Using view controllers wisely

Samuel Défago, hortis le studio

October 2011

{{ softshake }}

iOS

```
#ifdef ago
#endif
```

Who am I?

- iOS developer at hortis le studio (www.hortis.ch)
- Mail: defagos (at) gmail (dot) com
- Twitter: @defagos
- Blog: http://subjective-objective-c.blogspot.com/
- Open-source: https://github.com/defagos

Motivations

- VCs are fundamental iOS objects, ...
- ... are used and implemented every day...
- ... but are often misused and abused
- When correctly used, they lead to more reliable applications and superior code

Goals

Learn how to organize VCs in applications

Understand how to use containers

Avoid common pitfalls and errors

Misused? Abused?

- Messy implementation
- Incorrect resource management
- Often only slightly superior to Xcode templates (i.e. slightly superior to void)
- Poor separation of concerns, big balls of mud
- Monolithic application workflow

yep, here

Topics

Have a look in the corner to know which topic we are talking about

- Overview of view controllers & containers
- Using view controllers in an application
- Anatomy of containers

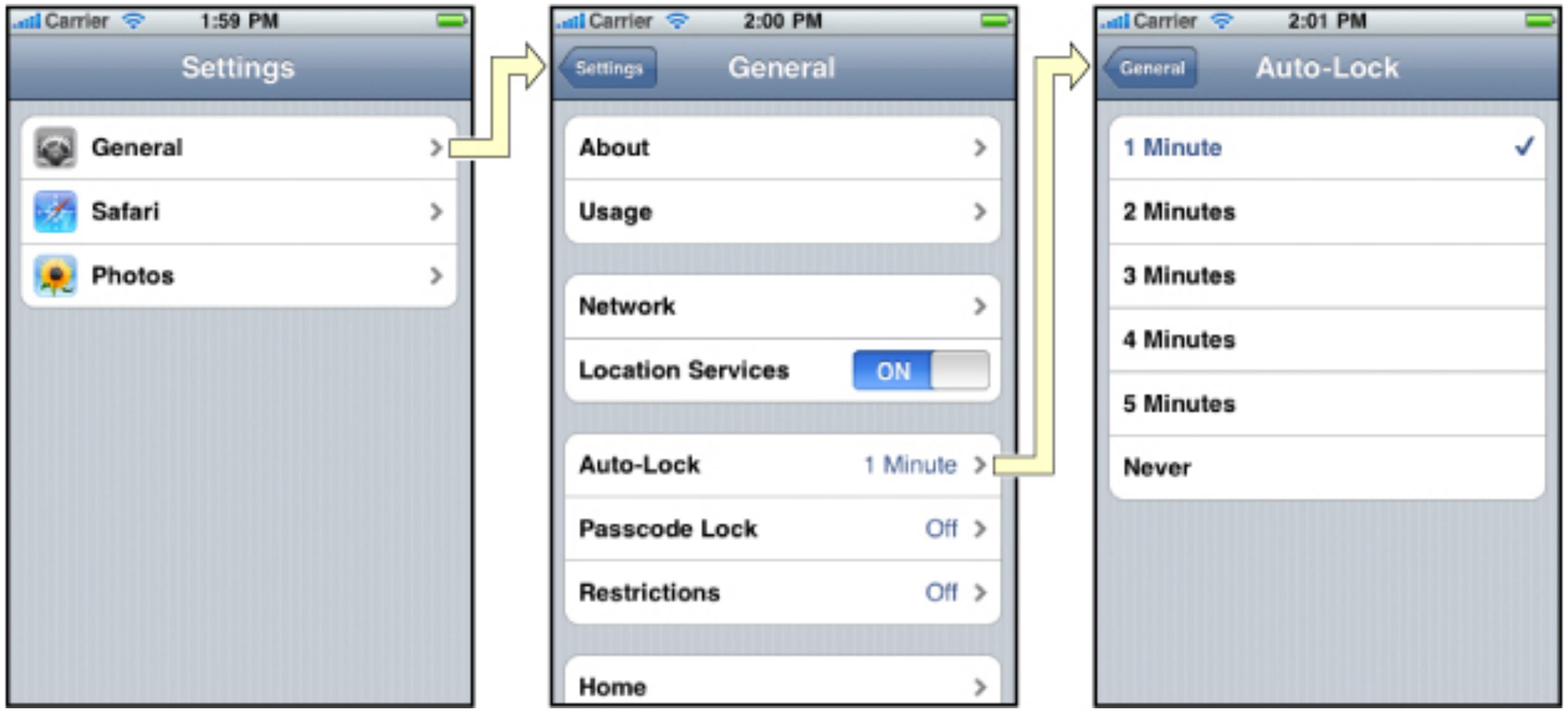
What is a view controller?

- is a controller: Bi-directional communication channel between view and model (MVC)
- has a view:View hierarchy management (lifecycle, orientation, low-memory conditions)
- Manages the content of one screen or of one part of a screen
- Subclass to create custom view controllers

What is a container?

- Is an object managing a set of view controllers
- In general used as-is, not meant to be subclassed
- Helps managing the view controller hierarchy, i.e. the application workflow...
- ... but only the workflow it implements

