

# What's New in Kext Development

Sign here please...

Session 707

**Jerry Cottingham**

Software Engineer Core OS IO Team

These are confidential sessions—please refrain from streaming, blogging, or taking pictures

# In This Session

- Kext Development Overview
- Developer ID program
- Your attention please

# Target Audience

# Target Audience

- You are here because you develop a kext

# Target Audience

- You are here because you develop a kext
- You know the top three reasons to avoid writing a kext

# Target Audience

- You are here because you develop a kext
- You know the top three reasons to avoid writing a kext
  - ✗ Kexts require wired memory

# Target Audience

- You are here because you develop a kext
- You know the top three reasons to avoid writing a kext
  - ❌ Kexts require wired memory
  - ❌ Mistakes are often fatal

# Target Audience

- You are here because you develop a kext
- You know the top three reasons to avoid writing a kext
  - ✗ Kexts require wired memory
  - ✗ Mistakes are often fatal
  - ✗ Debugging is harder



# Target Audience

- You are here because you develop a kext
- You know the top three reasons to avoid writing a kext
  - ✗ Kexts require wired memory
  - ✗ Mistakes are often fatal
  - ✗ Debugging is harder
- OK, OK, if you must...

# Target Audience

- You are here because you develop a kext
- You know the top three reasons to avoid writing a kext
  - ❌ Kexts require wired memory
  - ❌ Mistakes are often fatal
  - ❌ Debugging is harder
- OK, OK, if you must...
  - Kernel Extension Programming Topics on [developer.apple.com](https://developer.apple.com/library/tech/notes/KernelExtensionProgrammingTopics/)

# Target Audience

- You are here because you develop a kext
- You know the top three reasons to avoid writing a kext
  - ❌ Kexts require wired memory
  - ❌ Mistakes are often fatal
  - ❌ Debugging is harder
- OK, OK, if you must...
  - Kernel Extension Programming Topics on [developer.apple.com](https://developer.apple.com/library/tech/notes/KernelExtensionProgrammingTopics/)
  - Kernel Programming Guide on [developer.apple.com](https://developer.apple.com/library/tech/notes/KernelProgrammingGuide/)

# Target Audience

- You are here because you develop a kext
- You know the top three reasons to avoid writing a kext
  - ✗ Kexts require wired memory
  - ✗ Mistakes are often fatal
  - ✗ Debugging is harder
- OK, OK, if you must...
  - Kernel Extension Programming Topics on [developer.apple.com](https://developer.apple.com/library/tech/notes/KernelExtensionProgrammingTopics/)
  - Kernel Programming Guide on [developer.apple.com](https://developer.apple.com/library/tech/notes/KernelProgrammingGuide/)

With great power comes great responsibility

# Kext Development Overview

What is a kext?

# Kext Development Overview

## What is a kext?

- A kext is a bundle that extends the kernel

# Kext Development Overview

## What is a kext?

- A kext is a bundle that extends the kernel
  - Kext = Kernel Extension

# Kext Development Overview

## What is a kext?

- A kext is a bundle that extends the kernel
  - Kext = Kernel Extension
  - Only available for OS X, not iOS



# Kext Development Overview

## What is a kext?

- A kext is a bundle that extends the kernel
  - Kext = Kernel Extension
  - Only available for OS X, not iOS
- Structured like any CFBundle

Name	Kind
▼ AppleSamplePCI.kext	Folder
▼ Contents	Folder
Info.plist	Property List
▼ MacOS	Folder
AppleSamplePCI	Unix Executable File

# Kext Development Overview

## What is a kext?

- A kext is a bundle that extends the kernel
  - Kext = Kernel Extension
  - Only available for OS X, not iOS
- Structured like any CFBundle
  - Binary in MacOS folder

Name	Kind
▼ AppleSamplePCI.kext	Folder
▼ Contents	Folder
Info.plist	Property List
▼ MacOS	Folder
AppleSamplePCI	Unix Executable File

# Kext Development Overview

## What is a kext?

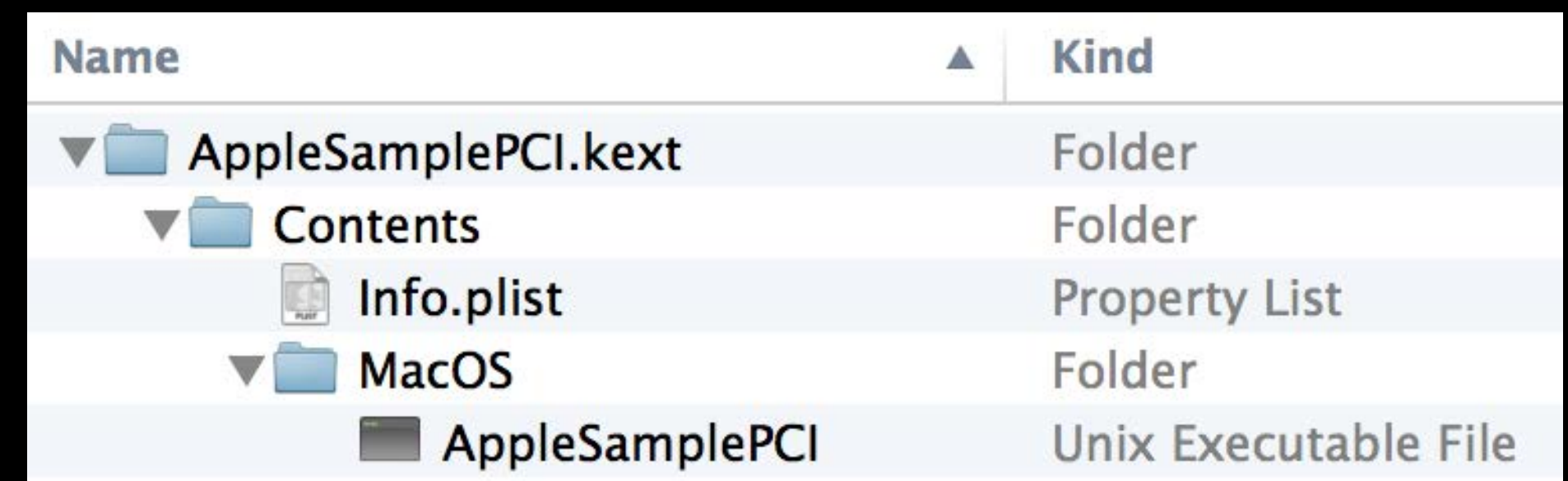
- A kext is a bundle that extends the kernel
  - Kext = Kernel Extension
  - Only available for OS X, not iOS
- Structured like any CFBundle
  - Binary in MacOS folder
  - Info.plist describes properties

Name	Kind
▼ AppleSamplePCI.kext	Folder
▼ Contents	Folder
Info.plist	Property List
▼ MacOS	Folder
AppleSamplePCI	Unix Executable File

# Kext Development Overview

## What is a kext?

- A kext is a bundle that extends the kernel
  - Kext = Kernel Extension
  - Only available for OS X, not iOS
- Structured like any CFBundle
  - Binary in MacOS folder
  - Info.plist describes properties
    - CFBundleIdentifier



The screenshot shows a file browser window displaying the contents of an AppleSamplePCI.kext bundle. The bundle is a folder containing a Contents folder, an Info.plist file, and a MacOS folder. The MacOS folder contains the AppleSamplePCI binary file.

Name	Kind
▼ AppleSamplePCI.kext	Folder
▼ Contents	Folder
Info.plist	Property List
▼ MacOS	Folder
AppleSamplePCI	Unix Executable File

# Kext Development Overview

## What is a kext?

- A kext is a bundle that extends the kernel
  - Kext = Kernel Extension
  - Only available for OS X, not iOS
- Structured like any CFBundle
  - Binary in MacOS folder
  - Info.plist describes properties
    - CFBundleIdentifier
    - CFBundleVersion

Name	Kind
▼ AppleSamplePCI.kext	Folder
▼ Contents	Folder
Info.plist	Property List
▼ MacOS	Folder
AppleSamplePCI	Unix Executable File

Very important to update this!

