CS193p

Spring 2010

Today's Topics

- Settings Bundle
- Image Picker
- Alerts
- NSTimer
- Localization

Settings Bundle

- Allows you to add UI to General Settings app
  Can keep your UI simpler, but don't "make" users go there!
- Works by modifying `NSUserDefaults` entries
- Only for rarely-changed settings
  Overriding default behavior.
  Configuration that you only do once (or once in a long while).
  Modifying the "skin" of your user-interface (only if that's not central to what your application does, of course).
  Rule of thumb: if the user never even found out that those settings were there, they could still successfully use your application.
- UI is limited and no code runs
  So it's really only for fairly simple settings

Settings Bundle

- What sorts of UI can you do?
  Text Field (text input)
  Multi-Value (user chooses from a list of values)
  Title (read-only version of Multi-Value)
  Slider (continuous floating point values)
  Toggle Switch (boolean)
  Group or Child Pane (navigates to a different settings bundle)

- Lots of limitations
  For example, you can't show a slider's exact value.
  And you can only put an image to the left or right of a slider.

- Done by editing a `.plist` in Xcode
  No good graphical editor for settings.
  Choose New File... and Settings Bundle under Resources to create one.
  Can't even drag in the files you need (have to use Finder).

Settings Bundle

- Demo
  Text Field
  Group
  Slider (continuous floating point values)

Image Picker

- How to get images from Camera or Library
- `UIImagePickerController` runs modally only
  Doesn't really work to use it any other way.
  Probably could install it directly into window, but only if custom overlay.
- Create, configure, then present modally
- What user can do depends on platform
  And on what you configure the picker to allow as a "source" of media
- There are (currently) three different "sources"
  Camera (can capture photo or video (on 3GS only))
  Photo Library (synced photos from iTunes)
  Saved Photos ("Camera Roll" on an iPhone)

Image Picker

- Lifecycle starts with `alloc/init`

```
UIImagePickerController *picker =
    [[UIImagePickerController alloc] init];
```

- Configure the source of the media

```
picker.sourceType = (UIImagePickerControllerSourceType)type;
UIImagePickerControllerSourceTypeCamera
UIImagePickerControllerSourceTypePhotoLibrary;
UIImagePickerControllerSourceTypeSavedPhotosAlbum;
```

Make sure you check that the source type you want is available on this device!

```
+ (BOOL)isSourceTypeAvailable:(UIImagePickerControllerSourceType);
```

Image Picker

- Next configure which media types are allowed

```
picker.mediaTypes = (NSArray *)mediaTypes; // of NSString
```

Defaults to `kUTTypeImage`, but you can get other types from the class method:

```
+ (NSArray *)availableMediaTypesForSourceType:(UIImagePickerSourceType)];
```

3GS will also support `kUTTypeMovie` for the `UIImagePickerSourceTypeCamera`.

- Set whether user can edit the captured data

```
picker.allowsEditing = (BOOL);
```

Images can be cropped and video can be cut down.

- Can limit video capture

```
picker.videoQuality = (UIImagePickerControllerQualityType);
picker.videoMaximumDuration = (NSTimeInterval);
```

Image Picker

- Overlay View

  `picker.cameraOverlayView = (UIView *);`

  Be sure to set the view's `frame` properly.
  Camera view is always full screen, so `[[UIScreen mainScreen] bounds]`.
  But if you have controls at the bottom, might want it to be smaller.

- If you don't want the normal camera controls

  `picker.showsCameraControls = (BOOL);`

  This will leave a blank area at the bottom because the camera's aspect ratio (4:3) is not the same as the screen's.

  With no controls, your overlay will want a "take picture" button.

  That button should send `- (void)takePicture` to the picker.

- Can zoom, translate the image while capturing

  `picker.cameraViewTransform = (CGAffineTransform);`

  For example, you could scale up photo to fit whole screen
  (i.e. not have the blank area at the bottom of the screen).

Image Picker

- To get notified of the chosen image, set `delegate`

```
picker.delegate = (id <UIImagePickerControllerDelegate>);
```

- Only one main method, but lots of information

```
- (void)imagePickerController:(UIImagePickerController *)picker
    didFinishPickingMediaWithInfo:(NSDictionary *)info
{
    // extract image/movie information here
    [self dismissModalViewControllerAnimated:YES];
    [picker release];
}
```

- The argument "`info`" has 5 pieces of information

```
UIImagePickerControllerMediaType // usually image or movie
UIImagePickerControllerOriginalImage; // UIImage *
UIImagePickerControllerEditedImage; // UIImage *
UIImagePickerControllerCropRect; // NSValue * (packaged CGRect)
UIImagePickerControllerMediaURL; // if video
```

Image Picker

- Also dismiss if user cancels

```
- (void)imagePickerControllerDidCancel:(UIImagePickerController *)picker
{
    [self dismissModalViewControllerAnimated:YES];
    [picker release];
}
```

Image Picker

- Saving chosen images (esp. from the camera)

  You can put them in the sandbox.

