A Case Study in Storyboards

Martin Winter

CocoaHeads Aachen, 29.3.2012

About me

Design + Development

“I was a Mac user when Apple was doomed”

Indie + freelancer since March 1, 2012!
(translation: I’m available for projects)

About this talk

Get you interested in using storyboards

Share practical experiences and tips

Case study

"BasisBibel" for German Bible Society

Universal app (iPhone + iPad)

Additional content, more than a simple reader, thus moderately complex UI

Case study

Video: A few features of the iPad version

Carrier
7:22 PM
100%
NT
Basis
Bibel
BasisBibel
Safari
Photos

What are storyboards?

Files containing UI definitions of multiple “scenes”

Scenes (view controllers) + transitions = flow
(≈ state machine)

Supplement standard nibs (can be mixed)

What are storyboards?

iOS 5 Library > General > What's New in iOS > iOS 5.0 > Storyboards

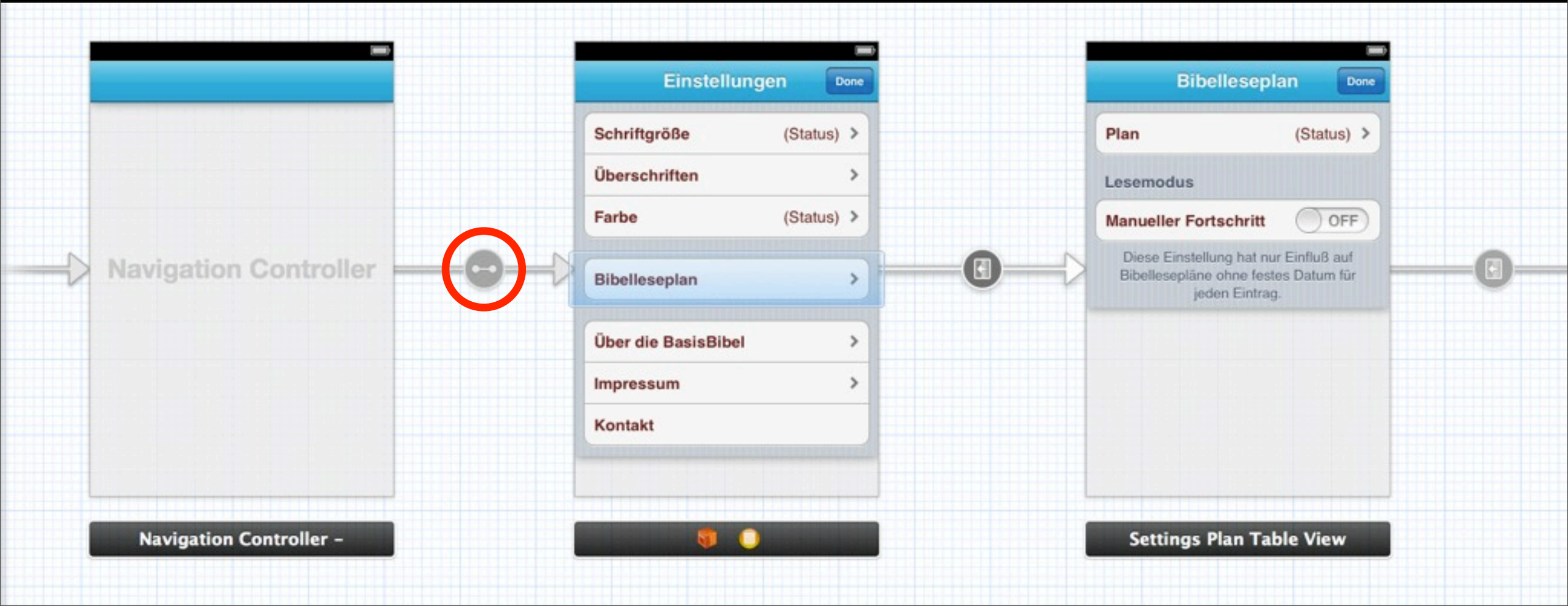
WWDC 2011, session 309:
"Introducing Interface Builder Storyboarding"

iDeveloper Live, episode 43:
"Didn't See That Mountain Coming!"
(Rich Warren)