UINavigationController



Location Services ON

Auto-Lock 1 Minute

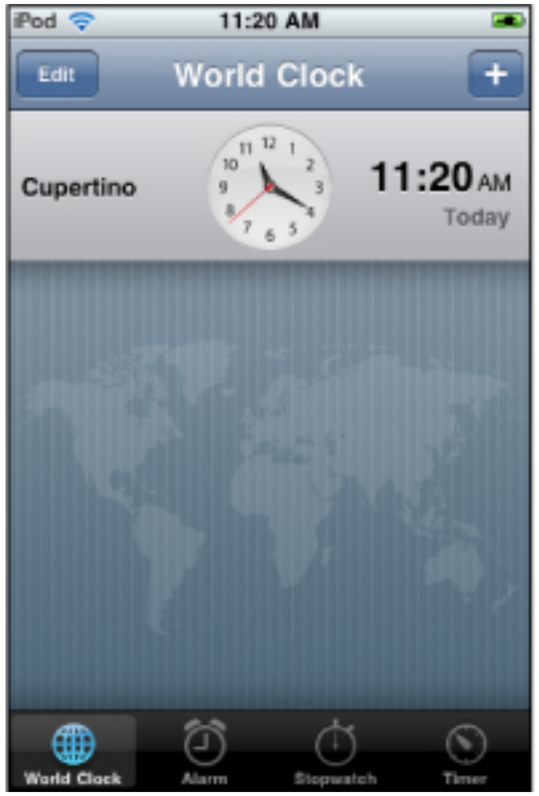
Passcode Lock Off

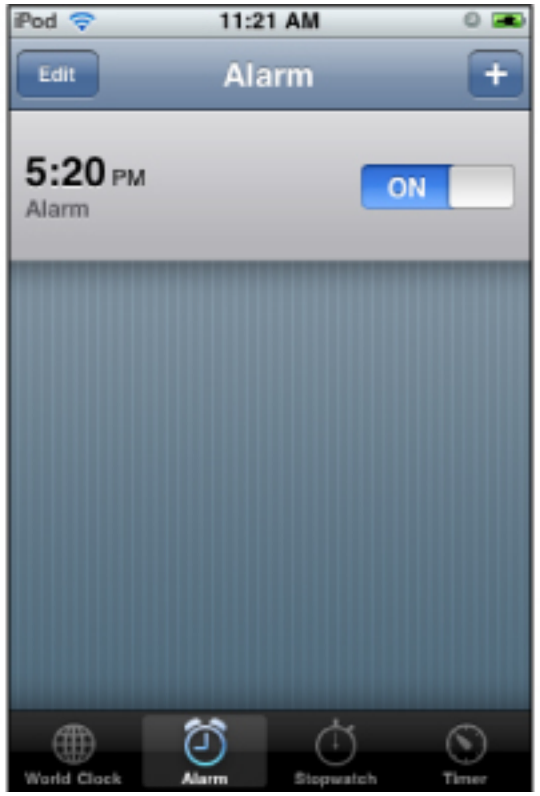
Restrictions Off

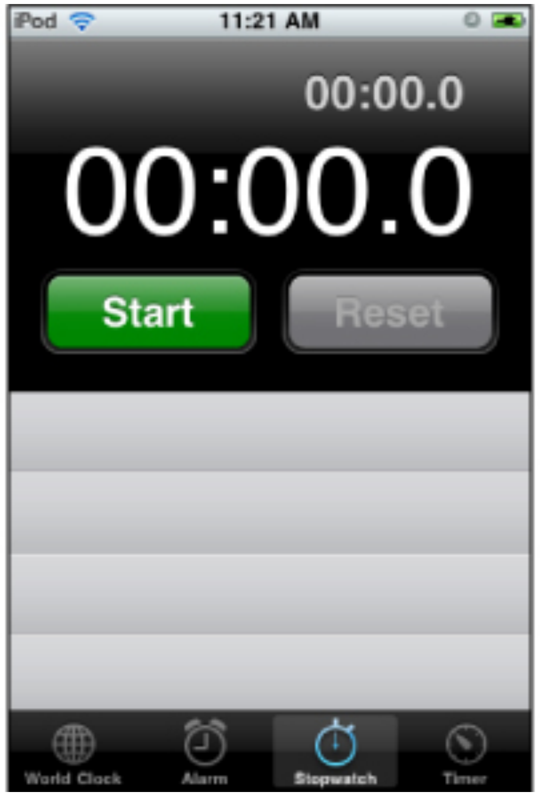
1 Minute ✓

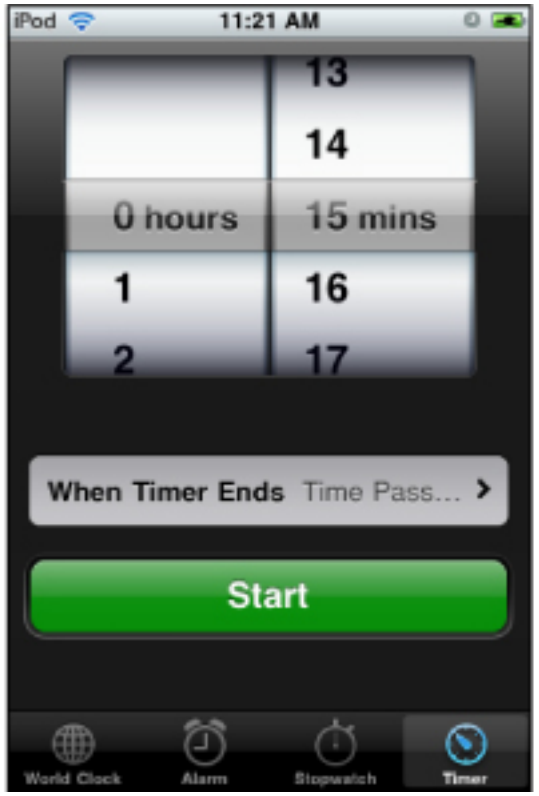
- Navigation bar at the top
- Push from right / left animations only

UITabBarController









- Tabs at the bottom
- Must be application root VC or root in modal

Criticism

- Few built-in containers...
- ... i.e. few well-supported workflows (navigation, tabs, modal, split content)
- This is annoying on iPad (emphasis on creative interfaces, use of both hands), less on iPhone (skinned interfaces, one hand usually suffices)

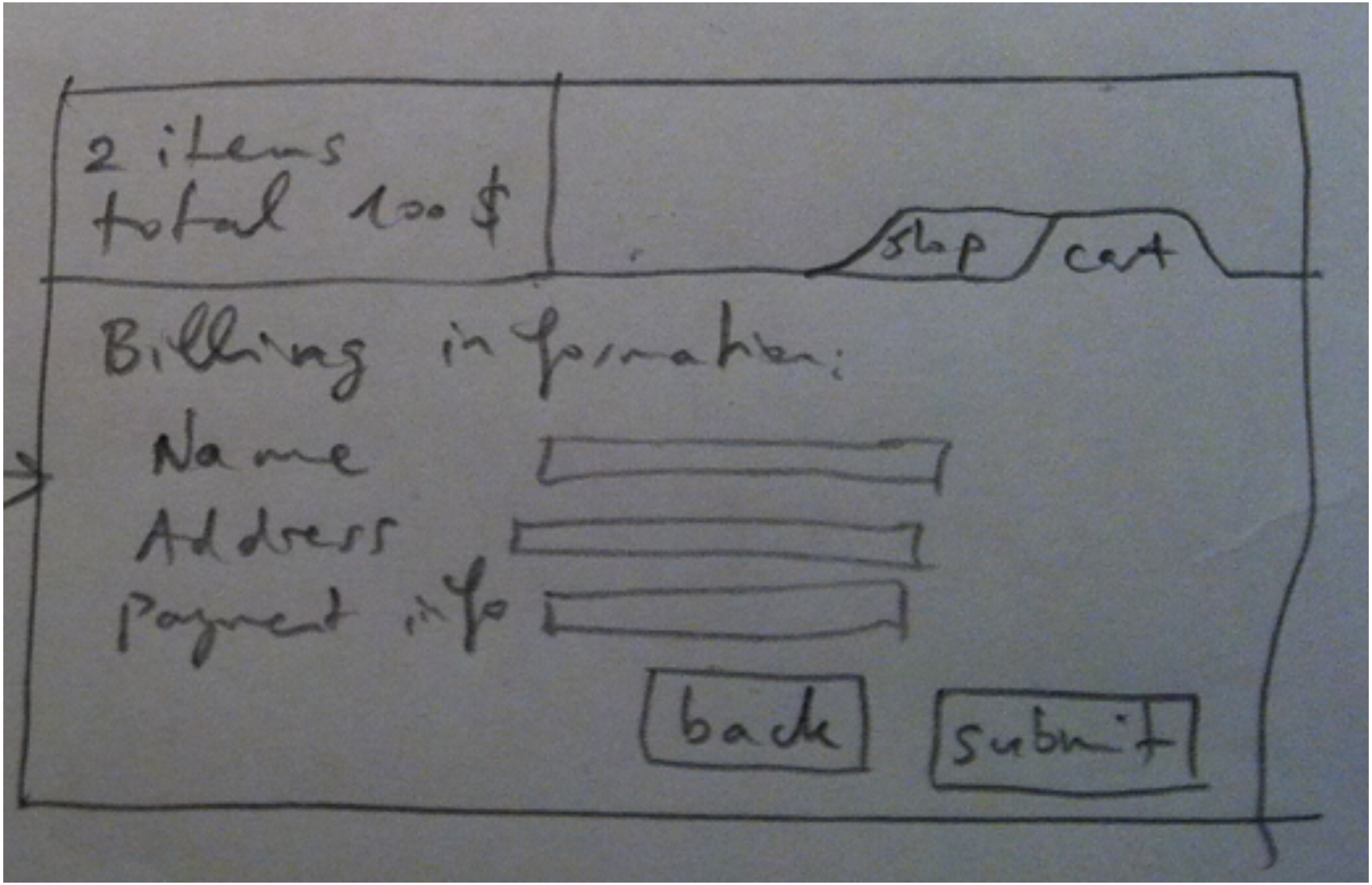
An iPad shopping application

- Application to buy items from some online store on the iPad
- Three people involved:
  - Product owner: Requirements
  - UI / UX designer: Design & interaction
  - Programmer: Write the application

