Making Your App World-Ready

Session 219

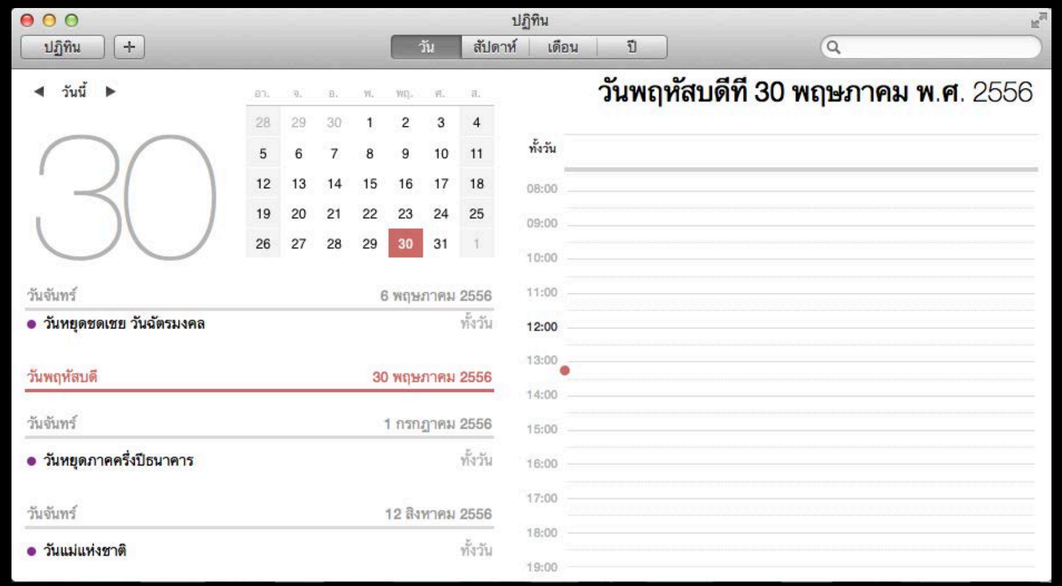
Douglas Davidson
Natural Languages

Albert Wan
Software Engineer

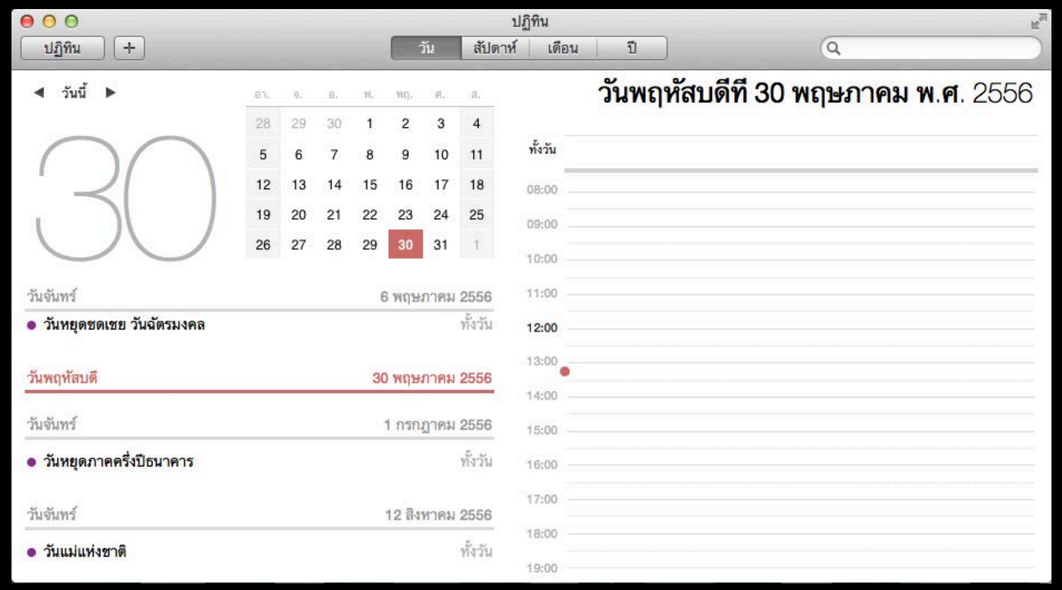
Nat Hillard
Software Engineer

Introduction

- Localization and internationalization
- Challenges and solutions
- What not to assume

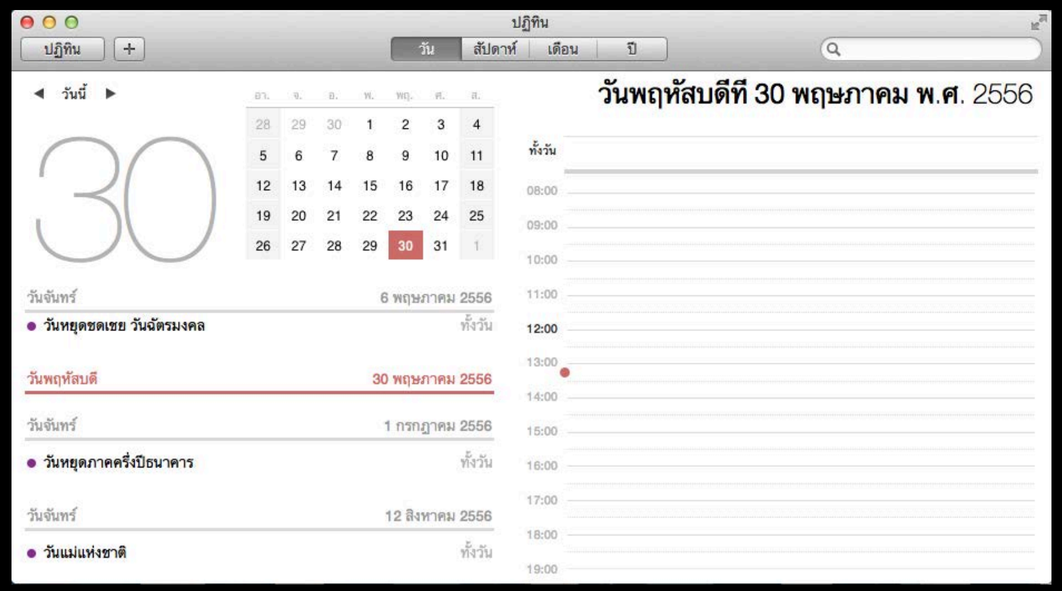

ปฏิทิน
ปฏิทิน
วัน
สัปดาห์
เดือน
ปี
วันนี้
30
อา. จ. อ. พ. พฤ. ศ. ส.
28 29 30 1 2 3 4
5 6 7 8 9 10 11
12 13 14 15 16 17 18
19 20 21 22 23 24 25
26 27 28 29 30 31 1
วันจันทร์ 6 พฤษภาคม 2556
วันหยุดชดเชย วันฉัตรมงคล ทั้งวัน
วันพฤหัสบดี 30 พฤษภาคม 2556
วันจันทร์ 1 กรกฎาคม 2556
วันหยุดภาคครึ่งปีธนาคาร ทั้งวัน
วันจันทร์ 12 สิงหาคม 2556
วันแม่แห่งชาติ ทั้งวัน
วันพฤหัสบดีที่ 30 พฤษภาคม พ.ศ. 2556
ทั้งวัน
08:00
09:00
10:00
11:00
12:00
13:00
14:00
15:00
16:00
17:00
18:00
19:00


시스템 환경설정
모두 보기
일반
데스크탑 및 화면 보호기
Dock
Mission Control
언어 및 지역
보안 및 개인 정보
Spotlight
알림
CD 및 DVD
디스플레이
에너지 절약
키보드
마우스
트랙패드
프린터 및 스캐너
사운드
iCloud
인터넷 계정
네트워크
Bluetooth
공유
사용자 및 그룹
유해 콘텐츠 차단
App Store
받아쓰기 및 말하기
날짜와 시간
시동 디스크
Time Machine
손쉬운 사용


ปฏิทิน
ปฏิทิน
วัน
สัปดาห์
เดือน
ปี
วันนี้
30
วันพฤหัสบดีที่ 30 พฤษภาคม พ.ศ. 2556
ทั้งวัน
วันจันทร์ 6 พฤษภาคม 2556
วันหยุดชดเชย วันฉัตรมงคล ทั้งวัน
วันพฤหัสบดี 30 พฤษภาคม 2556
วันจันทร์ 1 กรกฎาคม 2556
วันหยุดภาคครึ่งปีธนาคาร ทั้งวัน
วันจันทร์ 12 สิงหาคม 2556
วันแม่แห่งชาติ ทั้งวัน


시스템 환경설정
모두 보기
일반
데스크탑 및 화면 보호기
Dock
Mission Control
언어 및 지역
보안 및 개인 정보
Spotlight
알림
CD 및 DVD
디스플레이
에너지 절약
키보드
마우스
트랙패드
프린터 및 스캐너
사운드
iCloud
인터넷 계정
네트워크
Bluetooth
공유
App Store
받아쓰기 및 말하기
날짜와 시간
시동 디스크
Time Machine
손쉬운 사용

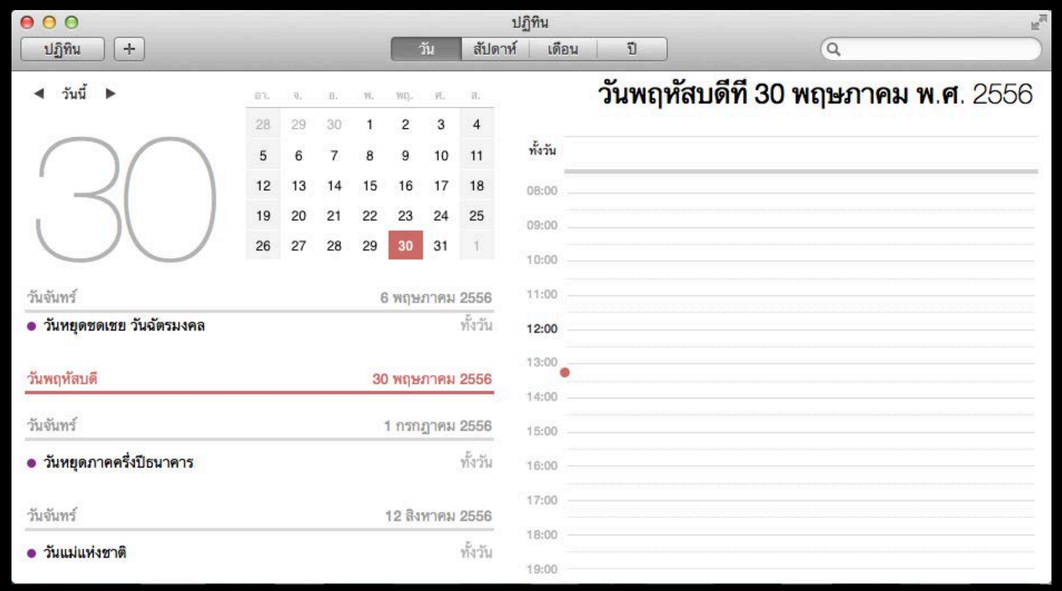

通用
多语言环境
语言
简体中文
语音控制
中文
键盘
9
区域格式
中国
日历
公历
区域格式示例
2013年1月5日星期六
上午12:34
408-5551212

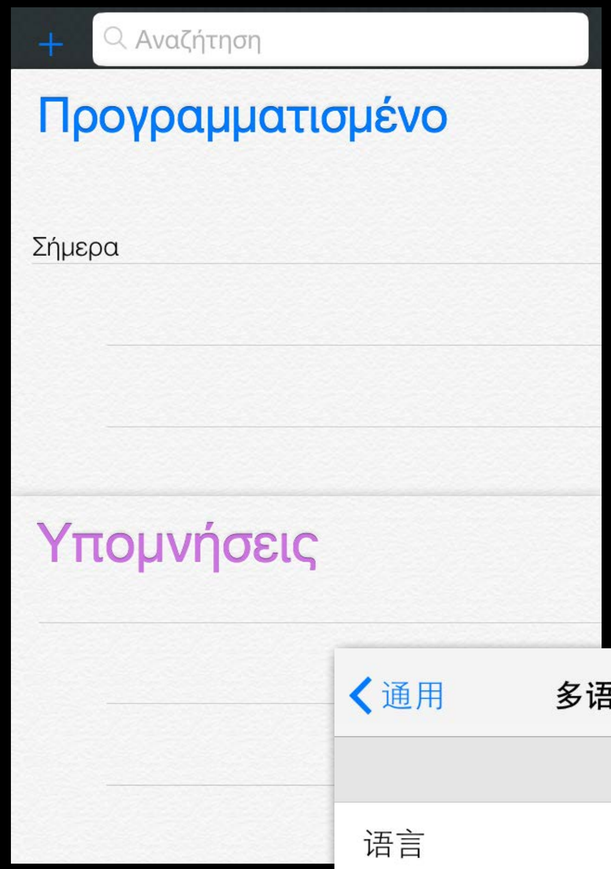

ปฏิทิน
ปฏิทิน
วัน
สัปดาห์
เดือน
ปี
วันนี้
30
วันพฤหัสบดีที่ 30 พฤษภาคม พ.ศ. 2556
วันจันทร์ 6 พฤษภาคม 2556
วันหยุดชดเชย วันฉัตรมงคล ทั้งวัน
วันพฤหัสบดี 30 พฤษภาคม 2556
วันจันทร์ 1 กรกฎาคม 2556
วันหยุดภาคครึ่งปีธนาคาร ทั้งวัน
วันจันทร์ 12 สิงหาคม 2556
วันแม่แห่งชาติ ทั้งวัน
ทั้งวัน


시스템 환경설정
모두 보기
일반
데스크탑 및 화면 보호기
Dock
Mission Control
언어 및 지역
보안 및 개인 정보
Spotlight
알림
CD 및 DVD
디스플레이
에너지 절약
키보드
마우스
트랙패드
프린터 및 스캐너
사운드
iCloud
인터넷 계정
네트워크
Bluetooth
공유
App Store
받아쓰기 및 말하기
날짜와 시간
시동 디스크
Time Machine
손쉬운 사용


Mai
M D M D F S S
27 28 29 30 31 1 2
Donnerstag 30. Mai 2013
10:09
11:00
12:00
13:00
14:00
Heute
Kalender
Eingang


通用
多语言环境
语言
简体中文
语音控制
中文
键盘
9
区域格式
中国
日历
公历
区域格式示例
2013年1月5日星期六
上午12:34
408-5551212

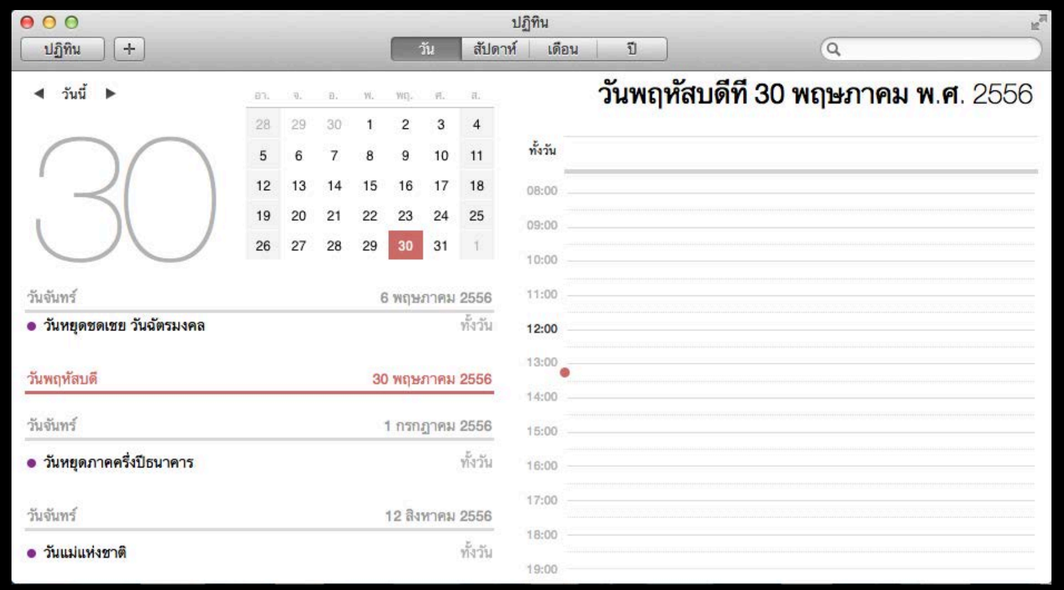

ปฏิทิน
วัน
สัปดาห์
เดือน
ปี
วันนี้
30
วันพฤหัสบดีที่ 30 พฤษภาคม พ.ศ. 2556
วันจันทร์ 6 พฤษภาคม 2556
วันหยุดชดเชย วันฉัตรมงคล
วันพฤหัสบดี 30 พฤษภาคม 2556
วันจันทร์ 1 กรกฎาคม 2556
วันหยุดภาคครึ่งปีธนาคาร
วันจันทร์ 12 สิงหาคม 2556
วันแม่แห่งชาติ


Αναζήτηση
Προγραμματισμένο
Σήμερα
Υπομνήσεις


시스템 환경설정
모두 보기
일반
데스크탑 및 화면 보호기
Dock
Mission Control
언어 및 지역
보안 및 개인 정보
Spotlight
알림
CD 및 DVD
디스플레이
에너지 절약
키보드
마우스
트랙패드
프린터 및 스캐너
사운드
iCloud
인터넷 계정
네트워크
Bluetooth
공유
App Store
받아쓰기 및 말하기
날짜와 시간
시동 디스크
Time Machine
손쉬운 사용


Mai
M D M D F S S
27 28 29 30 31 1 2
Donnerstag 30. Mai 2013
10:09
11:00
12:00
13:00
14:00
Heute
Kalender
Eingang

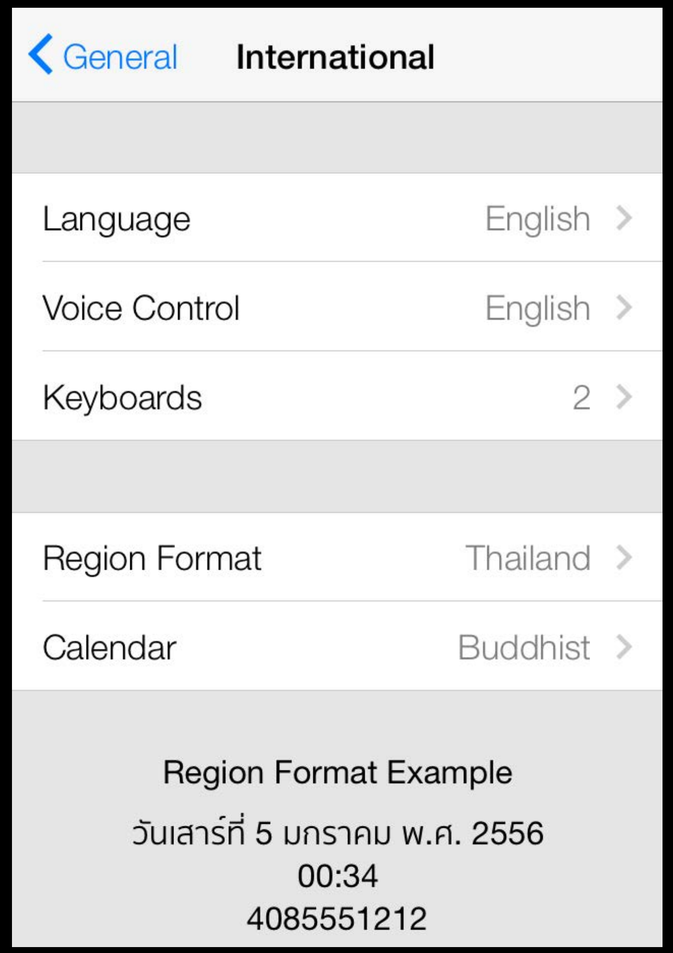

通用
多语言环境
语言
简体中文
语音控制
中文
键盘
9
区域格式
中国
日历
公历
区域格式示例
2013年1月5日星期六
上午12:34
408-5551212


