TABLET IN BRANCH

Project: How to Organise BranchTablet Project

TIB-Application: How to Design TIB App

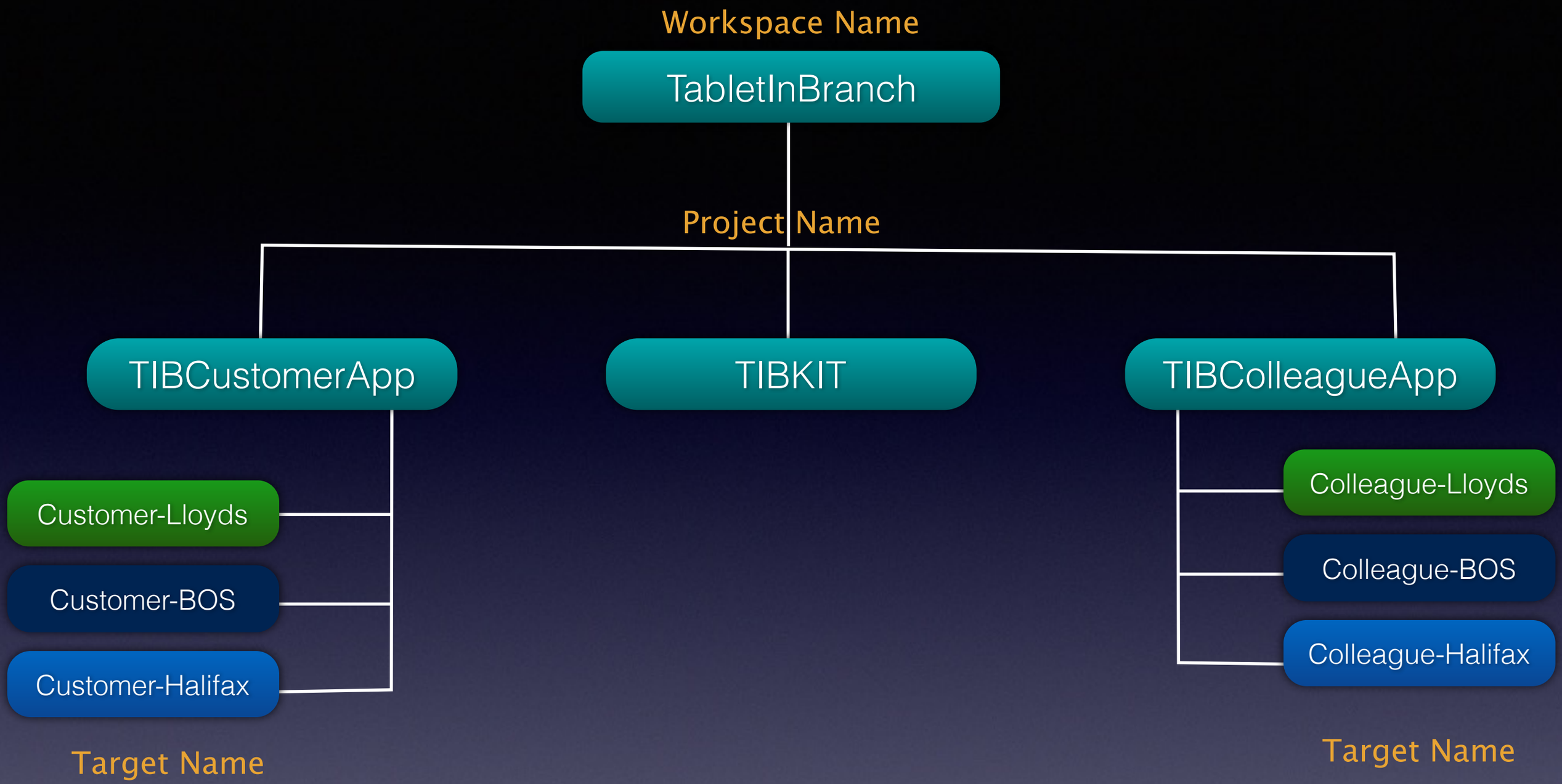
Resource: Manage Document, Resources and Architecture

Project: Tablet In Branch

How to Organise Tablet In Branch Project

Conventions

Structure



Once TIBLibrary will be mature. We will not push our code in Library without approval and sharing information to all

Conventions

Use Pascal Case for Flies,Folders and Class Name
Start with a Capital letter i.e. Controllers, MyClass

Use Camel Case for methods, properties and variables start with lower case i.e
setFirstName, userPassword, etc

Avoid using acronyms and **abbreviation** What the hell does it mean "usrPswLbl"? Yuck!

Prefixes It should consists of two or three uppercase letters and does not use underscores or "sub prefixes." Here are some examples:

Prefix	Cocoa Framework
NS	Foundation
AB	Address Book
IB	Interface Builder

Conventions

Class Name

- TIBNavigationBar
- CSLoginController
- CLLoginController

Prefix	TabletInBranch
TIB	Branding,Helper,Generic Controller (Common for Customer and Colleague App)
CS	Customer App
CL	Colleague App

Protocol Name / Category Name

Code	Commentary
NSLocking	Good Protocol Name
NSLock	Poor (seems like a name for a class)
NSObject+Branding	Category of OS Class
TIBNavigationBar+ExtendFunctionality	Category of Developer Class

A common convention is to use a gerund (“...ing”) form and +

Conventions

General Principles: Method Name

- It is good to be both clear and brief as possible, but clarity shouldn't suffer because of brevity:

Code	Commentary
insertObject:atIndex:	Good.
insert:at:	Not clear; what is being inserted? what does "at" signify?
removeObjectAtIndex:	Good.
removeObject:	Good, because it removes object referred to in argument.
remove:	Not clear; what is being removed?