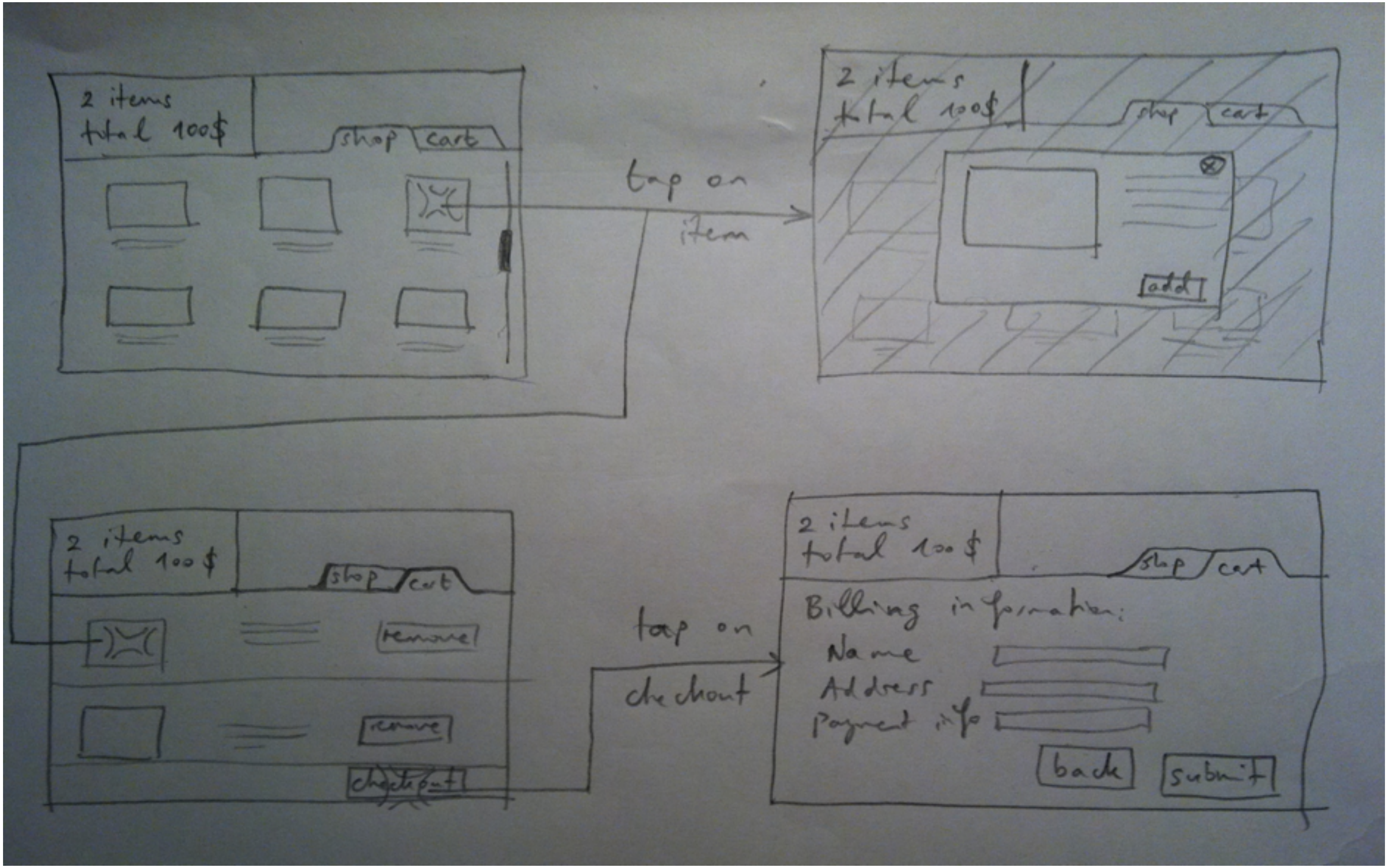
Product owner

- “I want the customer to be able to browse products and their descriptions”
- “I want the customer to be able to add items to its cart”
- “I want the customer to be able to access its cart and to submit an order”

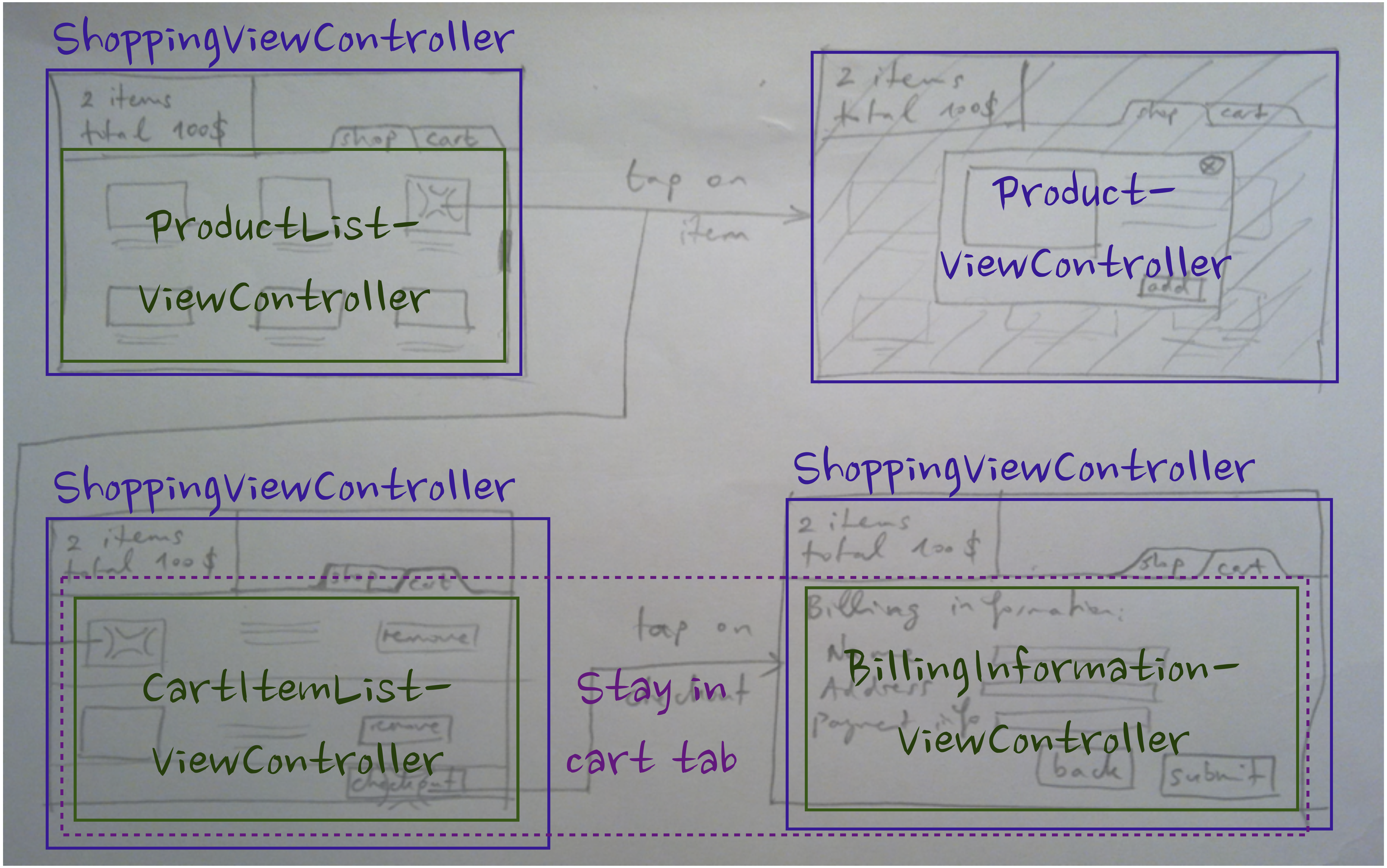
UI / UX designer (1)



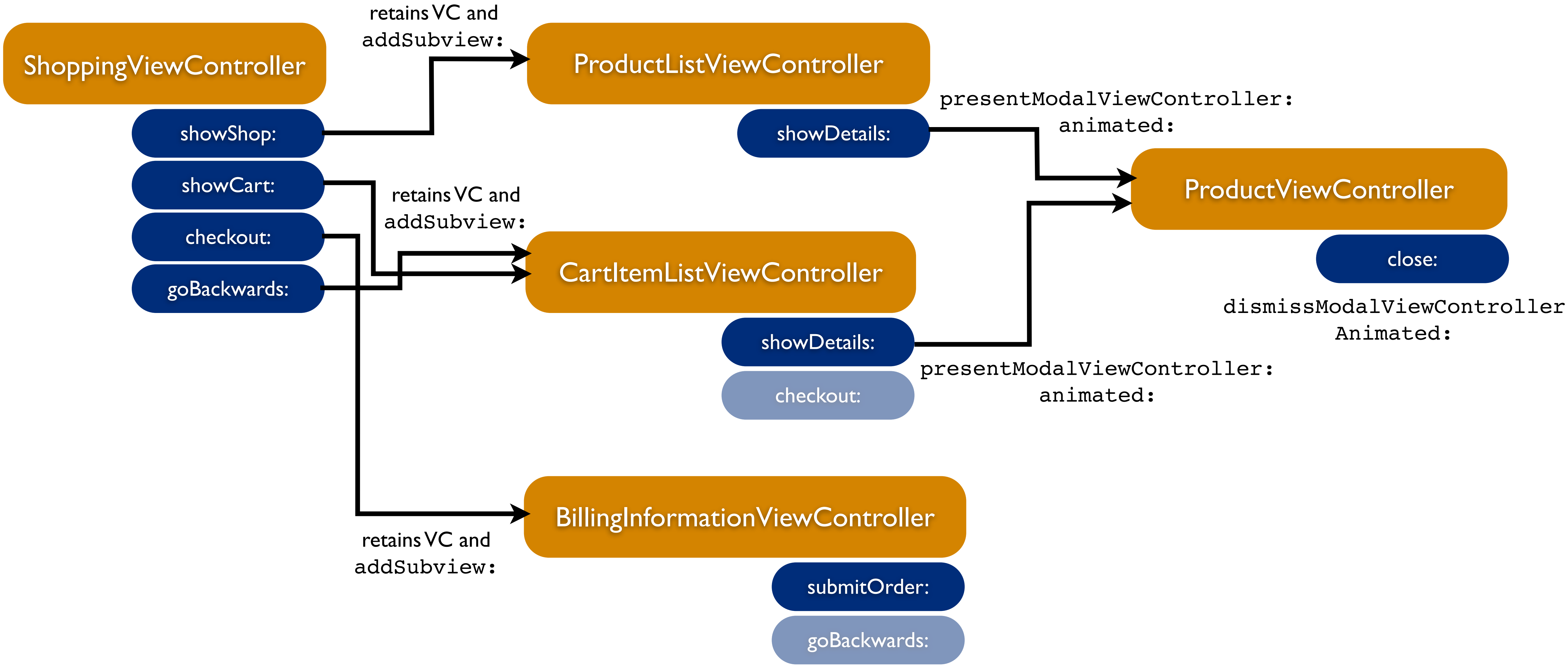
UI / UX designer (2)