ปฏิทิน
วัน
สัปดาห์
เดือน
ปี
วันนี้
30
วันพฤหัสบดีที่ 30 พฤษภาคม พ.ศ. 2556
วันจันทร์ 6 พฤษภาคม 2556
วันหยุดชดเชย วันฉัตรมงคล
ทั้งวัน
วันพฤหัสบดี 30 พฤษภาคม 2556
วันจันทร์ 1 กรกฎาคม 2556
วันหยุดภาคครึ่งปีธนาคาร
ทั้งวัน
วันจันทร์ 12 สิงหาคม 2556
วันแม่แห่งชาติ
ทั้งวัน

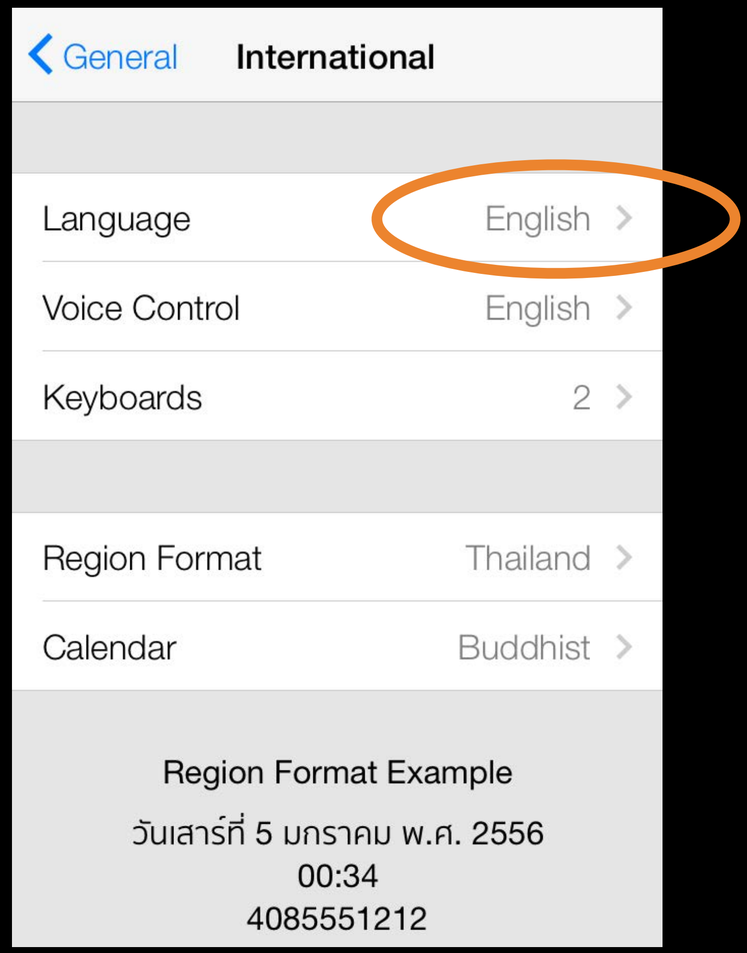
What You'll Learn

- Localization
- Locale data
- International text

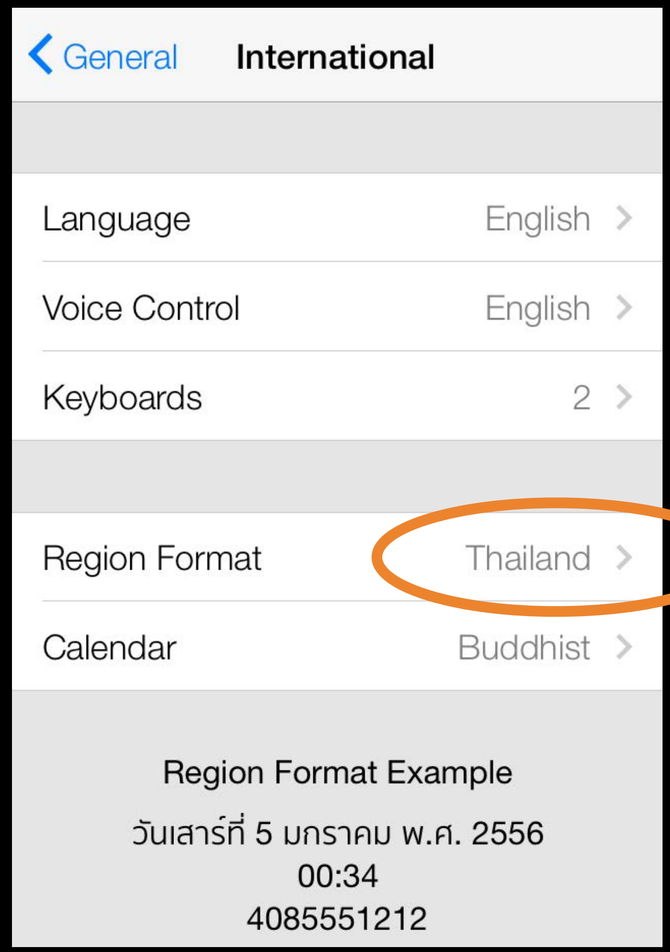
iOS Language and Locale Settings



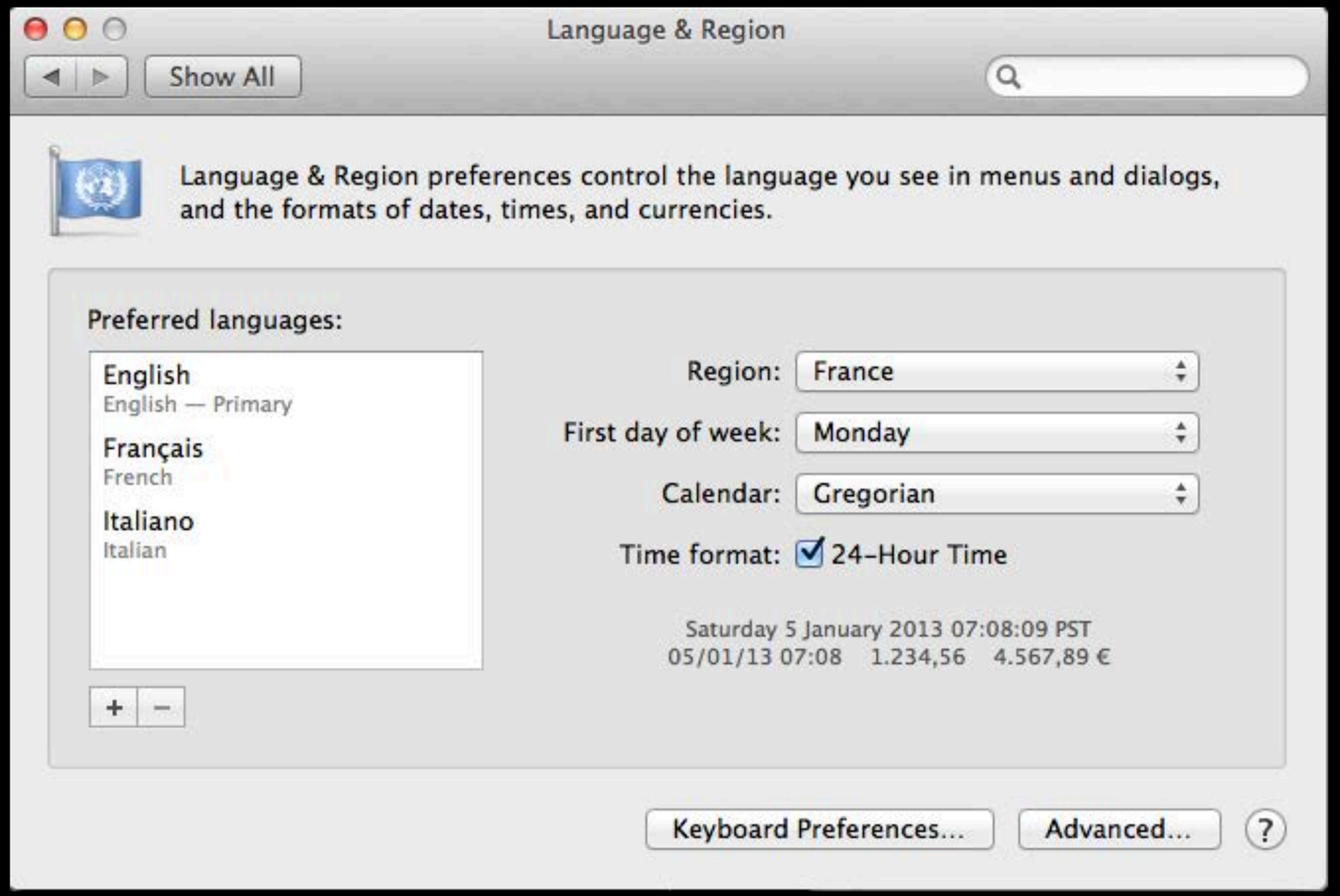
iOS Language Settings



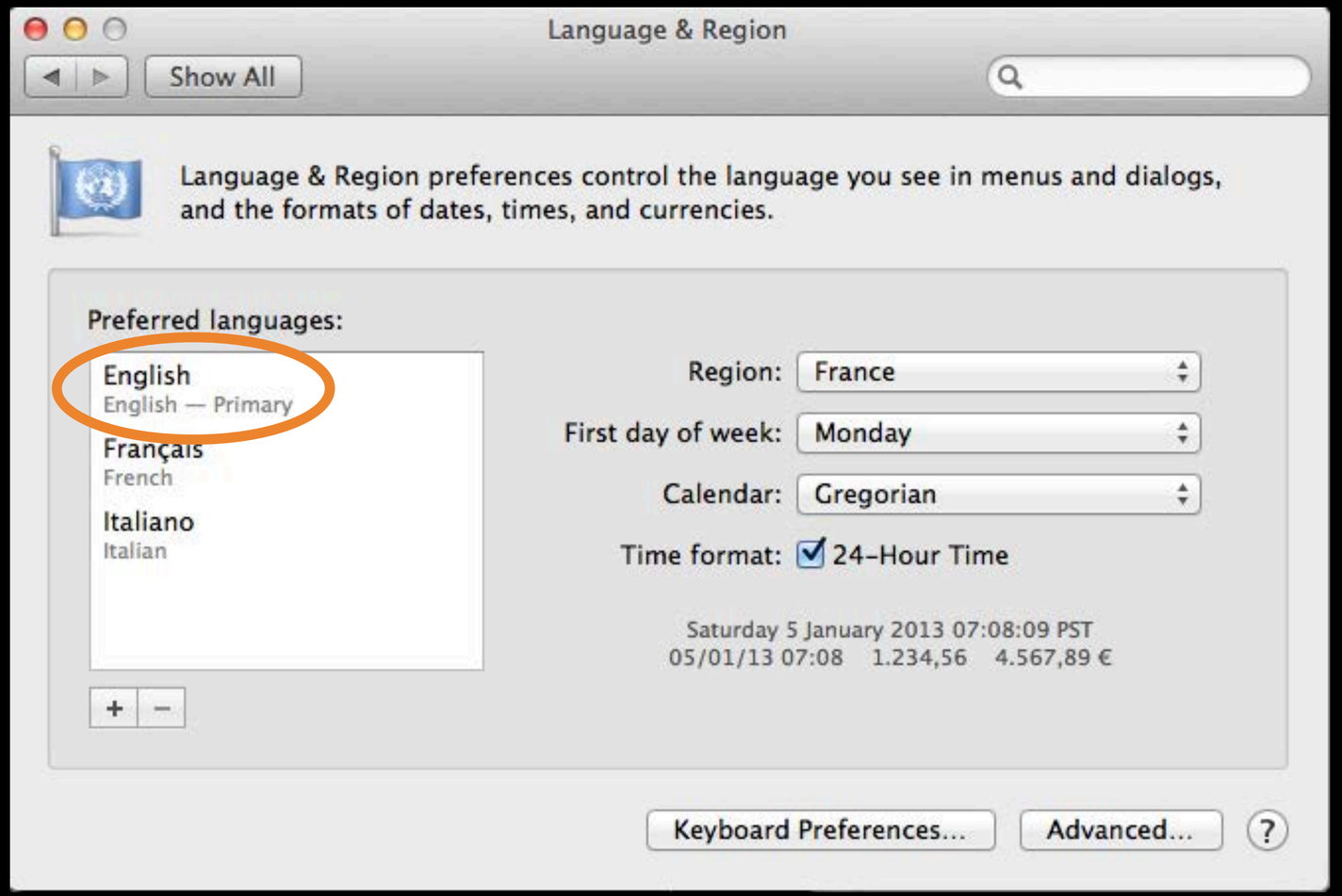
iOS Locale Settings



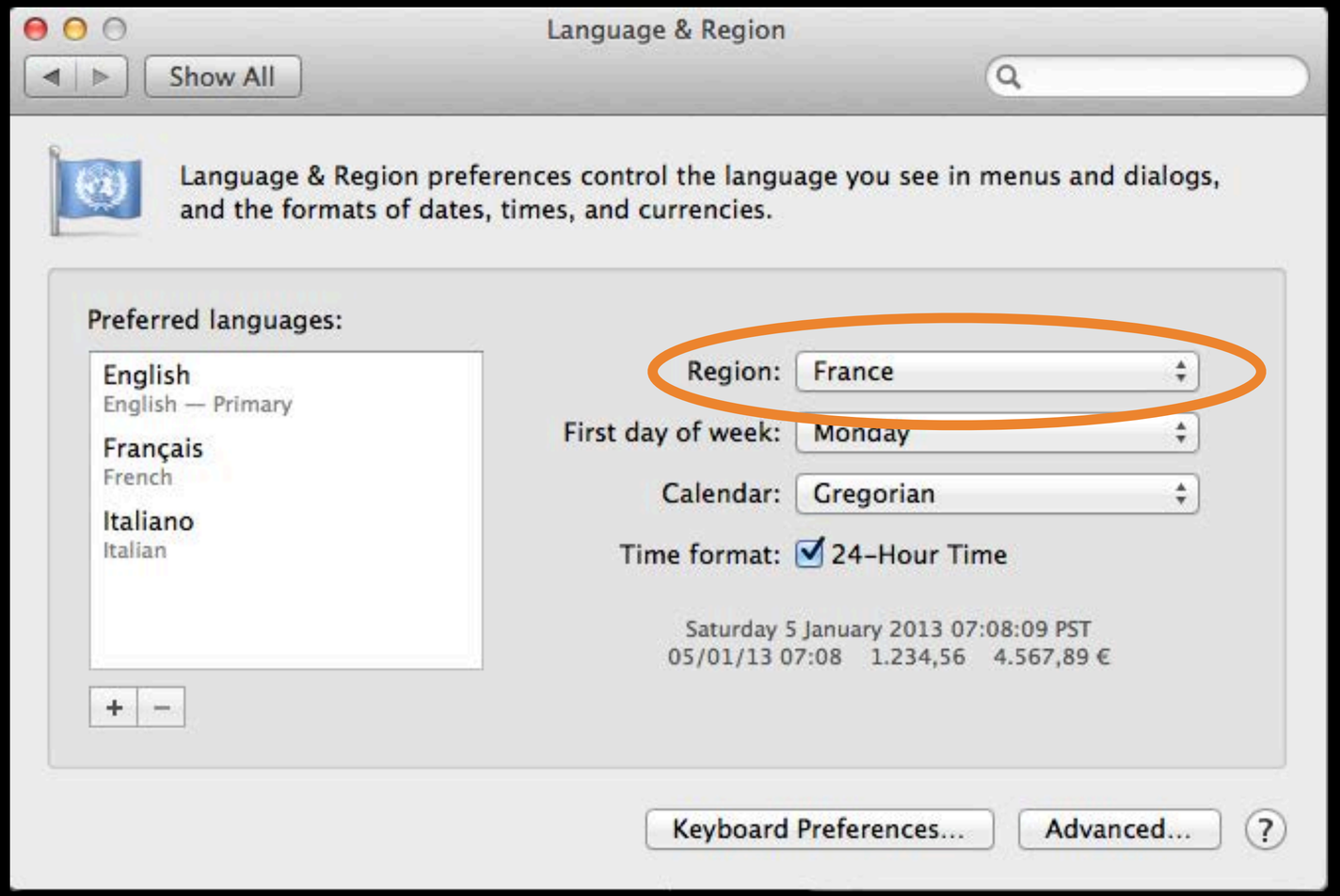
OS X Language and Locale Settings



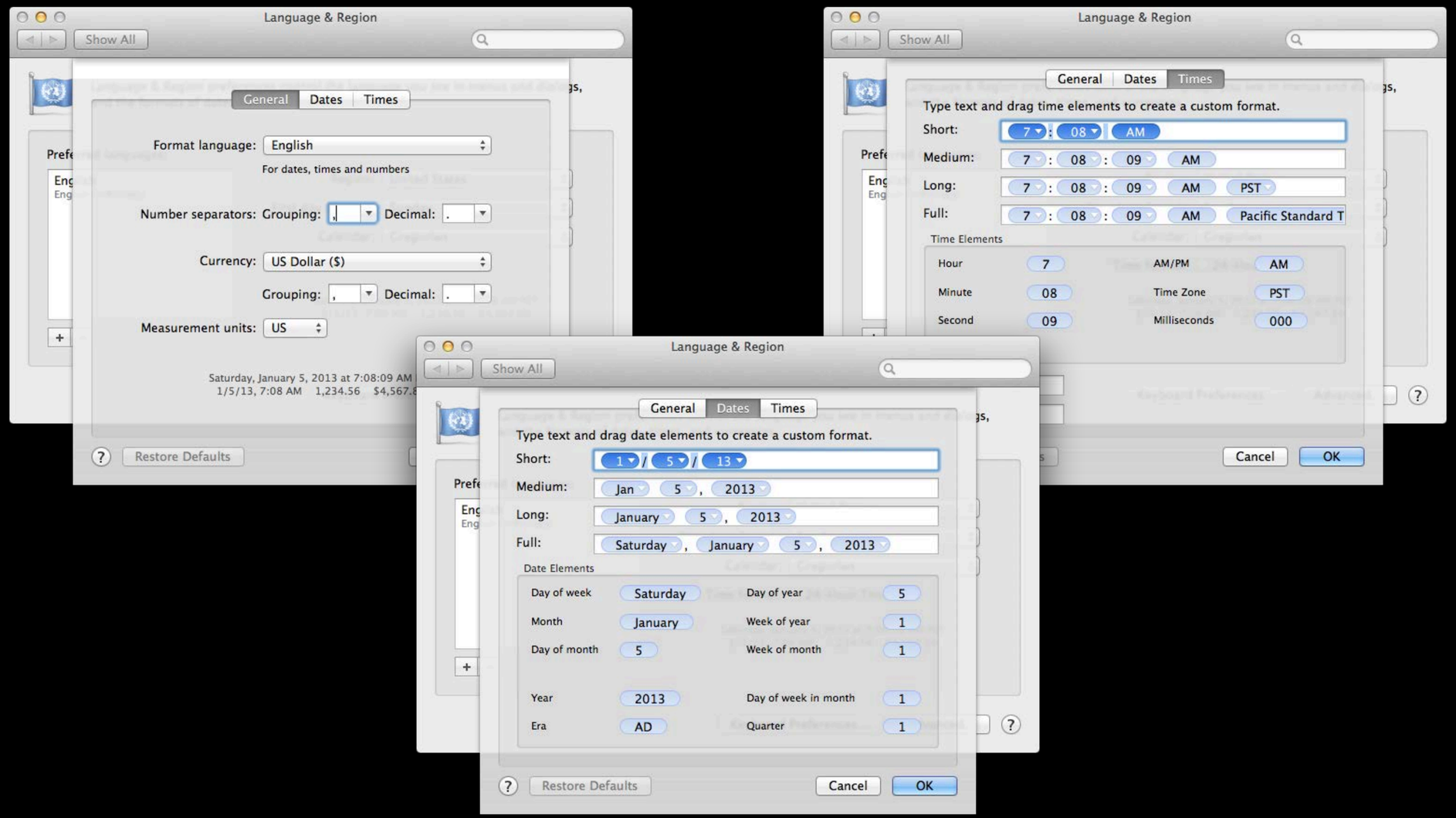
OS X Language Settings



OS X Locale Settings



OS X Detailed Locale Settings



Localization

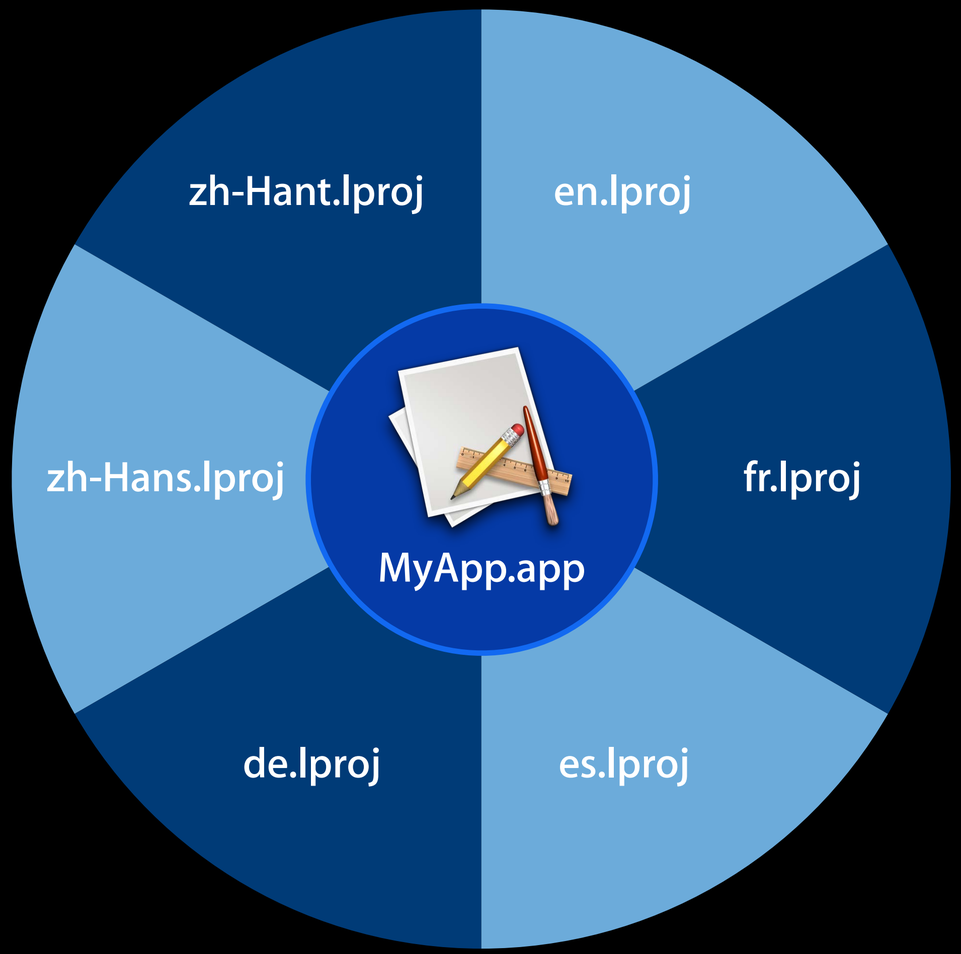
Albert Wan

Localization

- Translating your app into different languages for different markets
- Adapting your app to local norms

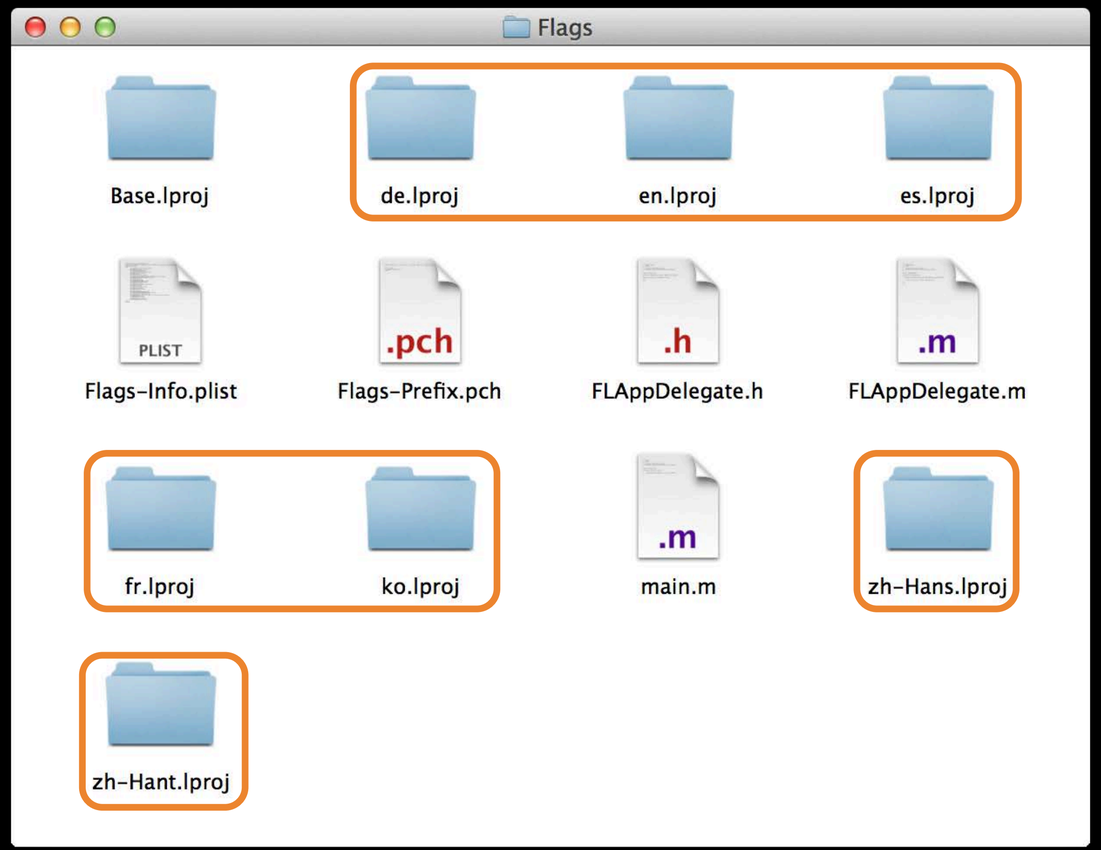
Single Binary, Multiple Localizations

Single Binary, Multiple Localizations



Language-Specific Project Directory

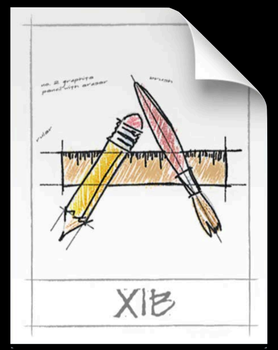
- Localizers only edit lproj folders
- These contain strings files and resource files for localization



Localizing Interface Files

