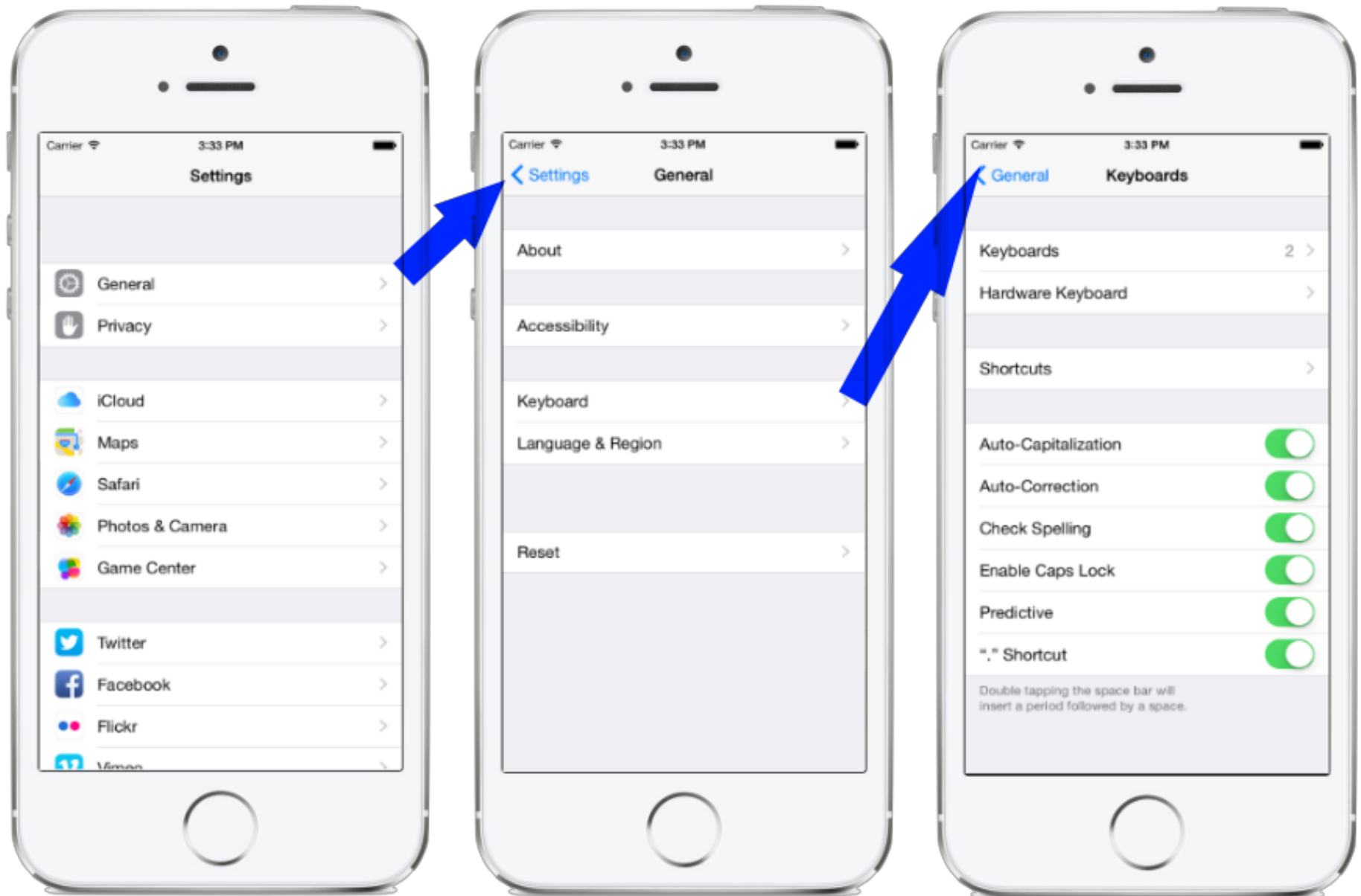
Základní VC

- UIViewController — vlastník "view".
- Table View Controller.
- Collection VC.
- Navigation VC.
- Page VC.
- Split VC.
- Popovers.

TabBar VC



Navigation VC



Dynamika VC, přepínání

- TabBar, UINavigationController.
- view, contentView, childVC.
- Hierarchie (strom) UIView.
- Předchozí VC: odebere se ze superView.
- Nový VC: přidá se do contentView, geometrie.

Životní cyklus zobrazování VC

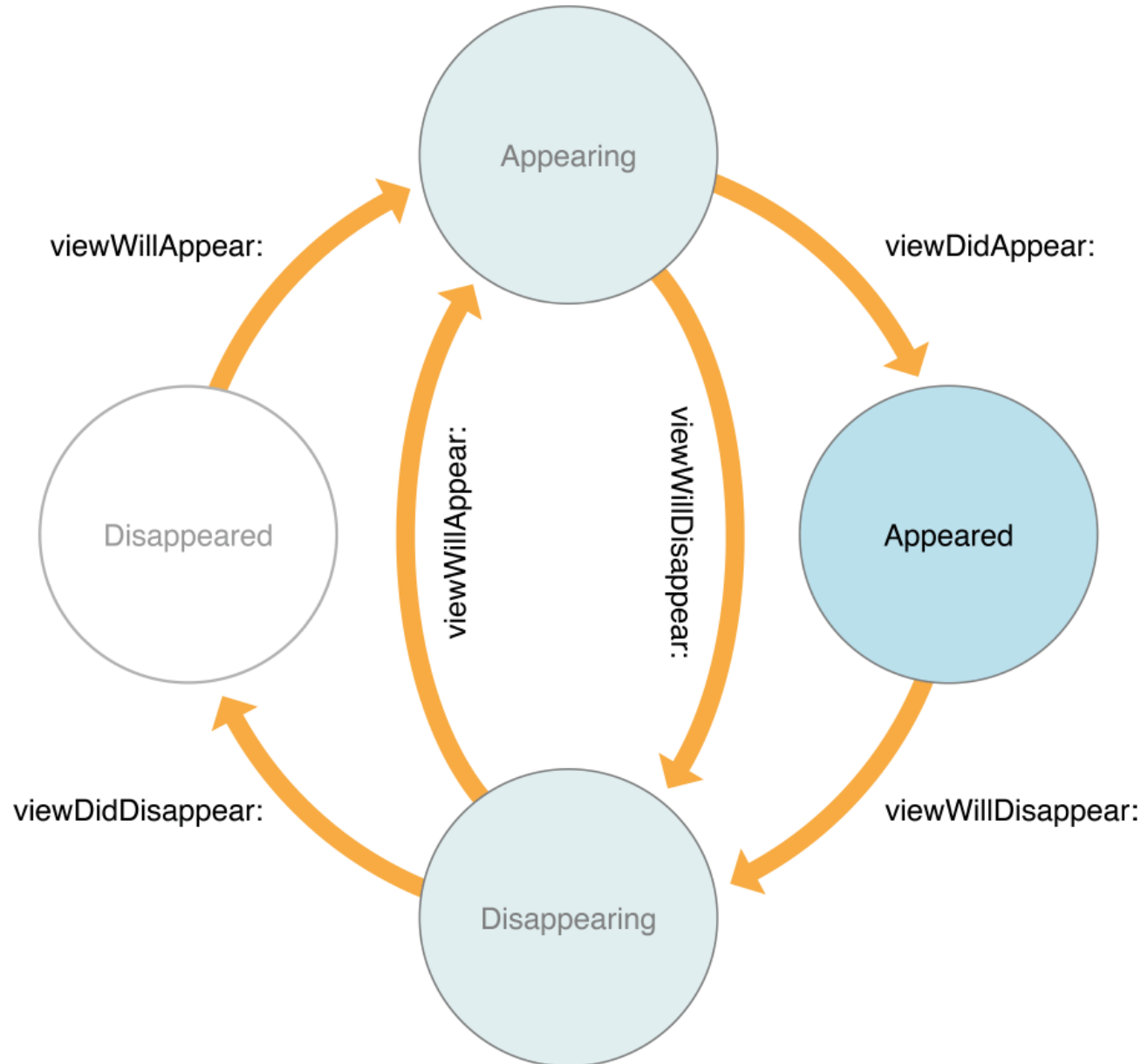
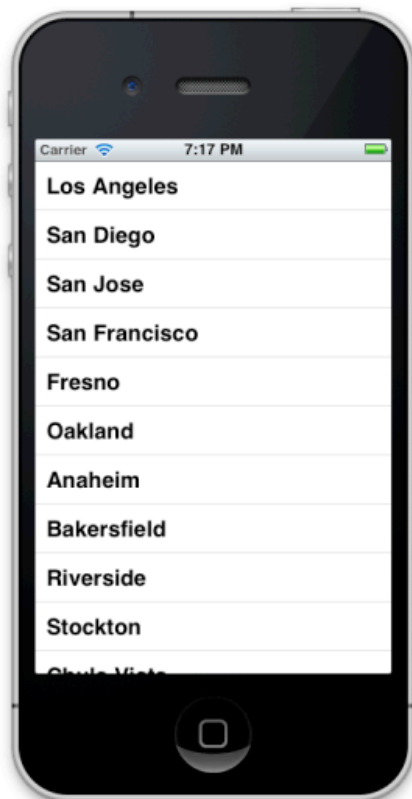
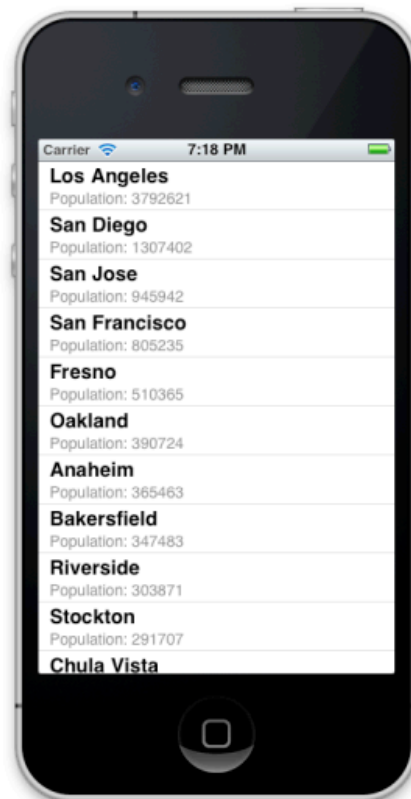