- In general, don't abbreviate names of things. Spell them out, even if they're long:

Code	Commentary
destinationSelection	Good.
destSel	Not clear.
setBackgroundColor:	Good.
setBkgdColor:	Not clear.

You may think an abbreviation is well-known, but it might not be, especially if the developer encountering your method or function name has a different cultural and linguistic background.

Conventions

Naming Properties and Data Types

```
@property (strong) NSString *title;
@property (assign, getter=isEditable) BOOL editable;
@synthesize showsTitle=_showsTitle;
```

- If its BOOL then use **“is”** prefix in its getter accessed for better understanding
- If you expect that your class will be subclassed, and that these subclasses will require direct access to the data, use the @protected directive.

In general, don’t use the **#define** preprocessor command to create constants. For integer constants, use **enumerations**, and for floating point constants use the **const** qualifier

Constants

Use **const** to create constants for **floating point values**. **otherwise**, use **enumeration**.

```
const float NSLightGray;
```

```
#ifdef DEBUG
```

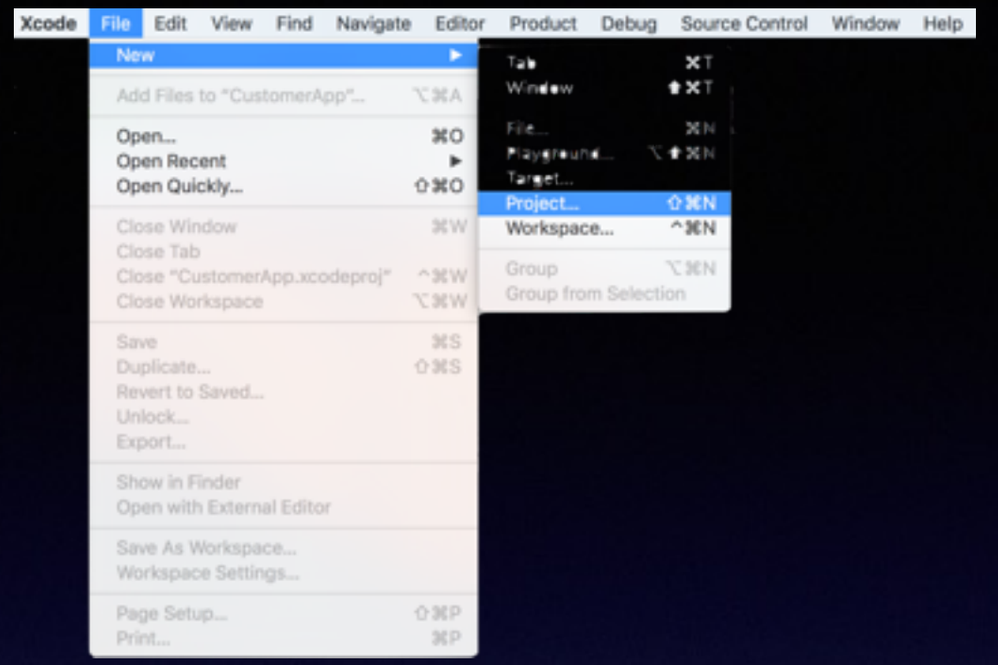
```
typedef enum {
    kActionSheetLogoutHelp,
    kActionSheetLogoutSettingsHelp,
    kActionSheetLogoutSettingsHelpSelectBusiness
} ActionSheetLayouts;
```

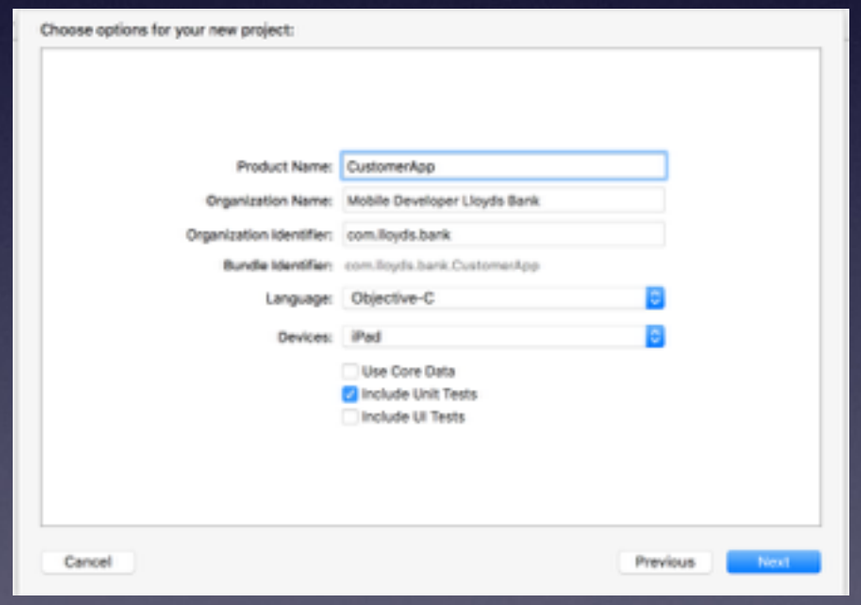
Use uppercase letters for symbols that the preprocessor evaluates in determining whether a block of code will be processed.