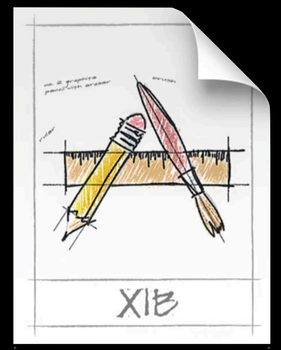
- Old style: localize each nib file
  - Localizers need to edit nib file
- New style: use Base Internationalization
  - Edit only one set of storyboards and nib files
  - A strings file is generated for each storyboard and nib file
  - Localizers only need to edit the strings file
- Auto Layout should be used with Base Internationalization

Base Internationalization



MainPage.xib

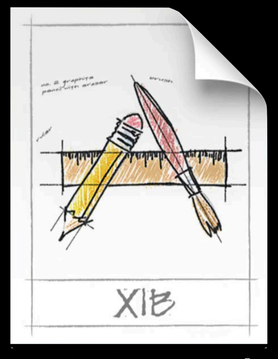
Base Internationalization



Base Internationalization File

MainPage.xib

Base Internationalization



MainPage.xib

Base Internationalization File

en.lproj

MainPage.strings

zh-Hans.lproj

MainPage.strings

ja.lproj

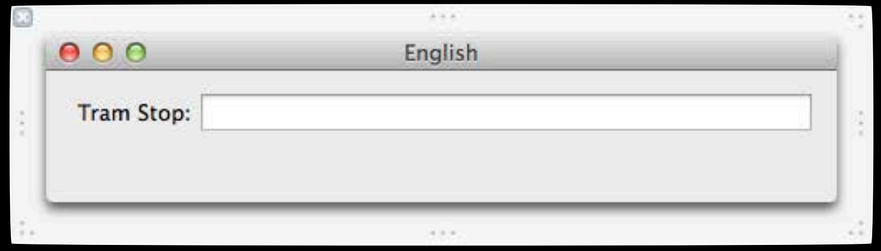
MainPage.strings

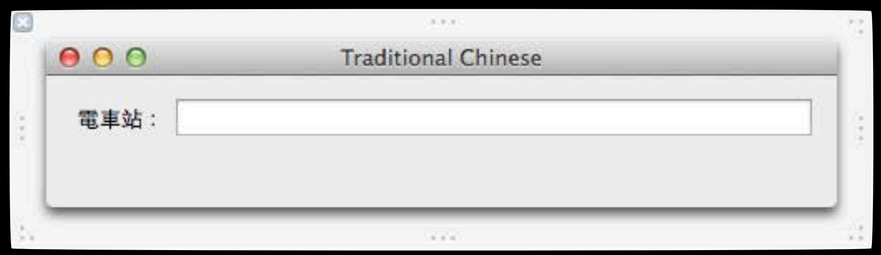
ko.lproj

MainPage.strings

Auto Layout

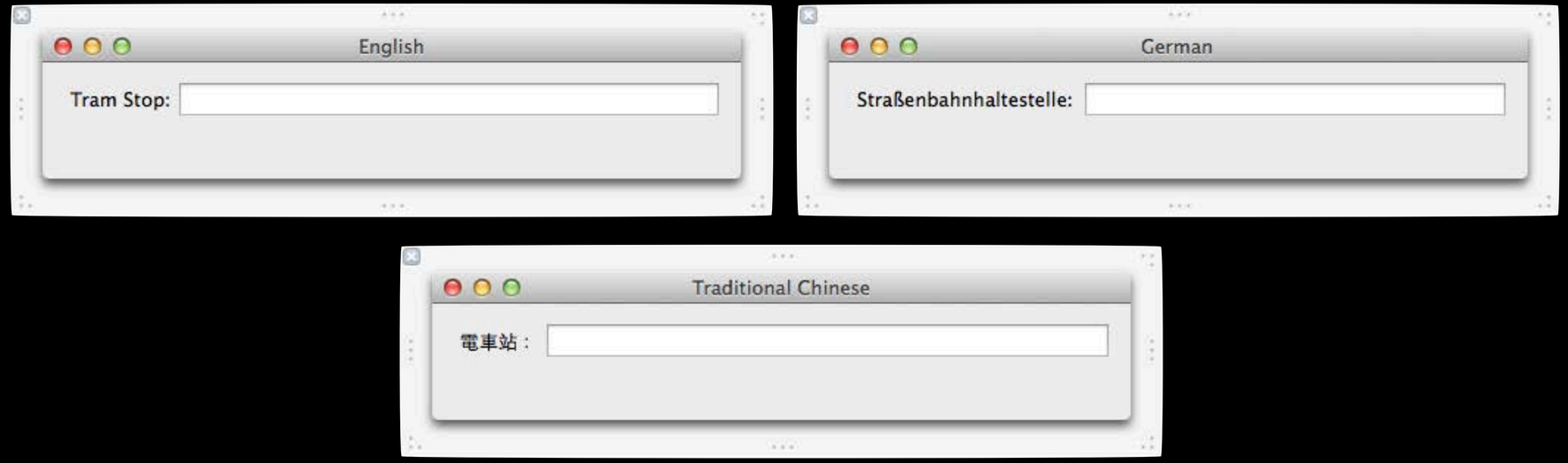
- Should be used for Base Internationalization
- Defines constraints for laying out elements in your user interface
- Allows elements with localized text to be resized appropriately





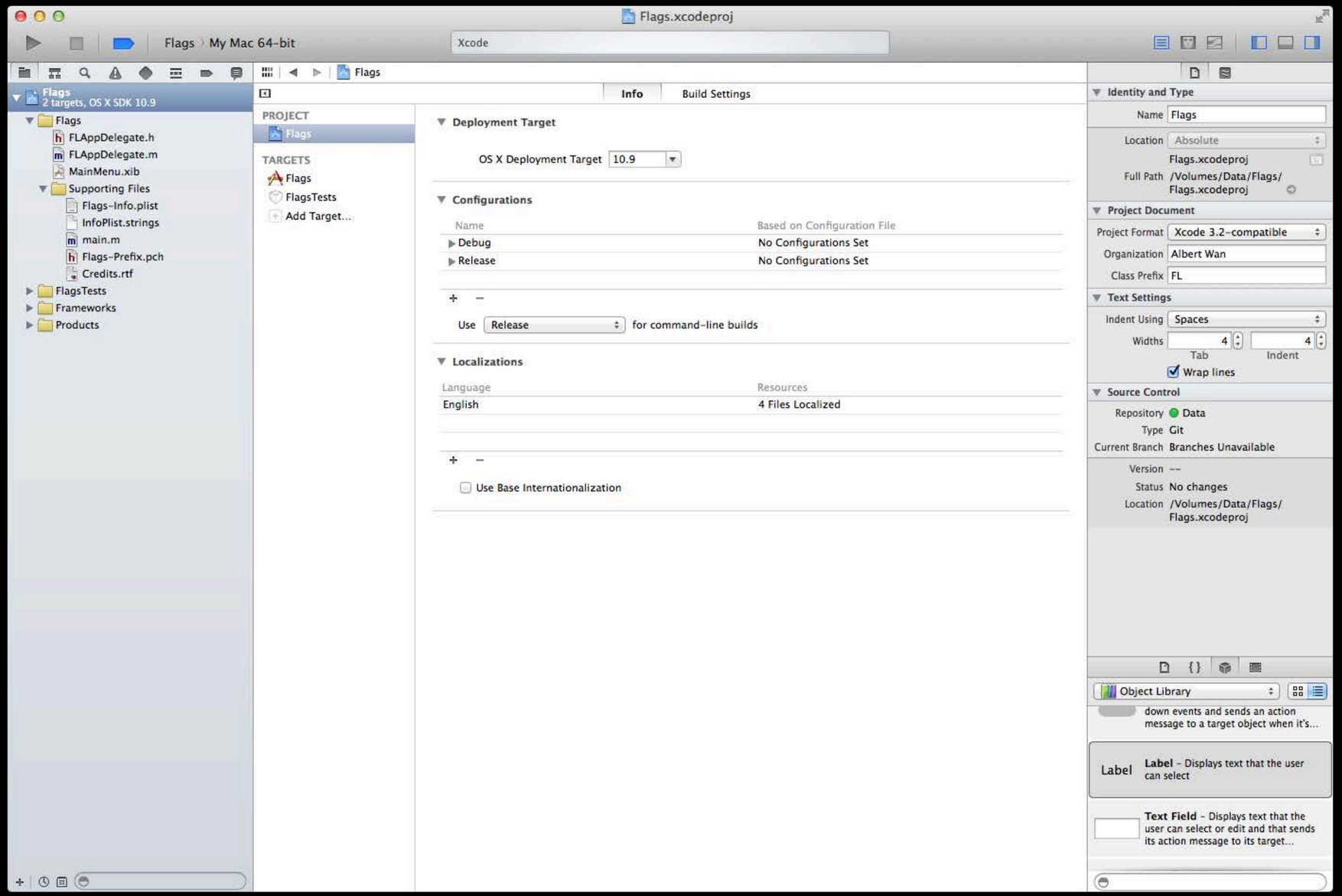
Auto Layout

- Should be used for Base Internationalization
- Defines constraints for laying out elements in your user interface
- Allows elements with localized text to be resized appropriately