# Kext Development Overview

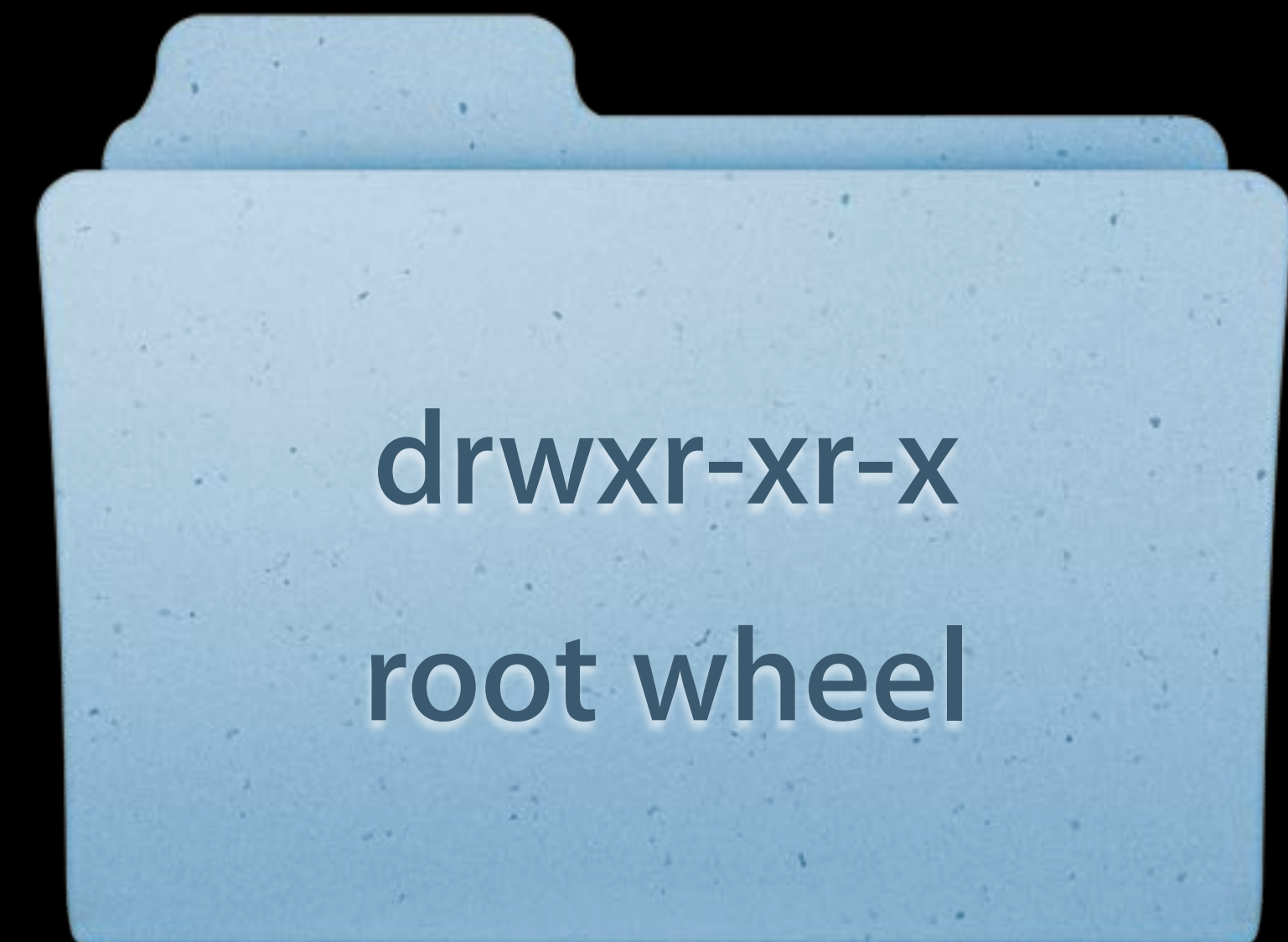
Protecting the kernel



# Kext Development Overview

## Protecting the kernel

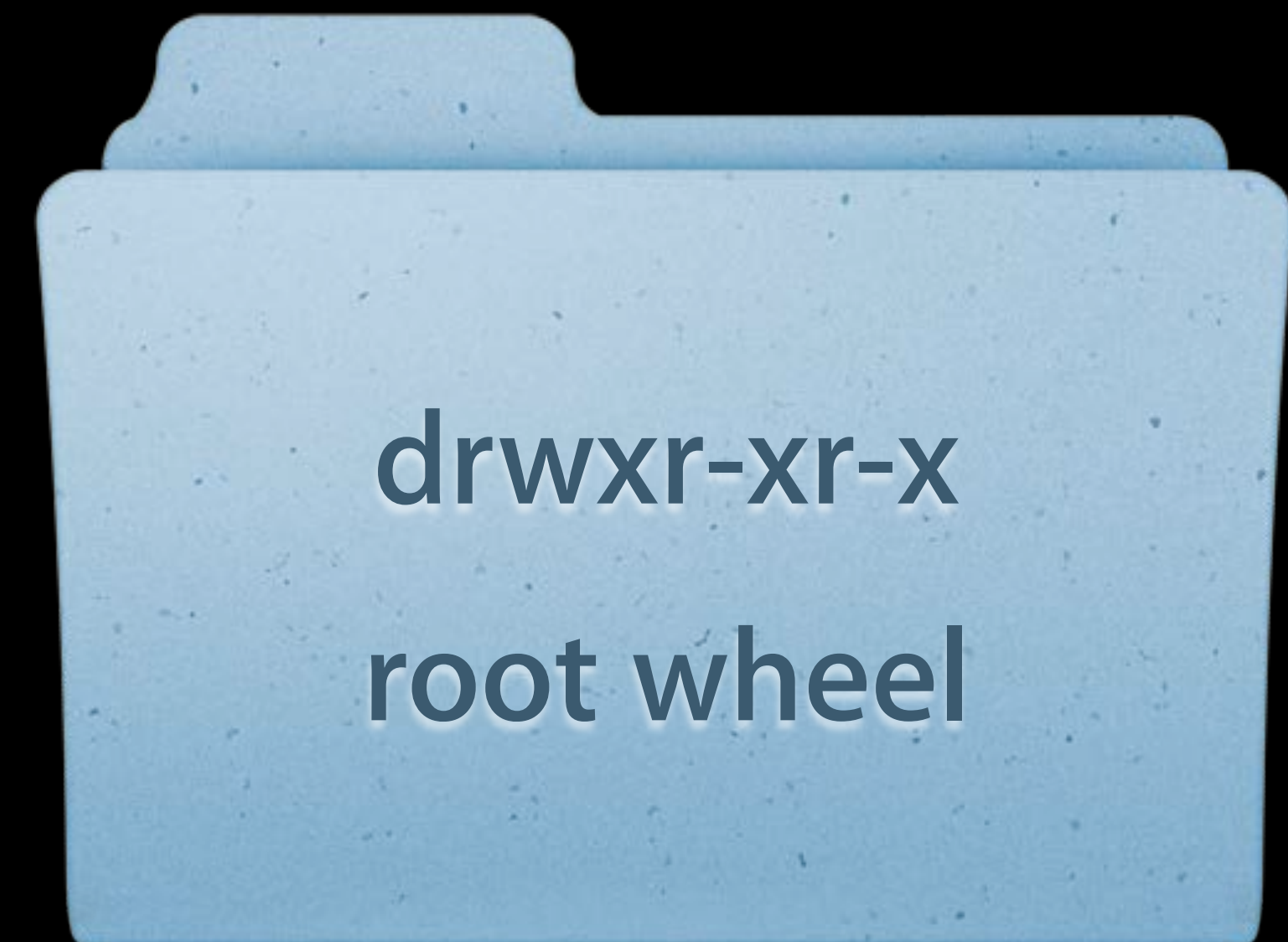
- Specific file/folder permissions



# Kext Development Overview

## Protecting the kernel

- Specific file/folder permissions
  - Kexts **must** be owned by "root," group "wheel"