Table View Controller

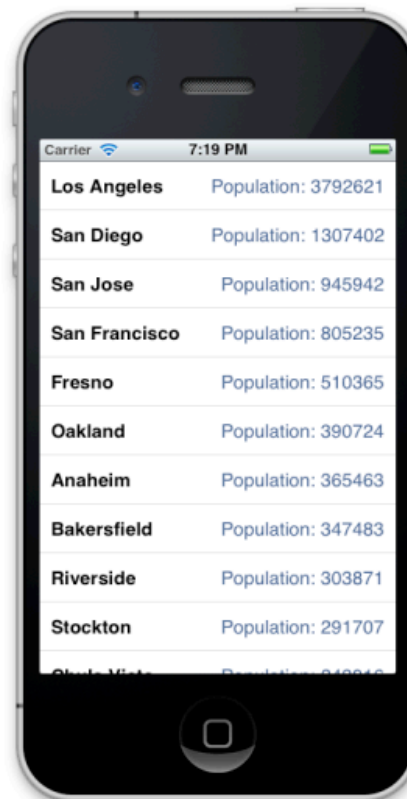
- Nejdůležitější prvek UI iOS.
- Seznam buněk (TableViewCell).



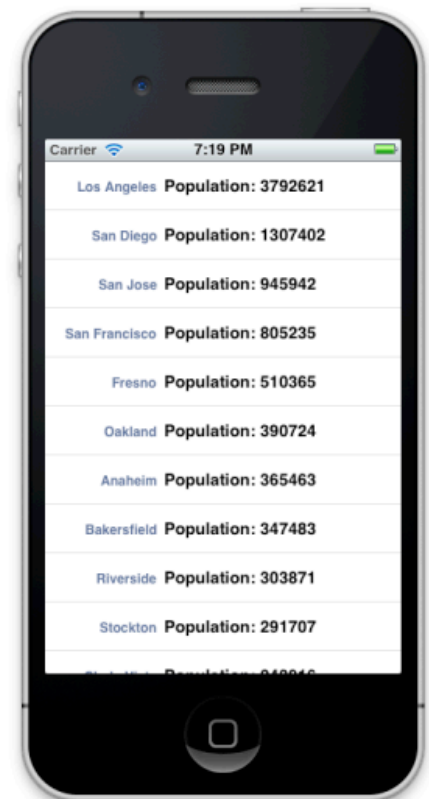
UITableViewCellStyleDefault



UITableViewCellStyleSubtitle

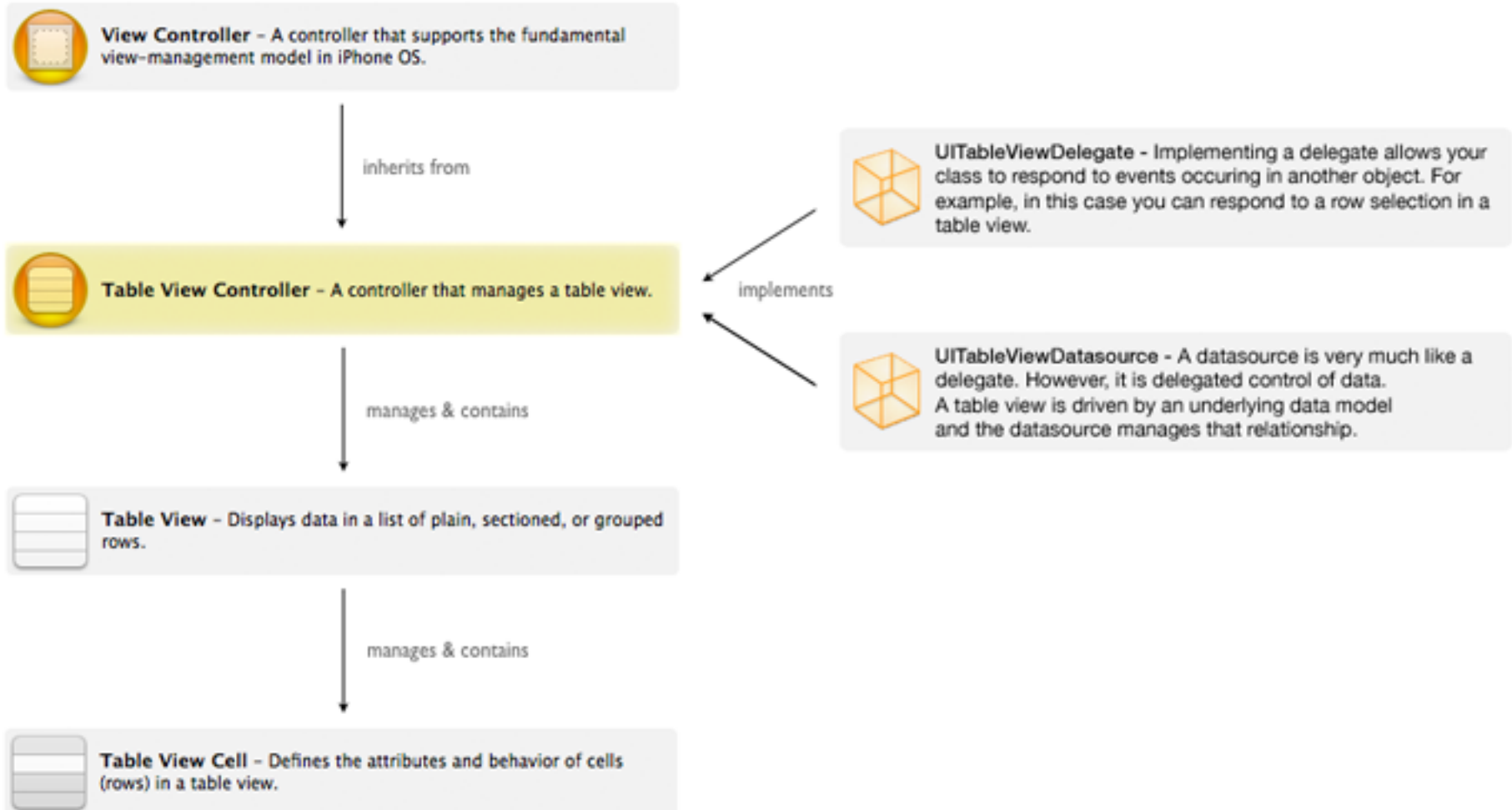


UITableViewCellStyleValue1