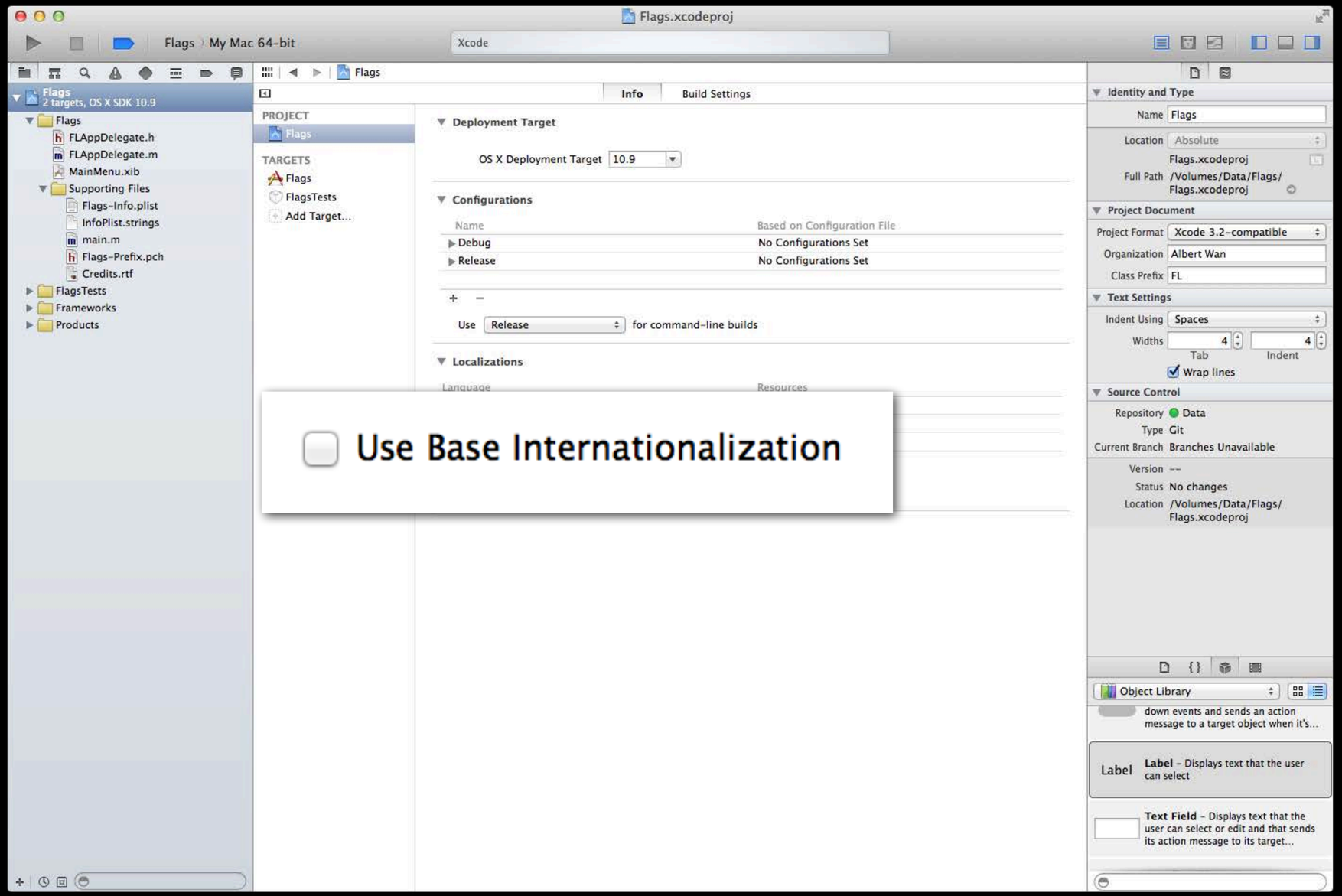
Using Base Internationalization

Getting started



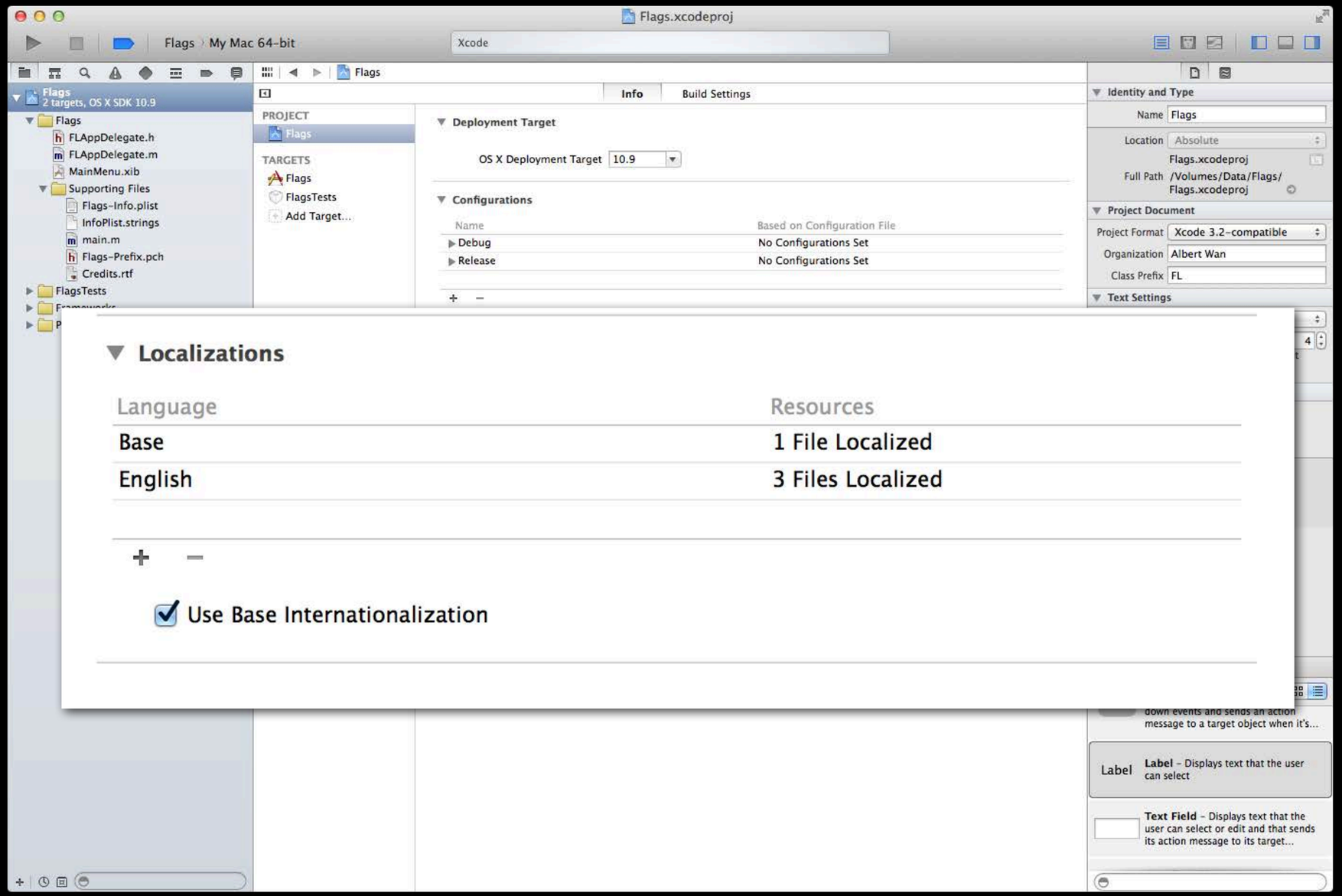
Using Base Internationalization

Getting started



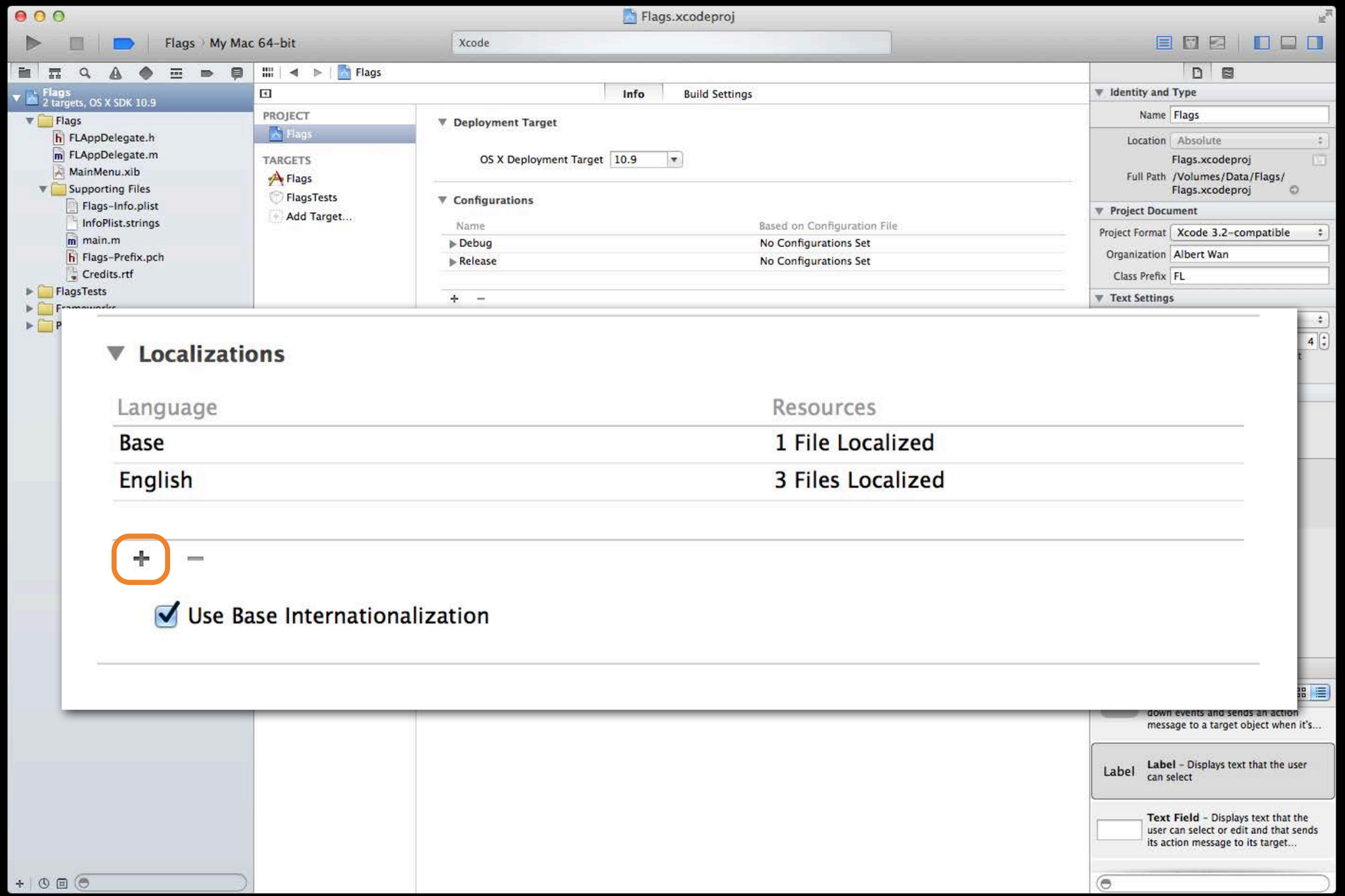
Using Base Internationalization

Getting started



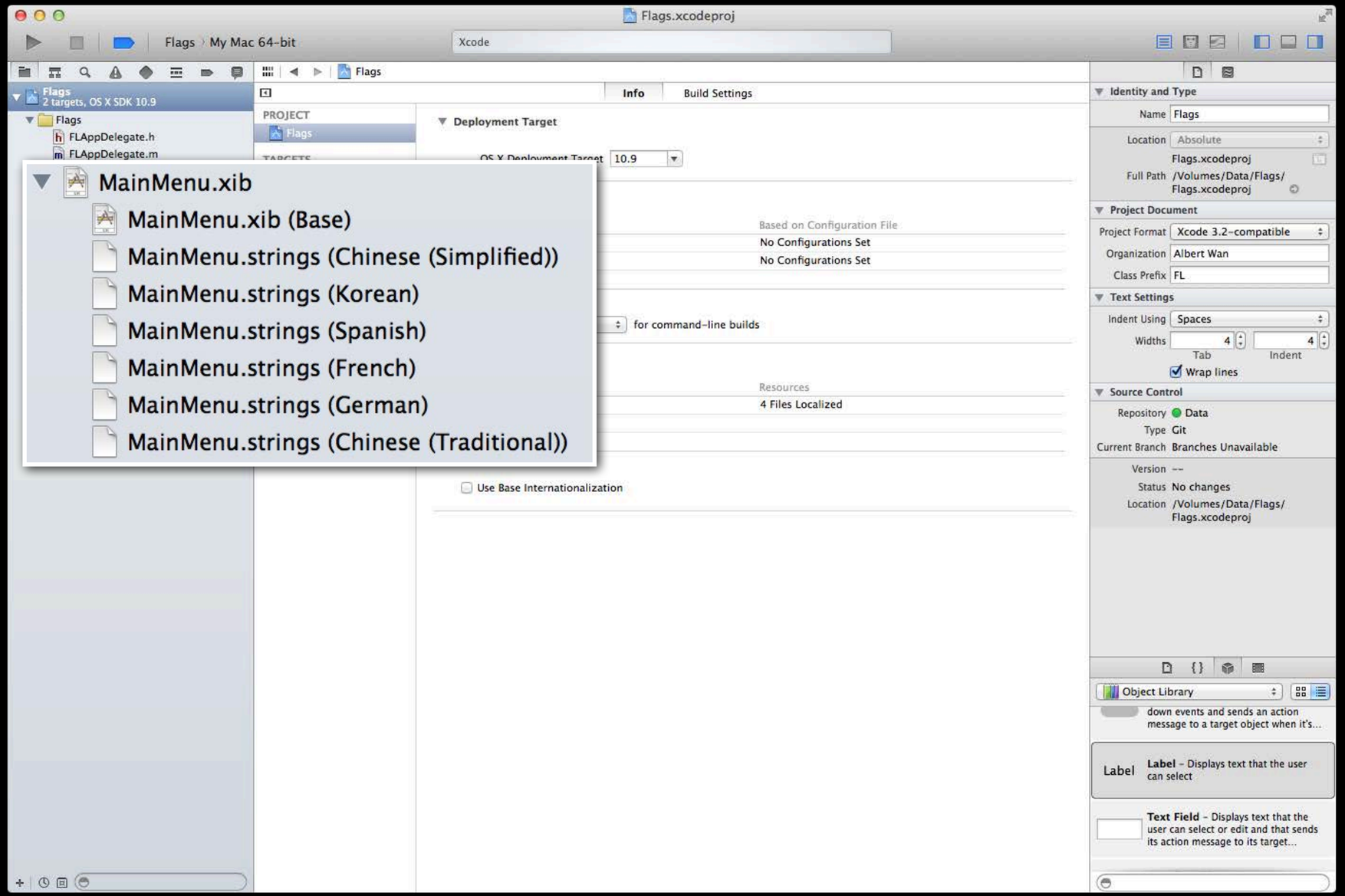
Using Base Internationalization

Getting started



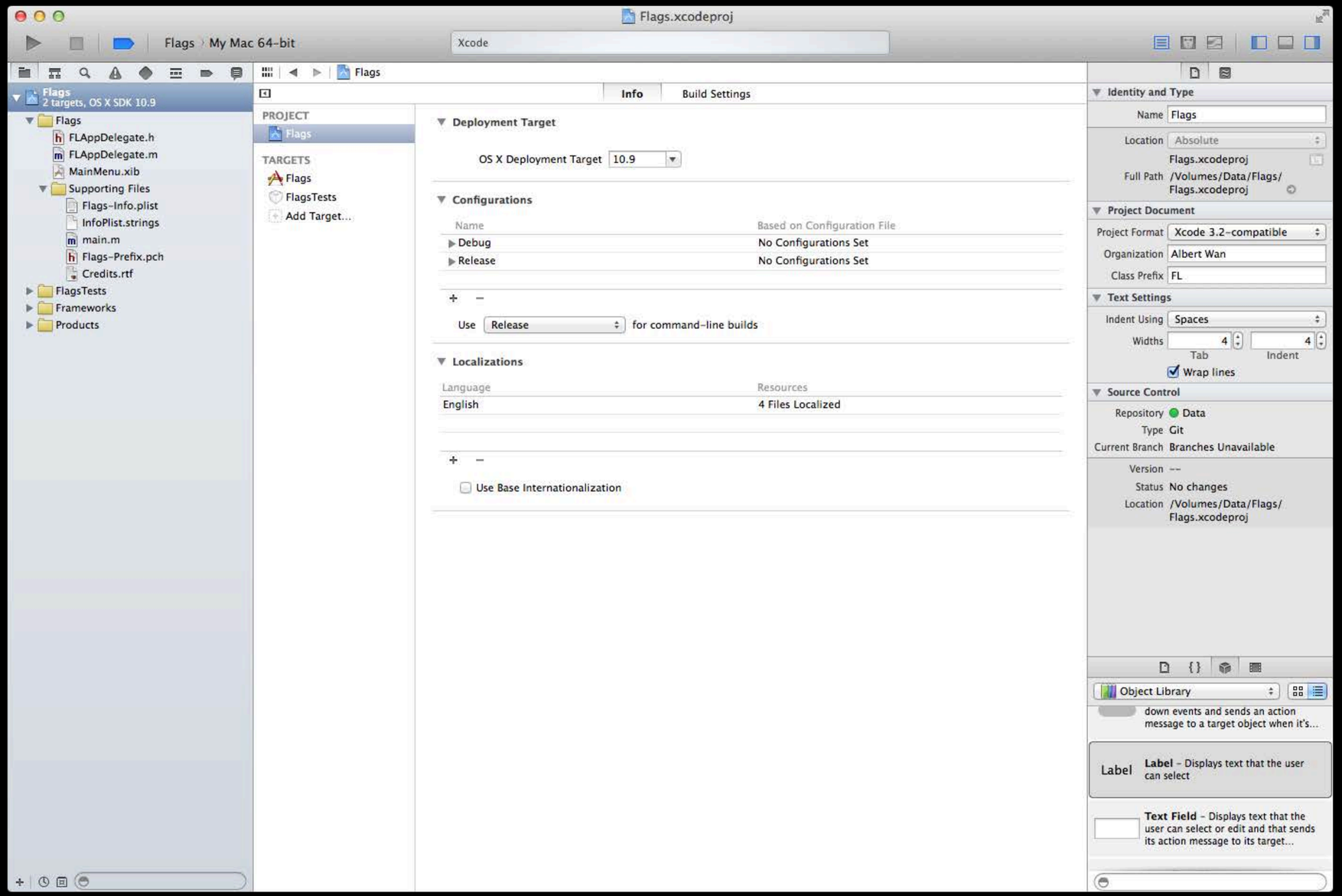
Using Base Internationalization

Getting started



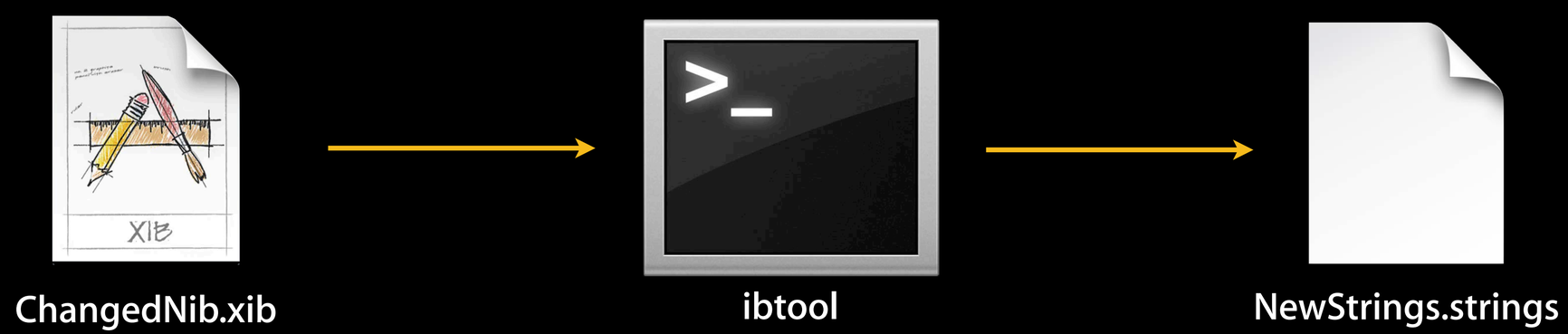
Using Base Internationalization

Getting started



Generating Localization Files

- Use `ibtool` every time you update your labels and text
- In the Base.lproj folder:
  - `ibtool ChangedNib.xib --generate-strings-file NewStrings.strings`
- Open the generated output file and copy all new string entries to `ChangedNib.strings` in each lproj



Pitfalls: Auto Layout

- Avoid using fixed widths in Auto Layout
- Prefer intrinsic content size
- Try out each localization to see if there are layout issues

The strings File

- Localized strings are stored as a table in a *.strings file

en.lproj/Localizable.strings

```
"Name" = "Name";
"Password" = "Password";
```

zh-Hans.lproj/Localizable.strings

```
"Name" = "名字";
"Password" = "密码";
```

es.lproj/Localizable.strings

```
"Name" = "Nombre";
"Password" = "Contraseña";
```

Strings in Code

- User visible text in your source code should use NSLocalizedString
- NSLocalizedString takes in a key and comment for the localizer
- Variants:

```
NSLocalizedString
NSLocalizedStringFromTable
NSLocalizedStringFromTableInBundle
NSLocalizedStringWithDefaultValue
```

NSLocalizedString

Key

```
NSLocalizedString(@"RunningDistance",
                  @"distance for a marathon");
```