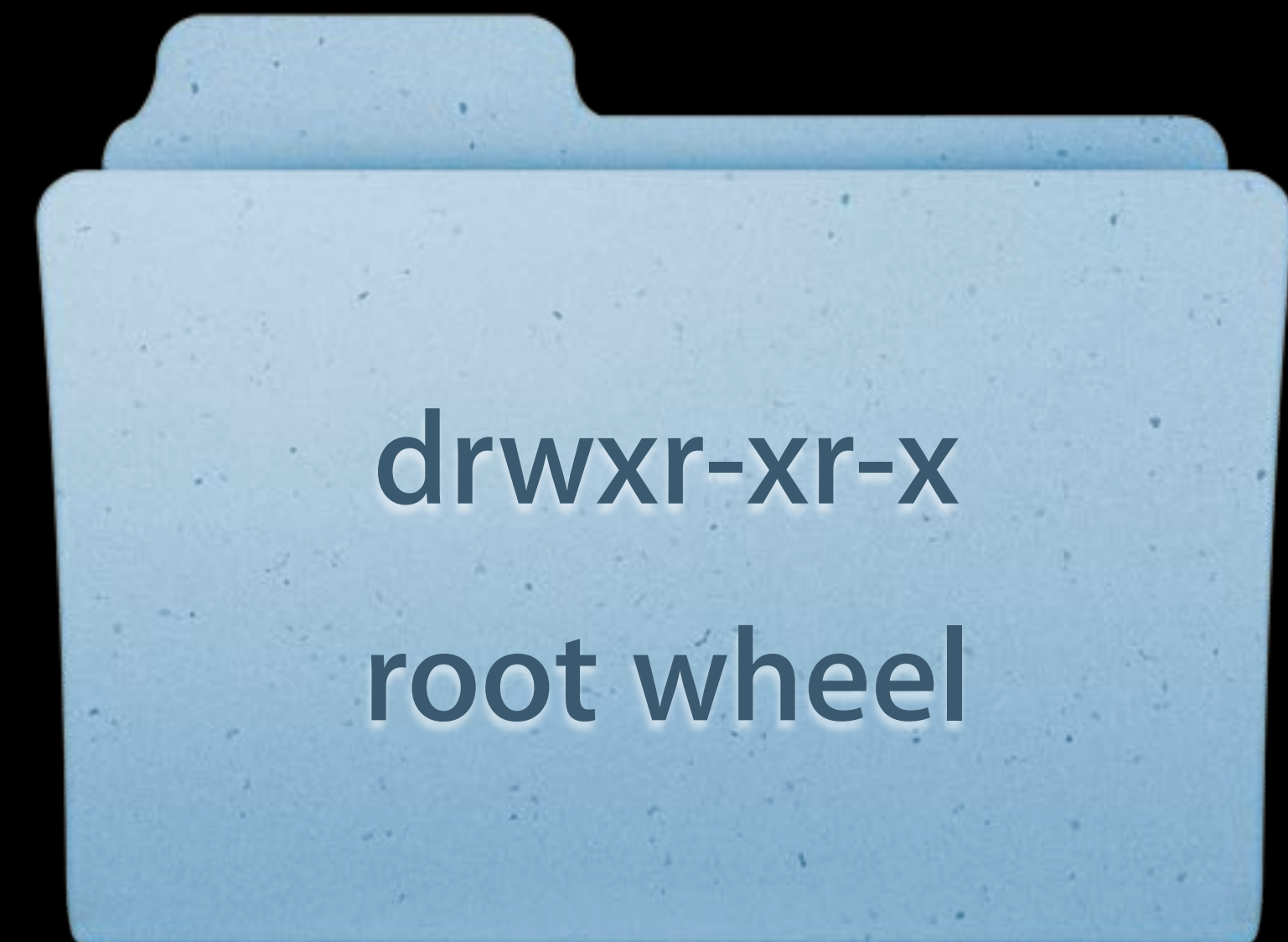




# Kext Development Overview

## Protecting the kernel

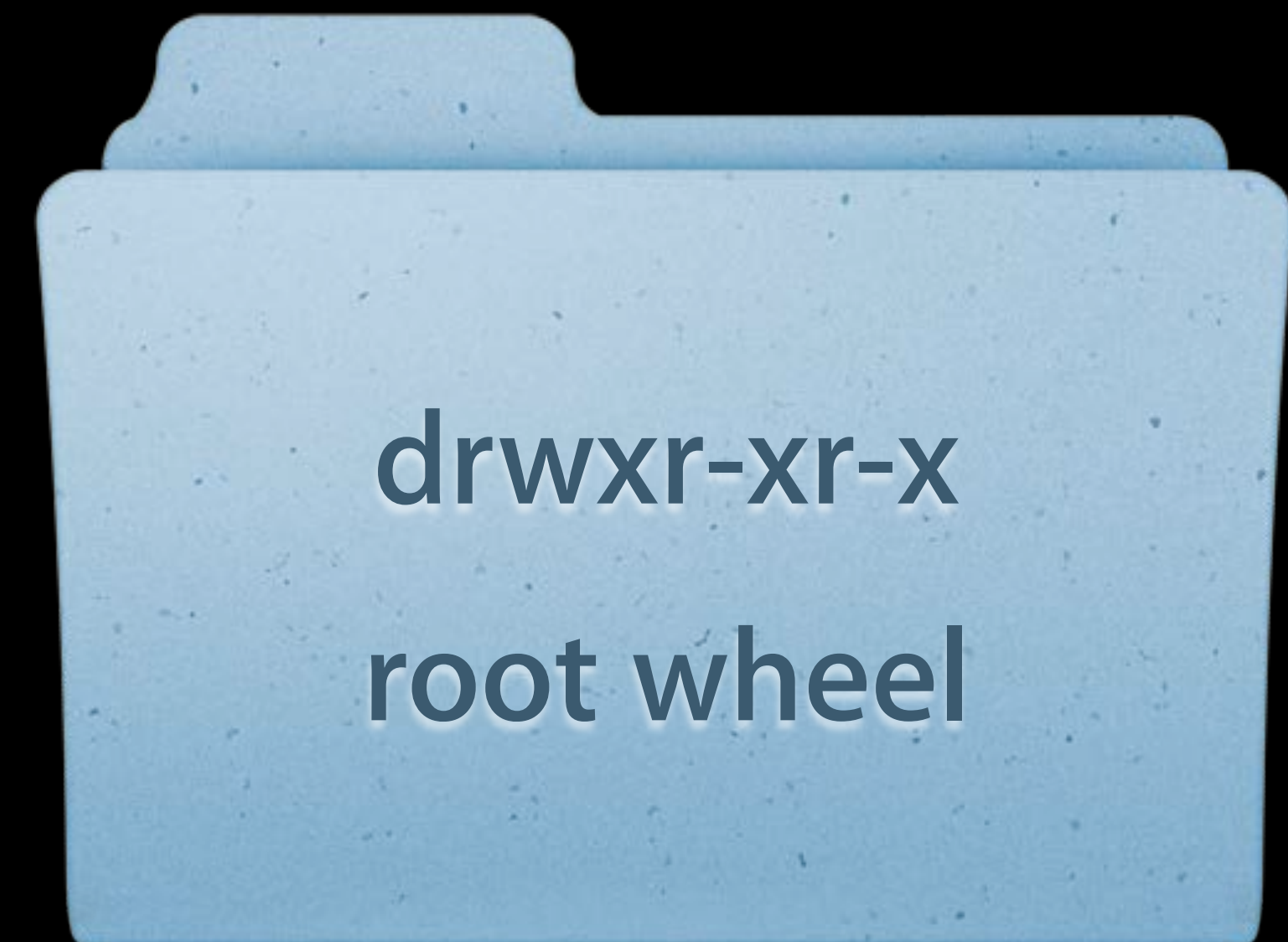
- Specific file/folder permissions
  - Kexts **must** be owned by "root," group "wheel"
  - Must **only** be writable by "root"



# Kext Development Overview

## Protecting the kernel

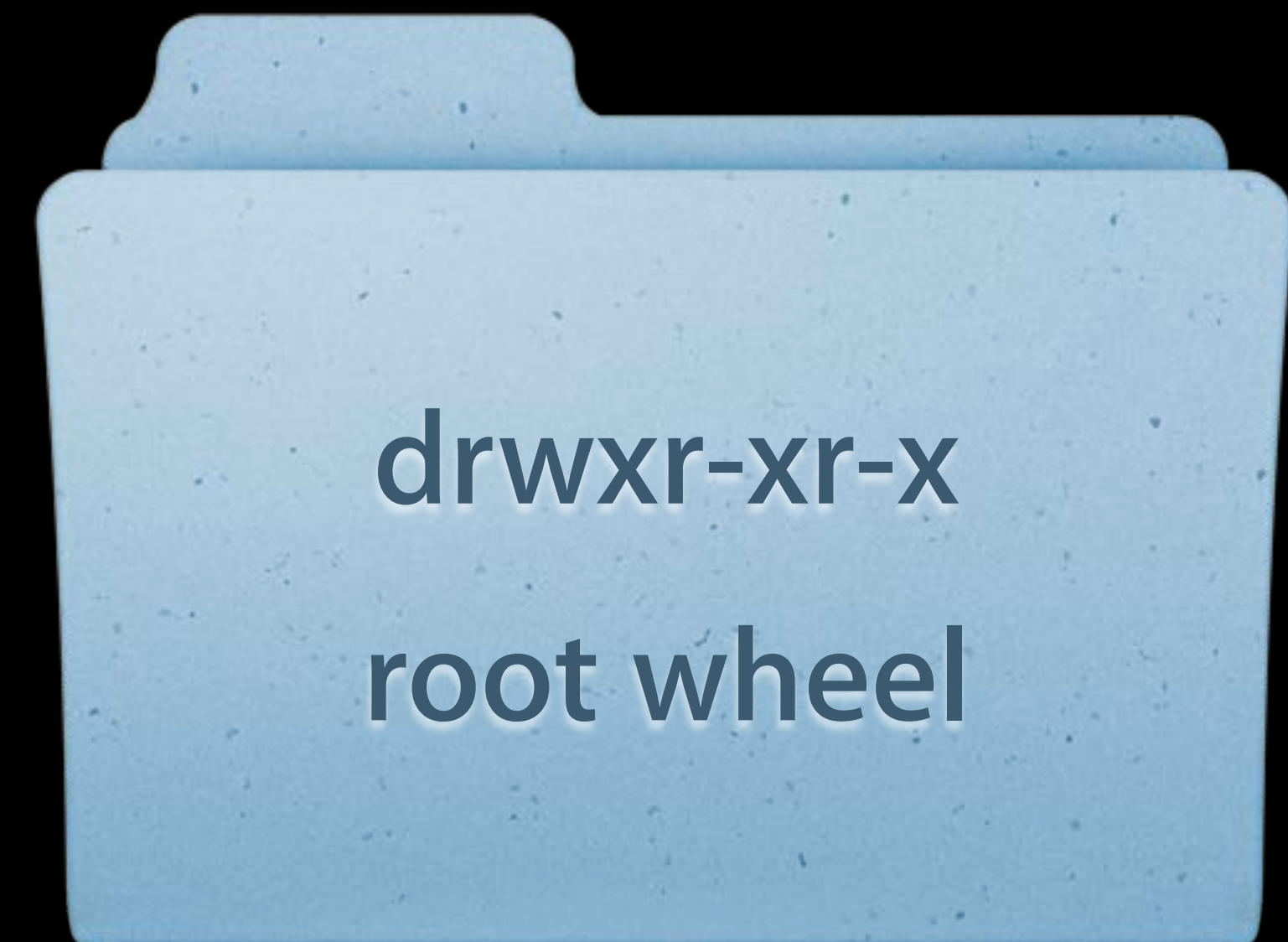
- Specific file/folder permissions
  - Kexts **must** be owned by "root," group "wheel"
  - Must **only** be writable by "root"
  - Permissions are deep (all items in bundle)



# Kext Development Overview

## Protecting the kernel

- Specific file/folder permissions
  - Kexts **must** be owned by "root," group "wheel"
  - Must **only** be writable by "root"
  - Permissions are deep (all items in bundle)
  - Incorrect permissions = no load

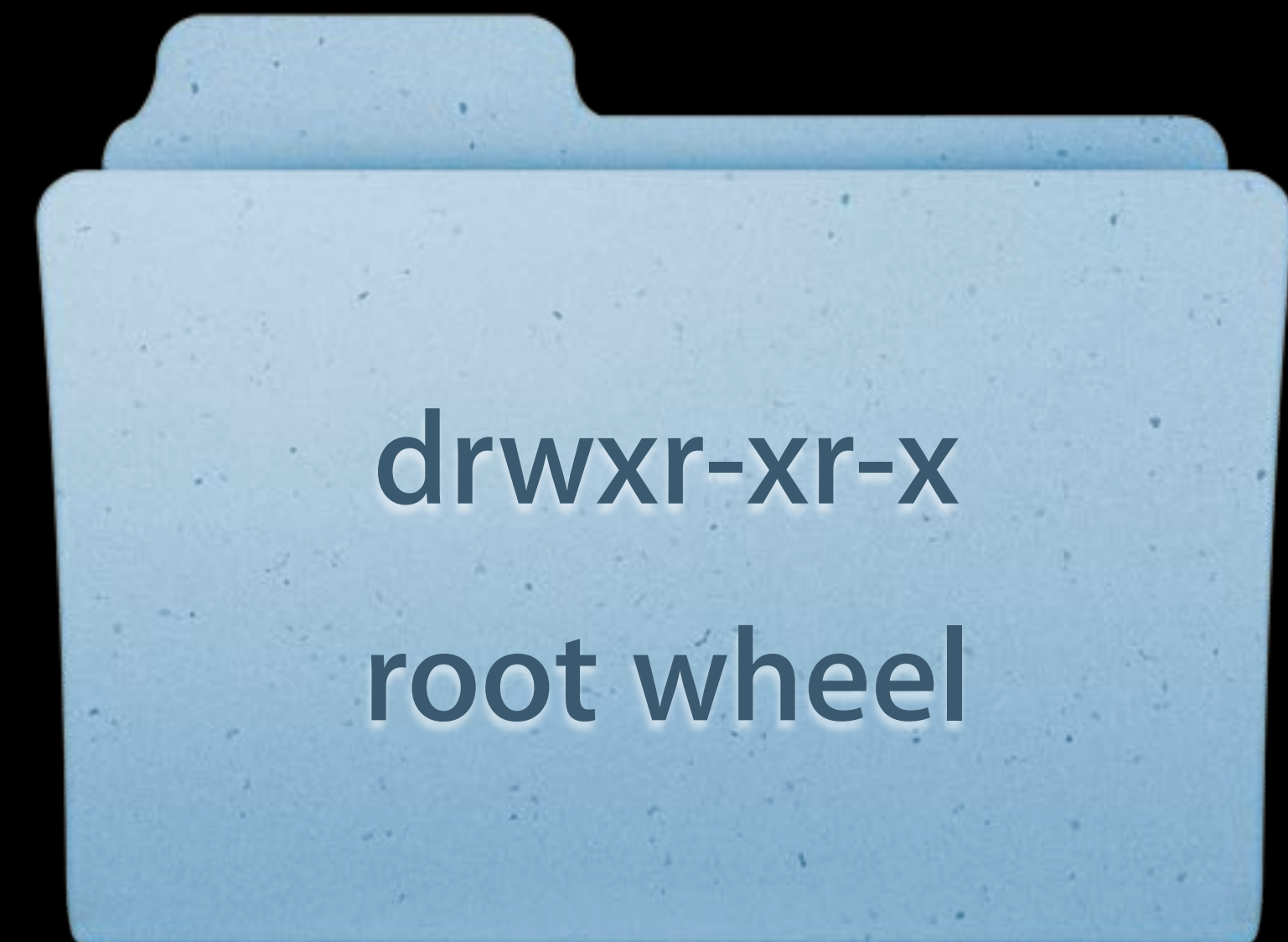




# Kext Development Overview

## Protecting the kernel

- Specific file/folder permissions
  - Kexts **must** be owned by "root," group "wheel"
  - Must **only** be writable by "root"
  - Permissions are deep (all items in bundle)
  - Incorrect permissions = no load
- Use **kextutil -tn** to verify your kext