UITableViewCellStyleValue2

UITableViewController

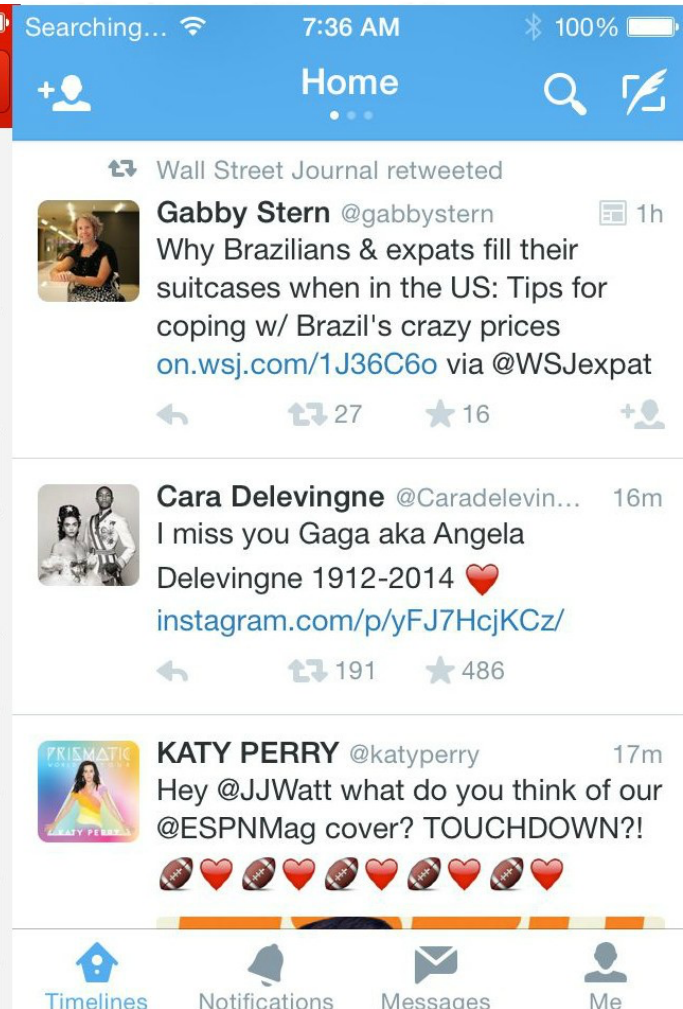
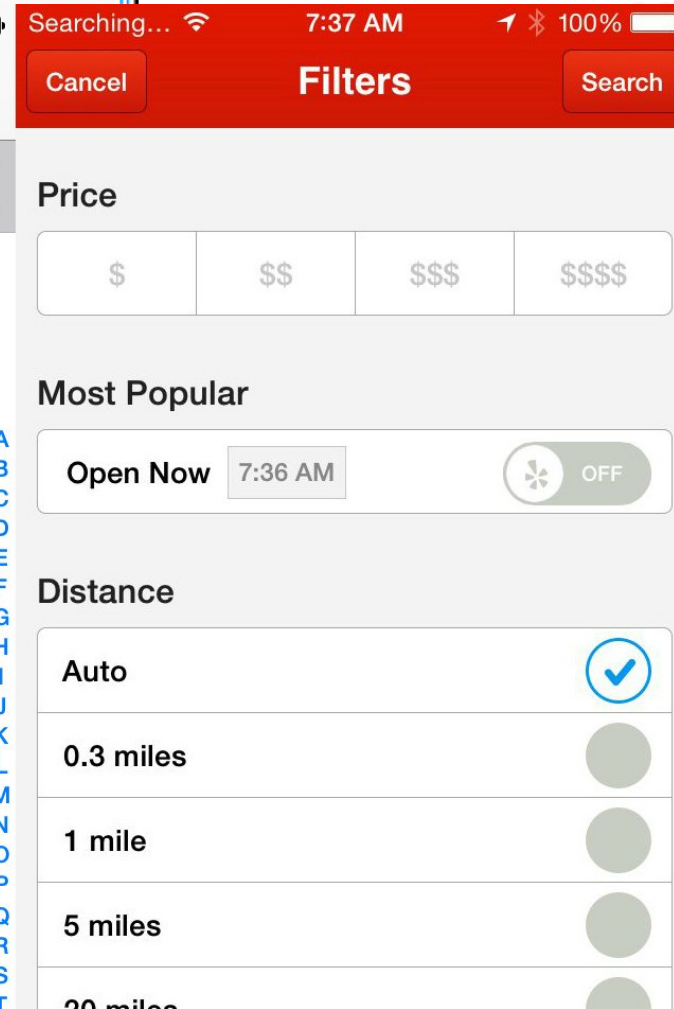
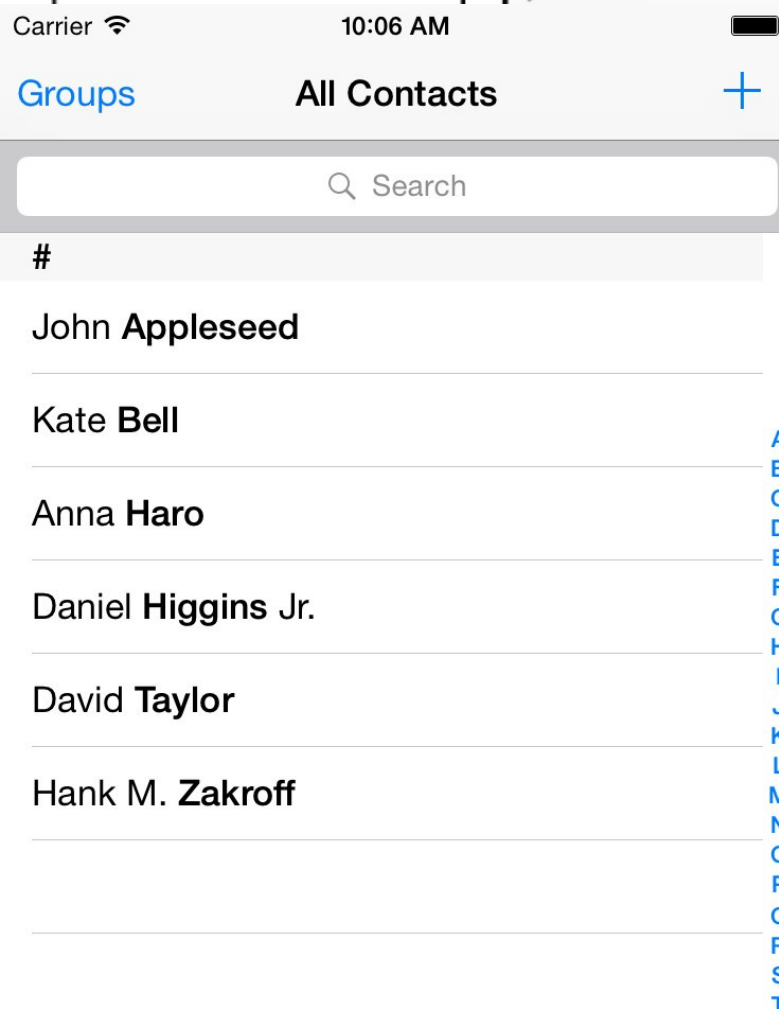
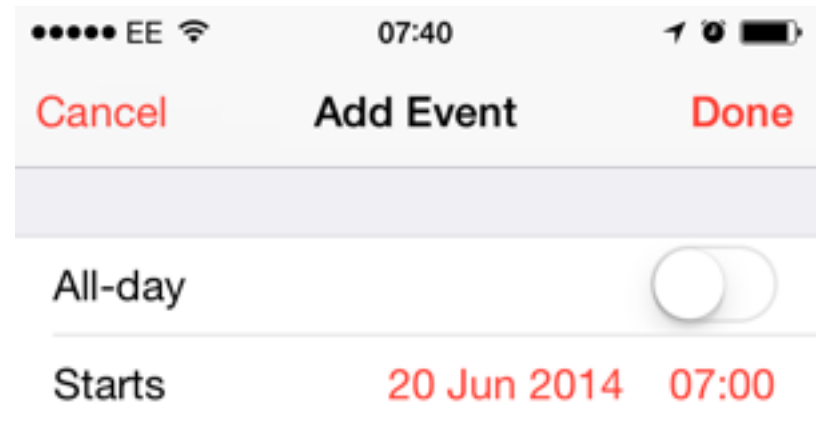
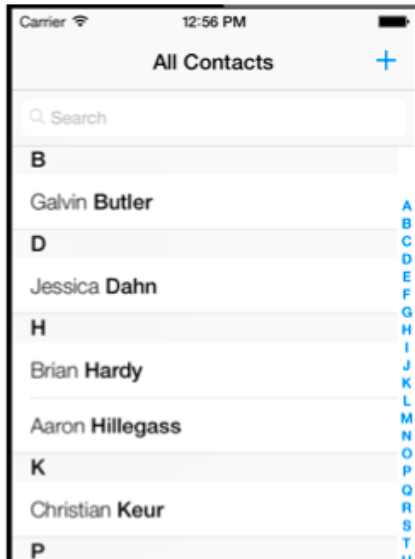
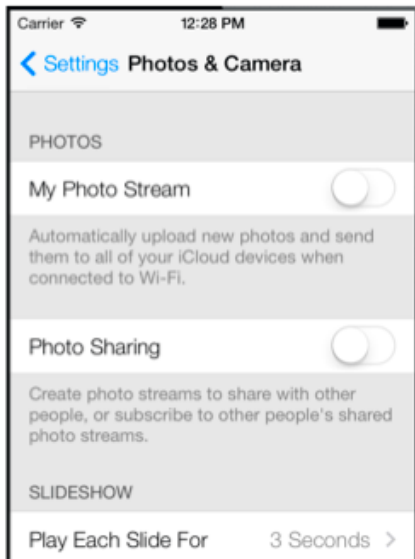