  Or you can store them in the Saved Photos Album.

```
UIImageWriteToSavedPhotosAlbum(UIImage *image,
                               id target, SEL selector,
                               void *context);
```

  The (optional) `selector` should look like this:

```
- (void)image:(UIImage *)image
    didFinishSavingWithError:(NSError *)error
    contextInfo:(void *)context;
```

- Saving chosen video

```
if (UIVideoAtPathIsCompatibleWithSavedPhotosAlbum(NSString *path)) {
   UISaveVideoAtPathToSavedPhotosAlbum(
      NSString *path, id target, SEL selector, void *context);
}

- (void)video:(NSString *)path didFinishSavingWithError:(NSError *)...
```

Image Picker

- Writing to the flash memory is slow

  And images are big.

  Think about showing an activity spinner or doing save in another thread.

  If you have an overlay view, you can put up spinner between the time the person presses your "take picture" button and the time you get the callback (or even after your callback saves the image).

Alerts

- Two kinds of "pop up and ask" mechanisms
  Alerts
  Action Sheets
- Alerts pop up in middle of screen
- Action sheets "slide in" from somewhere
  Usually bottom of screen on iPhone
- Alerts used for "exceptional notification"
  Something unusual or asynchronous happened
  e.g. "Connection request from peer. Allow? Deny?"
  e.g. "File save failed." or "Network fetch failed. Try again?"
- Action sheets are for "branching" decisions by user


Carrier
10:59 AM
Test
Choose Photo
Self Destruct!
Cancel
Photo Library
Saved Photos
Carrier
10:36 AM
Flickr Fetch
Network access failed!
Ignore
Try Again
Connect
Camera

Alerts

- Alert

```
UIAlertView *alert = [[UIAlertView alloc]
    initWithTitle:(NSString *)title
    message:(NSString *)message
    delegate:(id <UIAlertViewDelegate>)delegate
    cancelButtonTitle:(NSString *)cancelTitle
    otherButtonTitles:(NSString *)otherTitle, ..., nil];
[alert addButtonWithTitle:(NSString *)anotherTitle];
[alert show];
```

Delegate methods for will present, did present, will dismiss, did dismiss.
And, most importantly:

```
- (void)alertView:(UIAlertView *)alert
    clickedButtonAtIndex:(NSInteger)index
{
    NSString *chosen = [alert buttonTitleAtIndex:index];
    ...
}
```

The alert is automatically dismissed after this returns.

Alerts

- Action Sheet

```
UIActionSheet *sheet = [[UIActionSheet alloc]
    initWithTitle:(NSString *)title
    delegate:(id <UIAlertViewDelegate>)delegate
    cancelButtonTitle:(NSString *)cancelTitle
    destructiveButtonTitle:(NSString *)cancelTitle
    otherButtonTitles:(NSString *)otherTitle, ..., nil];
[sheet addButtonWithTitle:(NSString *)anotherTitle];
[sheet showInView:(UIView *)view];
[sheet showFromRect:(CGRect) inView:(UIView *) animated:(BOOL)];
[sheet showFromBarButtonItem:(UIBarButtonItem *) animated:(BOOL)];
[sheet showFromToolbar:(UIToolbar *)toolbar];
```

Same delegate method forms as UIAlertView:

```
- (void)actionSheet:(UIAlertView *)actionSheet
    clickedButtonAtIndex:(NSInteger)index
{
    NSString *chosen = [actionSheet buttonTitleAtIndex:index];
    ...
}
```

Alerts

- Other properties

```
@property BOOL visible;
@property NSInteger firstOtherButtonIndex;
@property NSInteger numberOfButtons;

@property UIActionSheetStyle actionSheetStyle;

UIActionSheetStyleAutomatic // takes on style of toolbar
UIActionSheetStyleDefault
UIActionSheetStyleBlackTranslucent
UIActionSheetStyleBlackOpaque
```

