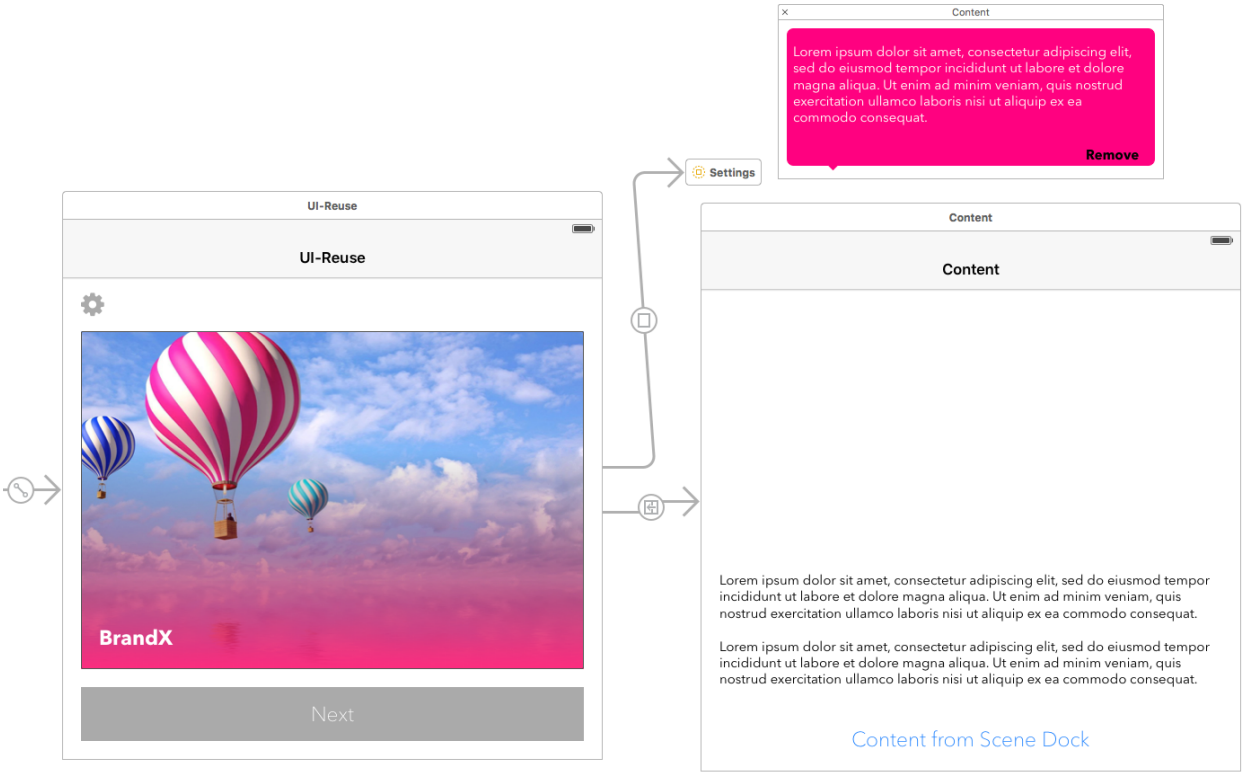
EWS iOS Best Practices

UI-Reuse Sample Application
- Segues and unwinding segues
- Storyboard references
- IBDesignable & IBInspectable
  - Shared Storyboard objects
    - UIFont
    - UIButton
    - UIImageView
- Content in Scene Dock
- Content in ViewControllers with custom Presentation Controller



sample application

General iOS Development Patterns

- Storyboard best practices
  - Use Multiple Storyboard to break work into modules
  - Use Storyboard references to connect other Storyboards
  - Add base and language specific strings to support Storyboard localization
  - Use Segues to connect Storyboard scenes

- Unwind segues to dismiss scenes
  - Programmatic where necessary
- Use AutoLayout over frame-based programmatic layout
- Use size classes instead of programmatic layout for device specific layout
- Use Dynamic Type to support type accessibility
- Animate NSLayoutConstraints
- Use StackViews over individual constraints
- Use template mode for tinting UIImages

- **Swift**
  - Use explicit optionals just don't unwrap them implicitly
  - Prefer object literals over verbose constructors
  - Avoid mutability (let over var)
  - Use pattern matching for complex flow control
  - Avoid shared state and singletons
  - Use guards when combining unwrapping and conditional checks
  - Don't use extension methods for code organization

- **How to construct View Controllers**
- Use a base class for common methods
  - Base setup and style hooks
- Use Interface Builder

- **View Controller organization**
  - Top level objects as strong references (not in view hierarchy)
  - Sub view hierarchy outlets as weak references
    - UI Elements
    - UI Constraints
  - Configuration settings in structs
  - Use dependency injection for ViewController dependencies

- **Table and CollectionViews**
  - Use TableViews for repeating content first over CollectionViews
  - Use AutoLayout instead of CollectionViews for repeating content who's layout might change or need to support different size classes
  - Subclass cells that will be reusable
  - Prefer static cells over dynamic prototypes where possible
  - Enums for cell construction, configuration and actions

- **Views**
  - Use a base class for common methods
  - Use IBDesignable & IBInspectable for Interface Builder re-use

- **Dependency Injection**
  - Use inversion of control for network services

- **Testing**
  - Include unit tests
    - Use mocks, stubs & verify expectations

- Include Integration tests
  - Network services