UITableViewController

- Implementuje dataSource a delegate protokoly tabulky.
- Uživatelský VC dědí z TVC, přepisuje potřebné metody dataSource a delegate protokolů.

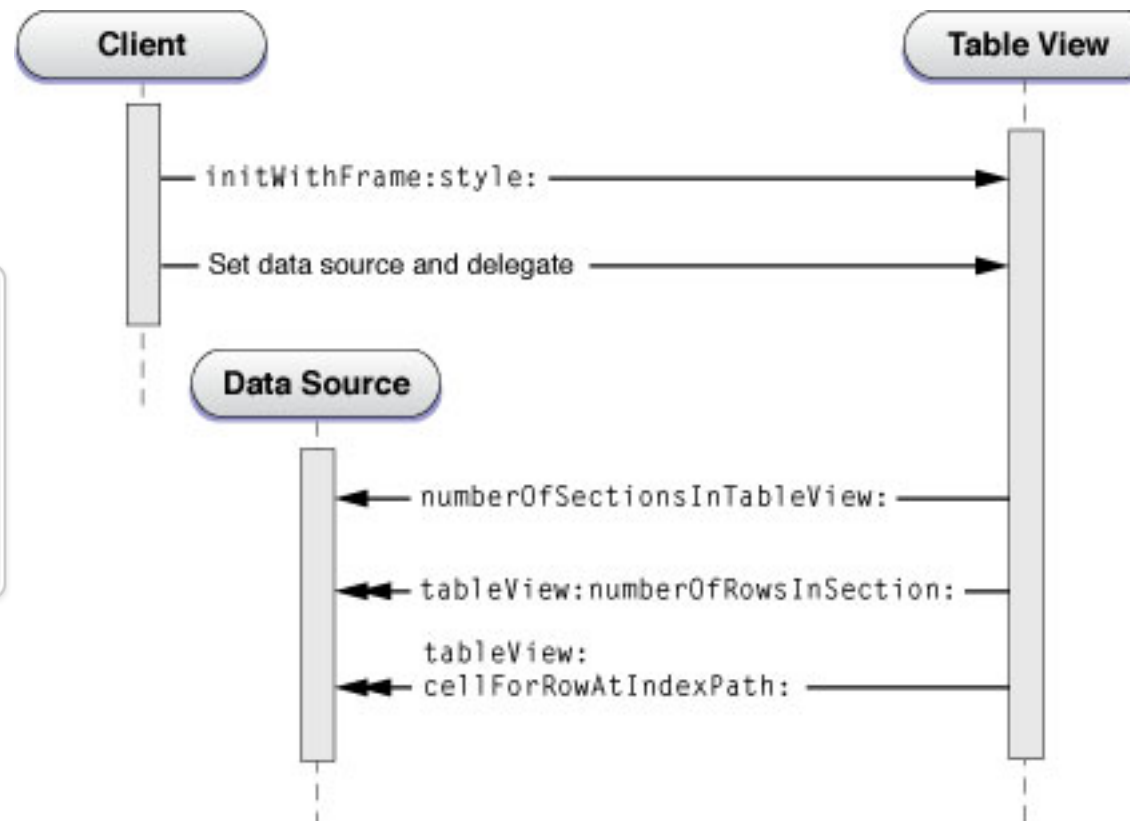
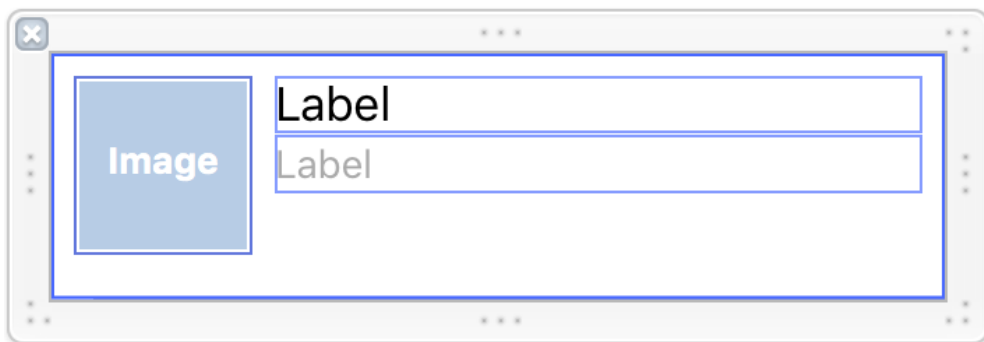
UITableView

- Odvozen od UIScrollView (přesahuje rámec obrazovky).
- Provádí rozmístění (layout) buněk (Cell) ve skupinách.
- Obecný heterogenní dynamický soupis buněk vertikálně řazených.