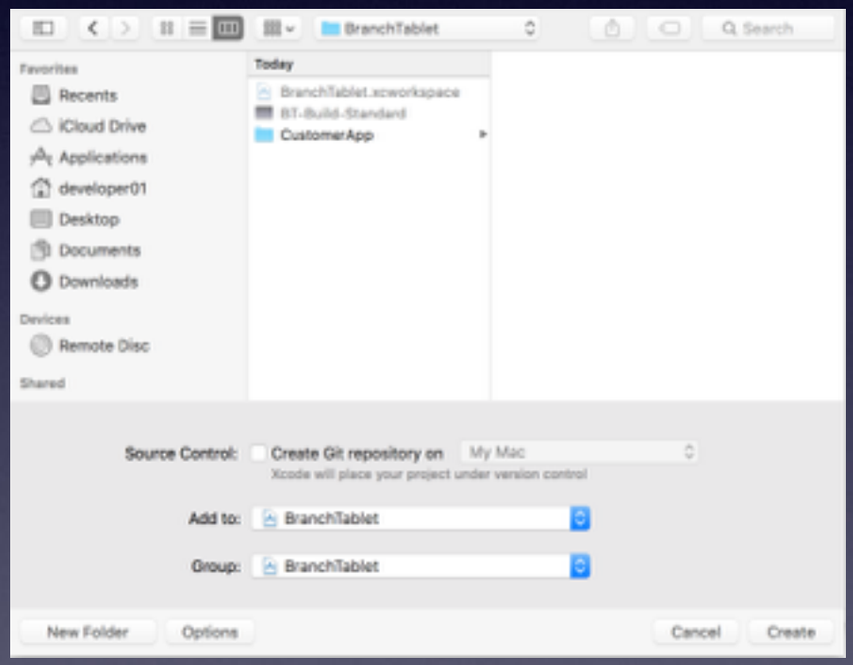
Structure

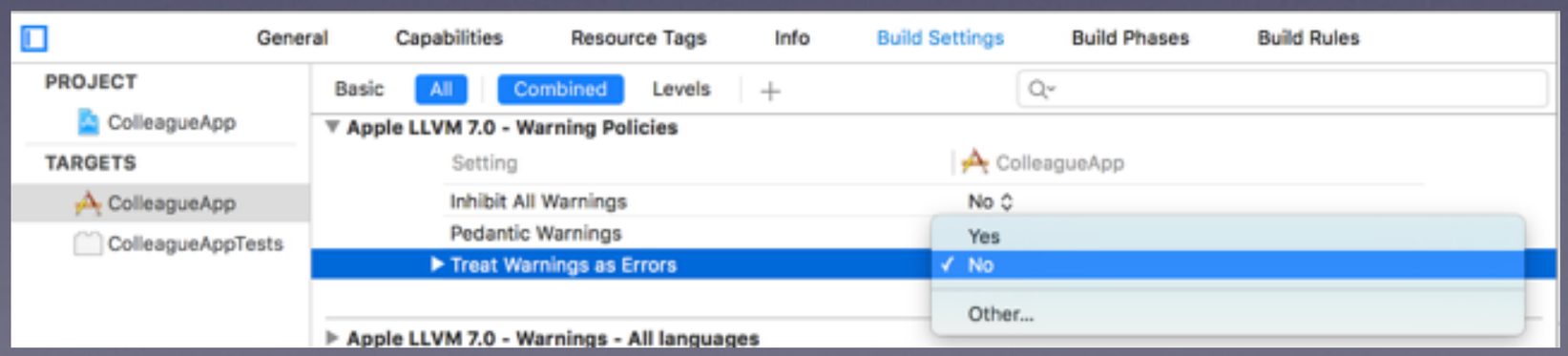
- **Create a Specific workspace** don't let Xcode do it for you
- **Setting up projects with correct name and prefix**
- **Configure Build Settings to improve quality**

  i.e. we can enable Static Analyser or Treat Warnings as Errors
- **Remove Auto-layout Warning**
- **Setting up projects with correct name and prefix**
- **Add App dynamic/Crash Reporter**









Structure

Process Need to Ready

- **Create a Build Automation to scripting common tasks** to compiling source code or to deploy artefacts with one command . Jenkins JOB should be ready with periodic build option initially. (TBD)Build Fail send the Email notification to all
- **Create a Version Automation** on each build to remove manual process to enter build no
- **Create a AdHocDistribution Build Configuration** So we can handle **Configuration** for different destination and Brand. Enterprise Developer License to distribute the App By EMM.
- **Integrate AppleDoc** inside the project. Wiki will be ready with development
- **EMM (AirWatch) console Configuration** So we can handle **device management, App Distribution, Policy control**
- **Create PED Configuration/Integration with Colleague App to access Card data**

Folders

- Put things in the right place and everything makes sense, unfortunately, Xcode doesn't help us
- Map all Xcode group folder to file system directory Xcode group folder don't represent physical folder
- Please remove Supporting Files group folder Who wants "Supporting Files" anymore? yuck!