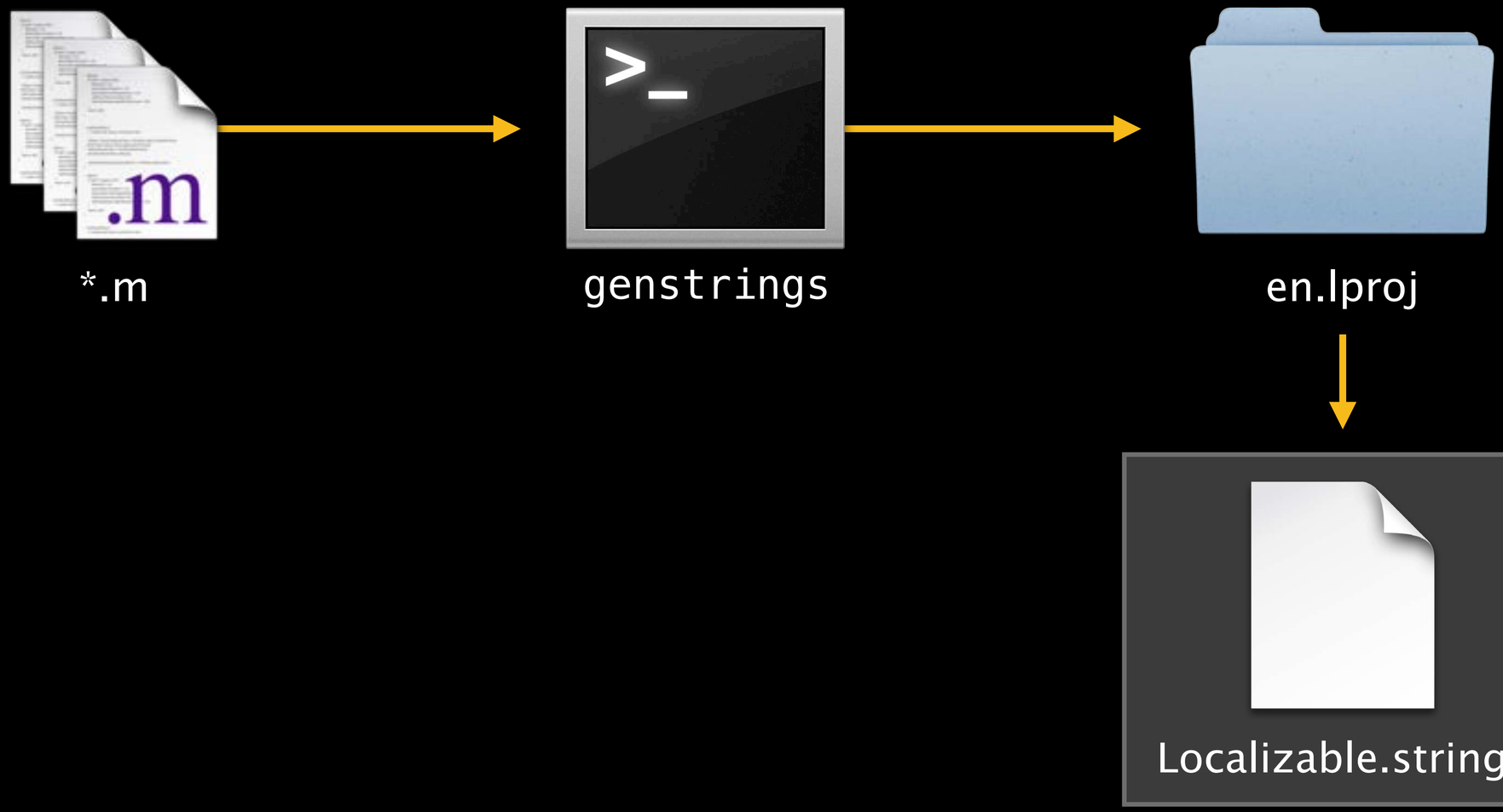
Comment for localizer

Using genstrings

- Creates strings file for code containing NSLocalizedString and variants
- Scriptable
- Customizable
- See genstrings man page

Using genstrings

```
find . -name \*.m | xargs genstrings -o en.lproj/
```



Using genstrings

```
NSLocalizedString(@"RunningDistance", @"distance for a marathon");
```

en.lproj/Localizable.strings

```
/* distance for a marathon */
"RunningDistance" = "RunningDistance";
```

ja.lproj/Localizable.strings

```
/* distance for a marathon */
"RunningDistance" = "RunningDistance";
```

Using genstrings

```
NSLocalizedString(@"RunningDistance", @"distance for a marathon");
```

en.lproj/Localizable.strings

```
/* distance for a marathon */
"RunningDistance" = "26.22 miles";
```

ja.lproj/Localizable.strings

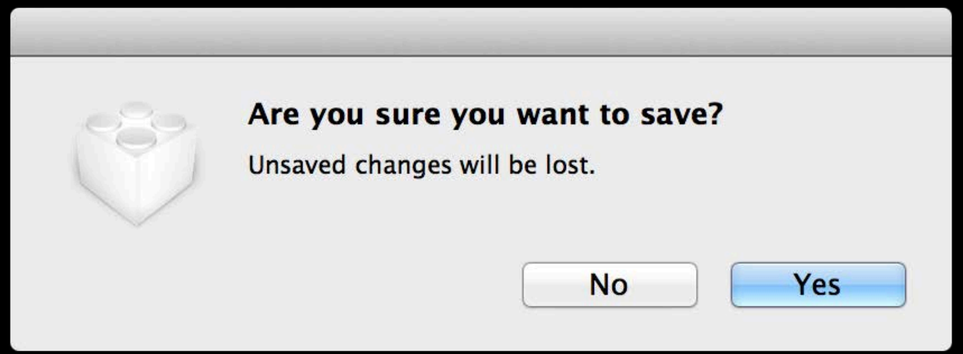
```
/* distance for a marathon */
"RunningDistance" = "42.20 キロメートル";
```

Pitfalls: NSLocalizedString

- Overloading keys

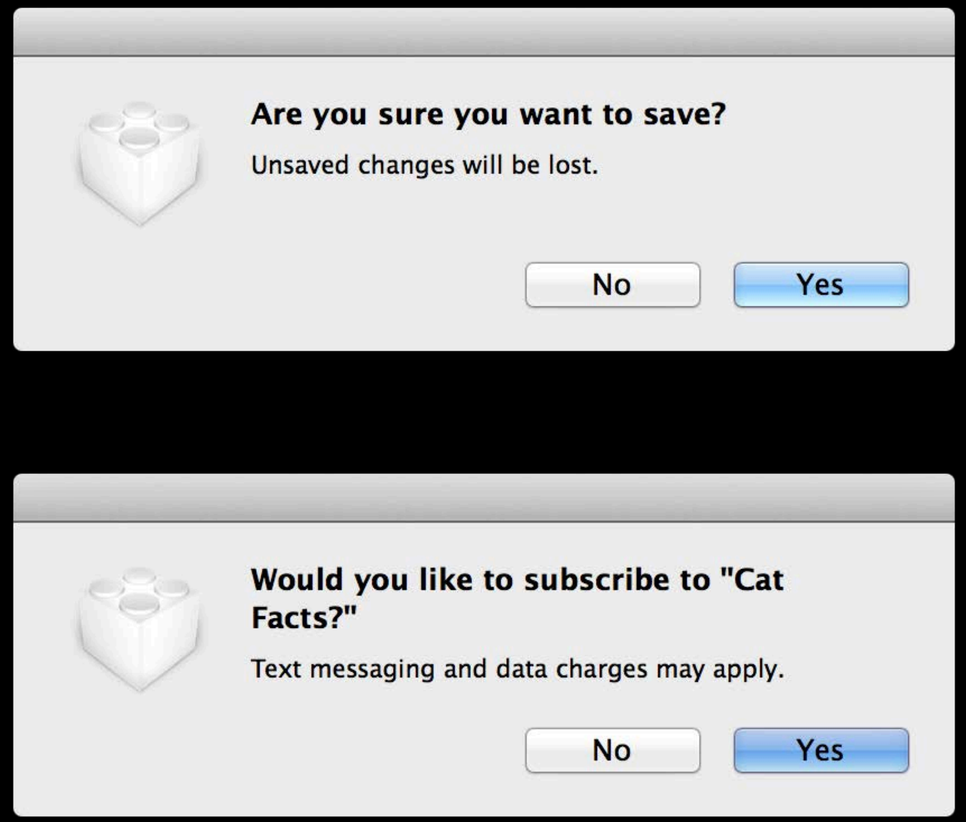
Pitfalls: NSLocalizedString

- Overloading keys



Pitfalls: NSLocalizedString

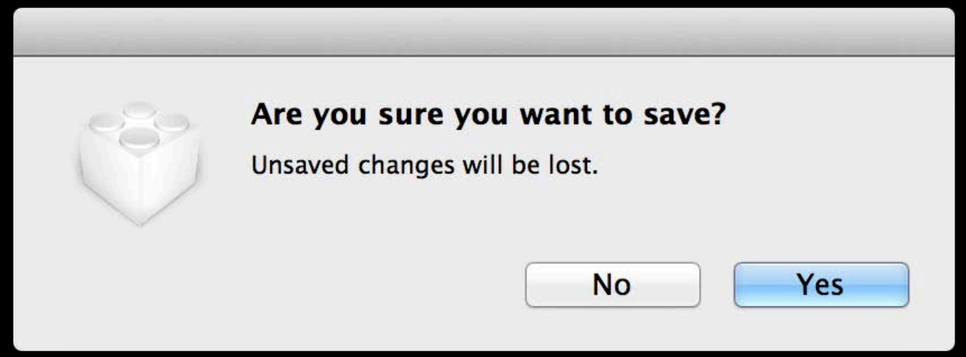
- Overloading keys



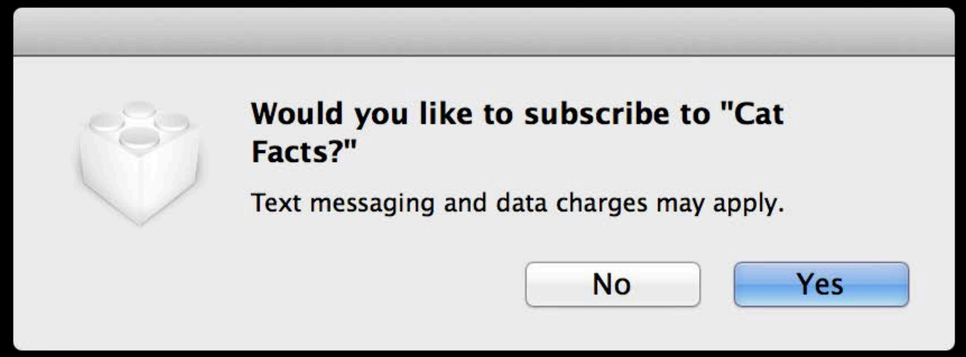
Pitfalls: NSLocalizedString

- Overloading keys

```
NSLocalizedString(@"Yes", @"Alert
Button Affirmative");
```



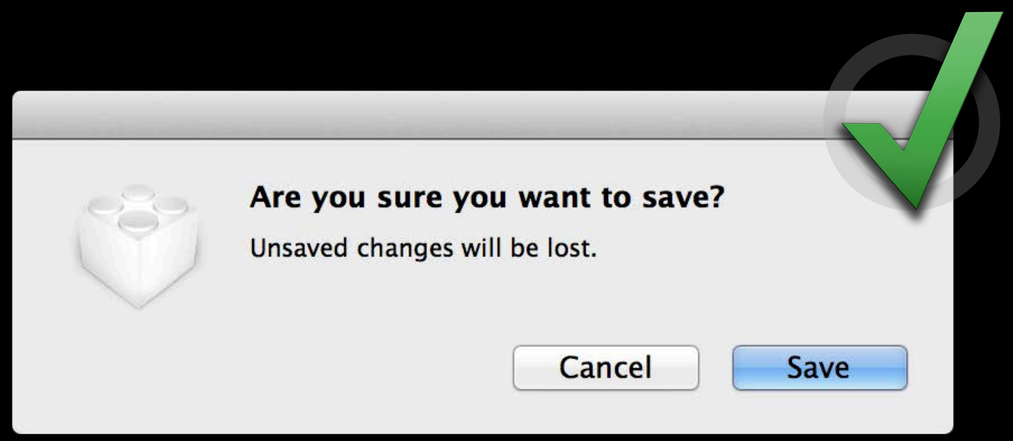

```
NSLocalizedString(@"No", @"Alert
Button Negative");
```



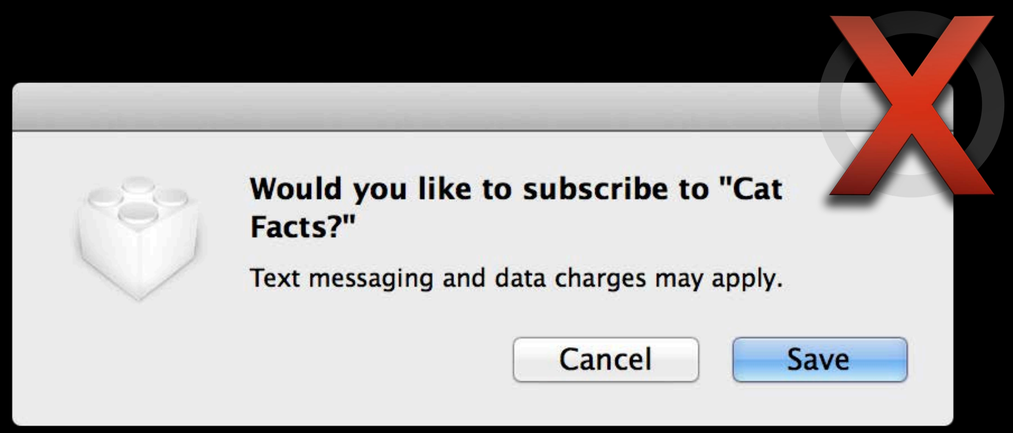
Pitfalls: NSLocalizedString

- Overloading keys

```
NSLocalizedString(@"Save",
@"Alert Button Affirmative");
```




```
NSLocalizedString(@"Cancel", @"Alert
Button Negative");
```

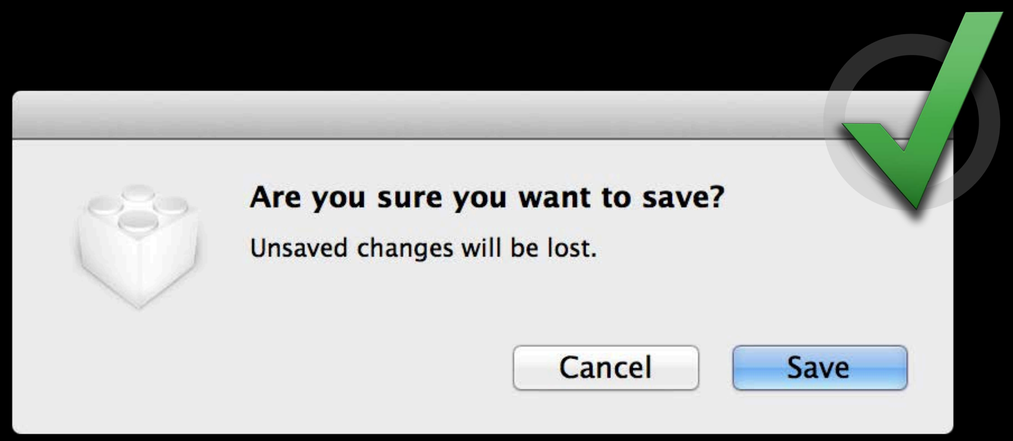


Pitfalls: NSLocalizedString

- Overloading keys

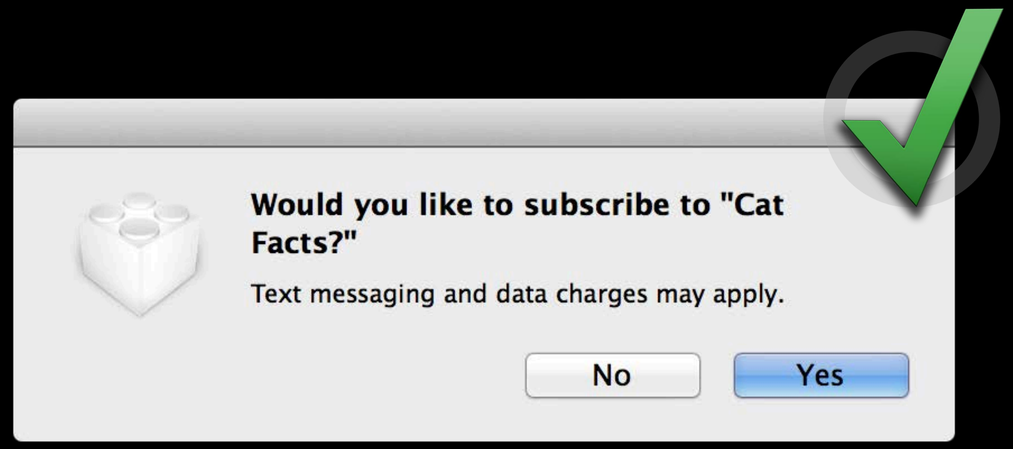
```
NSLocalizedString(@"AffirmSave",
@"Save Affirmative");

NSLocalizedString(@"CancelSave",
@"Cancel Save");
```




```
NSLocalizedString(
@"ConfirmSubscribe", @"User wants
cat facts");

NSLocalizedString(
@"DenySubscribe", @"User doesn’t
want cat facts");
```



Pitfalls: NSLocalizedString

```
/* No comment provided */
"Yes" = "Yes";
```

Insufficient comments

Pitfalls: NSLocalizedString

```
/* Ask if the user would like to sync with iCloud */
"ShouldSyncWithiCloud" = "Yes";
```

Insufficient comments

Pitfalls: NSLocalizedString

- Composing phrases together
- Grammatical number and gender of words may not agree

```
/* Go to next page/chapter */
"GoToNext” = "Go to next %@";
"chapter" = "chapter";
"page" = "page";
```

Pitfalls: NSLocalizedString

- Composing phrases together
- Grammatical number and gender of words may not agree

```
/* Go to next page/chapter */
"GoToNext” = "Go to next %@";
"chapter" = "chapter";
"page" = "page";
```

Go to next chapter
Go to next page

Pitfalls: NSLocalizedString

- Composing phrases together
- Grammatical number and gender of words may not agree

```
/* Go to next page/chapter */
"GoToNext” = "Go to next %@";
"chapter" = "chapter";
"page" = "page";
```

Go to next chapter Ir al siguiente capítulo
Go to next page

Pitfalls: NSLocalizedString

- Composing phrases together
- Grammatical number and gender of words may not agree

```
/* Go to next page/chapter */
"GoToNext” = "Go to next %@";
"chapter" = "chapter";
"page" = "page";
```

Go to next chapter Ir al siguiente capítulo
Go to next page Ir al siguiente página

Pitfalls: NSLocalizedString

- Composing phrases together
- Grammatical number and gender of words may not agree

```
/* Go to next chapter */
"GoToNextChapter" = "Go to next chapter";

/* Go to next page */
"GoToNextPage" = "Go to next page";
```

Go to next chapter Ir al siguiente capítulo
Go to next page Ir a la siguiente página

Using stringsdict

- A way to follow grammatical rules based on plurality and/or gender
  - `@"1 file remaining"`
  - `@"%d files remaining"`
- Plurals in general are hard to handle across languages
- Localized property list that encapsulates these rules

```
result = [NSString localizedStringWithFormat:
        NSLocalizedString(@"%d file(s) remaining",
        @"Message shown for remaining files"), n];
```

- Foundation release notes have a full description

Using stringsdict

- A way to follow grammatical rules based on plurality and/or gender
  - `@"1 file remaining"`
  - `@"%d files remaining"`
- Plurals in general are hard to handle across languages
- Localized property list that encapsulates these rules

```
result = [NSString localizedStringWithFormat:
        NSLocalizedString(@"%d file(s) remaining",
        @"Message shown for remaining files"), n];
```