Řízení — dataSource

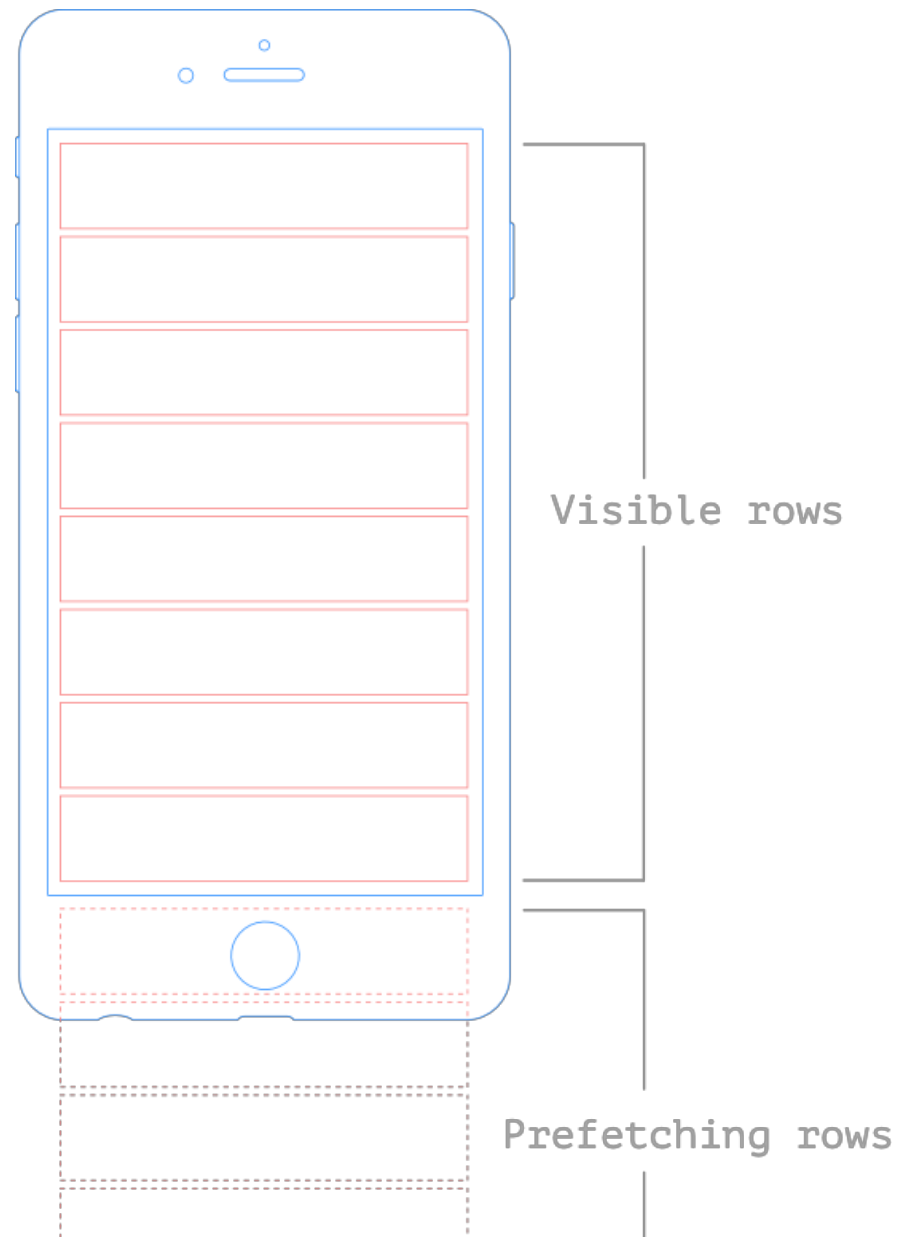
- TVC zprostředkovává obsah (Model) do View.
- Model zde vystupuje jako "dataSource" API:
 - počet sekcí, počet řádků v sekci.
 - sestavení TableViewCell pro zadanou souřadnici (IndexPath).



Řízení TVC, dynamika

- TVC zjistí datový rozsah (sekce, řádky).
- Předpokládá se, že viditelná je pouze část buněk.
- Ty jsou alokovány a inicializovány.
- Při posuvu tabulkou se dynamicky volá dataSource na doplňování obsahu buněk.
- Znovupoužití objektů buněk (pool). Identifikátory buněk.

Dynamika přístupu na dataSource



Demo

```
class SimpleTab: UITableViewController {  
    var seznam = ["Jeden", "Druhy", "Treti"];  
  
    override func tableView(_ tableView: UITableView,  
        numberOfRowsInSection section: Int) -> Int  
    {  
        return seznam.count  
    }  
  
    override func tableView(_ tableView: UITableView,  
        cellForRowAt indexPath: IndexPath) -> UITableViewCell  
    {  
        let cell = tableView.dequeueReusableCell(withIdentifier: "cell",  
for: indexPath)  
  
        cell.textLabel?.text = seznam[indexPath.row]  
  
        return cell  
    }  
}
```

Demo, oddělený dataSource

```
class MyArrayModel: NSObject, UITableViewDataSource {
    var seznam : [String]
    init(withStrings : [String]) {
        self.seznam = withStrings;
        super.init();
    }
    func tableView(_ tableView: UITableView,
                   numberOfRowsInSection section: Int) -> Int
    {
        return seznam.count
    }
    func tableView(_ tableView: UITableView,
                   cellForRowAt indexPath: IndexPath) -> UITableViewCell
    {
        let cell = tableView.dequeueReusableCell(withIdentifier: "cell",
for: indexPath)

        cell.textLabel?.text = seznam[indexPath.row]

        return cell
    }
}
```

Demo, oddělený dataSource

```
class ModelTab: UITableViewController {  
    override func viewDidLoad() {  
        super.viewDidLoad();  
        //  
        self.tableView.dataSource = MyArrayModel(withStrings:  
["1", "2", "3"]);  
    }  
}
```

Dynamika, posuv v Table

- Chod aplikace musí být hladký.
- Inicializace buněk může brzdit chod tabulky (rychlý posuv).
- Při náročnější inicializaci se přechází do bočních vláken (GCD).

```
override func tableView(_ tableView: UITableView, cellForRowAt
indexPath: IndexPath) -> UITableViewCell
{
    let cell = tableView.dequeueReusableCell(withIdentifier:
"tabik", for: indexPath)

    cell?.imageView?.image = placeholder;

    DispatchQueue.global().async {
        //
        if let loaded = myModel.load(cosi) {
            //
            if let ncell = tableView.cellForRow(at: indexPath) {
                //
                DispatchQueue.main.async {
                    ncell.imageView?.image = loaded;
                }
            }
        }
    }

    return cell;
}
```


TableView, Delegate

- Dostává zprávy o událostech nad tabulkou.
 - Označení buňky — will / did. Zrušení značení — will / did.
 - Typicky se provede přesun do dalšího VC (detail obsahu buňky).
- Geometrie buněk, dodatečné texty, ...
- Rozsáhlý protokol — UITableViewController ho implementuje.