Connections

Relationships, e. g. navigation controller–root

Segues, e. g. Push, Modal, Popover



Connections

Compositions, example: Branch

Video: Branched views on iPhone

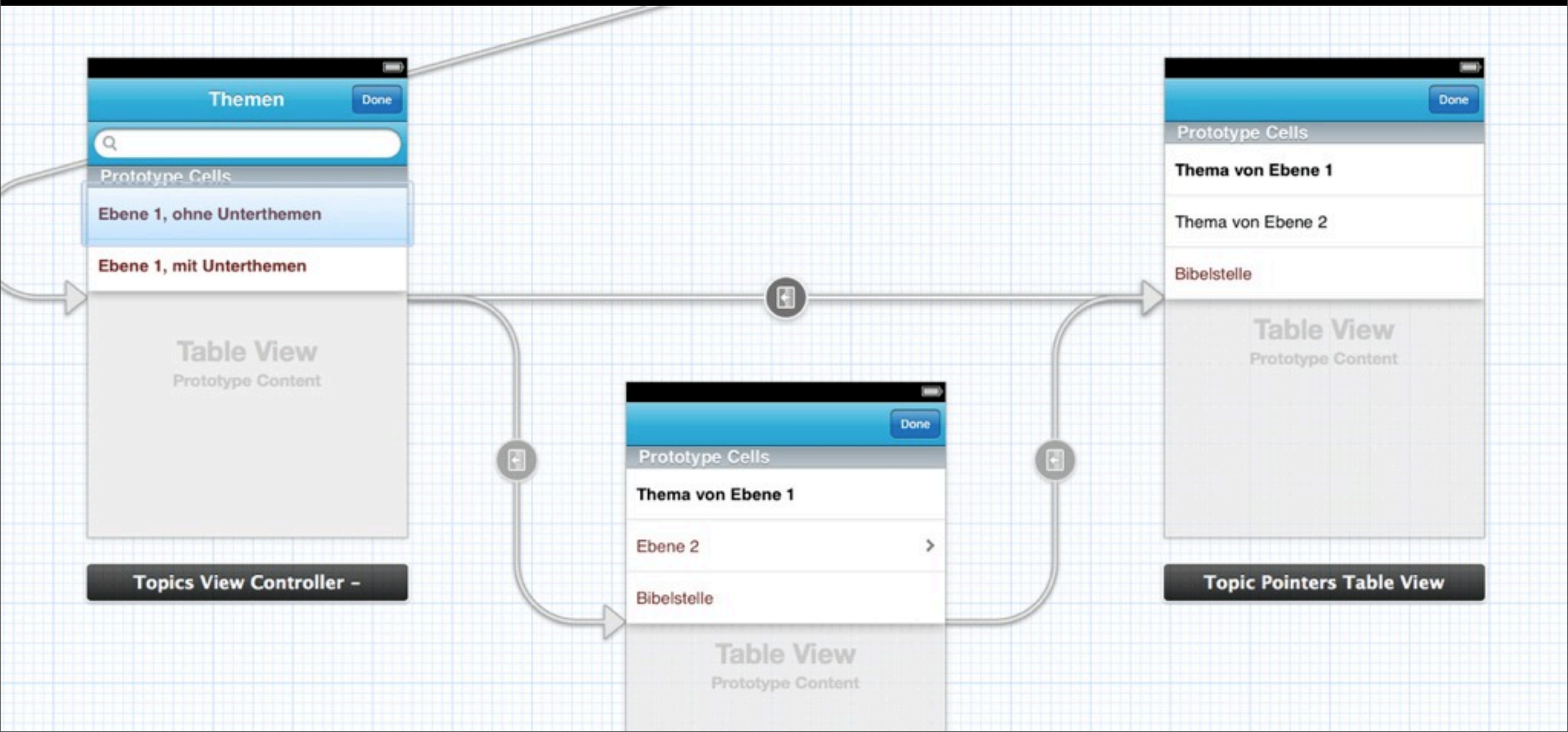
Carrier 6:56 PM

BasisBibel

Safari Photos

Connections

Compositions, example: Branch



Connections

More complex compositions possible,
such as loops and multiple in/out connections

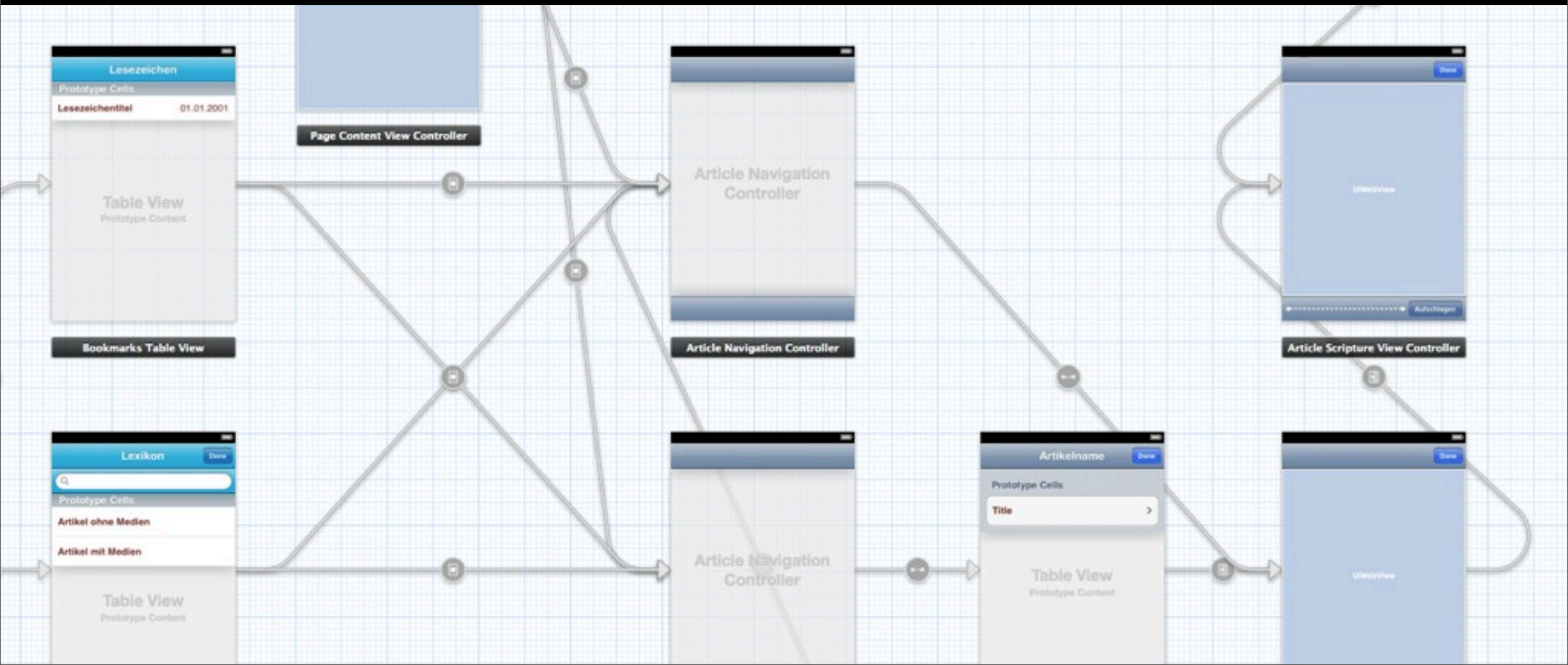
Video: Loops and multiple connections on iPhone