- Foundation release notes have a full description

What's New in Cocoa

Mission
Tuesday 3:15PM

Sample stringsdict File

en.lproj/Localizable.stringsdict

```
...
<key>NSStringLocalizedFormatKey</key>
<string>%#@files@</string>
<key>files</key>
<dict>
    <key>NSStringFormatSpecTypeKey</key>
    <string>NSStringPluralRuleType</key>
    <key>NSStringFormatValueTypeKey</key>
    <string>d</string>
    <key>one</key>
    <string>1 file remaining</string>
    <key>other</key>
    <string>%d files remaining</string>
</dict>
...
```

ru.lproj/Localizable.stringsdict

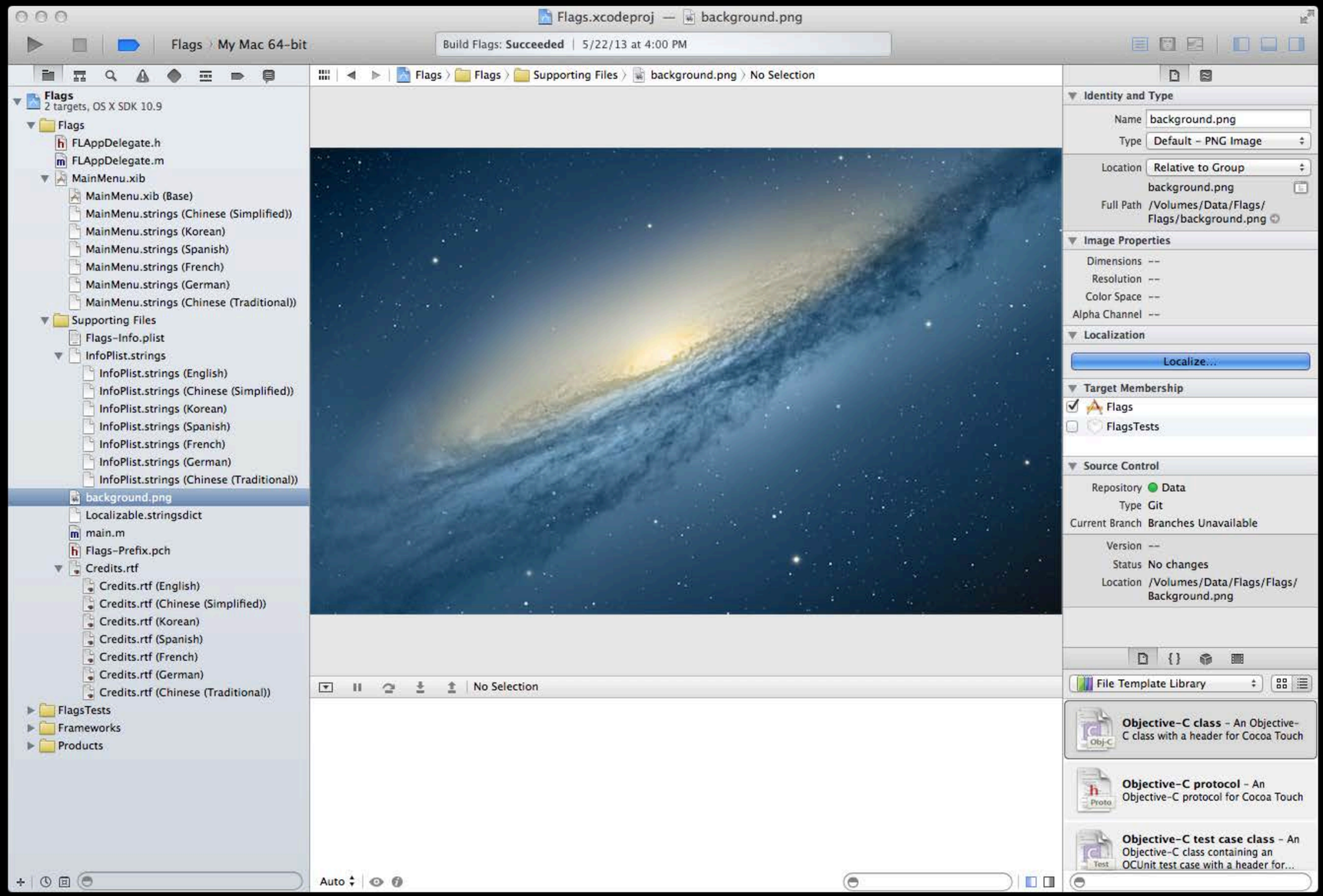
```
...
<dict>
    <key>NSStringFormatSpecTypeKey</key>
    <string>NSStringPluralRuleType</key>
    <key>NSStringFormatValueTypeKey</key>
    <string>d</string>
    <key>one</key>
    <string>Остался %d файл</string>
    <key>few</key>
    <string>Осталось %d файла</string>
    <key>many</key>
    <string>Осталось %d файлов</string>
    <key>other</key>
    <string>Осталось %d файла</string>
</dict>
...
```

Other Localized Resources

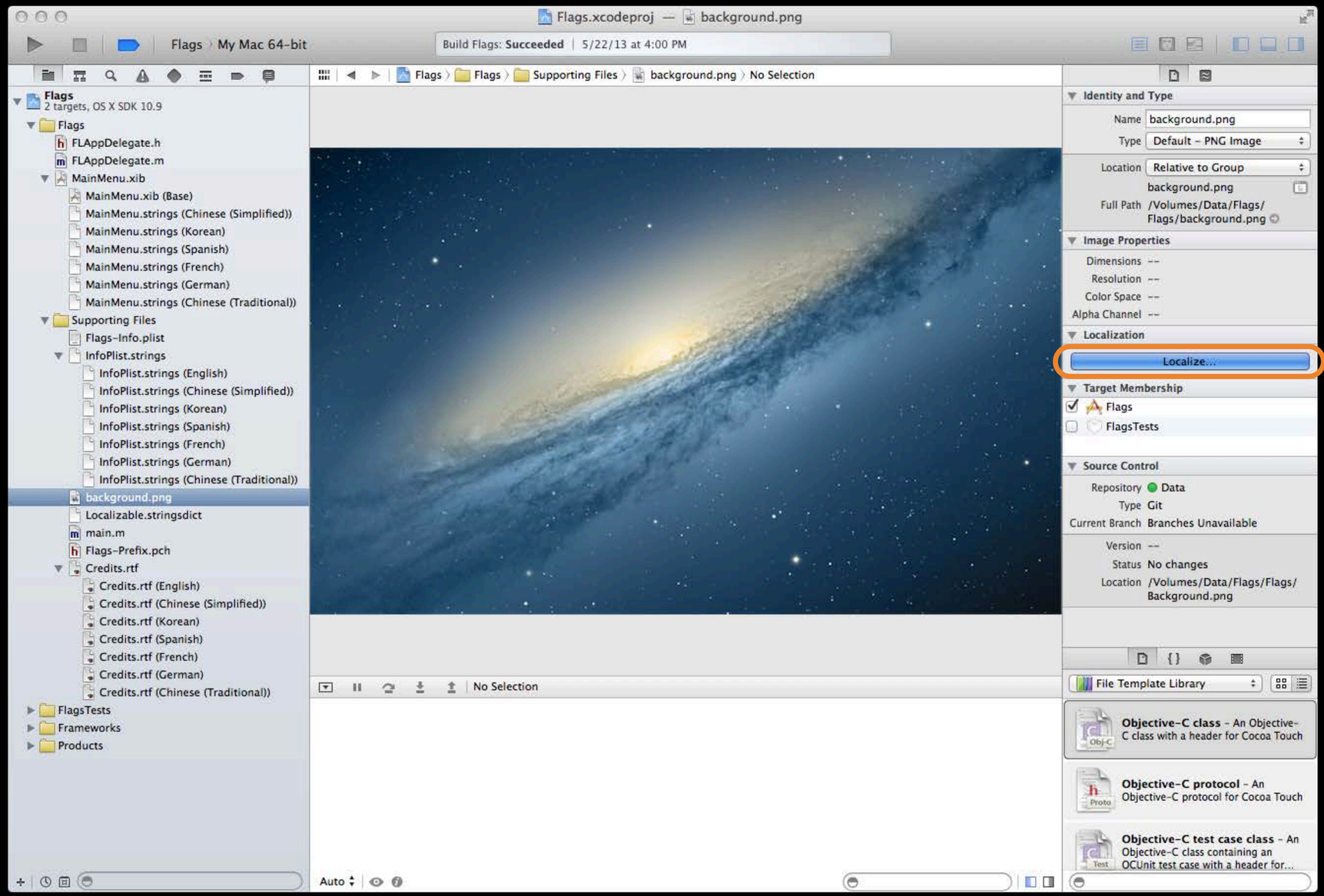
- Images, audio, and miscellaneous files can be localized
- Place the localized version into the respective lproj folder
- APIs will fetch from the localization currently running

```
[UIImage imageNamed:]
[NSImage imageNamed:]
[NSSound soundNamed:]
[NSBundle URLForResource:withExtension:]
```