Navigation VC, push VC

```
override func tableView(_ tableView: UITableView,
                        didSelectRowAt indexPath: IndexPath)
{
    //
    let story = UIStoryboard(name: "Main",
                             bundle: Bundle.main);

    let det = story.instantiateViewController(withIdentifier:
"jeden")

    self.navigationController?.pushViewController(det, animated:
true);
}
```

Editace



Dynamika obsahu tabulky

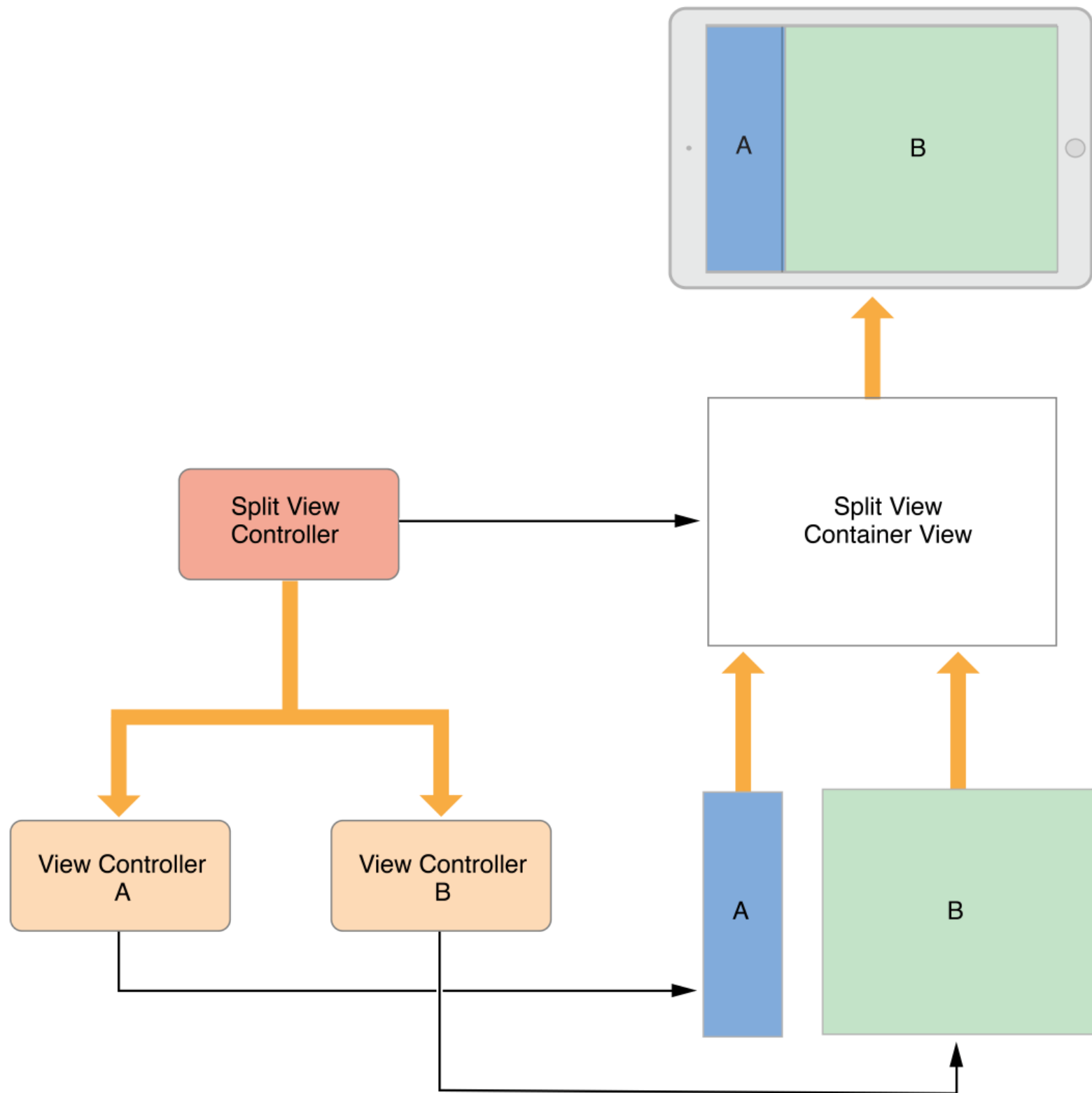
- Statické tabulky (formuláře).
- Dynamické tabulky — obsah se průběžně mění.
- Model v MVC tabulky je dynamický:
 - vkládání buněk,
 - rušení buněk,
 - přeuspořádání buněk.
- TableView implementuje (animované) změny obsahu tabulky. Řídí je Controller.

CoreData+TableView

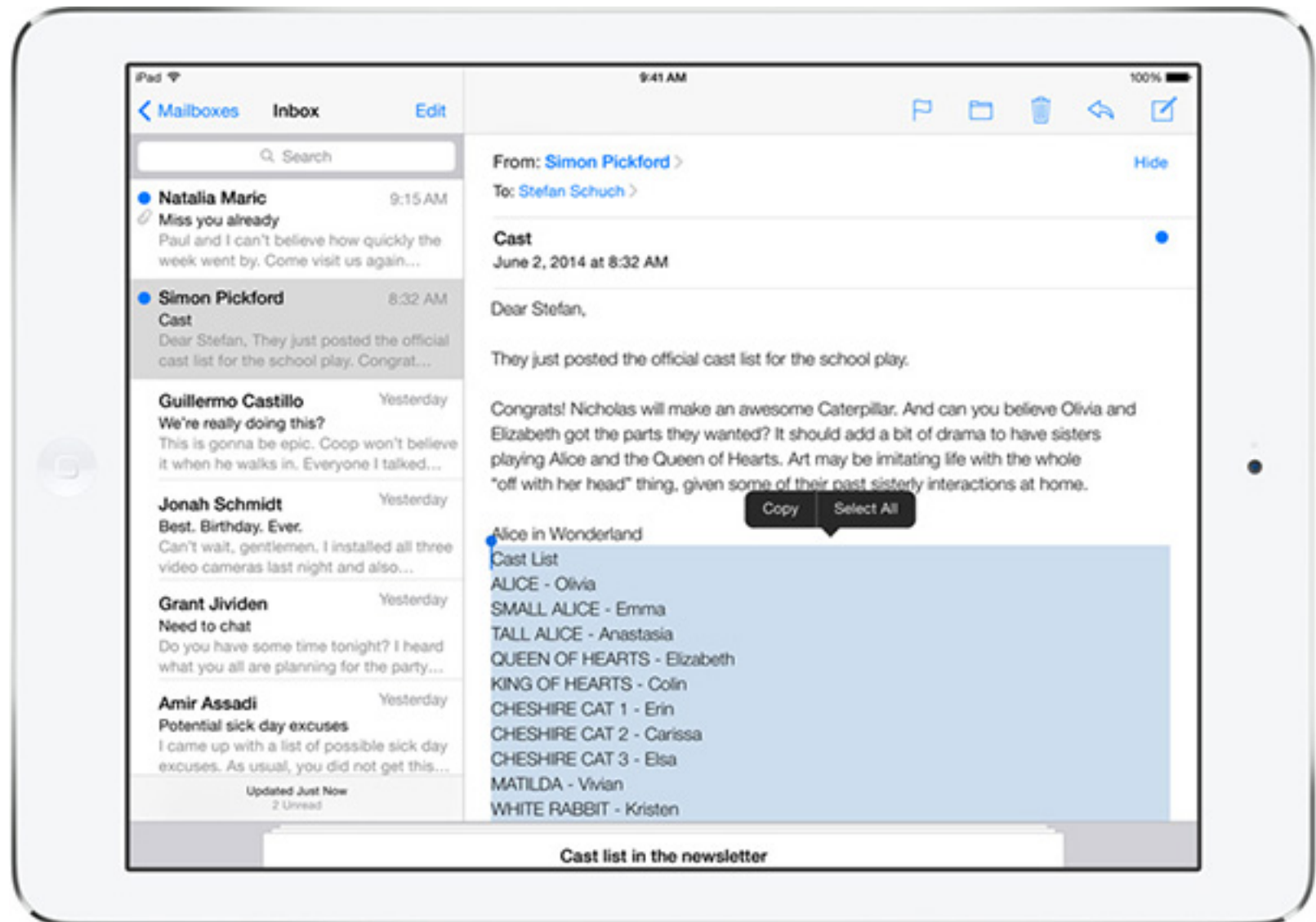
- Typická je kombinace obsahu DB tabulky (CoreData) a TableView.
- FetchResultsController — má delegáta přesměrovaného na Controller tabulky.