Image Picker/Alerts

- Demo

NSTimer

- Scheduled invocation of method
- Managed by the current thread's run loop
- Not truly "real time" (runs when run loop can)
- Starting a timer on the current run loop

```
NSTimer *timer = [NSTimer scheduledTimerWithTimeInterval:(NSTimeInterval)
    target:self selector:@selector(doSomething:) userInfo:(id)userInfo
    repeats:(BOOL)];

- (void)doSomething:(NSTimer *)timer
{
    id myUserInfo = timer.userInfo;
    // do whatever it is you want to do (don't take too long)
    if (imTiredOfThisRepeatingTimer) [timer invalidate]; // releases it
}
```

NSTimer

- Demo

Localization

- Two steps to making international applications
  - Internationalization (i18n)
  - Localization (l10n)
- Resources are taken from current locale's bundle
  - .xib files
  - string files
  - images, other files if you want
- Current locale is a system setting
- Command line tool for localizing .xibs -- ibtool
  - Generate strings file from .xib.
  - Localize a .xib using translated strings file.
  - Update an already-translated .xib from a master .xib with changes in it.
  - You can also "lock" .xib files (from IB) to reduce chance of breaking things.

Localization

- Strings (that are not in .xib files)

  `NSBundle` method to return a localized string

```
- (NSString *)localizedStringForKey:(NSString *)key
                              value:(NSString *)default
                              table:(NSString *)tableName];
```

  But it's better to use this C function:

```
NSLocalizedStringFromTable(
    NSString *key,
    NSString *tableName,
    NSString *comment);
```

  Note the "comment" that can be given to localizers.

  Also `NSLocalizedString()` without the table (uses `Localizable.strings`)

  Looks the key up in a .strings file from the appropriate language bundle.

- Then create .strings files for localizers to translate

  `genstrings *.m */*.m` (searches for `NSLocalizedString` calls)

Localization

- Getting other files in a localized way

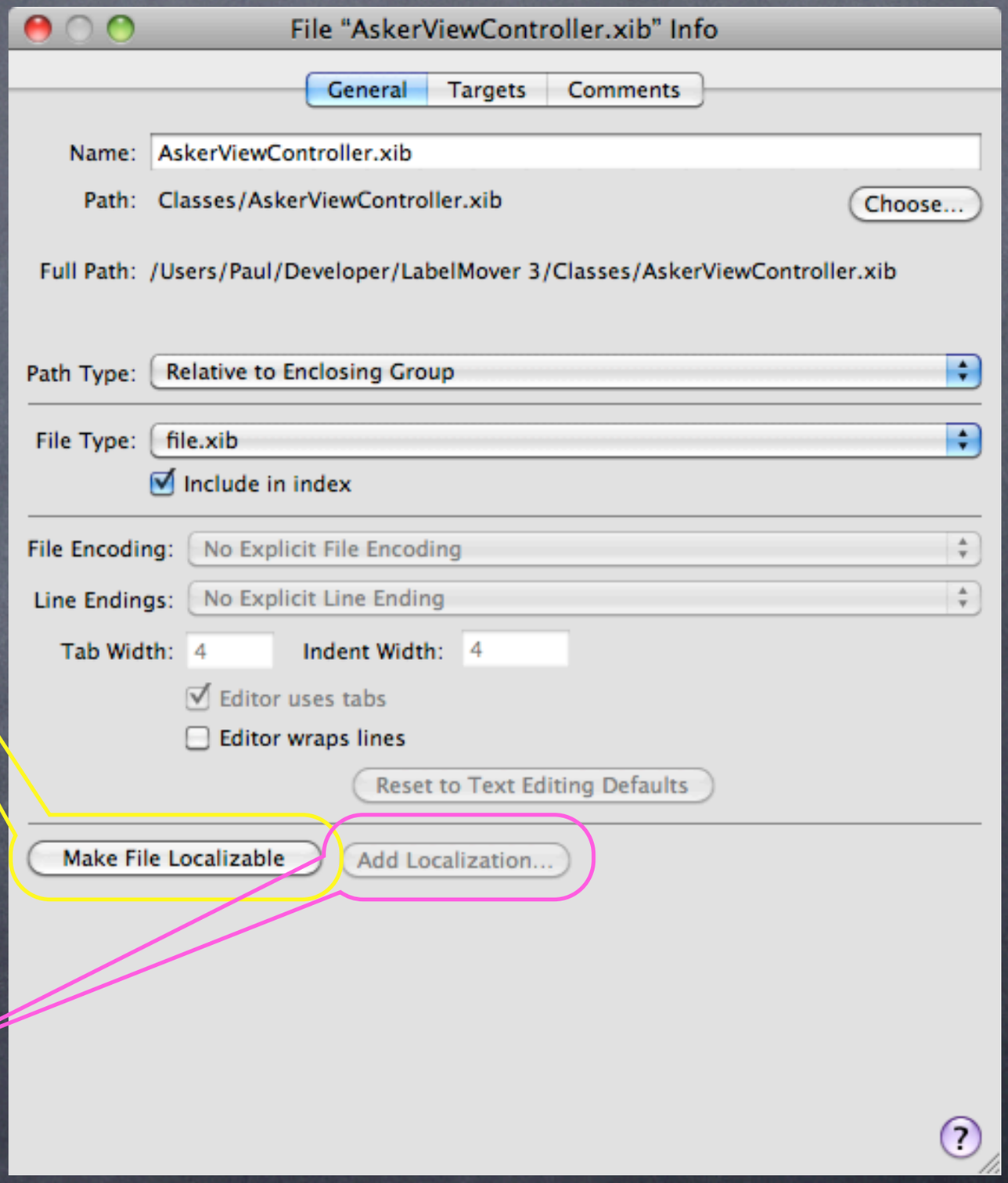
```
- (NSString *)pathForResource:(NSString *)resourceName
                       ofType:(NSString *)fileExtension];
```