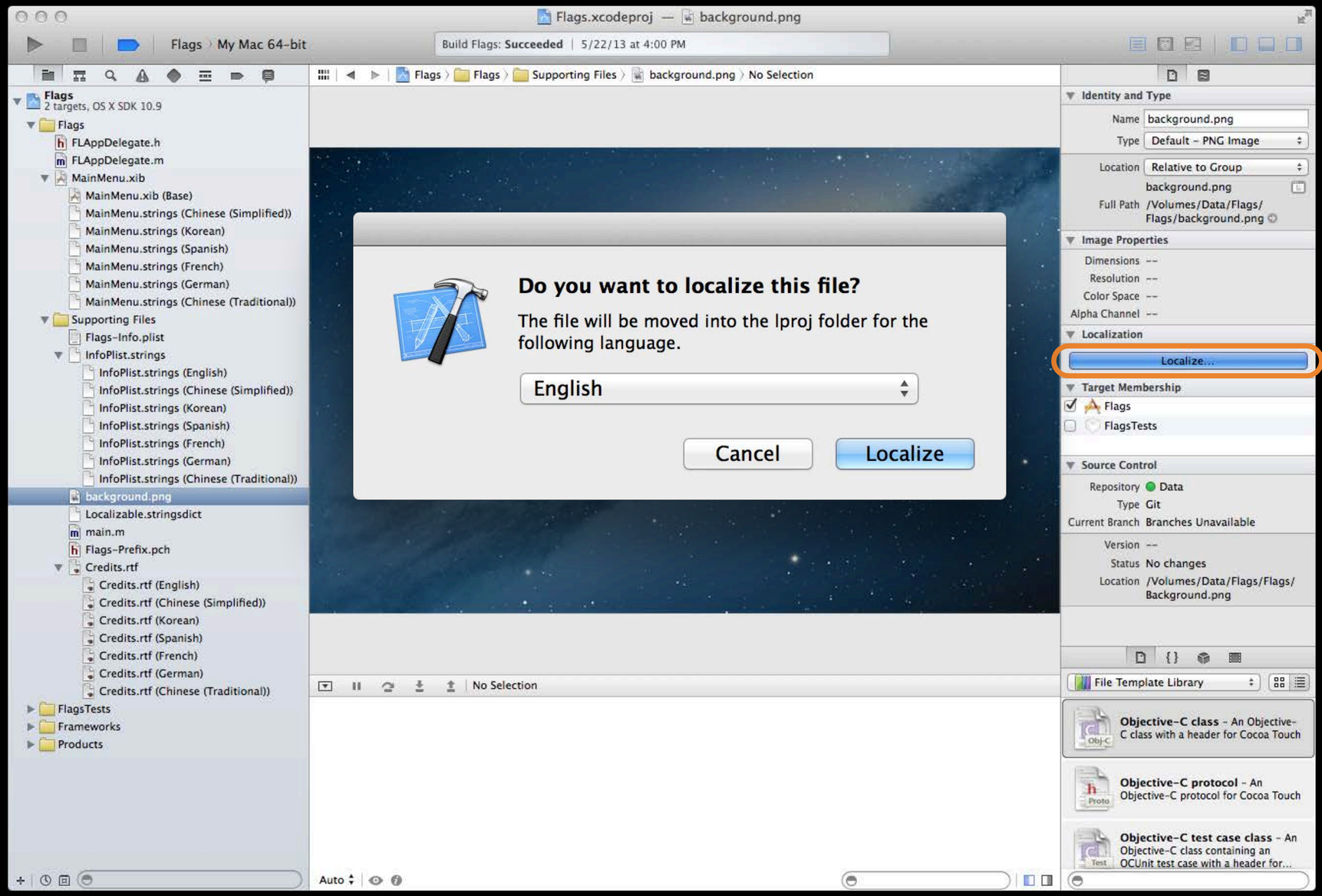
Other Localized Resources



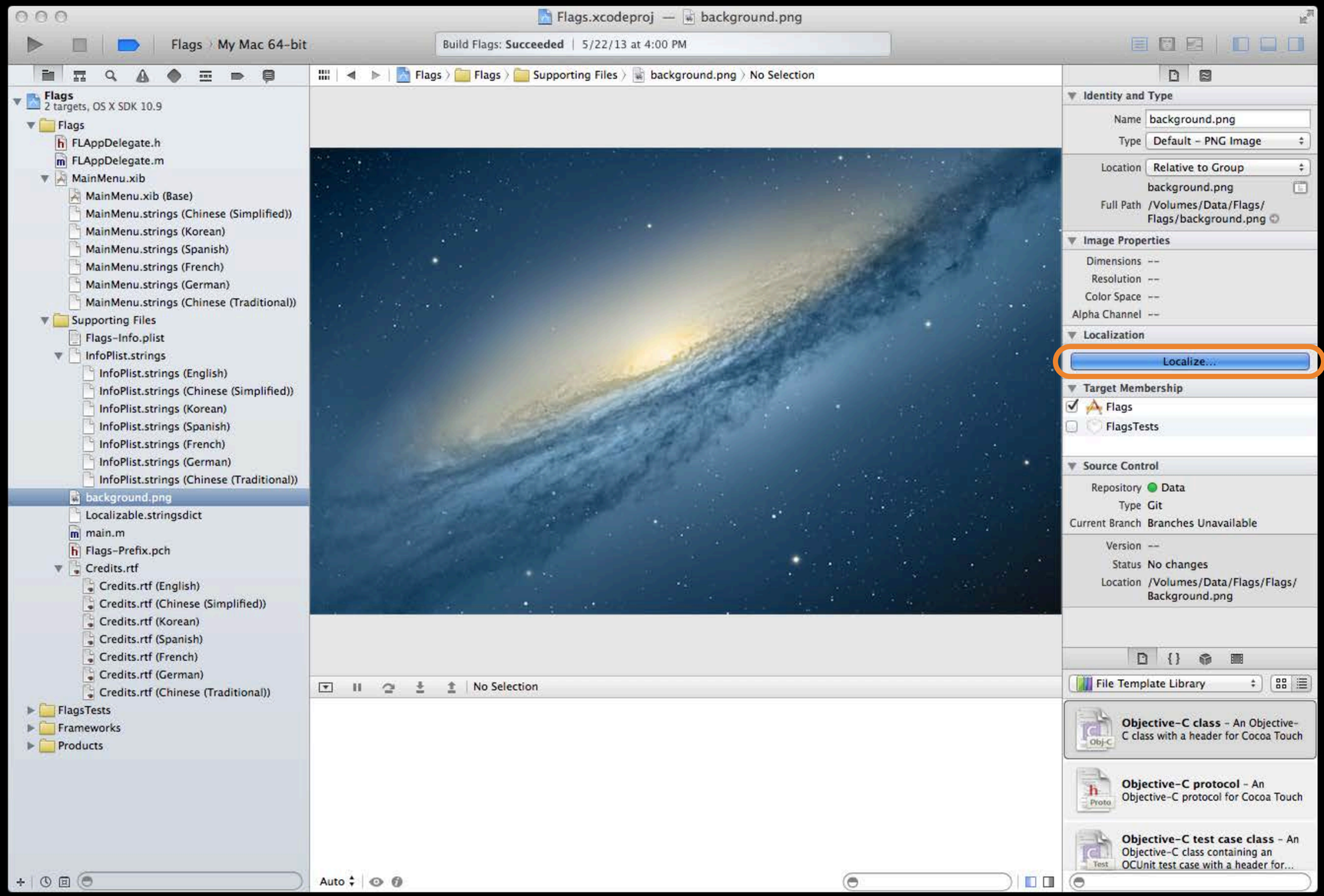
Other Localized Resources



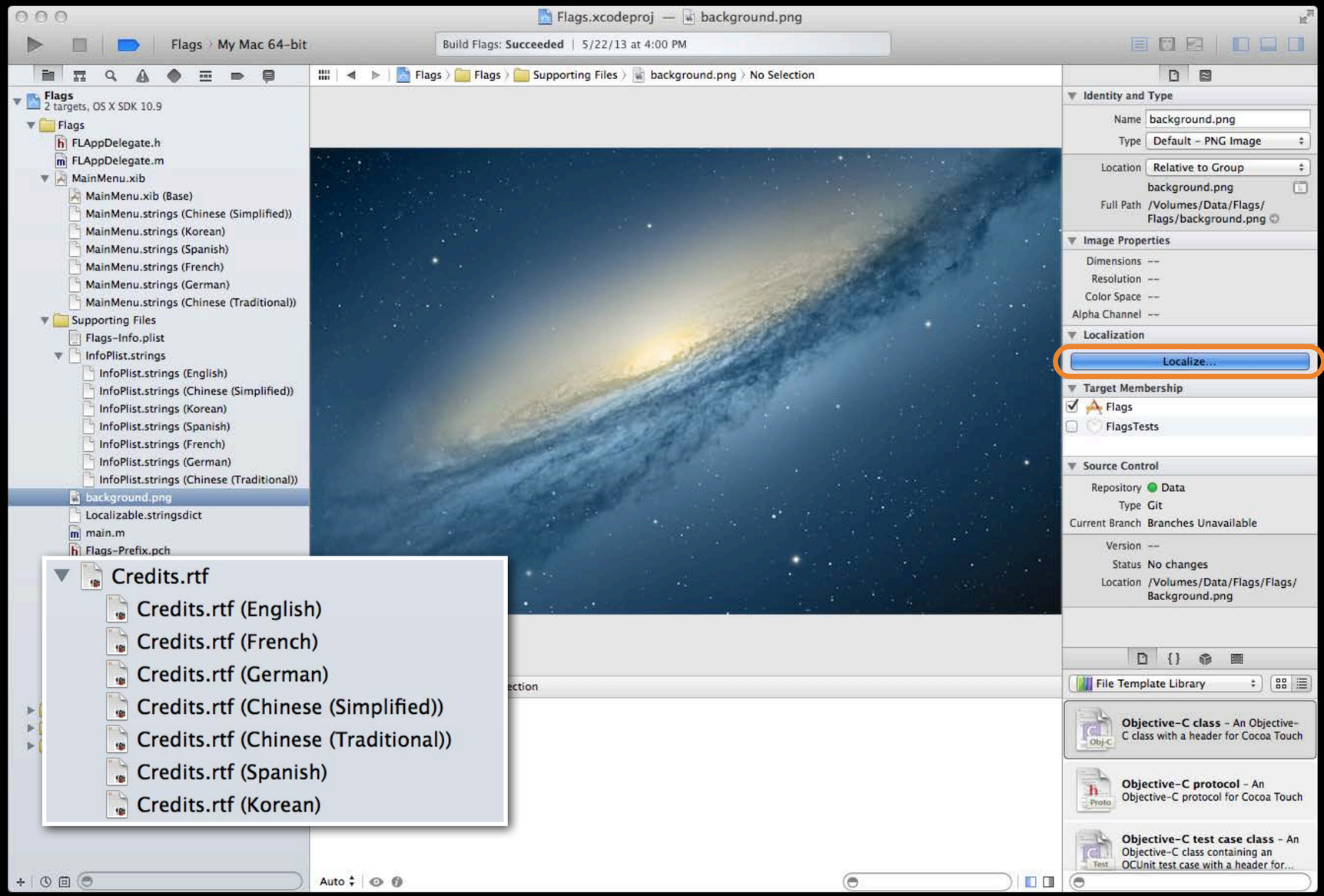
Other Localized Resources



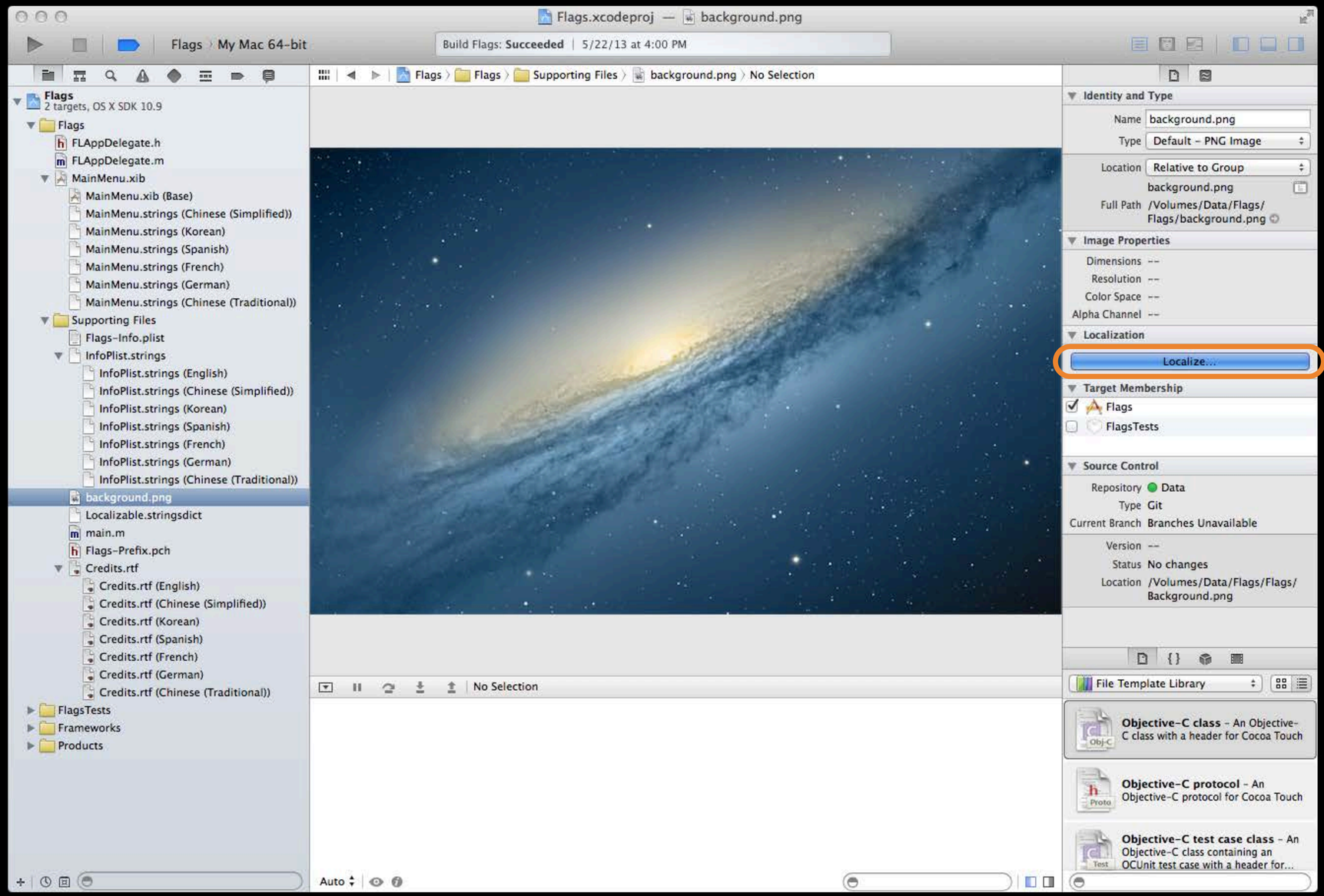
Other Localized Resources



Other Localized Resources



Other Localized Resources



Pitfalls: Images

- Text in images
- Culture and language specific references

Pitfalls: Images

- Text in images
- Culture and language specific references

Keyword Search

Pitfalls: Images

- Text in images
- Culture and language specific references

Keyword - Keyword
Stichwort - Headword
關鍵詞 - Focus word
संकेत शब्द - Indicative word

Keyword Search

Pitfalls: Images

- Text in images
- Culture and language specific references

Keyword - Keyword
Stichwort - Headword
關鍵詞 - Focus word
संकेत शब्द - Indicative word

Pitfalls: Images

- Text in images
- Culture and language specific references

Keyword - Keyword
Stichwort - Headword
關鍵詞 - Focus word
संकेत शब्द - Indicative word

Keyword Search

For Beginners

Pitfalls: Images

- Text in images
- Culture and language specific references

Keyword - Keyword
Stichwort - Headword
關鍵詞 - Focus word
संकेत शब्द - Indicative word

Testing Localization

- Change the system language to get the most accurate representation
- Quickly check your localization with the “AppleLanguages” argument

```
-AppleLanguages "(Korean)"
```

- Use pseudolocalization

Testing Localization

- Change the system language to get the most accurate representation
- Quickly check your localization with the “AppleLanguages” argument

```
–AppleLanguages "(Korean)"
```

- Use pseudolocalization

```
/* distance for a marathon */
"RunningDistance" = "[Ŕûüññîńǵ Dïšţ̊áäńçêè]";
```

Demo

Locale Data

Nat Hillard

Locale

Available APIs

Locale

Available APIs

API	Function
NSLocale	Obtain current region, format, etc.
NSDateFormatter	Format and parse dates and times
NSNumberFormatter	Format and parse numbers
NSCalendar	Current calendar and associated operations
NSTimeZone	Current timezone and associated operations
NSString	Sorting, searching, and more

Locale

What is it?

Locale

What is it?

- Set by the “Region Format” preference

Locale

What is it?

- Set by the “Region Format” preference
- Usually you will not deal with NSLocale object directly

Locale

What is it?

- Set by the “Region Format” preference
- Usually you will not deal with NSLocale object directly
- Locale vs. localization
  - “Locale” represents formatting standards
  - “Localization” refers to the UI Language

Locale

What is it?

- Set by the “Region Format” preference
- Usually you will not deal with NSLocale object directly
- Locale vs. localization
  - “Locale” represents formatting standards
  - “Localization” refers to the UI Language
- User’s locale and localization usually match, but not always

Date Formatting

NSDateFormatter

Date Formatting

NSDateFormatter

- Converts between NSDate objects and their string representations

Date Formatting

NSDateFormatter

- Converts between NSDate objects and their string representations
- Often attached explicitly to text fields in a nib

Date Formatting

NSDateFormatter

- Converts between NSDate objects and their string representations
- Often attached explicitly to text fields in a nib
- When you need to do this in code:

```
+[NSDateFormatter localizedStringFromDate:dateStyle:timeStyle:]
```

Date Formatting

Pre-set date styles

Date Formatting

Pre-set date styles

	Description	Date	Time
ShortStyle	Numeric only	6/10/13	11:03 AM
MediumStyle	Abbreviated text	Jun 10, 2013	11:03:15 AM
LongStyle	Full text	June 10, 2013	11:03:15 AM PDT
FullStyle	Complete details	Friday, June 10, 2013	11:03:15 AM Pacific Daylight Time
NoStyle	Output suppressed	-	-

Date Formatting

Pre-set date styles

Date Formatting

Pre-set date styles

```
[NSDateFormatter localizedStringFromDate:[NSDate date]
                               dateStyle:NSDateFormatterMediumStyle
                               timeStyle:NSDateFormatterShortStyle]
```

Date Formatting

Pre-set date styles

```
[NSDateFormatter localizedStringFromDate:[NSDate date]
                               dateStyle:NSDateFormatterMediumStyle
                               timeStyle:NSDateFormatterShortStyle]
```

Locale	Date	Time
English (U.S.)	Jun 6, 2013	10:14 AM
French (France)	6 Jun 2013	10:14
Chinese (China)	2013年6月6日	上午10:14

Date Formatting

Pre-set date styles

```
[NSDateFormatter localizedStringFromDate:[NSDate date]
                               dateStyle:NSDateFormatterMediumStyle
                               timeStyle:NSDateFormatterShortStyle]
```

Locale	Date	Time
English (U.S.)	Jun 6, 2013	10:14 AM
French (France)	6 Jun 2013	10:14
Chinese (China)	2013年6月6日	上午10:14

Date Formatting

Custom date and time styles

- When the default formats don’t meet your needs

Date Formatting

Custom date and time styles

- When the default formats don’t meet your needs
- Create NSDateFormatter instance

```
dateFormatter = [[NSDateFormatter alloc] init];
```

Date Formatting

Custom date and time styles

- When the default formats don’t meet your needs
- Create NSDateFormatter instance

```
dateFormatter = [[NSDateFormatter alloc] init];
```

- Create a format string

```
formatString = [NSDateFormatter
    dateFormatFromTemplate:@"dMMM" options:0
    locale:[NSLocale currentLocale]];
```

Date Formatting

Custom date and time styles

- When the default formats don’t meet your needs
- Create NSDateFormatter instance

```
dateFormatter = [[NSDateFormatter alloc] init];
```

- Create a format string

```
formatString = [NSDateFormatter
    dateFormatFromTemplate:@"dMMM" options:0
    locale:[NSLocale currentLocale]];
```

- Set the date format of the NSDateFormatter instance

```
[dateFormatter setDateFormat:formatString];
```

Pitfalls

Explicit format strings

```
[dateFormatter setDateFormat:@"MMM dd, yyyy"];
[dateFormatter stringFromDate:[NSDate date]];
```

Pitfalls

Explicit format strings

```
[dateFormatter setDateFormat:@"MMM dd, yyyy"];
[dateFormatter stringFromDate:[NSDate date]];
```

Locale	Date with format “MMM dd, yyyy”
English (U.S.)	June 06, 2013
French (France)	June 06, 2013
Chinese (China)	June 06, 2013

Pitfalls

Explicit format strings

```
[dateFormatter setDateFormat:@"MMM dd, yyyy"];
[dateFormatter stringFromDate:[NSDate date]];
```

Locale	Date with format “MMM dd, yyyy”
English (U.S.)	June 06, 2013
French (France)	June 06, 2013
Chinese (China)	June 06, 2013

Date Formatting

Custom date and time styles

Date Formatting

Custom date and time styles

```
formatString = [NSDateFormatter
    dateFormatFromTemplate:@"dMMM" options:0
    locale:[NSLocale currentLocale]];
[dateFormatter setDateFormat:@"dMMM"];
[dateFormatter stringFromDate:[NSDate date]];
```

Date Formatting

Custom date and time styles

```
formatString = [NSDateFormatter
    dateFormatFromTemplate:@"dMMM" options:0
    locale:[NSLocale currentLocale]];
[dateFormatter setDateFormat:@"dMMM"];
[dateFormatter stringFromDate:[NSDate date]];
```

Locale	Date with template “dMMM”
English (U.S.)	Jun 6
French (France)	6 Jun
Chinese (China)	6月6日

Date Formatting

Custom date and time styles

```
formatString = [NSDateFormatter
    dateFormatFromTemplate:@"dMMM" options:0
    locale:[NSLocale currentLocale]];
[dateFormatter setDateFormat:@"dMMM"];
[dateFormatter stringFromDate:[NSDate date]];
```

Locale	Date with template “dMMM”
English (U.S.)	Jun 6
French (France)	6 Jun
Chinese (China)	6月6日

Number Formatting

NSNumberFormatter

Number Formatting

NSNumberFormatter

- Locales differ in how they present numbers

Number Formatting

NSNumberFormatter

- Locales differ in how they present numbers
- Use NSNumberFormatter to display and parse numbers

Number Formatting

NSNumberFormatter

- Locales differ in how they present numbers
- Use NSNumberFormatter to display and parse numbers

Type	US English	Other
Decimal point and separator	1,234.56	1 234,56
Digits (not all use 0-9)	1,234.56	١٢٣٤.٥٦
Currency	$1,234.56	€1.234,56
Percentage	45%	٤٥٪
NaN, ∞, etc.	NaN	EiTa

Pitfalls

Explicit format strings

- Avoid `stringWithFormat:`, `printf`, etc.

```
[NSString stringWithFormat:@"%3@", myNumber];
```

Pitfalls

Explicit format strings

- Avoid `stringWithFormat:`, `printf`, etc.

```
[NSString stringWithFormat:@"%3@", myNumber];
```

Locale	Printf-Style Format String
English (U.S.)	241.23
Italian (Italy)	241.23
Arabic (Egypt)	241.23

Pitfalls

Explicit format strings

- Avoid `stringWithFormat:`, `printf`, etc.

```
[NSString stringWithFormat:@"%3@", myNumber];
```

Locale	Printf-Style Format String
English (U.S.)	241.23
Italian (Italy)	241.23
Arabic (Egypt)	241.23

Number Formatting

Using pre-set styles

Number Formatting

Using pre-set styles

- Legacy

```
+[NSString localizedStringWithFormat:]
```

Number Formatting

Using pre-set styles

- Legacy

```
+[NSString localizedStringWithFormat:]
```

- Going forward

```
+[NSNumberFormatter localizedStringFromNumber:myNumber
                                  numberStyle:NSNumberFormatterDecimalStyle];
```

Number Formatting

Using pre-set styles

- Legacy

```
+[NSString localizedStringWithFormat:]
```

- Going forward

```
+[NSNumberFormatter localizedStringFromNumber:myNumber
                                 numberStyle:NSNumberFormatterDecimalStyle];
```

Number Formatting

NSNumberFormatterStyle

Number Formatting

NSNumberFormatterStyle

@1234.56

DecimalStyle	1,234.56	1.234,56 (it_IT)
CurrencyStyle	$1,234.56	￥1,234.56 (zh_CN)
PercentStyle	123,456%	١٢٣٬٤٥٦٪ (ar_EG)
ScientificStyle	1.23456E+03	1,23456E3 (it_IT)
SpellOutStyle	one thousand two hundred thirty-four point five six	一千二百三十四点五六 (zh_CN)

NSLocale

Obtaining user locale

NSLocale

Obtaining user locale

- Standard APIs take locale into account

NSLocale

Obtaining user locale

- Standard APIs take locale into account
- To give as an argument to formatters

```
+currentLocale
+autoUpdatingCurrentLocale
```

NSLocale

Obtaining user locale

- Standard APIs take locale into account
- To give as an argument to formatters

```
+currentLocale
+autoUpdatingCurrentLocale
```

- Access NSLocale constants with objectForKey

NSLocale

Obtaining useful information

NSLocale

Obtaining useful information

- Does this locale use the metric system?

```
[locale objectForKey:NSLocaleUsesMetricSystem]
```

NSLocale

Obtaining useful information

- Does this locale use the metric system?

```
[locale objectForKey:NSLocaleUsesMetricSystem]
```

- What currency symbol does this locale use?

```
[locale objectForKey:NSLocaleCurrencySymbol]
```

NSLocale

Obtaining useful information

- Does this locale use the metric system?

```
[locale objectForKey:NSLocaleUsesMetricSystem]
```

- What currency symbol does this locale use?

```
[locale objectForKey:NSLocaleCurrencySymbol]
```

- Locale-sensitive quotes:

```
bQuote = [locale objectForKey:NSLocaleQuotationBeginDelimiterKey];
eQuote = [locale objectForKey:NSLocaleQuotationEndDelimiterKey];
```

NSLocale

Locale-sensitive quotes

NSLocale

Locale-sensitive quotes

```
quotedString = [NSString stringWithFormat:@"%@%@%@", bQuote, s, eQuote];
```

NSLocale

Locale-sensitive quotes

```
quotedString = [NSString stringWithFormat:@"%@%@%@", bQuote, s, eQuote];
```

Locale	Quoted String
Chinese (China)	“iPhone”
French (France)	«iPhone»
Japanese (Japan)	「iPhone」

Pitfalls

NSLocale

Pitfalls

NSLocale

- Don’t confuse locale with localization

Pitfalls

NSLocale

- Don’t confuse locale with localization

```
[[NSBundle mainBundle] preferredLocalizations][0];
```

  - Localization that the application is running in

Pitfalls

NSLocale

- Don’t confuse locale with localization

[[NSBundle mainBundle] preferredLocalizations][0];

  - Localization that the application is running in

[NSLocale currentLocale];

  - User-specified locale

Calendars

Regional variation

Calendars

Regional variation

Calendar Unit	Variant
Year	2011, 1432, 2554, 5771
Era	AD, Heisei
Number of months per year	12, 13, variable
Lengths of months	From 5 to 31 days
First day of week	Saturday, Sunday, Monday
When years change	昭和64年1月7日 → 平成1年1月8日

Calendars

NSCalendar

Calendars

NSCalendar

- Use NSCalendar for calendrical calculations

Calendars

NSCalendar

- Use NSCalendar for calendrical calculations
  - Number of days in month, weeks in year, etc.