Master-Detail ViewController

- Typicky především na iPadu.
- UISplitViewController — rozdělí obrazovku na dvě části.
- Master — VC s hlavním obsahem, typicky tabulka položek.
- Detail — VC s detailním pohledem na položku.

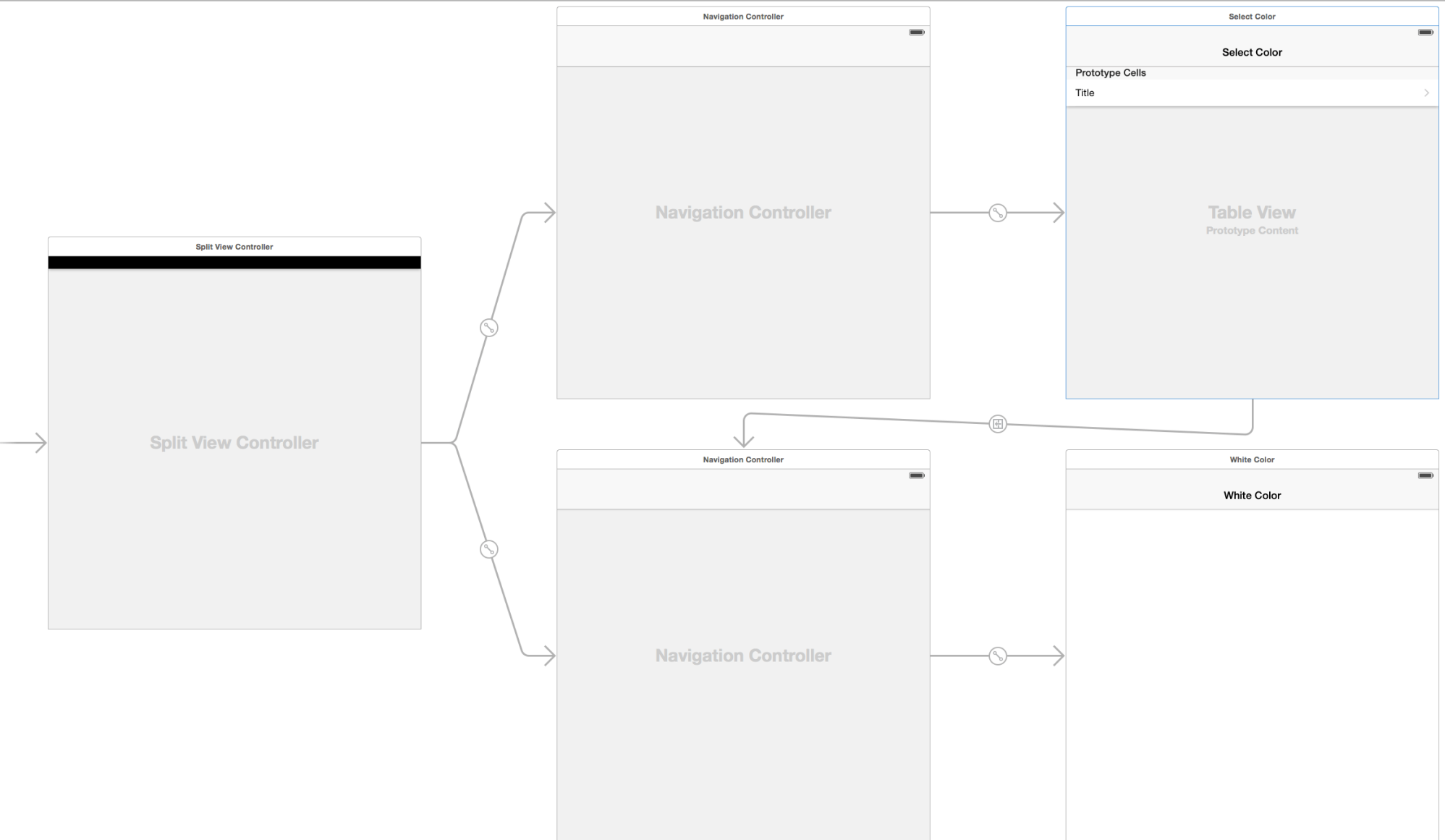


Mail app na iPadu



Architektura VC v M-D

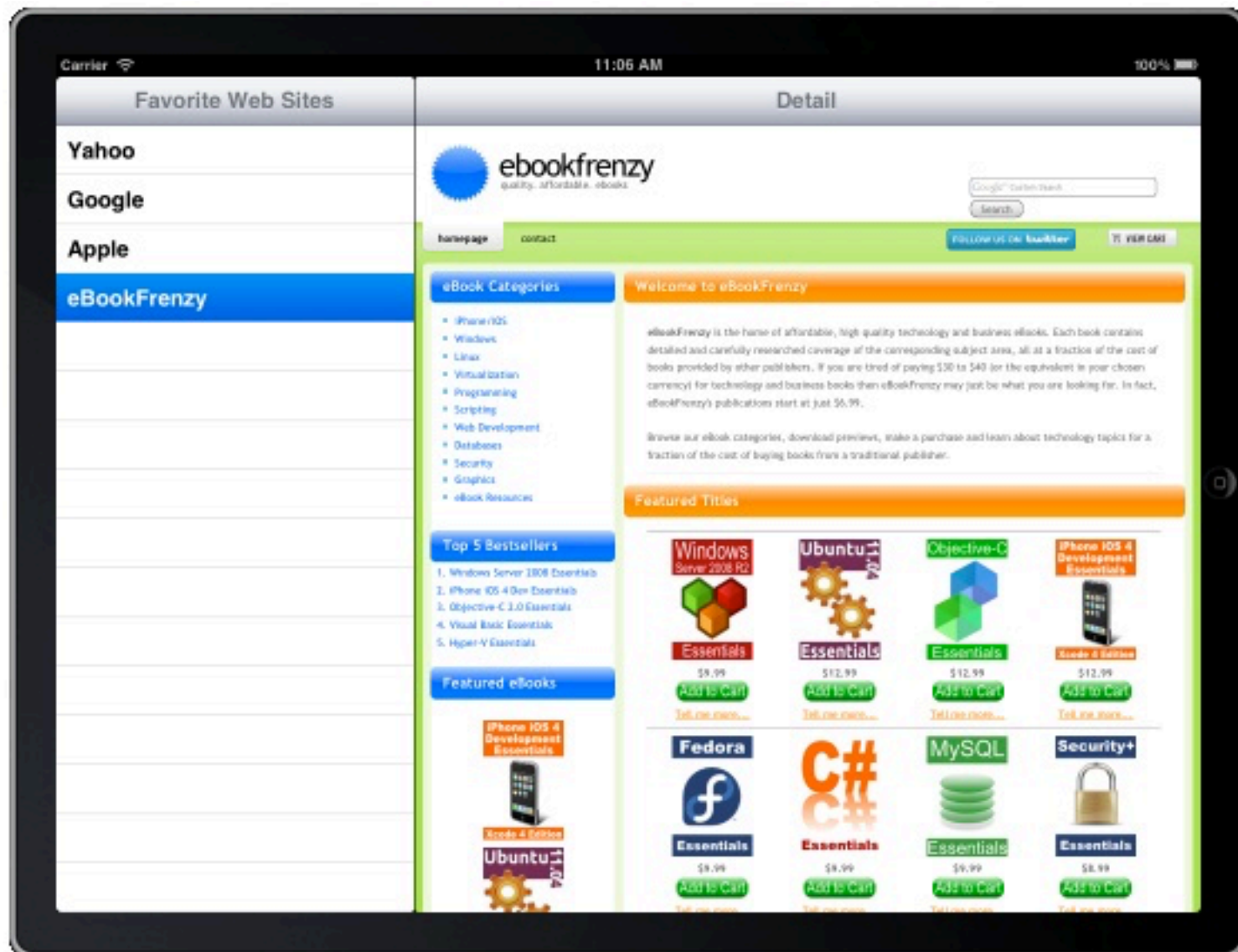
< > | SplitViewControllerDemo > SplitViewControllerDemo > Main.storyboard > Main.storyboard (Base) > Select Color Scene > Select Color