Programmer:VCs



Programmer: Workflow


ShoppingViewController
showShop:
showCart:
checkout:
goBackwards:
retains VC and addSubview:
ProductListViewController
showDetails:
presentModalViewController: animated:
ProductViewController
close:
dismissModalViewController Animated:
retains VC and addSubview:
CartItemListViewController
showDetails:
checkout:
presentModalViewController: animated:
BillingInformationViewController
submitOrder:
goBackwards:
retains VC and addSubview:

Demo

Criticism

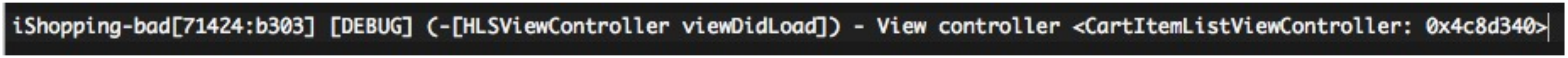
- No transparency when displaying product details
- ShoppingViewController manages almost all application workflow
- Seems to work, but does it REALLY work?

Analyzing the shopping application

- Use VCs logging all lifecycle and rotation events
- Navigate the application
- Rotate the device
- Simulate memory warnings
- Check the logs

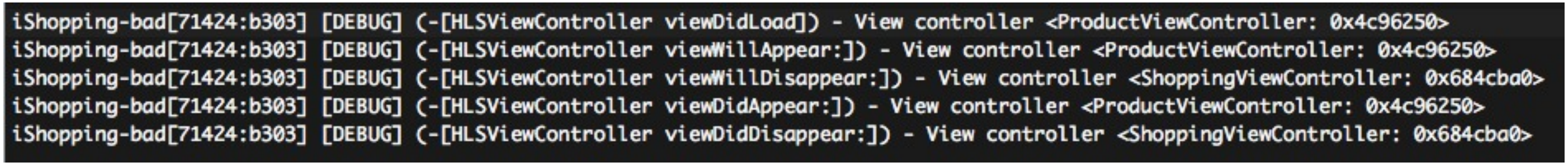
Log archaeology (1)

- Click on Shop tab, then on Cart tab:



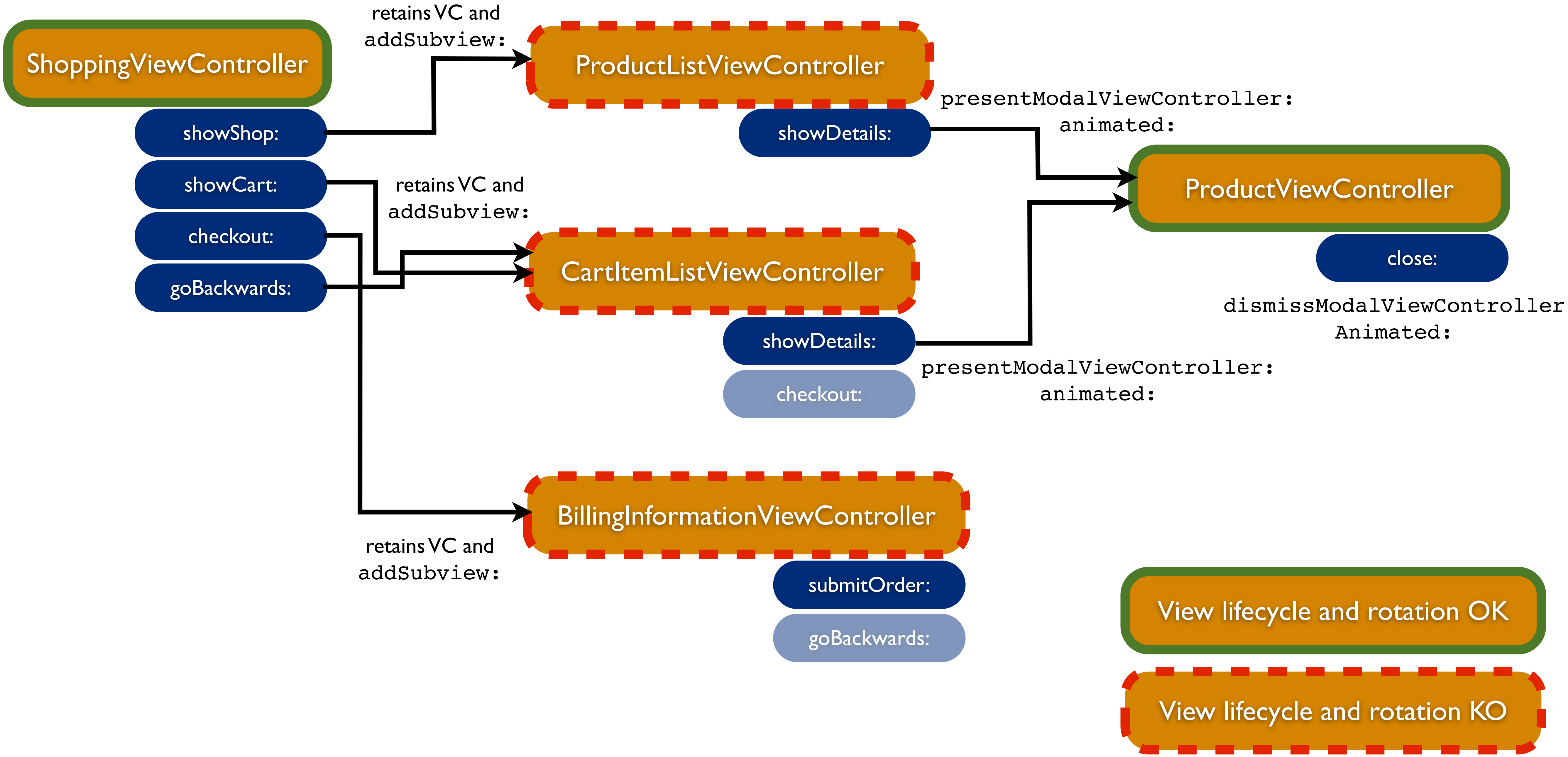
```
iShopping-bad[71424:b303] [DEBUG] (-[HLSViewController viewDidLoad]) - View controller <CartItemListViewController: 0x4c8d340>
```

- Show product details



```
iShopping-bad[71424:b303] [DEBUG] (-[HLSViewController viewDidLoad]) - View controller <ProductViewController: 0x4c96250>
iShopping-bad[71424:b303] [DEBUG] (-[HLSViewController viewWillAppear:]) - View controller <ProductViewController: 0x4c96250>
iShopping-bad[71424:b303] [DEBUG] (-[HLSViewController viewWillDisappear:]) - View controller <ShoppingViewController: 0x684cba0>
iShopping-bad[71424:b303] [DEBUG] (-[HLSViewController viewDidAppear:]) - View controller <ProductViewController: 0x4c96250>
iShopping-bad[71424:b303] [DEBUG] (-[HLSViewController viewDidDisappear:]) - View controller <ShoppingViewController: 0x684cba0>
```

Log archaeology (2)



Conclusion

- Creating a view controllers hierarchy using `addSubview:` leads to incorrect behavior
- View lifecycle and rotation events are not emitted automagically
- The root and modal view controllers have a proper behavior