Die Sterndeuter aus dem Osten

2 [1] Jesus wurde in Betlehem in Judäa geboren
zu der Zeit, als Herodes König war.
Sieh doch:
Es kamen Sterndeuter aus dem Osten nach Jerusalem.
[2] Sie fragten:
"Wo ist der neugeborene König der Juden?
Denn wir haben seinen Stern im Osten gesehen.
Wir sind gekommen, um ihn anzubeten.
[3] Als König Herodes das hörte,

Connections

More complex compositions possible,
such as loops and multiple in/out connections



Connections

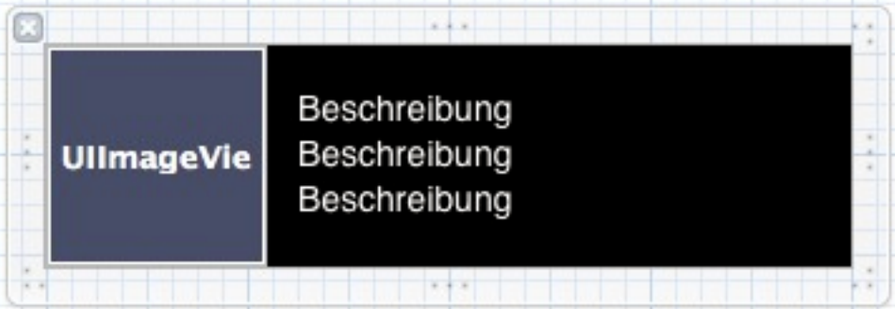
Configuration in `prepareForSegue:sender:`

Programmatic instantiation of view controllers
`instantiateViewControllerWithIdentifier:`

Programmatic performing of segues
`performSegueWithIdentifier:sender:`

Old-style nibs

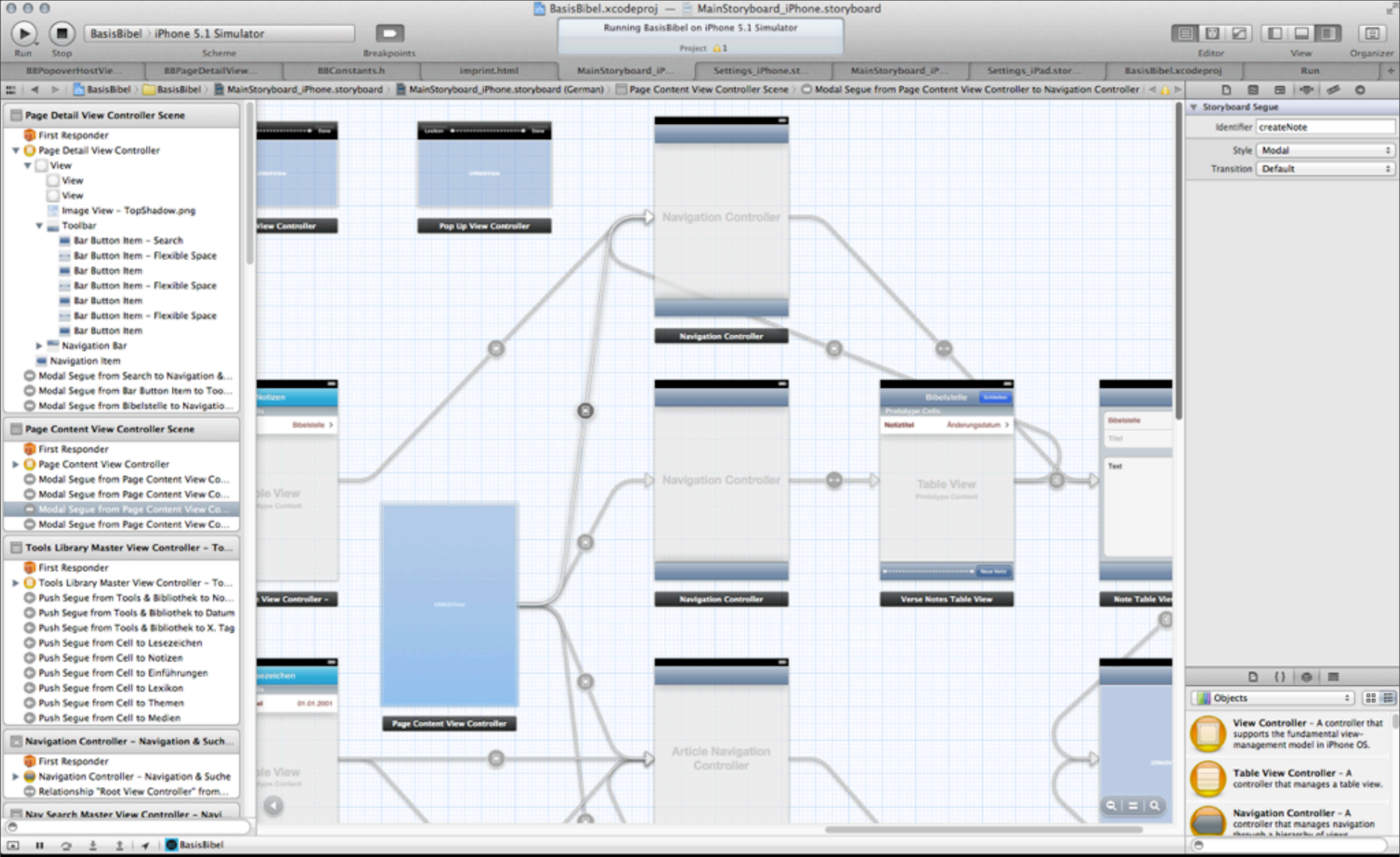
Classic use case: Subclassed table view cell