wAny hAny

이러한 구조

M-D v režimu Landscape

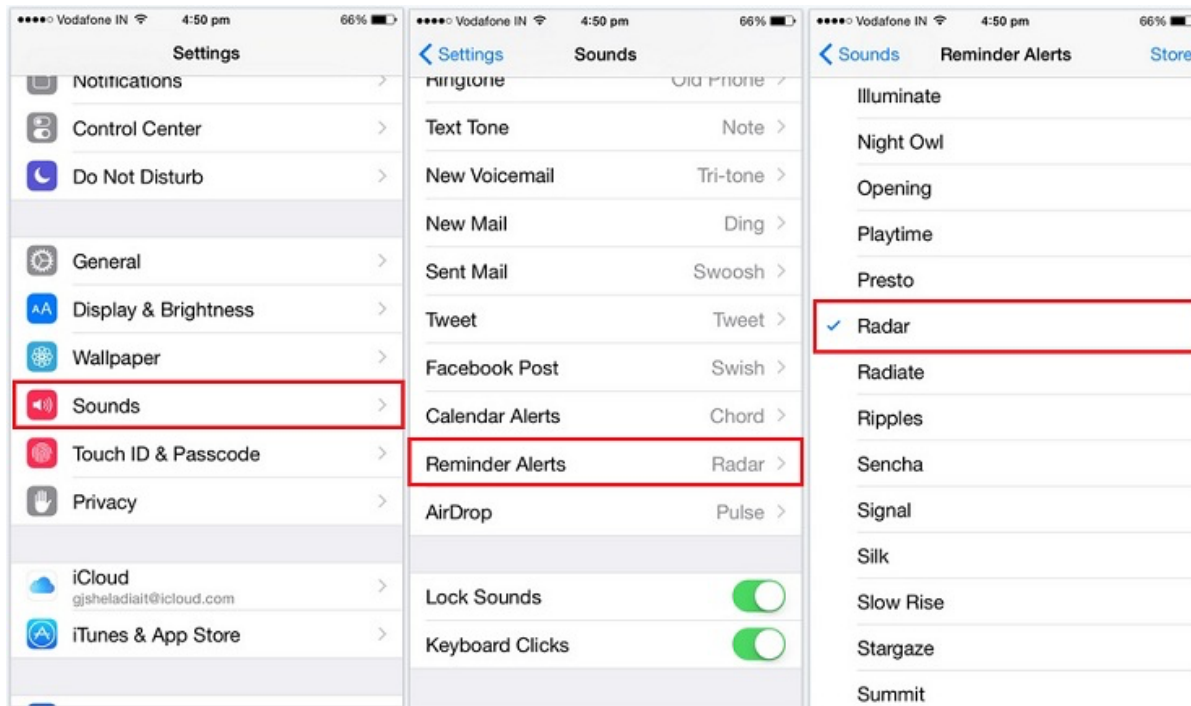
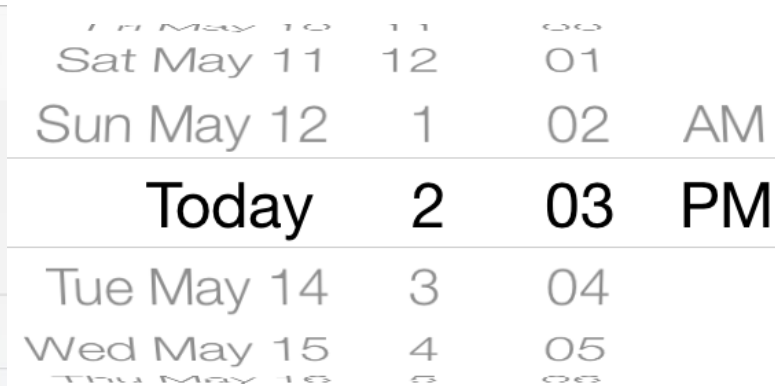
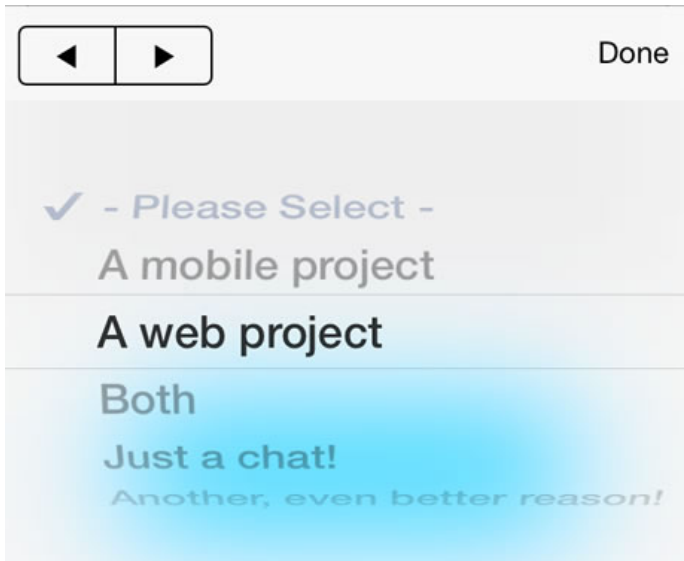


M-D v režimu Portrait





Pickery



Ekosystém zařízení

- Předpokládáme uživatele s více zařízeními (macOS, iOS).
- Potřeba přenášet soubory:
 - Soubory jsou pořizovány aplikacemi. Sandbox.
- Konfigurace aplikace: UserDefaults, iCloud Key-Value Store

Objektové kontejnery

- CoreData — interní OO DB. Synchronní.
- CloudKit — klient—server objektový kontejner.
 - Síťové operace.

Documents

- Strukturovaná data.
- Kódovací sub-systém (tar).
- Informace + metadata. Verzování.
- Synchronizace přes iCloud.

Paralelismus

- Vlákna.
- Operace.
- GCD.
- Fronty.

Herní engine

- SpriteKit.
- Herní "view" je součástí aplikace.
- Dynamika těles v čase. Diskrétní simulátor.

Závěr

- Apple měl / má obrovský přínos do stylu UI aplikací, počítačů a uživatelských zařízení.
- Dlouhodobá a stabilní koncepce.
- Programování zařízení Apple (IZA), léto, BP.