# Kext Development Overview

Protecting the kernel



# Kext Development Overview

## Protecting the kernel

- OS X 10.9 code signing verification for kexts



# Kext Development Overview

## Protecting the kernel

- OS X 10.9 code signing verification for kexts
  - OS X 10.9 all kext's signatures are verified



# Kext Development Overview

## Protecting the kernel



- OS X 10.9 code signing verification for kexts
  - OS X 10.9 all kext's signatures are verified
  - OS X 10.9 unsigned or invalid signatures are not fatal (with one exception)



# Kext Development Overview

## Protecting the kernel



- OS X 10.9 code signing verification for kexts
  - OS X 10.9 all kext's signatures are verified
  - OS X 10.9 unsigned or invalid signatures are not fatal (with one exception)
  - OS X 10.9 Signed kexts **will not load** on releases prior to OS X 10.8

# Kext Development Overview

## Protecting the kernel



- OS X 10.9 code signing verification for kexts
  - OS X 10.9 all kext's signatures are verified
  - OS X 10.9 unsigned or invalid signatures are not fatal (with one exception)
  - OS X 10.9 Signed kexts **will not load** on releases prior to OS X 10.8
  - Valid code signatures will eventually be mandatory for all kexts

# Kext Development Overview

## Protecting the kernel



- OS X 10.9 code signing verification for kexts
  - OS X 10.9 all kext's signatures are verified
  - OS X 10.9 unsigned or invalid signatures are not fatal (with one exception)
  - OS X 10.9 Signed kexts **will not load** on releases prior to OS X 10.8
  - Valid code signatures will eventually be mandatory for all kexts
- Use **kextutil -tn** to verify your kext

Be prepared!

# Kext Development Overview

Kext loading

# Kext Development Overview

## Kext loading

- Auto-load

# Kext Development Overview

## Kext loading

- Auto-load
  - IOKit Kexts auto-load when matching hardware detected

# Kext Development Overview

## Kext loading

- Auto-load
  - IOKit Kexts auto-load when matching hardware detected
  - Searches in `/System/Library/Extensions`

# Kext Development Overview

## Kext loading

- Auto-load
  - IOKit Kexts auto-load when matching hardware detected
  - Searches in `/System/Library/Extensions`
- On-demand



# Kext Development Overview

## Kext loading

- Auto-load
  - IOKit Kexts auto-load when matching hardware detected
  - Searches in /System/Library/Extensions
- On-demand
  - Load by explicit path  
`kextload ~/AppleSamplePCI.kext`

# Kext Development Overview

## Kext loading

- Auto-load
  - IOKit Kexts auto-load when matching hardware detected
  - Searches in /System/Library/Extensions
- On-demand
  - Load by explicit path  
`kextload ~/AppleSamplePCI.kext`
  - Load by CFBundleIdentifier  
`kextload -b com.example.apple-samplecode.driver.SamplePCI`

# Kext Development Overview

## Kext loading

- Auto-load
  - IOKit Kexts auto-load when matching hardware detected
  - Searches in /System/Library/Extensions
- On-demand
  - Load by explicit path  
`kextload ~/AppleSamplePCI.kext`
  - Load by CFBundleIdentifier  
`kextload -b com.example.apple-samplecode.driver.SamplePCI`
    - Searches in /System/Library/Extensions

# Kext Development Overview

## Kext loading

- Auto-load
  - IOKit Kexts auto-load when matching hardware detected
  - Searches in /System/Library/Extensions
- On-demand
  - Load by explicit path  
`kextload ~/AppleSamplePCI.kext`
  - Load by CFBundleIdentifier  
`kextload -b com.example.apple-samplecode.driver.SamplePCI`
    - Searches in /System/Library/Extensions
- Kernel cache build

# Kext Development Overview

## Kext loading

- Auto-load
  - IOKit Kexts auto-load when matching hardware detected
  - Searches in /System/Library/Extensions
- On-demand
  - Load by explicit path  
`kextload ~/AppleSamplePCI.kext`
  - Load by CFBundleIdentifier  
`kextload -b com.example.apple-samplecode.driver.SamplePCI`
    - Searches in /System/Library/Extensions
- Kernel cache build
  - Searches in /System/Library/Extensions

# Kext Development Overview

Kext loading

# Kext Development Overview

## Kext loading

- OS X 10.9 auto-load, on-demand load by CFBundleIdentifier, and kernel cache build



# Kext Development Overview



## Kext loading

- OS X 10.9 auto-load, on-demand load by CFBundleIdentifier, and kernel cache build
  - Searches in /System/Library/Extensions and /Library/Extensions



# Kext Development Overview



## Kext loading

- OS X 10.9 auto-load, on-demand load by CFBundleIdentifier, and kernel cache build
  - Searches in /System/Library/Extensions and /Library/Extensions
  - Must code sign kexts in /Library/Extensions

# Kext Development Overview

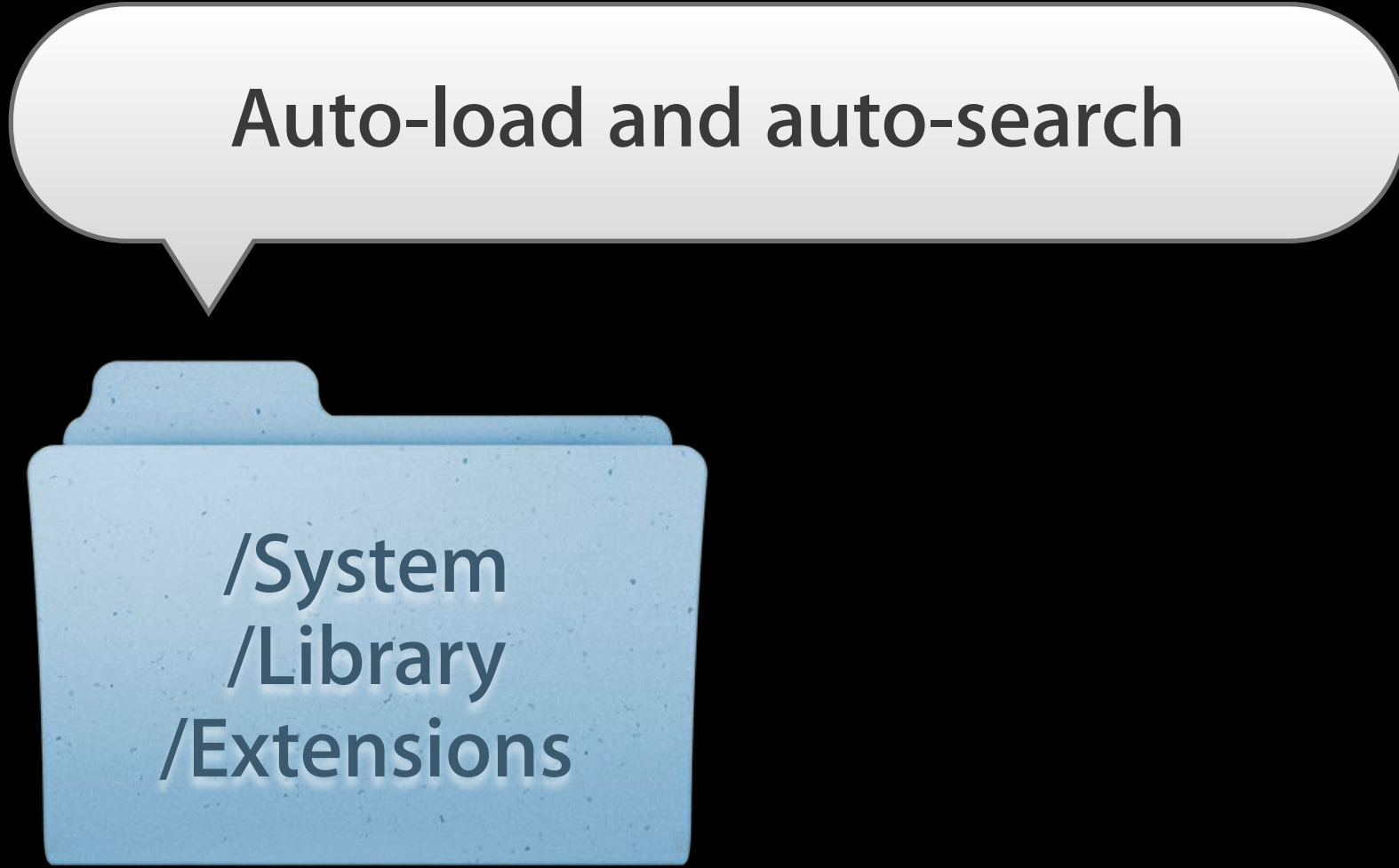
Where are your kexts installed?

# Kext Development Overview

Where are your kexts installed?