Localization

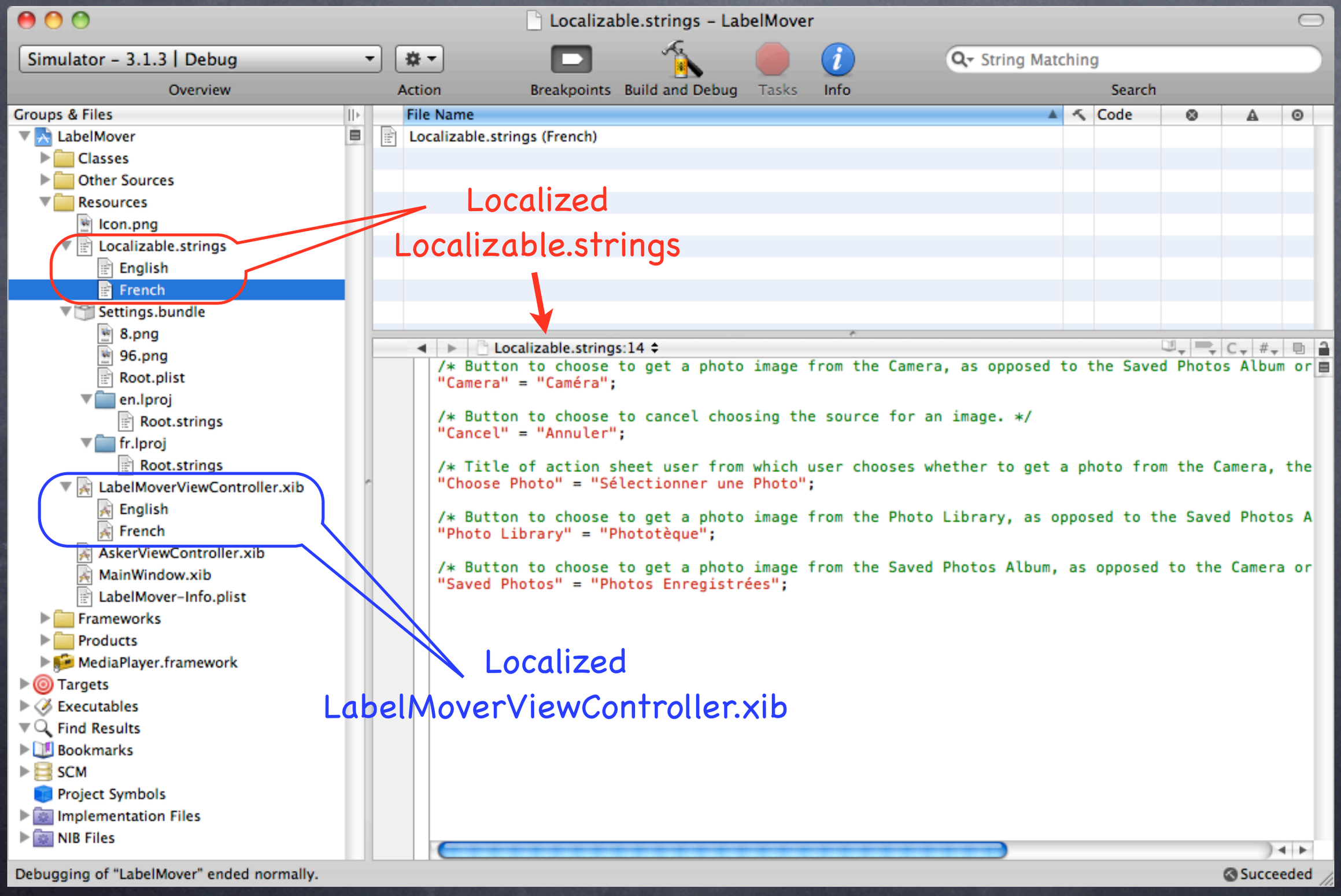
- Making a file localizable in Xcode

Inspect it
(right click, Get Info)
then choose
Make File Localizable