The good

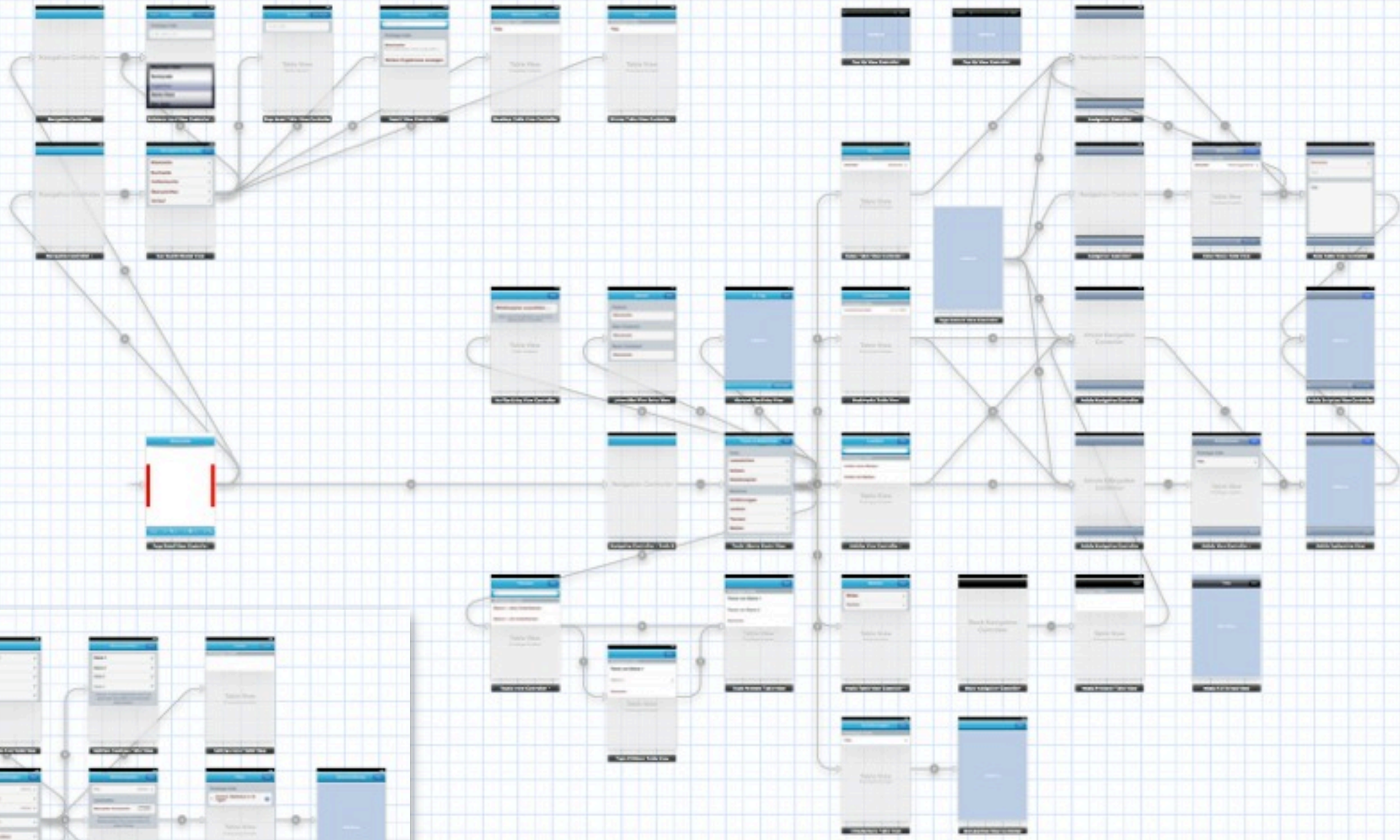
Visual overview of app's entire UI

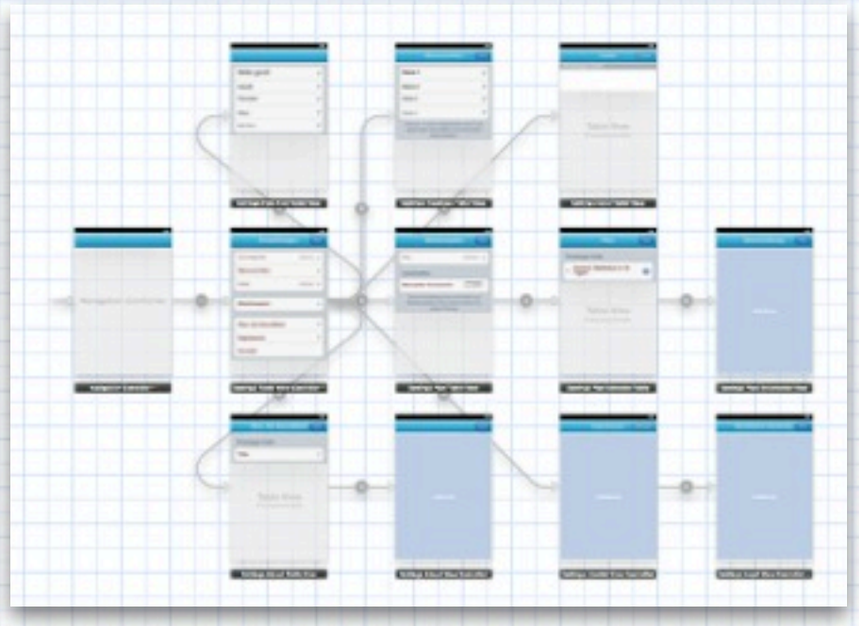
Better grasp of interactions and control flow



Storyboard in Xcode

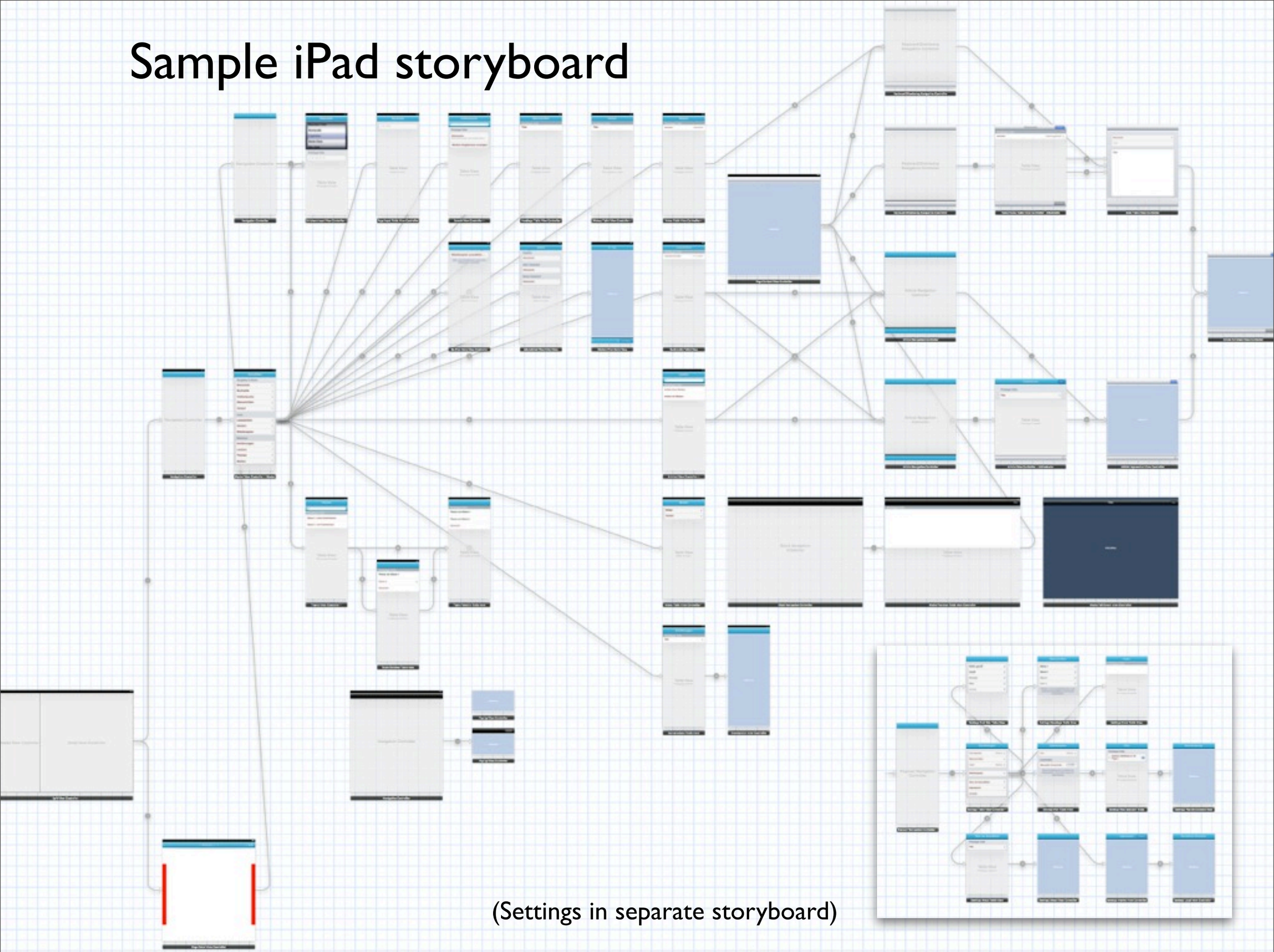
Sample iPhone storyboard





(Settings in separate storyboard)

Sample iPad storyboard



(Settings in separate storyboard)

The good

Visual overview of app's entire UI

Better grasp of interactions and control flow

Segues automatically instantiate view controllers (including popover controller!)