- **Application**: specific app related stuff like AppDelegate, main.m, .pch etc
- **Controllers**: view (.xib) and view controller stuff put together (obviously .h & .m) One (physical) folder for each Journey
- **Library**: specific application classes like helpers, base classes, services, etc
- **Models**: application domain models and entities
- **Images.xcassets**: all images One folder for each type of image:-Button, Background, Logo, iCON etc
- **Resources**: assets like images, fonts, sounds, videos, etc. One Folder for each Brand (Lloyds, BOS, Halifax) and One folder for each type of asset-:, fonts, sounds, videos, strings, plist, samples
- **Vendors**: third part libraries and frameworks - : EMM (Airwatch), PED(Powa)
- **Entitlements**: All Entitlement files
- **Plists, Protocol,WebTrend,etc**

Application: How to Design BranchTablet App
Design Pattern
Remember Before Code Start

TabletInBranch Application

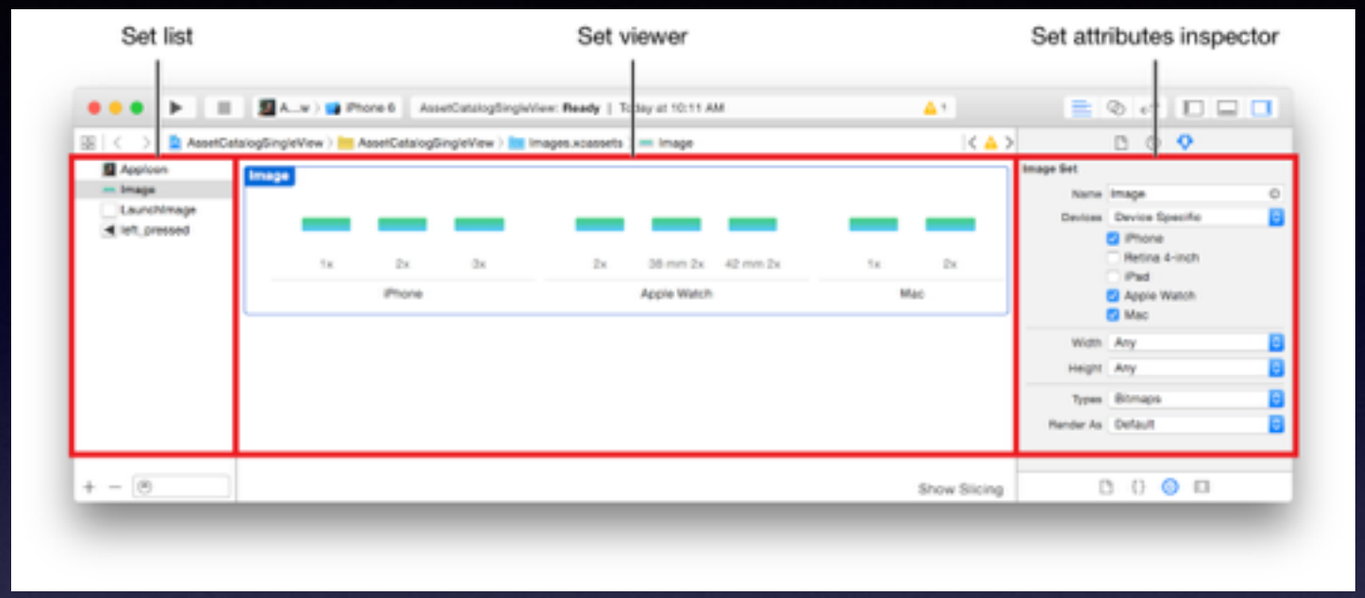
Design Pattern: Model-View-Controller-Store (MVCS)

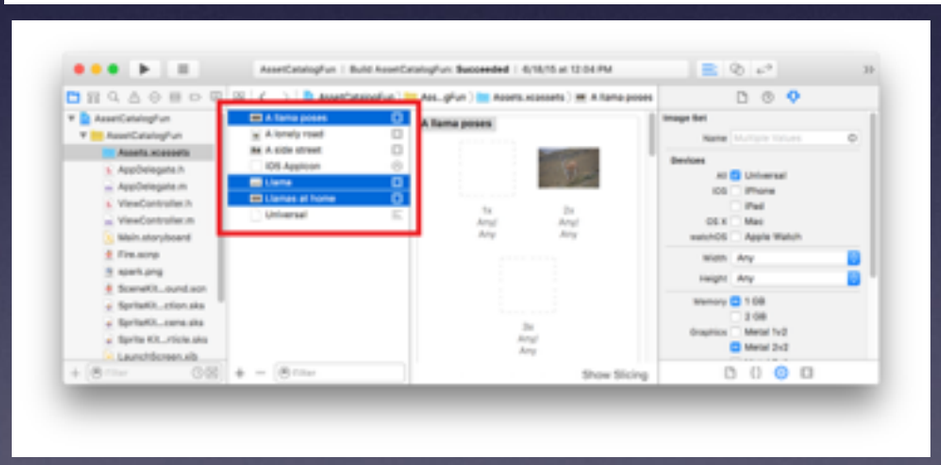
- This is the default Apple architecture (MVC), extended by a Store layer that vends Model instances and handles the networking, caching etc.
- Every Store exposes to the view controllers either RACSignals or void-returning methods with custom completion blocks.