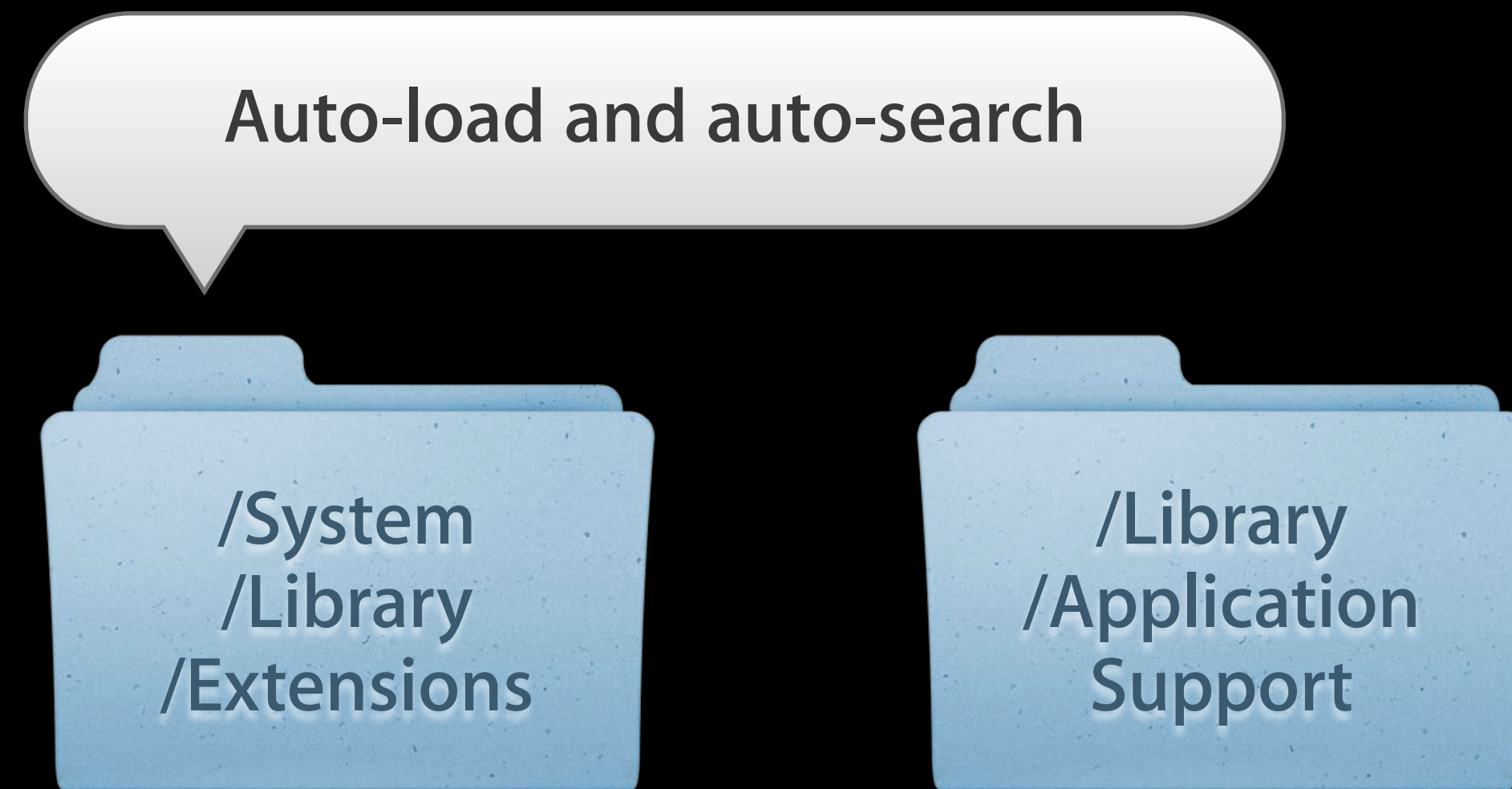
Auto-load and auto-search

/System  
/Library  
/Extensions



# Kext Development Overview

Where are your kexts installed?



# Kext Development Overview

Where are your kexts installed?

Auto-load and auto-search

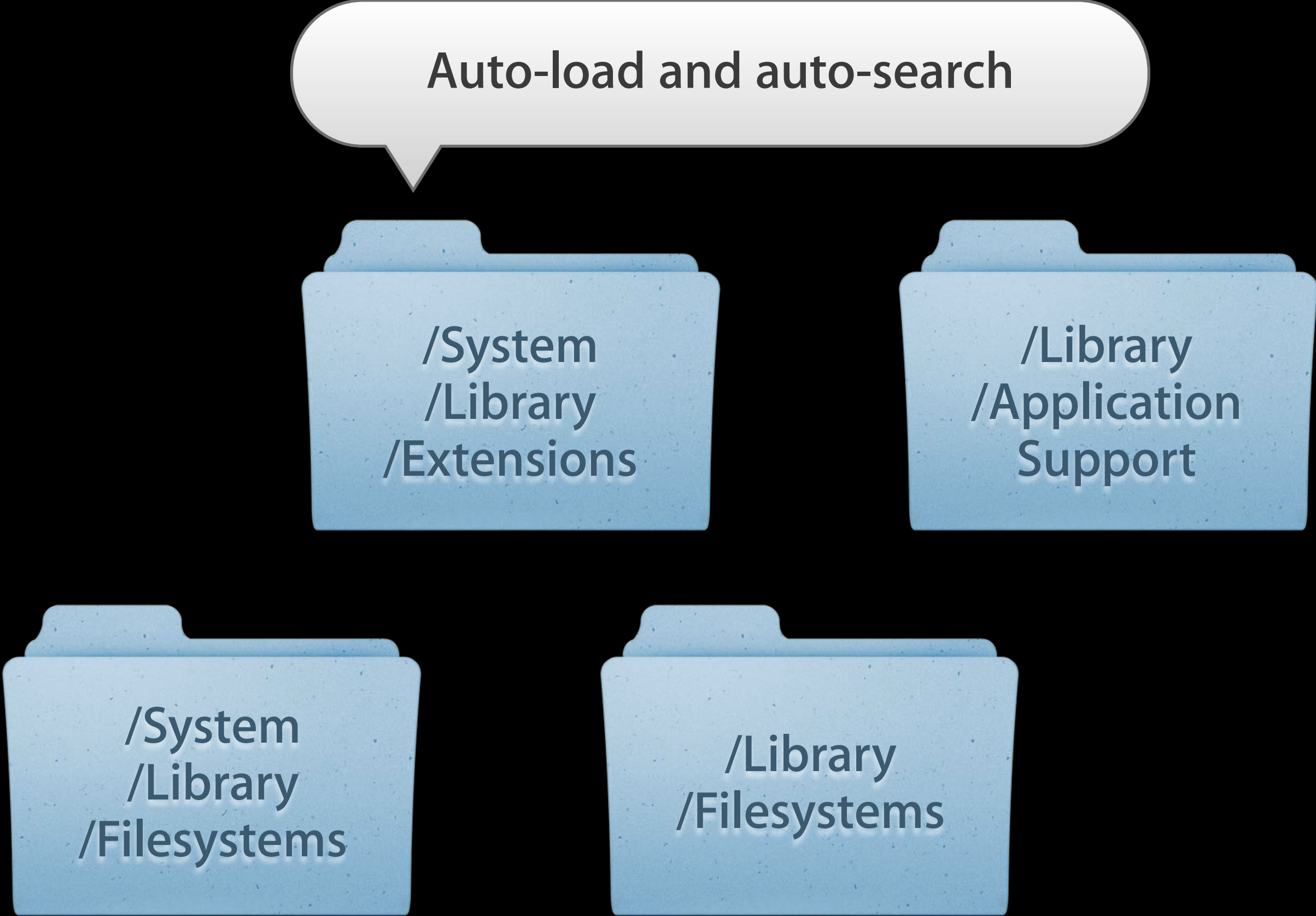
/System  
/Library  
/Extensions

/Library  
/Application  
Support

/System  
/Library  
/Filesystems

# Kext Development Overview

Where are your kexts installed?





# Kext Development Overview

Where are your kexts installed?