Custom segues (haven't tried this)

The good

Static table views! (not available in nibs)

Standard or custom cell types

Dynamic contents: View controller outlets!

BasisBibel.xcodeproj — Settings_iPhone.storyboard

Running BasisBibel on iPhone 5.1 Simulator

Project ⚠ 1

Editor View Organizer

MainStoryboard_iP... | Settings_iPhone.st... | MainStoryboard_iP... | Settings_iPad.stor... | BasisBibel.xcodeproj | Run

Settings Plan Table View Controller – Bibelleseplan Scene › Settings Plan Table View Controller – Bibelleseplan › Table View

ungen Done

(Status)

(Status)

el

ew Controller –

Bibelleseplan Done

Plan (Status)

Lesemodus

Manueller Fortschritt OFF

Diese Einstellung hat nur Einfluß auf Bibellesepläne ohne festes Datum für jeden Eintrag.

Table Vie

Dynamic Prototypes

Conte ✓ Static Cells

Sections 2

Style Grouped

Separator Single Line Etched

Default

Selection Single Selection

Editing No Selection During E...

Show Selection on Touch

Index Row Limit 0

Scroll View

Style Default

Scrollers Shows Horizontal Scrollers

Shows Vertical Scrollers

Scrolling Enabled

Paging Enabled

Direction Lock Enabled

Bounce Bounces

Bounce Horizontally

Bounce Vertically

Zoom 1 Min 1 Max

Touch Bounces Zoom

Delays Content Touches

Cancellable Content To...

View

Mode Scale To Fill

Tag 0

Interaction User Interaction Enabled

Multiple Touch

BasisBibel.xcodeproj — Settings_iPhone.storyboard

Running BasisBibel on iPhone 5.1 Simulator

Project ⚠ 1

Editor View Organizer

MainStoryboard_iP... | Settings_iPhone.st... | MainStoryboard_iP... | Settings_iPad.stor... | BasisBibel.xcodeproj | Run

Settings Plan Table View Cont... › Settings Plan Table View Cont... › Table View › Table View Section › Table View Cell

ungen Done

(Status)

(Status)

el

ew Controller –

Bibelleseplan Done

Plan (Status)

Lesemodus

Manueller Fortschritt OFF

Diese Einstellung hat nur Einfluß auf Bibellesepläne ohne festes Datum für jeden Eintrag.

Custom

Table Vie

Sty

Imag

Basic

✓ Right Detail

Left Detail

Subtitle

Identifier Reuse Identifier

Selection Gray

Accessory Disclosure Indicator

Editing Acc. None

Indentation 1 Level 0 Width

Indent While Editing

Shows Re-order Controls

View

Mode Scale To Fill

Tag 0

Interaction User Interaction Enabled

Multiple Touch

Alpha 1

Background Default

Drawing Opaque Hidden

Clears Graphics Context

Clip Subviews

Autoresize Subviews

Stretching 0 X 0 Y

1 Width 1 Height

BasisBibel.xcodeproj — Settings_iPhone.storyboard

Running BasisBibel on iPhone 5.1 Simulator

Project ⚠ 1

Editor View Organizer

MainStoryboard_iP... | Settings_iPhone.st... | MainStoryboard_iP... | Settings_iPad.stor... | BasisBibel.xcodeproj | Run

Plan Table Vie... › Settings Plan Table Vie... › Table View › Table View Section › Table View Cell › Label – (Status)

ungen Done

(Status)

(Status)

el

ew Controller –

Bibelleseplan Done

Plan (Status)

Lesemodus

Manueller Fortschritt OFF

Diese Einstellung hat nur Einfluß auf Bibellesepläne ohne festes Datum für jeden Eintrag.

Referencing Outlets

planNameLabel — Settings Plan T...

New Referencing Outlet

Referencing Outlet Collections

New Referencing Outlet Collection

The good

Static table views! (not available in nibs)

Standard or custom cell types

Dynamic contents: View controller outlets!