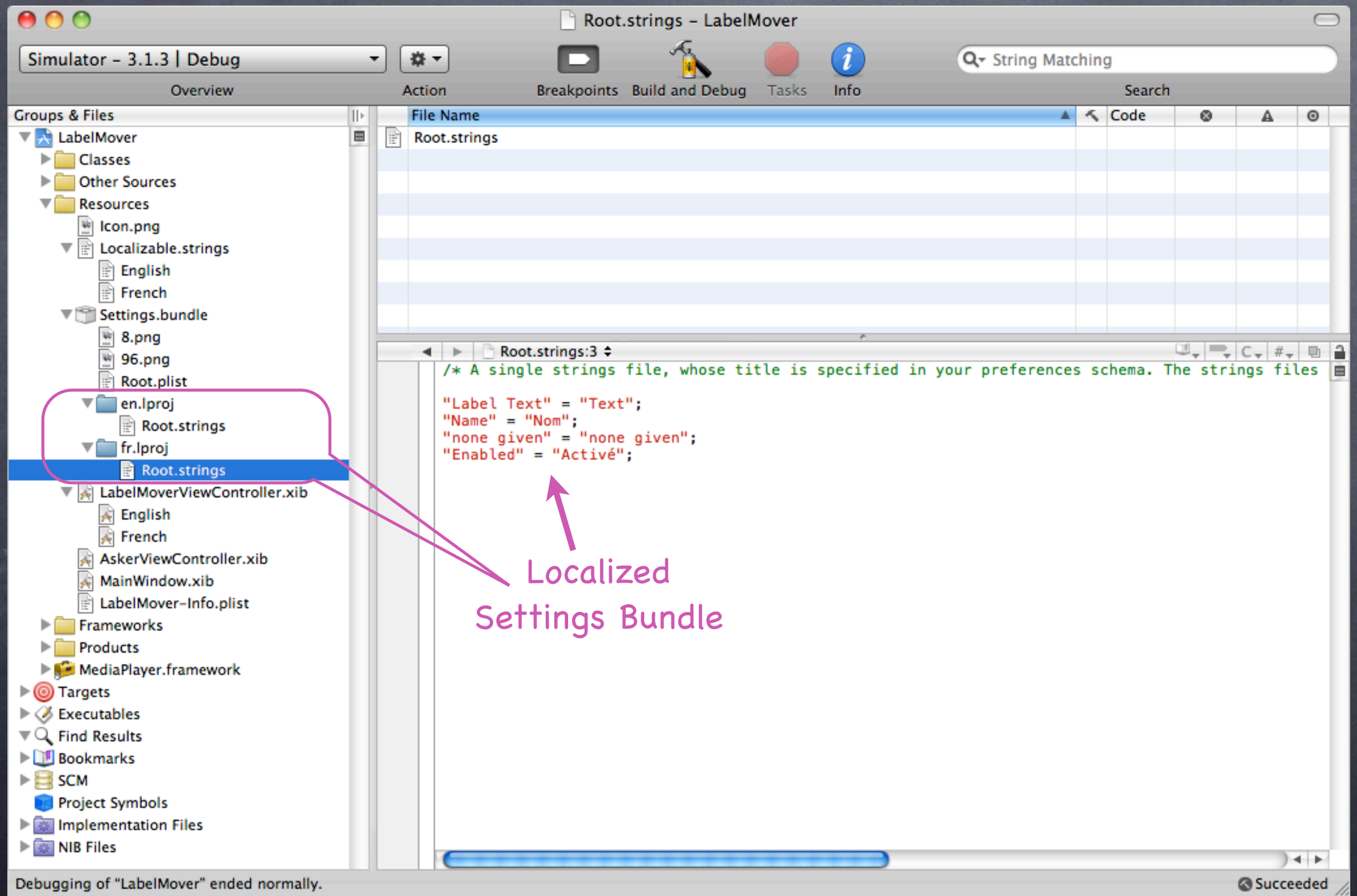
Then Add Localization...
for each subsequent language
you want to support.



Localization



Localization



Localization

- Demo (time permitting)