Auto-load and auto-search

/System  
/Library  
/Extensions

/Library  
/Application  
Support

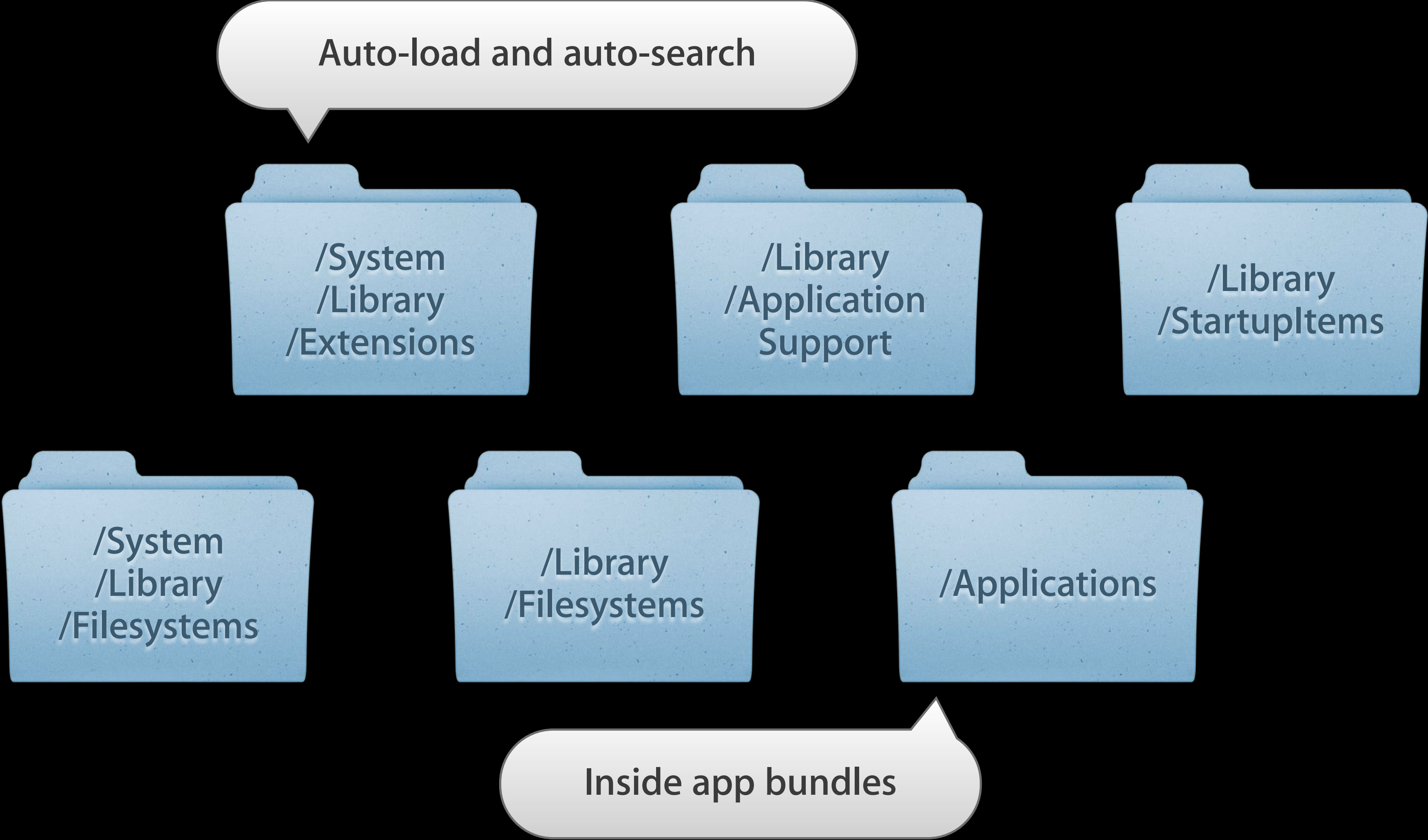
/Library  
/StartupItems

/System  
/Library  
/Filesystems

/Library  
/Filesystems

# Kext Development Overview

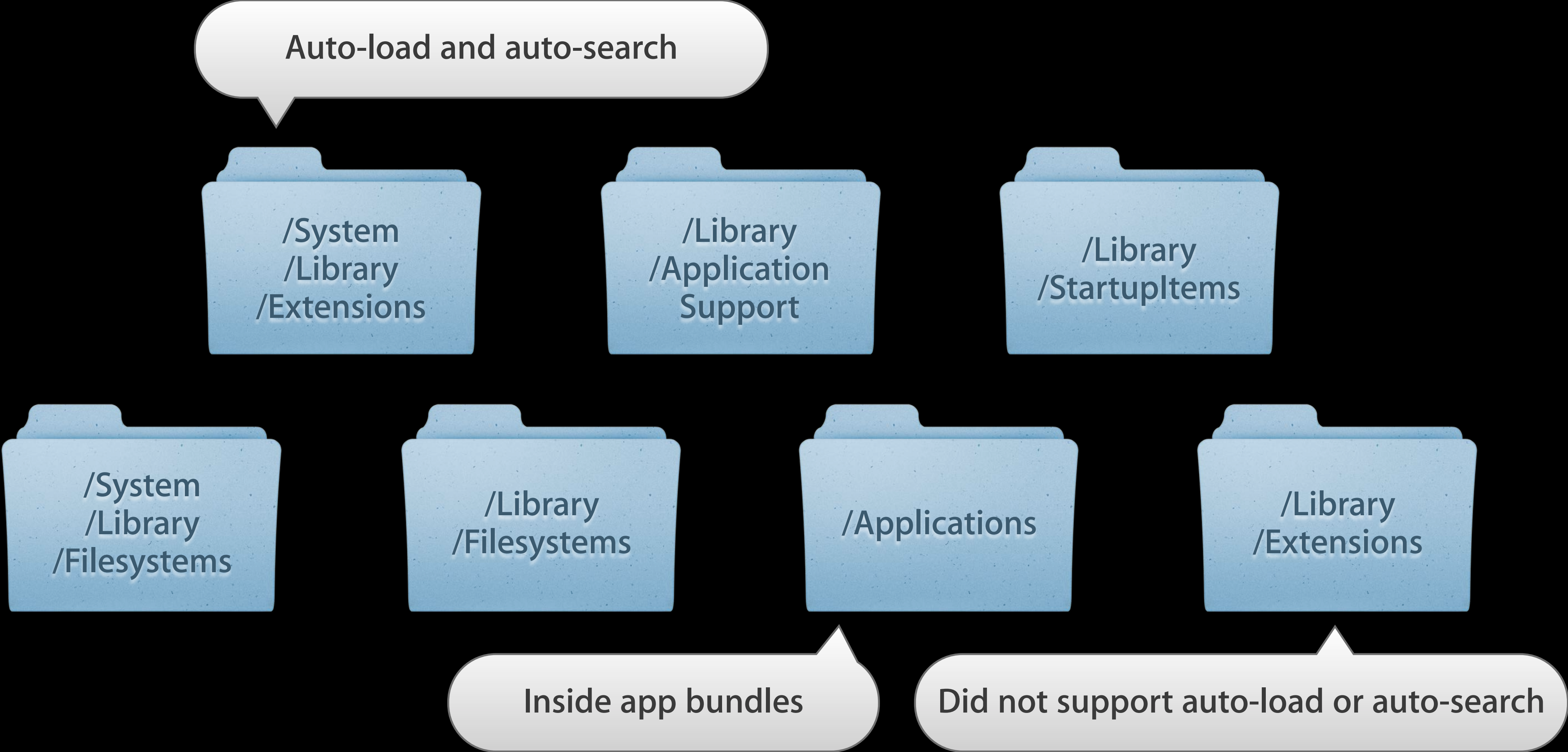
## Where are your kexts installed?





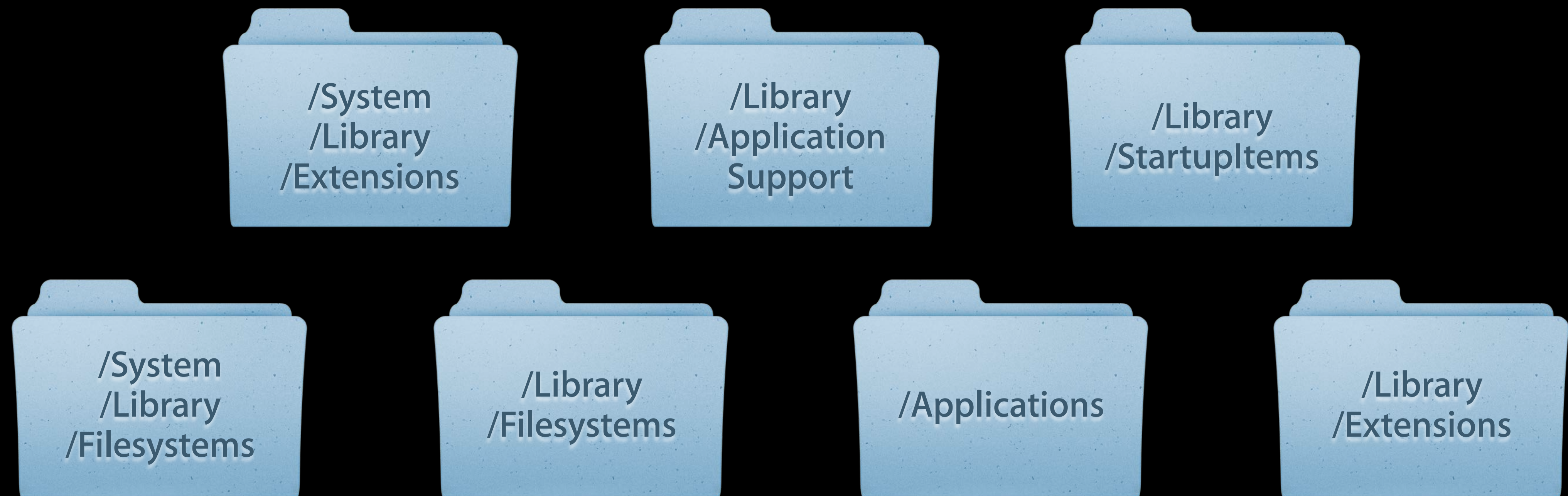
# Kext Development Overview

## Where are your kexts installed?



# Kext Development Overview

OS X 10.9—Where are your kexts installed?





# Kext Development Overview

OS X 10.9—Where are your kexts installed?



Auto-load and auto-search

/System  
/Library  
/Extensions

/Library  
/Application  
Support

/Library  
/StartupItems

/System  
/Library  
/Filesystems

/Library  
/Filesystems

/Applications

/Library  
/Extensions

NEW! Auto-load and auto-search

# Kext Development Overview

OS X 10.9—Where are your kexts installed?



Auto-load and auto-search

/System  
/Library  
/Extensions

/Library  
/Application  
Support

/Library  
/StartupItems

Must be signed!

/System  
/Library  
/Filesystems

/Library  
/Filesystems

/Applications

/Library  
/Extensions

NEW! Auto-load and auto-search

# Kext Development Overview

Where we want your kexts installed

# Kext Development Overview

Where we want your kexts installed

- Auto-load, kexts required for rooting, or auto-searching of kexts



# Kext Development Overview

## Where we want your kexts installed

- Auto-load, kexts required for rooting, or auto-searching of kexts
  - OS X 10.9, /Library/Extensions MUST be signed

# Kext Development Overview

## Where we want your kexts installed

- Auto-load, kexts required for rooting, or auto-searching of kexts
  - OS X 10.9, /Library/Extensions MUST be signed
  - OS X 10.9, /System/Library/Extensions (for compatibility)



# Kext Development Overview

## Where we want your kexts installed

- Auto-load, kexts required for rooting, or auto-searching of kexts
  - OS X 10.9, /Library/Extensions MUST be signed
  - OS X 10.9, /System/Library/Extensions (for compatibility)
  - OS X 10.8 and earlier, install unsigned kext in /System/Library/Extensions

# Kext Development Overview

## Where we want your kexts installed

- Auto-load, kexts required for rooting, or auto-searching of kexts
  - OS X 10.9, /Library/Extensions MUST be signed
  - OS X 10.9, /System/Library/Extensions (for compatibility)
  - OS X 10.8 and earlier, install unsigned kext in /System/Library/Extensions
- All other kexts

# Kext Development Overview

## Where we want your kexts installed

- Auto-load, kexts required for rooting, or auto-searching of kexts
  - OS X 10.9, /Library/Extensions MUST be signed
  - OS X 10.9, /System/Library/Extensions (for compatibility)
  - OS X 10.8 and earlier, install unsigned kext in /System/Library/Extensions
- All other kexts
  - Signed kexts CAN go into /Library/Extensions

# Kext Development Overview

## Where we want your kexts installed

- Auto-load, kexts required for rooting, or auto-searching of kexts
  - OS X 10.9, /Library/Extensions MUST be signed
  - OS X 10.9, /System/Library/Extensions (for compatibility)
  - OS X 10.8 and earlier, install unsigned kext in /System/Library/Extensions
- All other kexts
  - Signed kexts CAN go into /Library/Extensions
  - Do not install anywhere in /System

Future proof your kext

# Kext Development Overview

## Where we want your kexts installed


- Auto-load, kexts required for rooting, or auto-searching of kexts
  - OS X 10.9, /Library/Extensions MUST be signed
  - OS X 10.9, /System/Library/Extensions (for compatibility)
  - OS X 10.8 and earlier, install unsigned kext in /System/Library/Extensions
- All other kexts
  - Signed kexts CAN go into /Library/Extensions
  - Do not install anywhere in /System
  - Other common locations still OK



Future proof your kext

# Kext Development Overview

## Where we want your kexts installed

- Auto-load, kexts required for rooting, or auto-searching of kexts
  - OS X 10.9, /Library/Extensions MUST be signed
  - OS X 10.9, /System/Library/Extensions (for compatibility)
  - OS X 10.8 and earlier, install unsigned kext in /System/Library/Extensions
- All other kexts
  - Signed kexts CAN go into /Library/Extensions
  - Do not install anywhere in /System 
  - Other common locations still OK
- May need to install two kexts for compatibility with multiple releases

# Developer ID Program

Sign here please...

# Developer ID Program

Sign here please...

- Part of Mac Development Program



# Developer ID Program

Sign here please...

- Part of Mac Development Program
- Allows distribution of applications outside of Mac App Store

# Developer ID Program

Sign here please...