Calendars

NSCalendar

- Use NSCalendar for calendrical calculations
  - Number of days in month, weeks in year, etc.
  - Components of a date

Calendars

NSCalendar

- Use NSCalendar for calendrical calculations
  - Number of days in month, weeks in year, etc.
  - Components of a date
  - Delta computations

Calendars

NSCalendar

- Use NSCalendar for calendrical calculations
  - Number of days in month, weeks in year, etc.
  - Components of a date
  - Delta computations
- Not to be confused with NSDate

Calendars

NSCalendar

- Use NSCalendar for calendrical calculations
  - Number of days in month, weeks in year, etc.
  - Components of a date
  - Delta computations
- Not to be confused with NSDate
  - NSDate is simply a point in time

Calendars

NSCalendar

- Use NSCalendar for calendrical calculations
  - Number of days in month, weeks in year, etc.
  - Components of a date
  - Delta computations
- Not to be confused with NSDate
  - NSDate is simply a point in time
  - Must be interpreted through the lens of an NSCalendar

Calendars

Components of a date

Calendars

Components of a date

```
NSDateComponents *components = [[NSCalendar currentCalendar]
                                  components:NSDayCalendarUnit   |
                                             NSMonthCalendarUnit |
                                             NSYearCalendarUnit  |
                                             NSEraCalendarUnit
                                  fromDate:[NSDate date]];
```

Calendars

Components of a date

```
NSDateComponents *components = [[NSCalendar currentCalendar]
                                  components:NSDayCalendarUnit   |
                                             NSMonthCalendarUnit |
                                             NSYearCalendarUnit  |
                                             NSEraCalendarUnit
                                  fromDate:[NSDate date]];

NSInteger day   = [components day];
NSInteger month = [components month];
NSInteger year  = [components year];
NSInteger era   = [components era];
```

Pitfalls

Calendrical calculations

Pitfalls

Calendrical calculations

- Common Mistakes

Pitfalls

Calendrical calculations

- Common Mistakes
  - +1 day ≠ +86,400 seconds

Pitfalls

Calendrical calculations

- Common Mistakes
  - +1 day ≠ +86,400 seconds
  - +1 month ≠ +30 days

Pitfalls

Calendrical calculations

- Common Mistakes
  - +1 day ≠ +86,400 seconds
  - +1 month ≠ +30 days
  - +1 year ≠ +525,600 minutes

Pitfalls

Calendrical calculations

- Common Mistakes
  - +1 day ≠ +86,400 seconds
  - +1 month ≠ +30 days
  - +1 year ≠ +525,600 minutes

Demo

International Text

Doug Davidson

International Text

International Text

鉴于对人类家庭所有成员的固有尊严及其平等的和不移的权利的承认

International Text

鉴于对人类家庭所有成员的固有尊严及其平等的和不移的权利的承认

Все люди рождаются свободными и равными

International Text

鉴于对人类家庭所有成员的固有尊严及其平等的和不移的权利的承认

Все люди рождаются свободными и равными

Ὅλοι οι ἄνθρωποι γεννιούνται ελεύθεροι

International Text

כל בני אדם נולדו בני חורין ושווים בערכם ובזכויותיהם

鉴于对人类家庭所有成员的固有尊严及其平等的和不移的权利的承认

Все люди рождаются свободными и равными

Ὅλοι οι ἄνθρωποι γεννιούνται ελεύθεροι

International Text

כל בני אדם נולדו בני חורין ושווים בערכם ובזכויותיהם

鉴于对人类家庭所有成员的固有尊严及其平等的和不移的权利的承认

모든 인간은 태어날 때부터 자유로우며 그 존엄과 권리에 있어 동등하다

Все люди рождаются свободными и равными

Ὅλοι οι ἄνθρωποι γεννιούνται ελεύθεροι

International Text

כל בני אדם נולדו בני חורין ושווים בערכם ובזכויותיהם

鉴于对人类家庭所有成员的固有尊严及其平等的和不移的权利的承认

모든 인간은 태어날 때부터 자유로우며 그 존엄과 권리에 있어 동등하다

Все люди рождаются свободными и равными

Ὅλοι οι ἄνθρωποι γεννιούνται ελεύθεροι

มนุษย์ทั้งหลายเกิดมามีอิสระและเสมอภาคกันในเกียรติศักด[เกียรติศักดิ์]และสิทธิ

International Text

כל בני אדם נולדו בני חורין ושווים בערכם ובזכויותיהם

𐑣𐑟𐑤 𐑛𐑣𐑚�part 𐑣𐑞𐑡𐑤𐑐�part 𐑛𐑝 𐑪𐑣𐑕𐑖𐑩 𐑪�part𐑖𐑮 𐑖𐑜𐑙𐑩 𐑞𐑮𐑑

鉴于对人类家庭所有成员的固有尊严及其平等的和不移的权利的承认

모든 인간은 태어날 때부터 자유로우며 그 존엄과 권리에 있어 동등하다

Все люди рождаются свободными и равными

Ὅλοι οι ἄνθρωποι γεννιούνται ελεύθεροι

มนุษย์ทั้งหลายเกิดมามีอิสระและเสมอภาคกันในเกียรติศักด[เกียรติศักดิ์]และสิทธิ

International Text

כל בני אדם נולדו בני חורין ושווים בערכם ובזכויותיהם

ᏂᎦᏓ ᎠᏂᏴᏫ ᏂᎨᎫᏓᎸᎾ ᎠᎴ ᎤᏂᏠᏱ ᎤᎾᏕᎿ ᏚᏳᎧᏛ ᎨᏒᎢ

鉴于对人类家庭所有成员的固有尊严及其平等的和不移的权利的承认

सभी मनुष्यों को गौरव और अधिकारों

모든 인간은 태어날 때부터 자유로우며 그 존엄과 권리에 있어 동등하다

Все люди рождаются свободными и равными

'Ολοι οι άνθρωποι γεννιούνται ελεύθεροι

มนุษย์ทั้งหลายเกิดมามีอิสระและเสมอภาคกันในเกียรติศักด[เกียรติศักดิ์]และสิทธิ

International Text

כל בני אדם נולדו בני חורין ושווים בערכם ובזכויותיהם

𐑣𐑮𐑤 𐑛𐑣𐑚�part 𐑣𐑦𐑡𐑤𐑨�part 𐑛𐑯 𐑷𐑣𐑯𐑩 𐑷�part𐑕𐑑𐑭 𐑕𐑜𐑙𐑯 𐑦𐑮𐑑

鉴于对人类家庭所有成员的固有尊严及其平等的和不移的权利的承认

सभी मनुष्यों को गौरव और अधिकारों يولد جميع الناس أحرارًا متساوين في الكرامة والحقوق

모든 인간은 태어날 때부터 자유로우며 그 존엄과 권리에 있어 동등하다

Все люди рождаются свободными и равными

Ὅλοι οι ἄνθρωποι γεννιούνται ελεύθεροι

มนุษย์ทั้งหลายเกิดมามีอิสระและเสมอภาคกันในเกียรติศักด[เกียรติศักดิ์]และสิทธิ

International Text

כל בני אדם נולדו בני חורין ושווים בערכם ובזכויותיהם

འཇིག་མིའི་རིགས་རྒྱུད་ཡོངས་ལ་སྐྱེས་ཙམ་ཉིད་ནས་ཆེ་མཐོངས་དང་།

ᏂᎯᏓ ᎠᏂᏴᎾ ᏂᎨᏣᏘᏱ ᎠᎴ ᎤᏂᏰᎸᏯ ᎤᎾᏕᏔᎵ ᏚᏳᎧᏛ ᎨᏒᎢ

鉴于对人类家庭所有成员的固有尊严及其平等的和不移的权利的承认

सभी मनुष्यों को गौरव और अधिकारों

يولد جميع الناس أحرارًا متساوين في الكرامة والحقوق

모든 인간은 태어날 때부터 자유로우며 그 존엄과 권리에 있어 동등하다

Все люди рождаются свободными и равными

Ὅλοι οι ἄνθρωποι γεννιούνται ελεύθεροι

มนุษย์ทั้งหลายเกิดมามีอิสระและเสมอภาคกันในเกียรติศักด[เกียรติศักดิ์]และสิทธิ

International Text

- Use Unicode and NSString for text
- Use appropriate string APIs for iteration, searching, and sorting
- Use standard views and controls for text input and display

Using Unicode

Using Unicode

- Standard for encoding characters from all the world's writing systems

Using Unicode

- Standard for encoding characters from all the world’s writing systems
- NSString holds Unicode string exposed as UTF-16

Using Unicode

- Standard for encoding characters from all the world's writing systems
- NSString holds Unicode string exposed as UTF-16
- What the user sees as a character is variable in length

Using Unicode

- Standard for encoding characters from all the world's writing systems
- NSString holds Unicode string exposed as UTF-16
- What the user sees as a character is variable in length
- Operate on strings and ranges of characters within strings

Composed Character Sequences

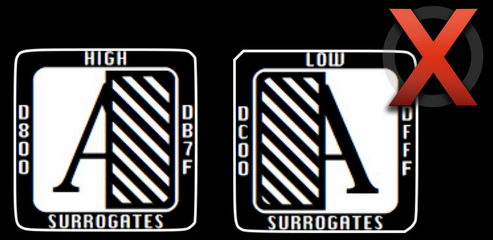
`-[NSString rangeOfComposedCharacterSequenceAtIndex:]`

	UTF-16	UTF-32
揚	D85F DF2E	27F2E
각	1100 1161 11A8	01100 01161 011A8
😄	D83D DE04	1F604
🇫🇷	D83C DDEB D83C DDF7	1F1EB 1F1F7

Composed Character Sequences

`-[NSString rangeOfComposedCharacterSequenceAtIndex:]`

	UTF-16	UTF-32
撟	D85F DF2E	27F2E
각	1100 1161 11A8	01100 01161 011A8
😄	D83D DE04	1F604
🇫🇷	D83C DDEB D83C DDF7	1F1EB 1F1F7



String APIs: Iteration

- Operate on ranges of characters within strings
- Iterate by character cluster, word, sentence, paragraph

```
-[NSString enumerateSubstringsInRange:options:usingBlock:]
```

String APIs: Iteration

- Operate on ranges of characters within strings
- Iterate by character cluster, word, sentence, paragraph

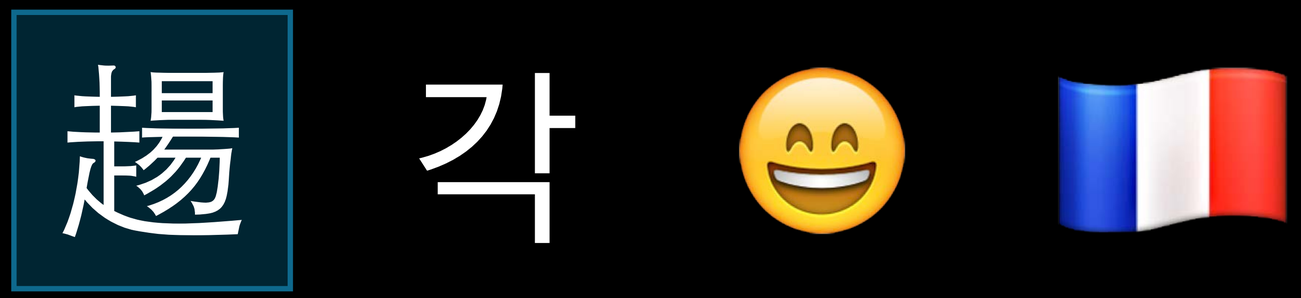
```
-[NSString enumerateSubstringsInRange:options:usingBlock:]
NSStringEnumerationByComposedCharacterSequences
```

趨　각 😄 🇫🇷

String APIs: Iteration

- Operate on ranges of characters within strings
- Iterate by character cluster, word, sentence, paragraph

```
-[NSString enumerateSubstringsInRange:options:usingBlock:]
NSStringEnumerationByComposedCharacterSequences
```



String APIs: Iteration

- Operate on ranges of characters within strings
- Iterate by character cluster, word, sentence, paragraph

```
-[NSString enumerateSubstringsInRange:options:usingBlock:]
NSStringEnumerationByComposedCharacterSequences
```

趨 각  

String APIs: Iteration

- Operate on ranges of characters within strings
- Iterate by character cluster, word, sentence, paragraph

```
-[NSString enumerateSubstringsInRange:options:usingBlock:]
NSStringEnumerationByComposedCharacterSequences
```

趨　각  

String APIs: Iteration

- Operate on ranges of characters within strings
- Iterate by character cluster, word, sentence, paragraph

`-[NSString enumerateSubstringsInRange:options:usingBlock:]`

`NSStringEnumerationByComposedCharacterSequences`

趨　각  

String APIs: Iteration

- Operate on ranges of characters within strings
- Iterate by character cluster, word, sentence, paragraph

```
-[NSString enumerateSubstringsInRange:options:usingBlock:]
NSStringEnumerationByComposedCharacterSequences
NSStringEnumerationByWords
```

Say “正しい日本語です”!

String APIs: Iteration

- Operate on ranges of characters within strings
- Iterate by character cluster, word, sentence, paragraph

```
-[NSString enumerateSubstringsInRange:options:usingBlock:]
NSStringEnumerationByComposedCharacterSequences
NSStringEnumerationByWords
```

Say “正しい日本語です”!

String APIs: Iteration

- Operate on ranges of characters within strings
- Iterate by character cluster, word, sentence, paragraph

```
-[NSString enumerateSubstringsInRange:options:usingBlock:]
NSStringEnumerationByComposedCharacterSequences
NSStringEnumerationByWords
```

Say “正しい日本語です”!

String APIs: Iteration

- Operate on ranges of characters within strings
- Iterate by character cluster, word, sentence, paragraph

```
-[NSString enumerateSubstringsInRange:options:usingBlock:]
NSStringEnumerationByComposedCharacterSequences
NSStringEnumerationByWords
```

Say “正しい日本語です”!

String APIs: Iteration

- Operate on ranges of characters within strings
- Iterate by character cluster, word, sentence, paragraph

```
-[NSString enumerateSubstringsInRange:options:usingBlock:]
NSStringEnumerationByComposedCharacterSequences
NSStringEnumerationByWords
```

Say “正しい日本語です”!

String APIs: Searching

String APIs: Searching

```
[NSString rangeOfString:options:range:locale:]
```

String APIs: Searching

```
[NSString rangeOfString:options:range:locale:]
  NSCaseInsensitiveSearch
```

String APIs: Searching

```
[NSString rangeOfString:options:range:locale:]
  NSCaseInsensitiveSearch
  NSDiacriticInsensitiveSearch
```

String APIs: Searching

```
[NSString rangeOfString:options:range:locale:]
  NSCaseInsensitiveSearch
  NSDiacriticInsensitiveSearch
  NSBackwardsSearch
```

String APIs: Searching

```
[NSString rangeOfString:options:range:locale:]
  NSCaseInsensitiveSearch
  NSDiacriticInsensitiveSearch
  NSBackwardsSearch
  NSAnchoredSearch
```

String APIs: Sorting

String APIs: Sorting

- Different languages and regions have different standard sort orders

String APIs: Sorting

- Different languages and regions have different standard sort orders
- Sometimes diacritics are significant, sometimes not

String APIs: Sorting

- Different languages and regions have different standard sort orders
- Sometimes diacritics are significant, sometimes not
- Sometimes multiple letters need to be considered together

String APIs: Sorting

- Different languages and regions have different standard sort orders
- Sometimes diacritics are significant, sometimes not
- Sometimes multiple letters need to be considered together
- Use localized comparison for user-visible sorting

```
-[NSString compare:]
-[NSString localizedStandardCompare:]
```

String APIs: Sorting

- Different languages and regions have different standard sort orders
- Sometimes diacritics are significant, sometimes not
- Sometimes multiple letters need to be considered together
- Use localized comparison for user-visible sorting

❌ `-[NSString compare:]`

`-[NSString localizedStandardCompare:]`

String APIs: Sorting

- Different languages and regions have different standard sort orders
- Sometimes diacritics are significant, sometimes not
- Sometimes multiple letters need to be considered together
- Use localized comparison for user-visible sorting

❌ `-[NSString compare:]`

✅ `-[NSString localizedStandardCompare:]`

Sorting

`compare:`
Locale-Independent

Aachen
Älmhult
Ångström
Guzman
Günter
Ozzie
Özer
archie
de Moivre
Ørsted
吴用
宋江
林冲
花荣

Sorting

`localizedStandardCompare:`
US English

Aachen
Älmhult
Ångström
archie
de Moivre
Günter
Guzman
Ørsted
Özer
Ozzie
吴用
宋江
林冲
花荣

Sorting

`localizedStandardCompare:`
Danish

archie
de Moivre
Guzman
Günter
Ozzie
Älmhult
Ørsted
Özer
Aachen
Ångström
吴用
宋江
林冲
花荣

Sorting

`localizedStandardCompare:`
Chinese

花荣
林冲
宋江
吴用
Aachen
Älmhult
Ångström
archie
de Moivre
Günter
Guzman
Ørsted
Özer
Ozzie

Text Display

Text Display

- Glyph is the smallest unit of displayable text in a font

Text Display

- Glyph is the smallest unit of displayable text in a font
- Mapping from characters to glyphs is many-to-many

Text Display

- Glyph is the smallest unit of displayable text in a font
- Mapping from characters to glyphs is many-to-many
- Ordering and positioning of glyphs in a line is complex

Text Display

- Glyph is the smallest unit of displayable text in a font
- Mapping from characters to glyphs is many-to-many
- Ordering and positioning of glyphs in a line is complex
- Standard views and controls handle Unicode layout and display

Text Display

- Glyph is the smallest unit of displayable text in a font
- Mapping from characters to glyphs is many-to-many
- Ordering and positioning of glyphs in a line is complex
- Standard views and controls handle Unicode layout and display
- If you must do your own, use Text APIs

Text Display

- Glyph is the smallest unit of displayable text in a font
- Mapping from characters to glyphs is many-to-many
- Ordering and positioning of glyphs in a line is complex
- Standard views and controls handle Unicode layout and display
- If you must do your own, use Text APIs

Introducing Text Kit	Presidio Wednesday 2:00PM
Advanced Text Layouts and Effects with Text Kit	Mission Thursday 2:00PM
Using Fonts with Text Kit	Presidio Friday 9:00AM

Bidirectional Text

Bidirectional Text

Text Input

我们今天qu kan dian ying

1 去看电影	2 去	3 取	4 区	5 曲	
娶	屈	渠	觑	趋	趣
驱	蛆	躯	戌	龋	衢
诎	蕖	苣	璩	劬	黢
氍	癯	麴	蘧	蛐	鸲
阒	祛	眍	蠷	覷	磲

Frequency Radical Stroke Emoji Structure

Text Input


Notes
Done
May 29, 2013, 1:24 PM
我们今天qukandianying
去看电影
去
取
区
曲
Q W E R T Y U I O P
A S D F G H J K L
Z X C V B N M
123
选定
确认

Text Input

Avenue des

Champs-Élyse

Text Input

Text Input

- Input methods don’t always directly insert text letter by letter

Text Input

- Input methods don’t always directly insert text letter by letter
- Complex input methods first insert preliminary (“marked”) text, then convert it to final form which user confirms

Text Input

- Input methods don’t always directly insert text letter by letter
- Complex input methods first insert preliminary (“marked”) text, then convert it to final form which user confirms
- Operate on text as it changes, not keystroke by keystroke

Text Input

- Input methods don’t always directly insert text letter by letter
- Complex input methods first insert preliminary (“marked”) text, then convert it to final form which user confirms
- Operate on text as it changes, not keystroke by keystroke
- Be aware of marked text

Names, Addresses, Phone Numbers

Names, Addresses, Phone Numbers

- Personal names have many different forms

Names, Addresses, Phone Numbers

- Personal names have many different forms
- Different orderings of names

Names, Addresses, Phone Numbers

- Personal names have many different forms
- Different orderings of names
- Phone number and address formats differ

Names, Addresses, Phone Numbers

- Personal names have many different forms
- Different orderings of names
- Phone number and address formats differ
- Use free-form data as much as possible

Names, Addresses, Phone Numbers

- Personal names have many different forms
- Different orderings of names
- Phone number and address formats differ
- Use free-form data as much as possible
- Data detectors can help (NSDataDetector)

More Information

Jake Behrens
App Frameworks Evangelist
behrens@apple.com

Documentation

https://developer.apple.com/library/ios/#documentation/MacOSX/Conceptual/BPInternational/BPInternational.html
https://developer.apple.com/library/ios/#documentation/CoreFoundation/Conceptual/CFLocales/CFLocales.html
https://developer.apple.com/library/ios/#documentation/Cocoa/Conceptual/DataFormatting/DataFormatting.html
https://developer.apple.com/library/ios/#referencelibrary/GettingStarted/RoadMapiOS/chapters/InternationalizeYourApp/InternationalizeYourApp/InternationalizeYourApp.html

Apple Developer Forums

http://devforums.apple.com

Related Sessions

What's New in Cocoa	Mission Tuesday 3:15PM	
Taking Control of Auto Layout in Xcode 5	Presidio Wednesday 10:15AM	
Introducing Text Kit	Presidio Wednesday 2:00PM	
Advanced Text Layouts and Effects with Text Kit	Mission Thursday 2:00PM	
Using Fonts with Text Kit	Presidio Friday 9:00AM	
Solutions to Common Date and Time Challenges	Marina Friday 11:30AM	

Labs

Text Kit and Core Text Lab	Frameworks Lab A Thursday 4:30PM
Internationalization Lab	Frameworks Lab B Friday 11:30AM

Summary

- Localization
  - Make your interfaces localizable
  - Use Base Localization and Auto Layout where possible
  - Use strings files and genstrings for strings in code

Summary

- Locale data
  - Use formatters for user-visible times, dates, and numbers
  - Use templates if necessary to customize them
  - Use NSCalendar for calendrical calculations

Summary

- International text
  - Use Unicode and NSString for text
  - Use appropriate string APIs for iteration, searching, and sorting
  - Use standard views and controls for text input and display

WWDC2013