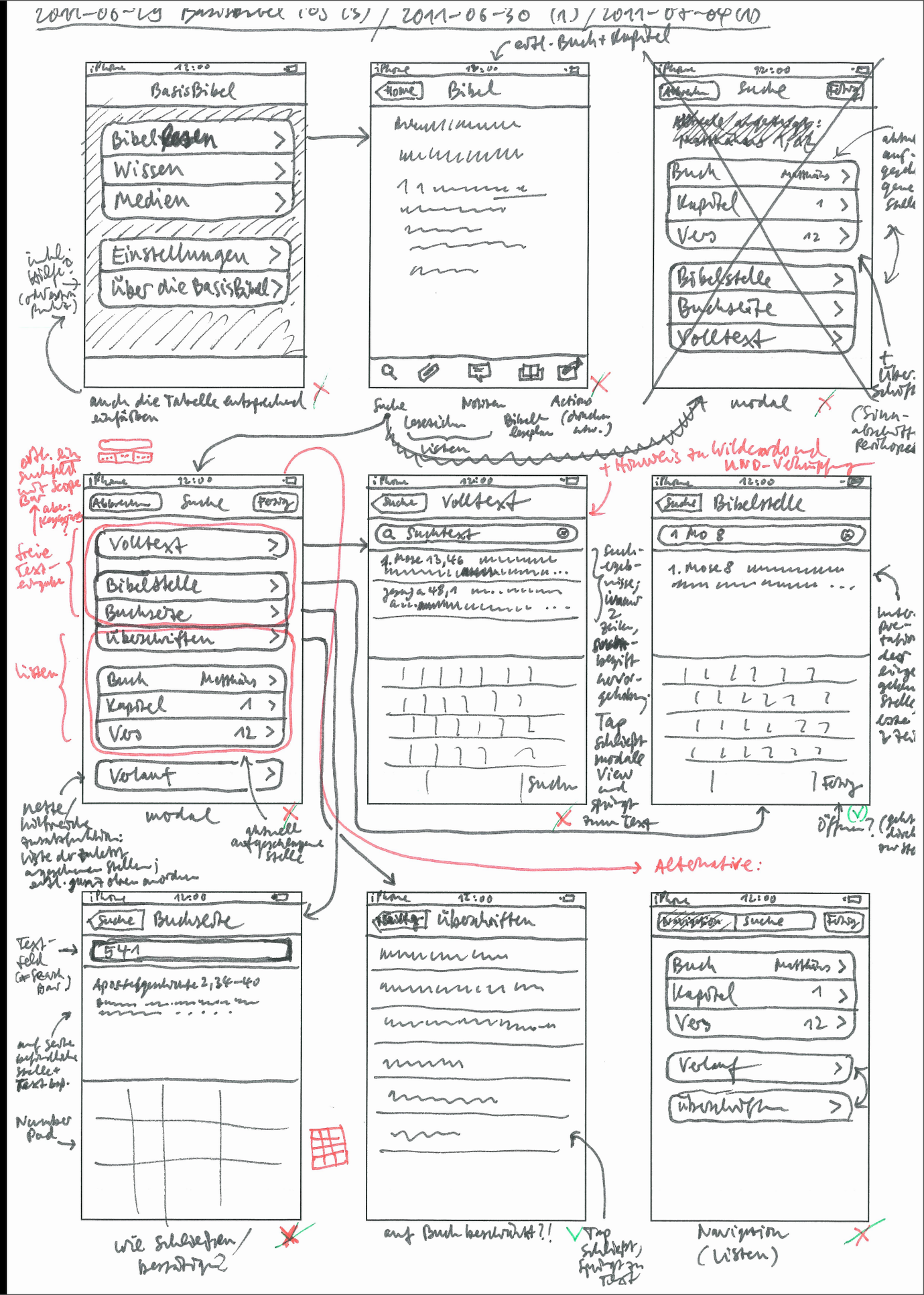
Combine with automatic cell loading:

```
UITableViewCell *cell = [tableView
    dequeueReusableCellWithIdentifier:CellIdentifier];

if (!cell)
{
    cell = [[UITableViewCell alloc]
              initWithStyle:UITableViewCellStyleDefault
            reuseIdentifier:CellIdentifier];
}
```