- Part of Mac Development Program
- Allows distribution of applications outside of Mac App Store
- Special Developer ID certificate for signing applications

# Developer ID Program

Sign here please...



- Part of Mac Development Program
- Allows distribution of applications outside of Mac App Store
- Special Developer ID certificate for signing applications
- New with OS X 10.9, certificate for signing applications and kexts

# Developer ID Program

Sign here please...



# Developer ID Program

Sign here please...

- Getting your Developer ID certificate



# Developer ID Program

Sign here please...



- Getting your Developer ID certificate

1. Go to <https://developer.apple.com/resources/developer-id/>

# Developer ID Program

Sign here please...



- Getting your Developer ID certificate
  1. Go to <https://developer.apple.com/resources/developer-id/>
  2. Click on link to request Developer ID certificate for signing kexts



# Developer ID Program

Sign here please...



- Getting your Developer ID certificate
  1. Go to <https://developer.apple.com/resources/developer-id/>
  2. Click on link to request Developer ID certificate for signing kexts
  3. Fill out form

# Developer ID Program

Sign here please...



- Getting your Developer ID certificate
  1. Go to <https://developer.apple.com/resources/developer-id/>
  2. Click on link to request Developer ID certificate for signing kexts
  3. Fill out form
  4. Once form is approved, go to request a Developer ID certificate in the "Certificates, Identifier, and Profiles" area of Member Center

# Developer ID Program

Sign here please...



Select the certificates you want to generate.



**Important:** Xcode is the preferred method for requesting and automatically installing these certificates on a Mac. [Learn how](#)

**Developer ID Application and Kernel Extension**

This certificate is used to code sign your app or kernel extension for distribution outside of the Mac App Store.

**Developer ID Installer**

This certificate is used to sign your app's Installer Package for distribution outside of the Mac App Store.

# Developer ID Program

Sign here please...

# Developer ID Program

Sign here please...

- Part of Mac Development Program
- Allows distribution of applications outside of Mac App Store
- Special Developer ID certificate for signing applications
- New with OS X 10.9, certificate for signing applications, and kexts
- For more info
  - <https://developer.apple.com/resources/developer-id/>
  - See session 702 from WWDC 2012 (Gatekeeper and Developer ID)

# Your Attention Please

In OS X 10.9

# Your Attention Please

In OS X 10.9

- Kexts in /Library/Extensions **will not** load if they are unsigned or have a signature verification error



# Your Attention Please

## In OS X 10.9

- Kexts in /Library/Extensions **will not** load if they are unsigned or have a signature verification error
- User **will** see the “no load” alert dialog



Invalid signature or unsigned kexts!

This means YOU!

# Your Attention Please

In OS X 10.9

# Your Attention Please

In OS X 10.9

- Kexts outside /Library/Extensions **will** load if they are unsigned or have a signature verification error

# Your Attention Please

## In OS X 10.9

- Kexts outside /Library/Extensions **will** load if they are unsigned or have a signature verification error
- But user **will** see the “unidentified developer” alert dialog



This means YOU!

# Your Attention Please

In OS X 10.9

# Your Attention Please

In OS X 10.9

- Code signature verification warnings and error messages

# Your Attention Please

In OS X 10.9

- Code signature verification warnings and error messages
- Most common code signature verification error codes



# Your Attention Please

In OS X 10.9

- Code signature verification warnings and error messages
- Most common code signature verification error codes
  - -67030 : something in kext bundle was modified

# Your Attention Please

## In OS X 10.9

- Code signature verification warnings and error messages
- Most common code signature verification error codes
  - -67030 : something in kext bundle was modified
  - -67062 : kext is not code signed

# Your Attention Please

## In OS X 10.9

- Code signature verification warnings and error messages
- Most common code signature verification error codes
  - -67030 : something in kext bundle was modified
  - -67062 : kext is not code signed
  - Code signing error codes are in Security.framework:
    - `#include <Security/CSCommon.h>`

# Your Attention Please

## In OS X 10.9

- Code signature verification warnings and error messages
- Most common code signature verification error codes
  - -67030 : something in kext bundle was modified
  - -67062 : kext is not code signed
  - Code signing error codes are in Security.framework:
    - `#include <Security/CSCommon.h>`

After modified Info.plist

```
com.apple.kextd[40]: WARNING - Invalid signature -67030 for kext URL =  
"file:///Users/local/AppleSamplePCI.kext/", ID = "com.example.apple-  
samplecode.driver.SamplePCI"
```

# Summary

# Summary

- Sign your kext with Developer ID certificate

# Summary

- Sign your kext with Developer ID certificate
- Get out of `/System/Library/Extensions`



# Summary

- Sign your kext with Developer ID certificate
- Get out of /System/Library/Extensions
  - In fact ALL of /System

Talk to us if you think you need to be here!

# Summary

- Sign your kext with Developer ID certificate
- Get out of /System/Library/Extensions
  - In fact ALL of /System
- Get into /Library/Extensions

Talk to us if you think you need to be here!

# What's New in Kext Development

A sign of the times

Session 707

**Dean Reece**

Core OS IO Team Manager

These are confidential sessions—please refrain from streaming, blogging, or taking pictures

# Building a Signed Kext for OS X 10.9

## The tools

- Use OS X 10.8.3 or later
  - Required for correct signing

# Building a Signed Kext for OS X 10.9

## The tools

- Use OS X 10.8.3 or later
  - Required for correct signing
- Use Xcode 4.6 or later
  - Xcode 5.0 preferred

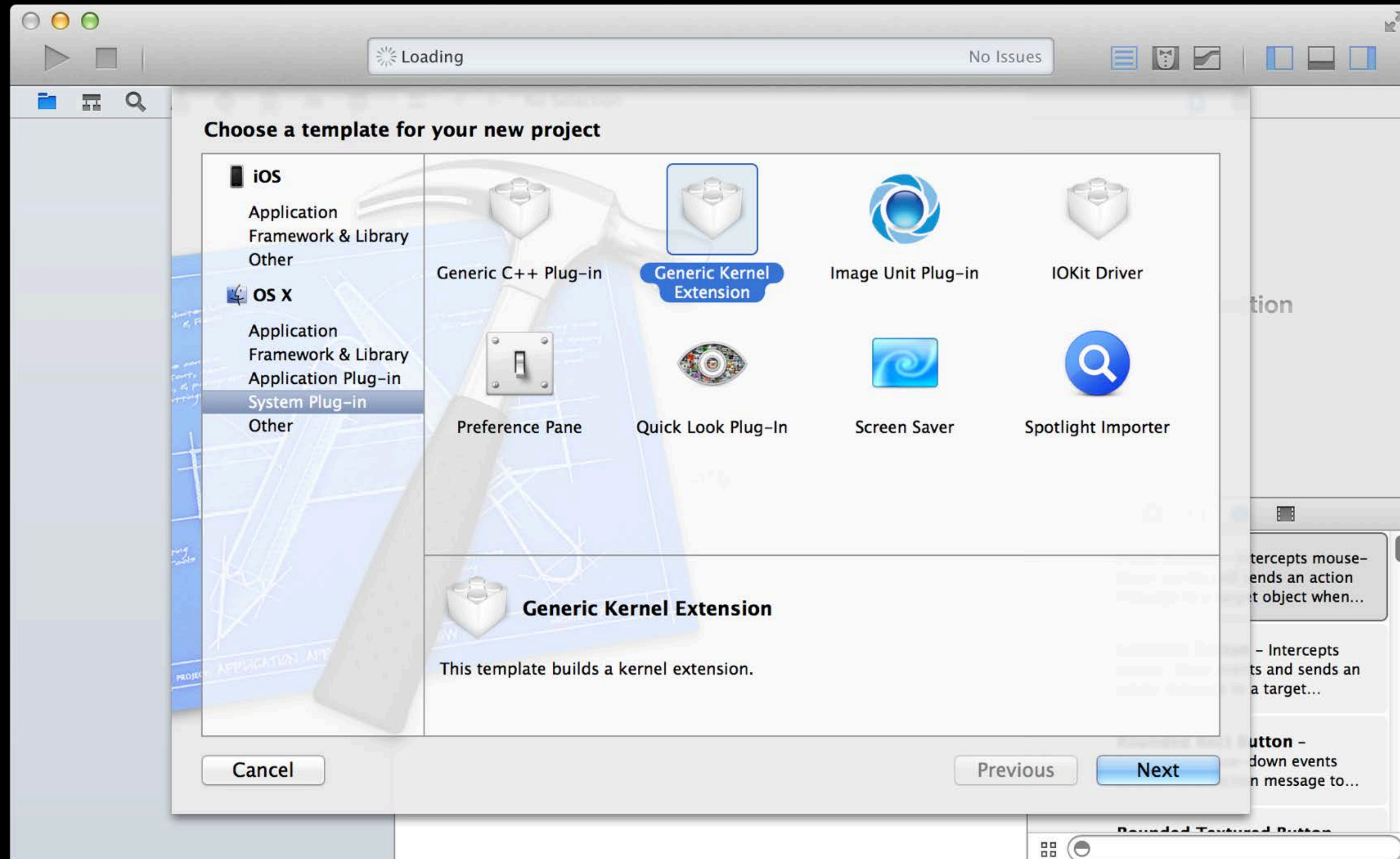
# Building a Signed Kext for OS X 10.9

## The tools

- Use OS X 10.8.3 or later
  - Required for correct signing
- Use Xcode 4.6 or later
  - Xcode 5.0 preferred
- Requires Developer ID certificate for Applications and Kernel Extensions
  - Acquire from Apple Developer website
  - Install in your keychain