Always use containers to manage your view controller hierarchy

Custom containers

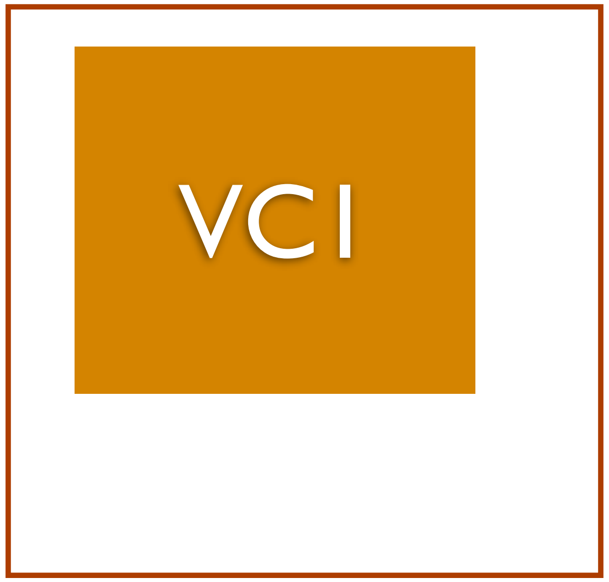
- Custom containers are needed for more creative workflows
- Using custom containers must be easy...
- ... but writing custom containers is hard

Elementary containers

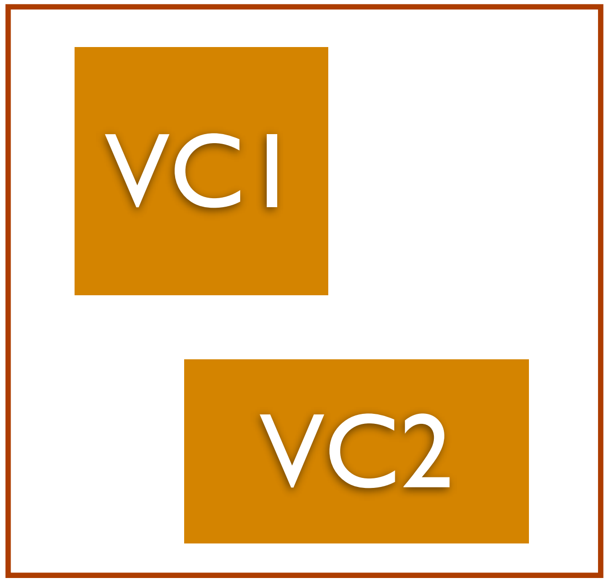
- Writing containers is hard...
- ... so write few correct elementary containers...
- ... and combine them to create new workflows...
- ... or even new containers!

Placeholders

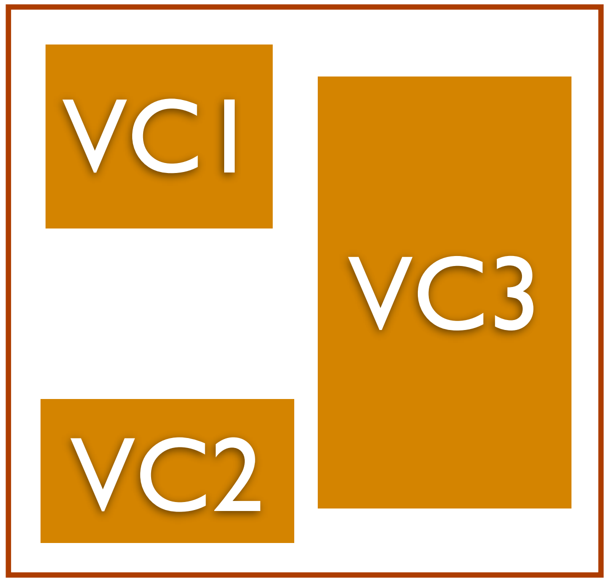
- Embed some other view controllers
- Display VCs simultaneously
- Can display other UI elements



1-placeholder



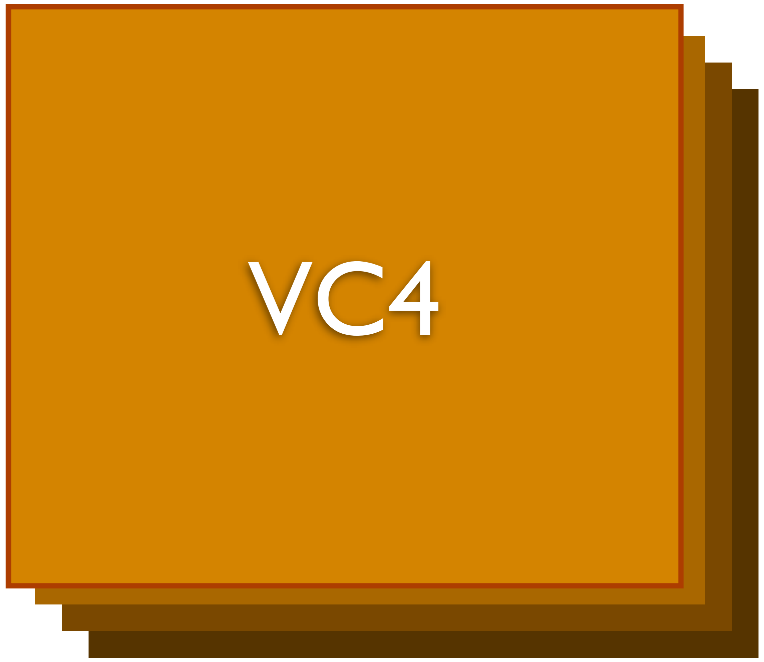
2-placeholder



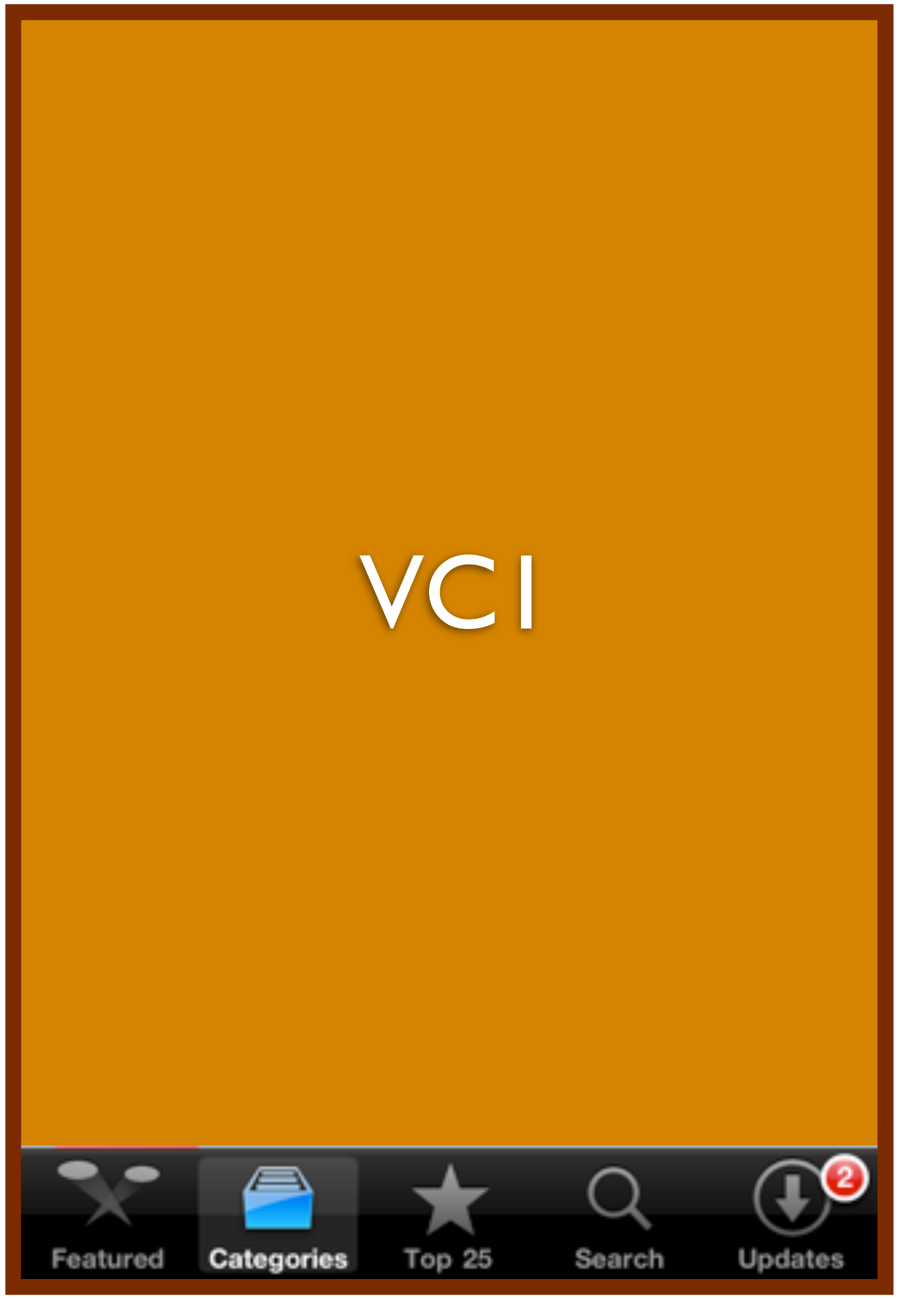
3-placeholder

Stack

- Manage a stack of view controllers
- Usually only display the top VC
- Fill the whole view, no other UI elements