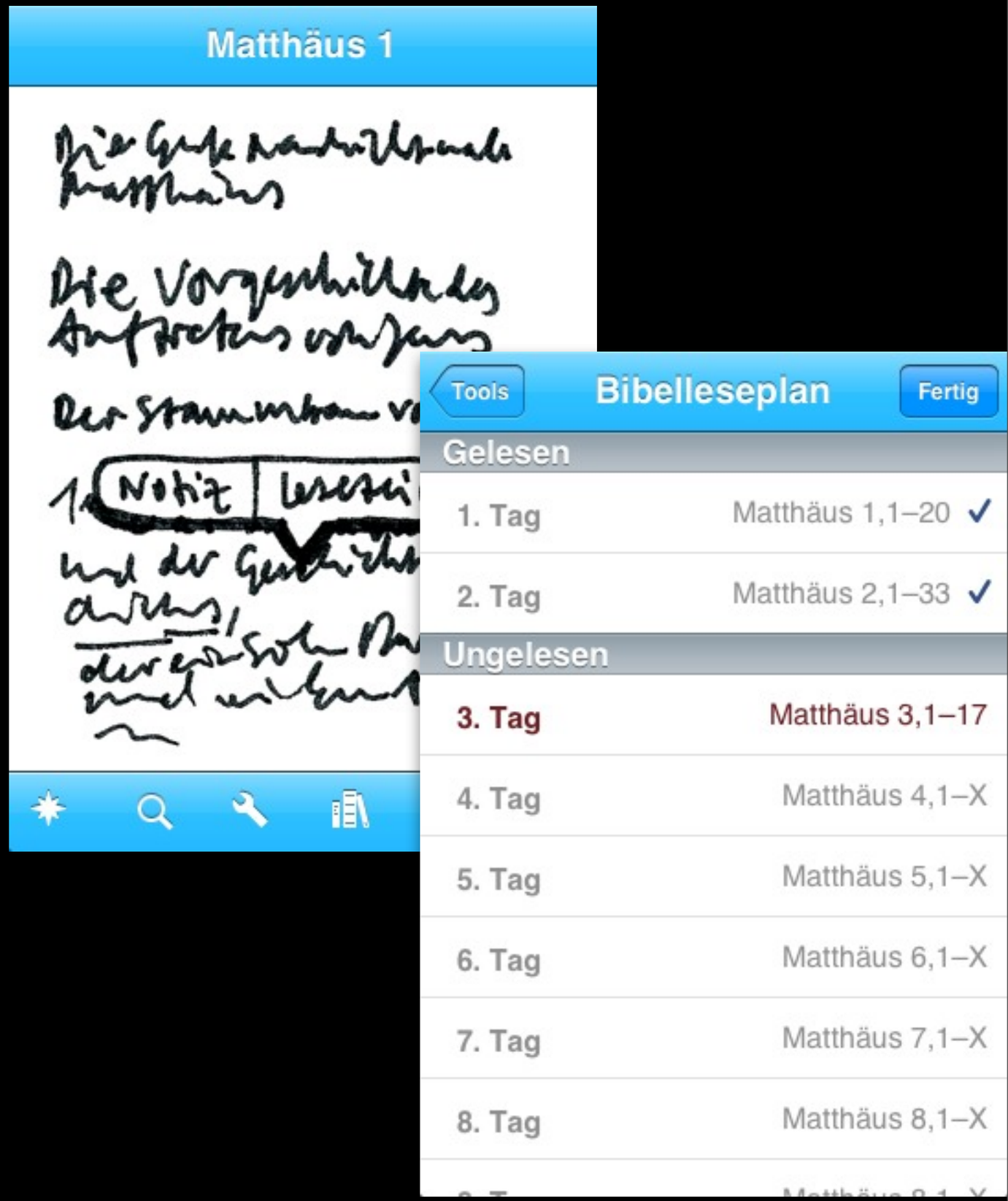
The good

IB Storyboards turn Xcode into a tool for quick storyboarding and prototyping



The good

IB Storyboards turn Xcode into a tool for quick storyboarding and prototyping



The bad

iOS 5 only

Universal app = redundancy = bugs

Storyboards can become complex quickly
(however: may be split into several files!)

Xcode = slow (however: SSDs are fast)

Xcode = buggy (blank document, switch to fix)

The bad

No way to cancel in `prepareForSegue:`

Workaround: Place condition logic in `tableView:didSelectRowForIndexPath:` and call `performSegueWithIdentifier:`

Downside: We're writing code again!

Proposal: `shouldPerformSegueWithIdentifier:`

The bad

No auto-alignment for connections at useful angles (45°, 90°) to make layout tidy

Difficult to have table views share a superview with others (workaround: add to plain UIViewController and wire up manually)

No way to add views without view controller

No access to application delegate

No autoselection of current scene in list view

Tips and patterns

Embedding shifts existing objects to the right, so move out of the way first

Use proper string constants for identifiers:

```
extern NSString *const kMySegue;         // .h
NSString *const kMySegue = @”MySegue”;  // .m, IB
```

Strive to intentionally align scenes and “untangle” layout (shortest paths)

Freeform size (e. g., for popovers)

Inferred vs. explicit metrics (e. g., when several transitions point to a single view controller)

Tips and patterns

Modal view controllers: Presentation is easy, dismissal not … Define a category:

```
@interface UIViewController (UIViewController_Dismissal)
- (IBAction)dismiss:(id)sender;
@end

@implementation UIViewController (UIViewController_Dismissal)
- (IBAction)dismiss:(id)sender
{
    if (self.presentingViewController)
    {
        [self.presentingViewController
         dismissModalViewControllerAnimated:YES];
    }
}
```

Tips and patterns

Complex storyboards can be split across several files

Disadvantage: Segues between them must be done programmatically (especially tedious for popovers)

Sample code (simplified):
Modal presentation on iPhone, popover on iPad

Tips and patterns

`prepareForSegue:` adapted from https://devforums.apple.com/message/478671

Workaround no longer necessary, but still useful to keep reference to popover controller

Tips and patterns

```
- (void)prepareForSegue:(UIStoryboardSegue *)segue
                 sender:(id)sender
{

    if (self.popoverController.popoverVisible) {
        [self.popoverController dismissPopoverAnimated:YES];
    }

    if ([segue isKindOfClass:[UIStoryboardPopoverSegue class]]) {

        UIStoryboardPopoverSegue *popoverSegue =
            (UIStoryboardPopoverSegue *)segue;

        UIPopoverController *thePopoverController =
            [popoverSegue popoverController];

        [thePopoverController setDelegate:self];
        self.popoverController = thePopoverController;
    }
}
```

Tips and patterns

`performSegueToSettingsStoryboard` outlines programmatic transition to another storyboard

iPhone: Modal presentation

iPad: Popover

Tips and patterns

```
- (void)performSegueToSettingsStoryboard
{

    if (self.modalViewController) {

        // Try again until no modal view controller is active.
        [self
performSelector:@selector(performSegueToSettingsStoryboard)
         withObject:nil
         afterDelay:0.1];

        return;
}
```

Tips and patterns

```
UIUserInterfaceIdiom idiom = UI_USER_INTERFACE_IDIOM();

BOOL isPhone = (idiom == UIUserInterfaceIdiomPhone);

NSString *idiomSuffix = (isPhone) ? @"iPhone" : @"iPad";

NSString *storyboardName = [NSString
 stringWithFormat:@"Settings_%@", idiomSuffix];

UIStoryboard *settingsStoryboard = [UIStoryboard
 storyboardWithName:storyboardName
             bundle:nil];

id destinationViewController = [settingsStoryboard
 instantiateInitialViewController];
```

Tips and patterns

```
if (isPhone) {
    [self presentViewController:destinationViewController
                       animated:YES
                     completion:nil];
}

else {
    if (self.popoverController.popoverVisible) {
        [self.popoverController dismissPopoverAnimated:YES];
    }

    self.popoverController = [[UIPopoverController alloc]
     initWithContentViewController:destinationViewController];

    [self.popoverController
     presentPopoverFromBarButtonItem:self.settingsButton
            permittedArrowDirections:UIPopoverArrowDirectionAny
                            animated:YES];
    }
}
```

Thank you!

developer@martinwinter.com

@martinwinter