TabletInBranch Application

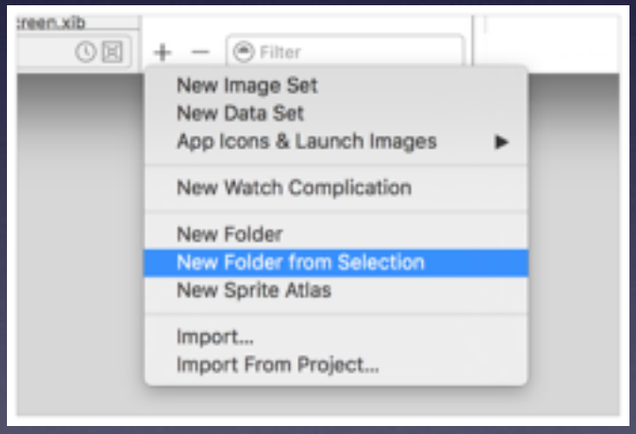
Asset Management: Asset Catalog

- Use asset catalogs to simplify management of images that are used by your app as part of its user interface. it can include: Folders, Image sets, icon and launch image

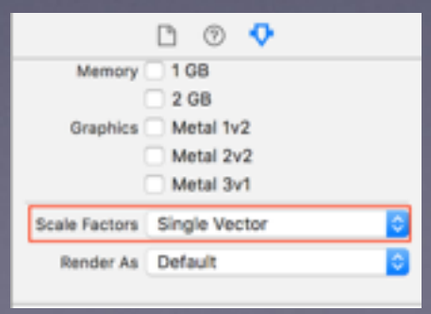




- Group Assets in Folder and rename accordingly



- We will use only **Original Vector Images** (PDF) and it would be produced by Designer. Its not required but good for future point of view. If business come with more device requirement



TabletInBranch Application

Localization:

- Keep all user strings in localization files right from the beginning
- File name should be Localizable.strings and each target should have its own file

Content should be like below. Avoid hard coded string

AppSign_NoPendingRequest_title"="You have no pending requests”;

"NGA-I-40-01a_T2" = "Business Mobile Banking”; YUCK!

Story Board:

- Use more then one storyboards to graphically lay out the user’s path through your iOS
- Separate Story board for customer and colleague App.
- Use xib for Custom UI Controls (Suggested.. developer can share their view in detail)

Auto Layout:

- Branch Table will be supported for Landscape Only. But we should be ready for all Orientation for future scope, it may extend. So we will use **Auto layout**. It dynamically calculates the size and position of all the views in your view hierarchy, based on constraints placed on those views..
- We can use Autolayout with constraints or with out constraints on the basis of Visual design. Please make sure App should be UI should be **Pixel Perfect as per VD**

Pixel Perfect as per VD & Style Guide

TabletInBranch Application

Auto Layout: Programmatically Creating Constraints

// Setting a fixed distance between two buttons: Button_2.leading = 1.0 * Button_1.trailing + 8.0

// Aligning the leading edge of two buttons: Button_1.leading = 1.0 * Button_2.leading + 0.0

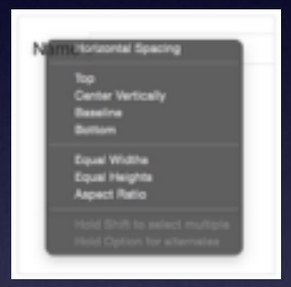
// Center a view in its superview

View.centerX = 1.0 * Superview.centerX + 0.0

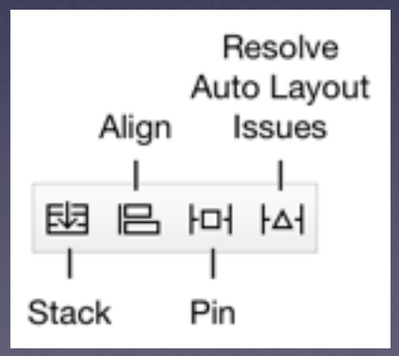
View.centerY = 1.0 * Superview.centerY + 0.0

Auto Layout: Creating Constraints with Interface Builder

When you release the mouse see—>



Interface builder provides four Auto Layout tools in the bottom-right corner of the Editor window. These are the Stack, Align, Pin, and Resolve Auto Layout Issues tools.



Please Open the X-Code and look after click on these options how the screen look like

All are experience so hope you aware with actual implementation. If not Please go through the below link or **shout immediately**

https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/AutolayoutPG/index.html#//apple_ref/doc/uid/TP40010853-CH7-SW1

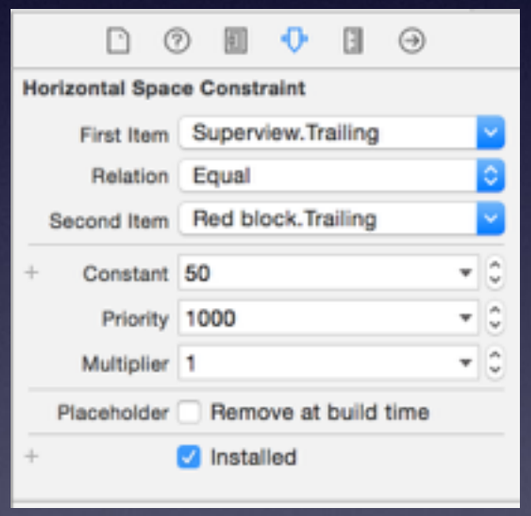
TabletInBranch Application

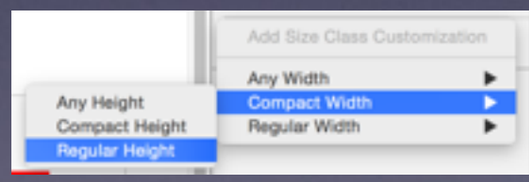
Size Classes:

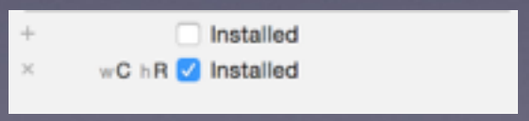
We will enable the size classes for future scope of tablet in branch. Its not required as per current requirement but scope will extend in future so please enable the size classes so that in future we can use Multiple Size Classes. Good to have