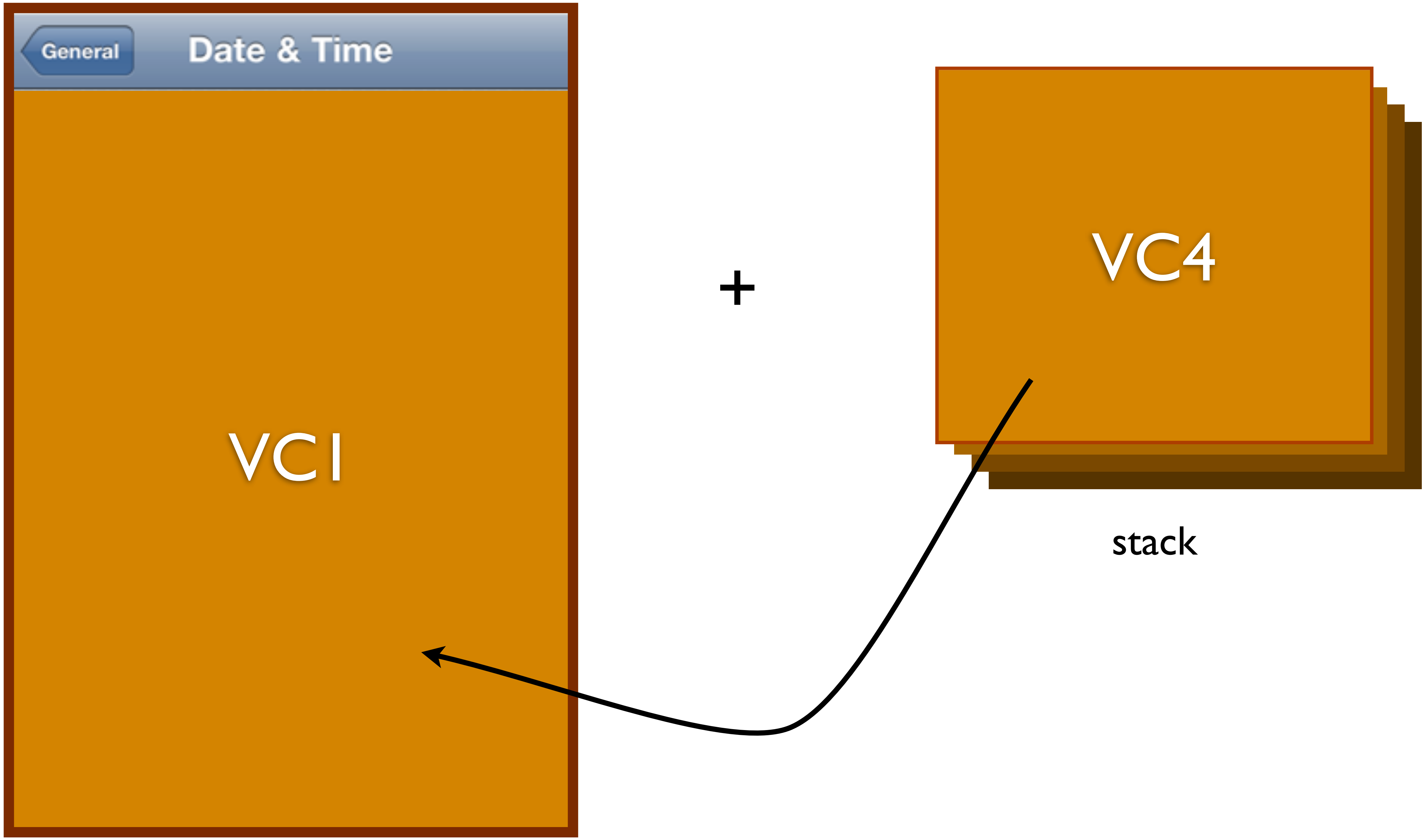
Combinations (1)



= custom UITabBarController

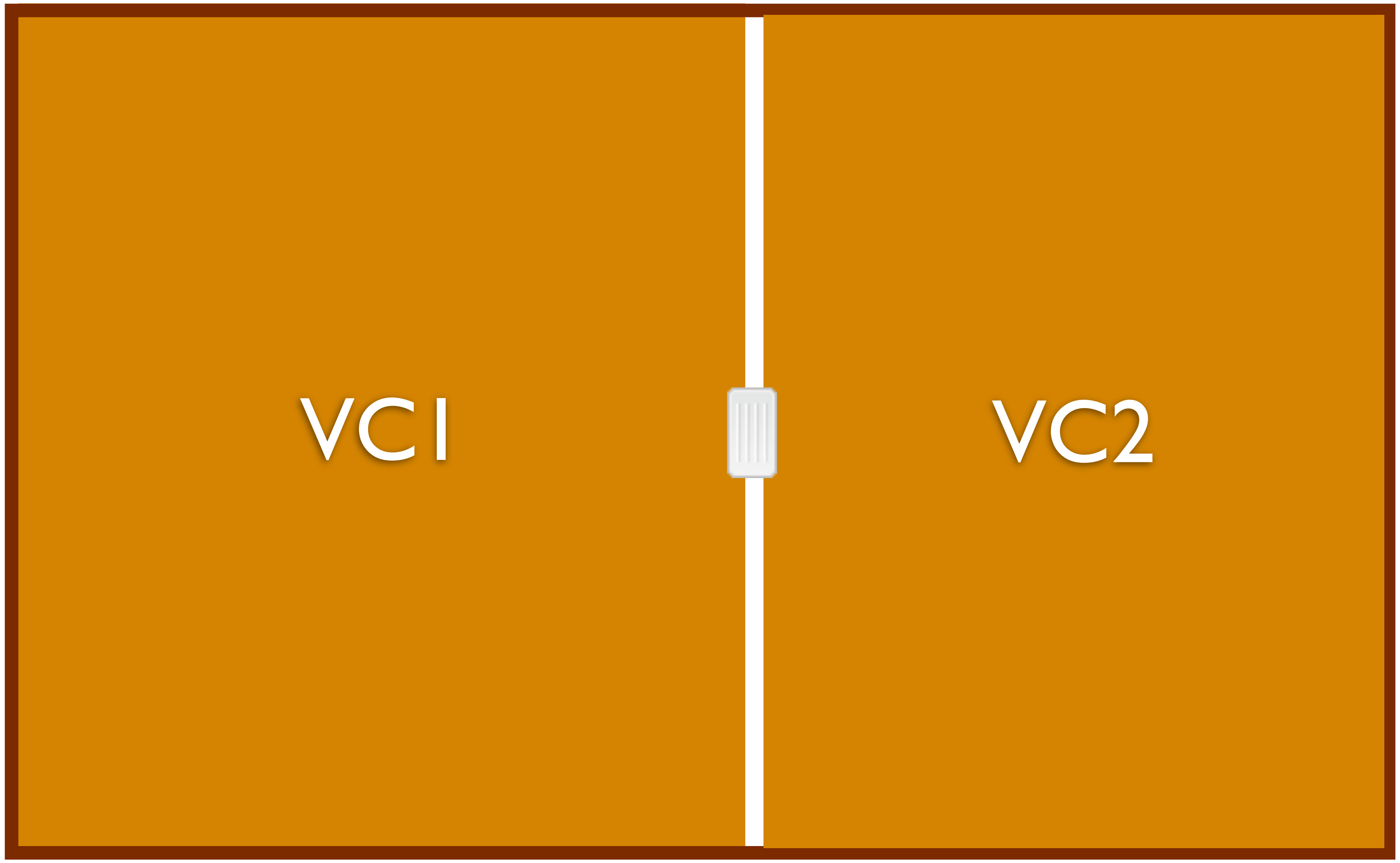
1-placeholder

Combinations (2)



= custom UINavigationController

Combinations (3)