Some quick background on size classes: There are currently two size classes – horizontal and vertical, and each one comes in two sizes – regular and compact. The current orientation of the device can be described as a combination of the sizes:

Horizontal regular, vertical regular: iPad in either orientation
Horizontal compact, vertical regular: iPhone portrait
Horizontal regular, vertical compact: no current device
Horizontal compact, vertical compact: iPhone landscape







At the bottom of the Interface Builder window, there's now a control that allows you to switch between each combination

TabletInBranch Application

Size Classes:

You can use the a new callback method that's called as the interface changes:

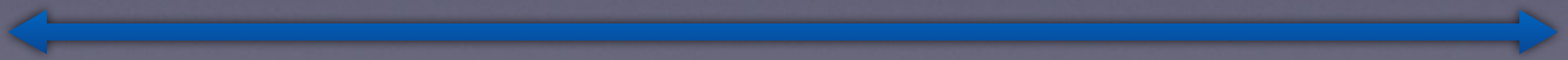
```
-(void)willTransitionToTraitCollection:(UITraitCollection *)newCollection
          withTransitionCoordinator:(id<UIViewControllerTransitionCoordinator>)coordinator
{
    NSLog(@"Branch Tablet = %@", newCollection);
}
```

This will display the trait collection that each orientation triggers:

```
Branch Tablet = <UITraitCollection: 0x7fa983c13a10; _UITraitNameUserInterfaceIdiom = Phone,
_UITraitNameDisplayScale = 2.000000, _UITraitNameHorizontalSizeClass = Compact,
_UITraitNameVerticalSizeClass = Regular, _UITraitNameTouchLevel = 0, _UITraitNameInteractionModel =
1>
```

All are experience so hope you aware with actual implementation. If not Please go through the below link or **shout immediately**

http://adoptioncurve.net/archives/2014/08/working-with-size-classes-in-interface-builder/



https://developer.apple.com/library/ios/recipes/xcode_help-IB_adaptive_sizes/chapters/AboutAdaptiveSizeDesign.html#//apple_ref/doc/uid/TP40014436-CH6-SW1

TabletInBranch Application

Environment Selection Like NGA:

- No need any example. Please copy paste. BackendSelectionViewController

Logging inside Project

- We will not use NSLOG any where in our project. We will create pre-processor like NGA Project Please take the reference from there

```
#ifdef LOGGING_IS_ENABLED
    #define LOG(format, ...) __log(format, ## __VA_ARGS__)
#else
    #define LOG(...)
#endif
```

```
#ifdef PLOGGING_IS_ENABLED
#define PLOG(format, ...) __log(format, ## __VA_ARGS__)
#else
#define PLOG(...)

#endif
```

Branding:

- Branding should contain Font, Color, size style etc. Take a look NGA for reference

Create xcconfig

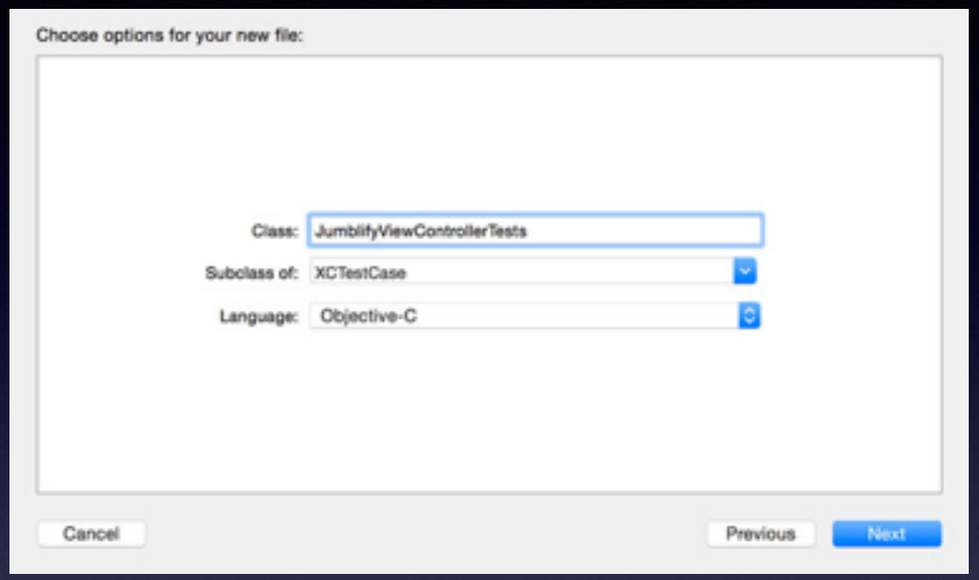
- To handle Debug/Tester/Release Preprocesses

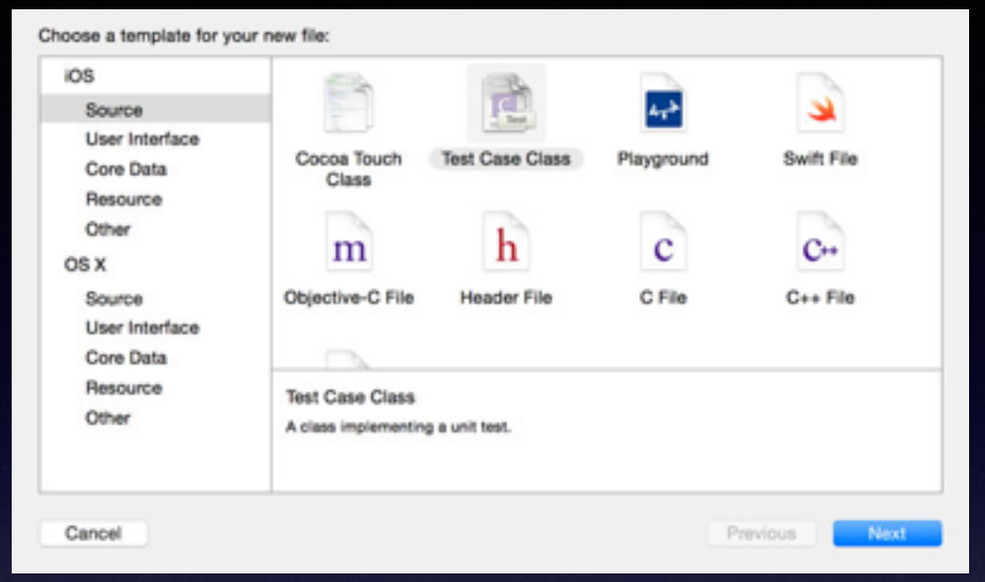
Crash Log Enable

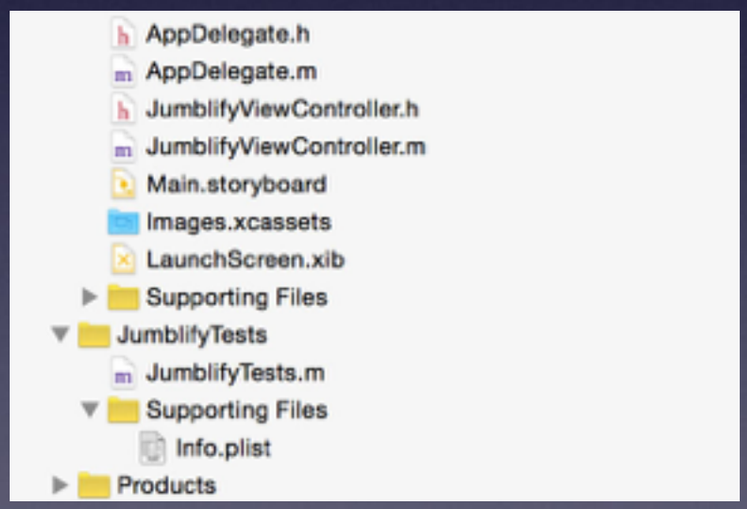
- We will Enable Crash reporter either Appdynamics / PLCrashreporter