VC1

VC2

2-placeholder

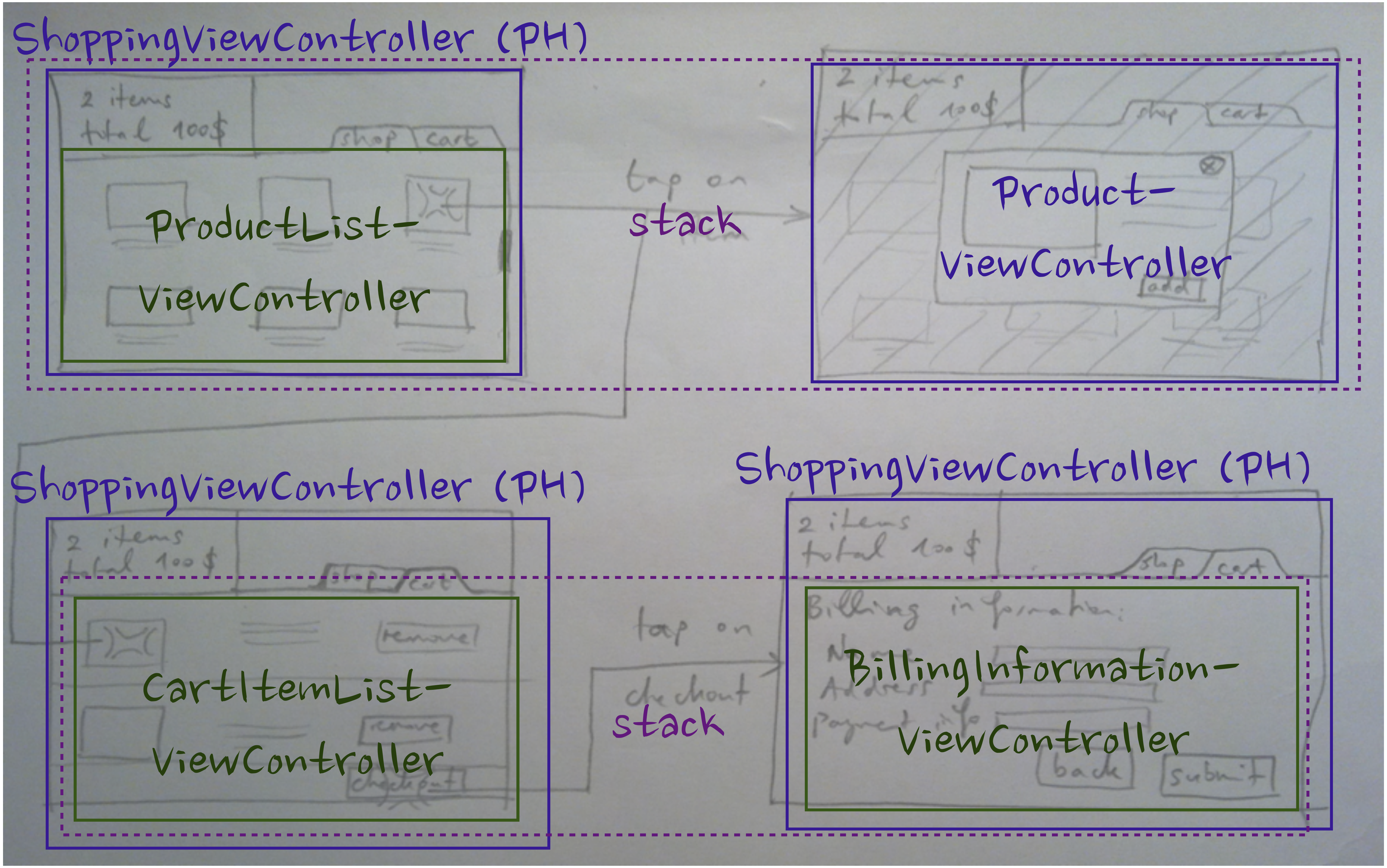
= resizable UISplitViewController

Implementation

- CoconutKit (https://github.com/defagos/CoconutKit):
  - HLSPlaceholderViewController
  - HLSStackController
- Technical details: http://subjective-objective-c.blogspot.com/2011/08/writing-high-quality-view-controller.html

Demo

iShopping revisited



ShoppingViewController (PH)

ProductList-ViewController

stack

Product-ViewController

ShoppingViewController (PH)

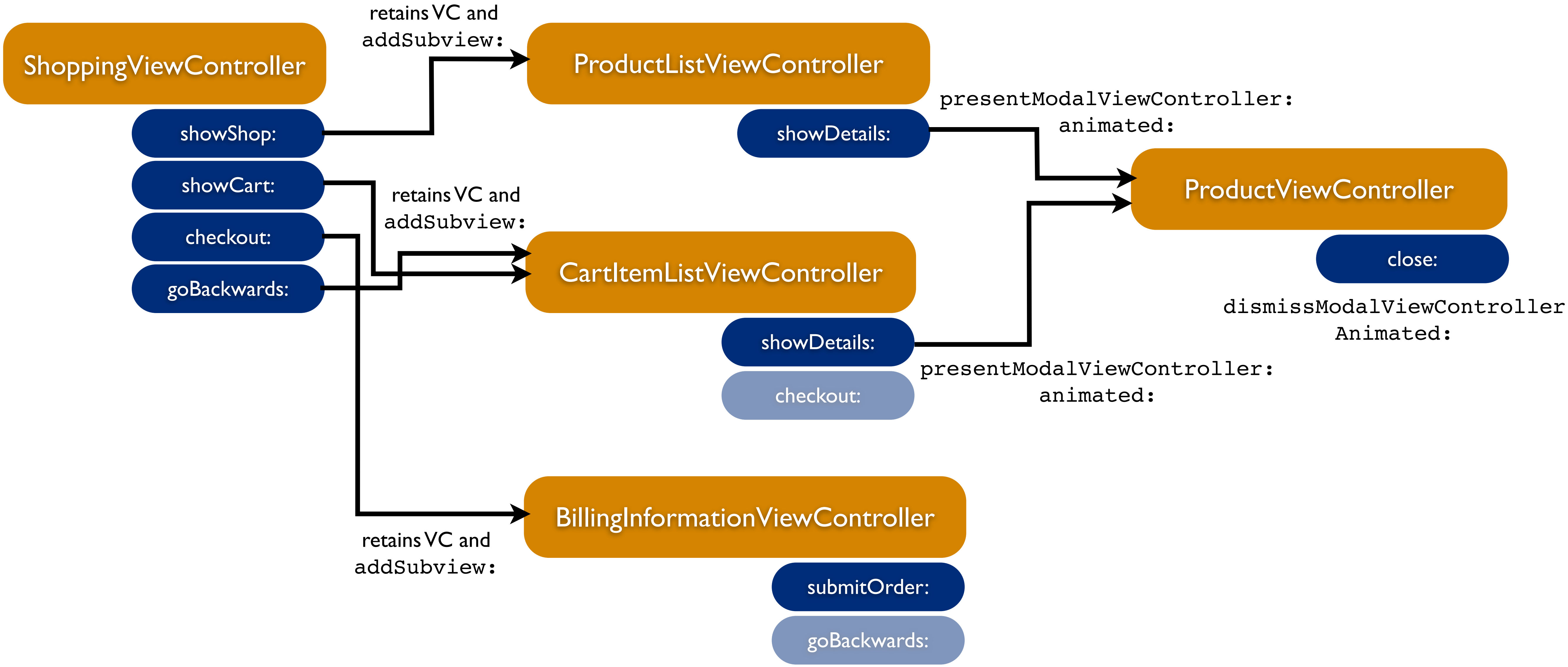
CartItemList-ViewController

stack

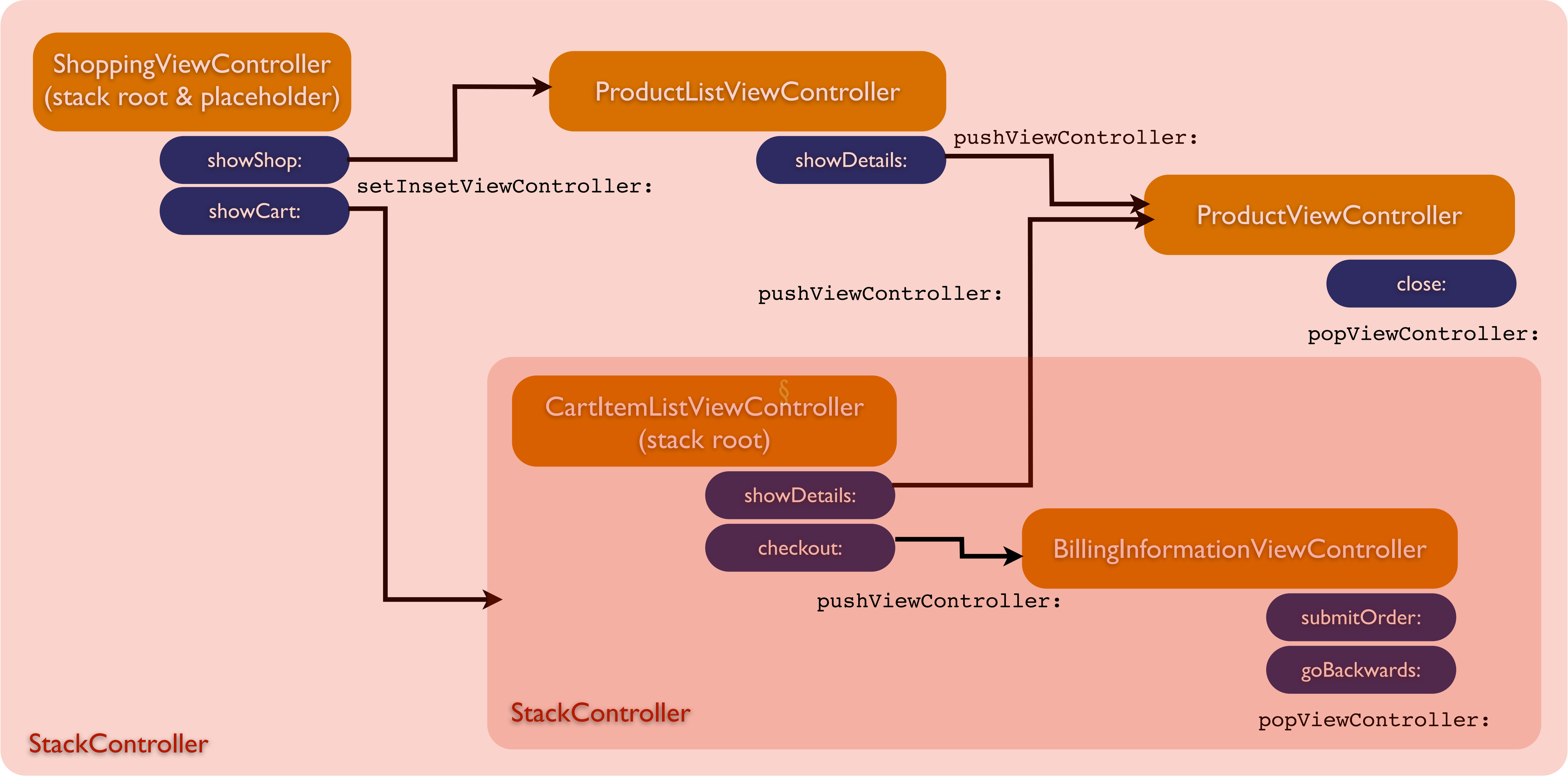
ShoppingViewController (PH)

BillingInformation-ViewController

Buggy workflow



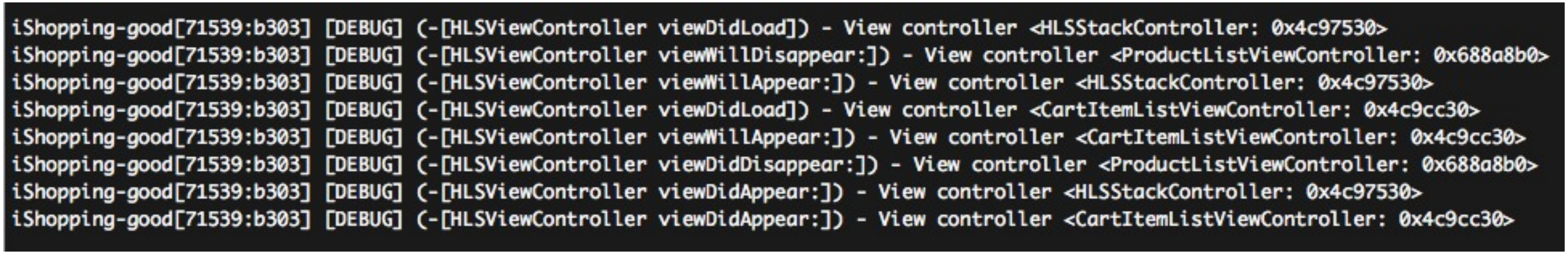
Fixed workflow



Demo

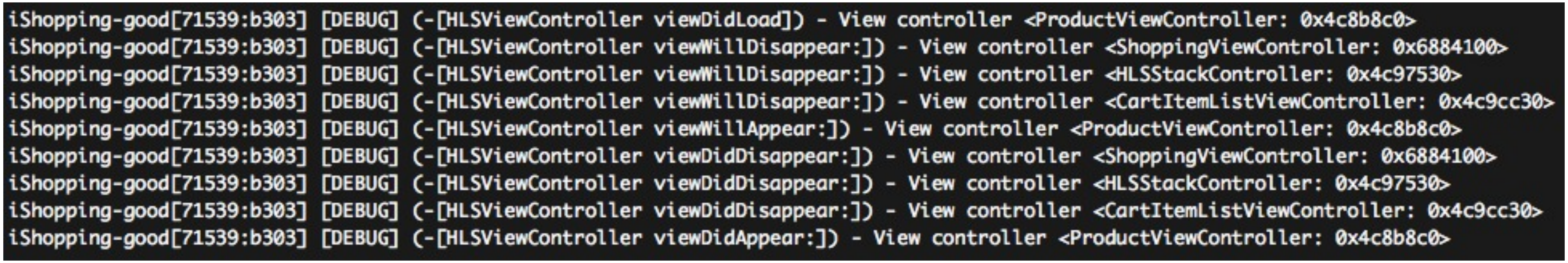
Log archaeology

- Click on Cart tab when on Shop tab:



```
iShopping-good[71539:b303] [DEBUG] (-[HLSViewController viewDidLoad]) - View controller <HLSStackController: 0x4c97530>
iShopping-good[71539:b303] [DEBUG] (-[HLSViewController viewWillDisappear:]) - View controller <ProductListViewController: 0x688a8b0>
iShopping-good[71539:b303] [DEBUG] (-[HLSViewController viewWillAppear:]) - View controller <HLSStackController: 0x4c97530>
iShopping-good[71539:b303] [DEBUG] (-[HLSViewController viewDidLoad]) - View controller <CartItemListViewController: 0x4c9cc30>
iShopping-good[71539:b303] [DEBUG] (-[HLSViewController viewWillAppear:]) - View controller <CartItemListViewController: 0x4c9cc30>
iShopping-good[71539:b303] [DEBUG] (-[HLSViewController viewDidDisappear:]) - View controller <ProductListViewController: 0x688a8b0>
iShopping-good[71539:b303] [DEBUG] (-[HLSViewController viewDidAppear:]) - View controller <HLSStackController: 0x4c97530>
iShopping-good[71539:b303] [DEBUG] (-[HLSViewController viewDidAppear:]) - View controller <CartItemListViewController: 0x4c9cc30>
```

- Show product details



```
iShopping-good[71539:b303] [DEBUG] (-[HLSViewController viewDidLoad]) - View controller <ProductViewController: 0x4c8b8c0>
iShopping-good[71539:b303] [DEBUG] (-[HLSViewController viewWillDisappear:]) - View controller <ShoppingViewController: 0x6884100>
iShopping-good[71539:b303] [DEBUG] (-[HLSViewController viewWillDisappear:]) - View controller <HLSStackController: 0x4c97530>
iShopping-good[71539:b303] [DEBUG] (-[HLSViewController viewWillDisappear:]) - View controller <CartItemListViewController: 0x4c9cc30>
iShopping-good[71539:b303] [DEBUG] (-[HLSViewController viewWillAppear:]) - View controller <ProductViewController: 0x4c8b8c0>
iShopping-good[71539:b303] [DEBUG] (-[HLSViewController viewDidDisappear:]) - View controller <ShoppingViewController: 0x6884100>
iShopping-good[71539:b303] [DEBUG] (-[HLSViewController viewDidDisappear:]) - View controller <HLSStackController: 0x4c97530>
iShopping-good[71539:b303] [DEBUG] (-[HLSViewController viewDidDisappear:]) - View controller <CartItemListViewController: 0x4c9cc30>
iShopping-good[71539:b303] [DEBUG] (-[HLSViewController viewDidAppear:]) - View controller <ProductViewController: 0x4c8b8c0>
```

Summary

- Never ever display a view controller by adding its view as subview directly
- Design your view controller workflow, and manage it using view controller containers
- Use the view controller lifecycle correctly to insert your code where most appropriate

Thanks for your attention

Questions?