TabletInBranch Application

Unit Test Cases (XCTest) In Progress

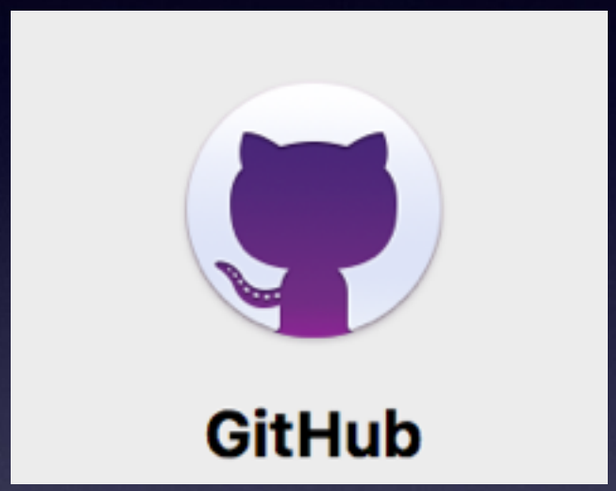




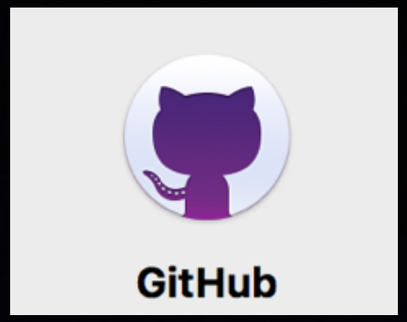


http://code.tutsplus.com/tutorials/introduction-to-testing-on-ios--cms-22394

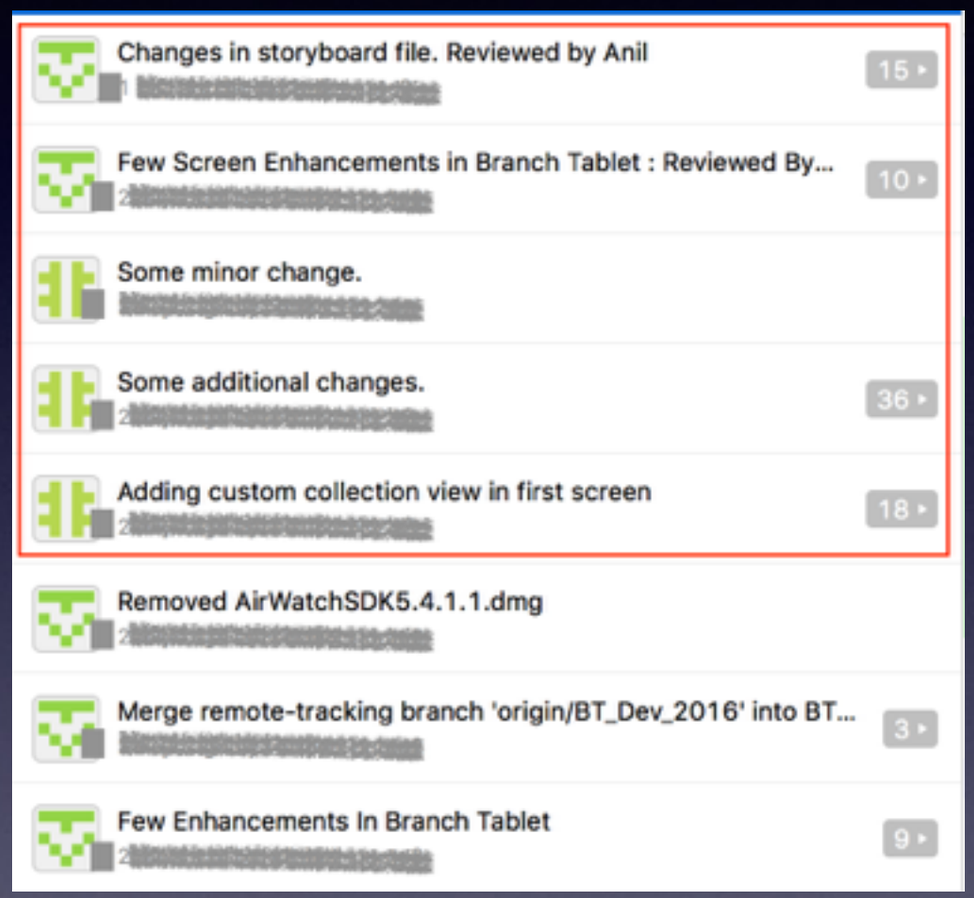
Good……Coding is done Time to make commit…
Thank God….. GYAN KHATAM



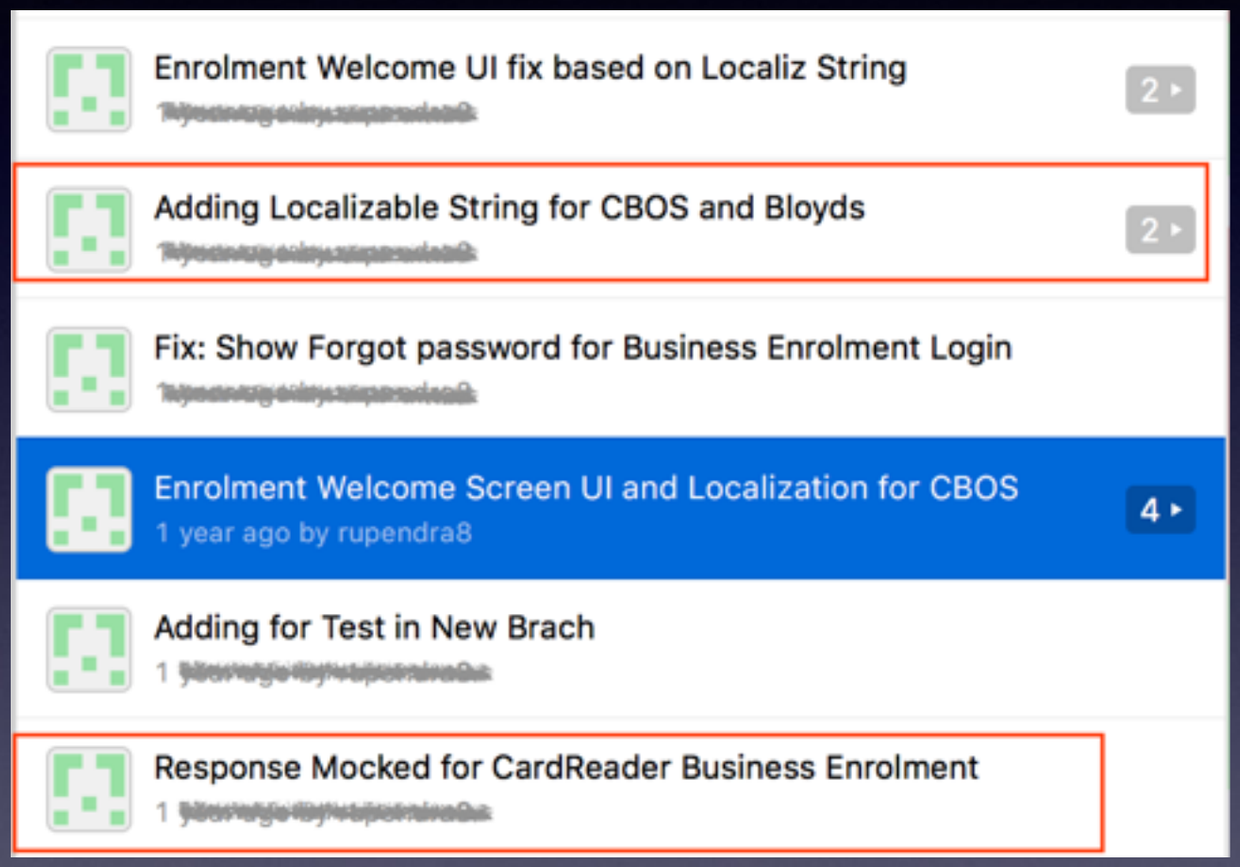
We will use GIT for revision Control



Meaning Full Message and Description : While Commit the code never writ the message like below.



Message is not clear that what changes are done in red mark commit



Its not best message. But Clear what is done in code. Its just an example. Our message should be much crisp and clear

TabletInBranch Team

Viresh

Anil

Nishant

GangaRaju

Rupendra

Himanshu

Vaishali

Tarun

Thanks