# Building a Signed Kext for OS X 10.9

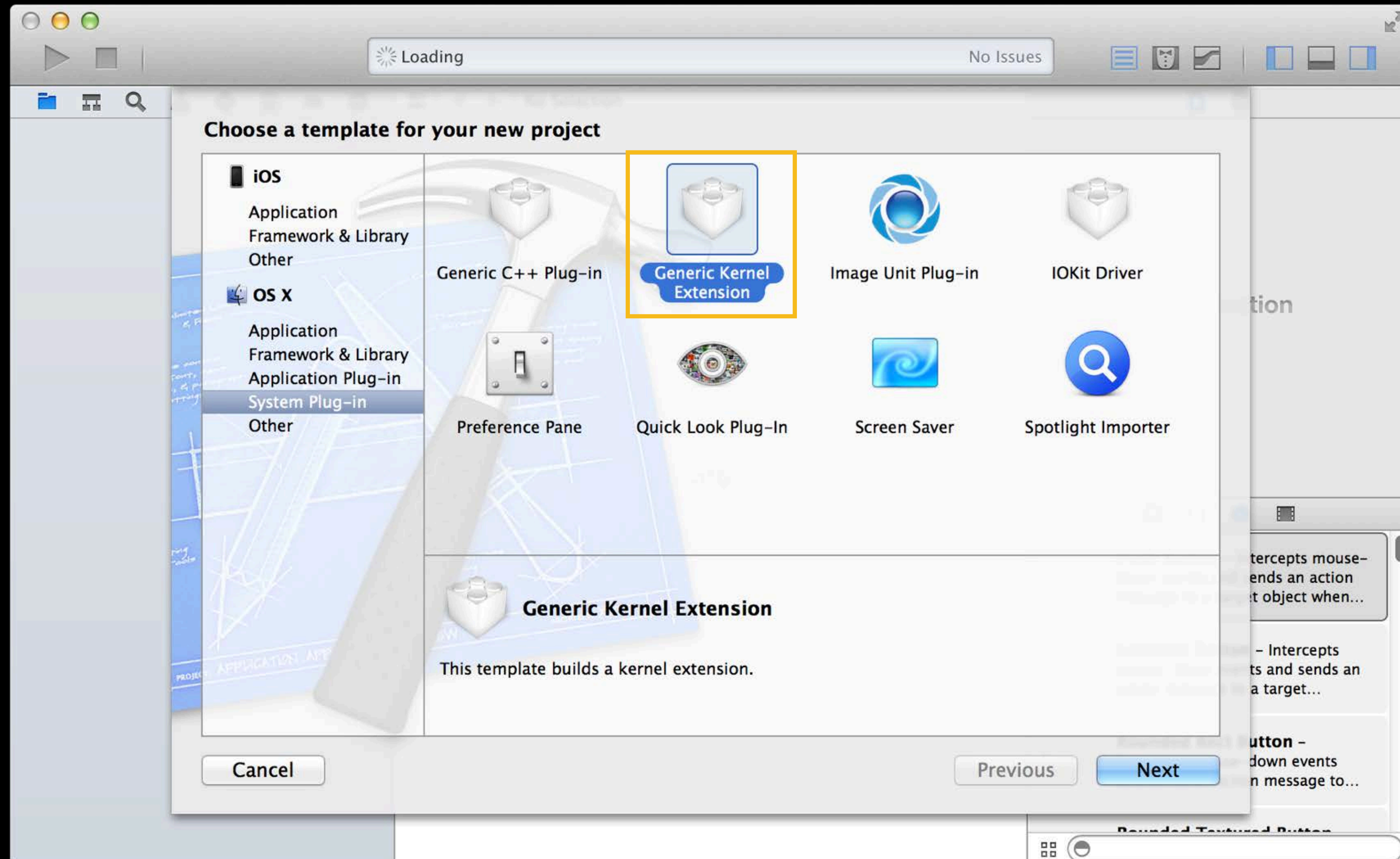
## New Xcode project





# Building a Signed Kext for OS X 10.9

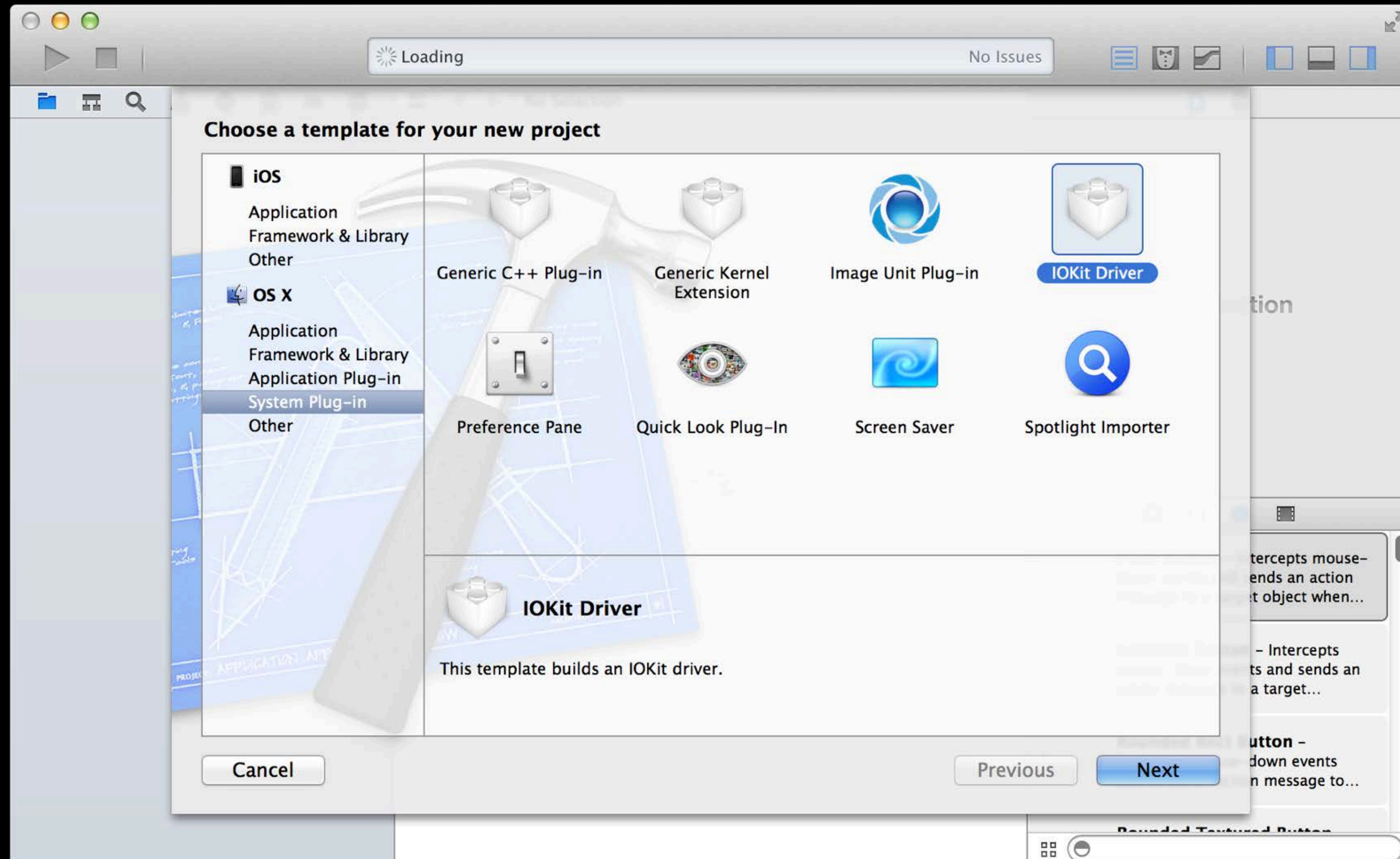
## New Xcode project





# Building a Signed Kext for OS X 10.9

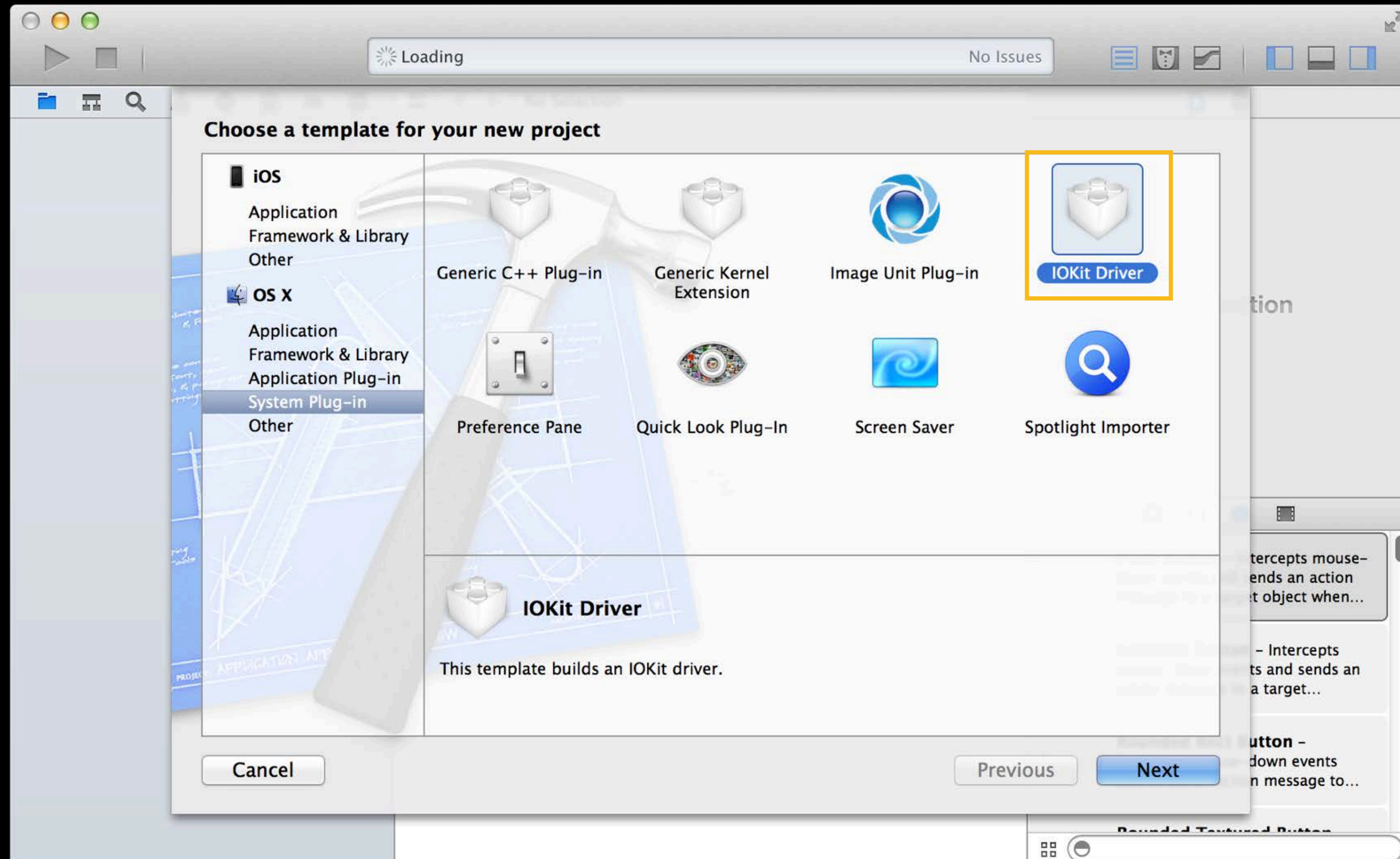
## New Xcode project





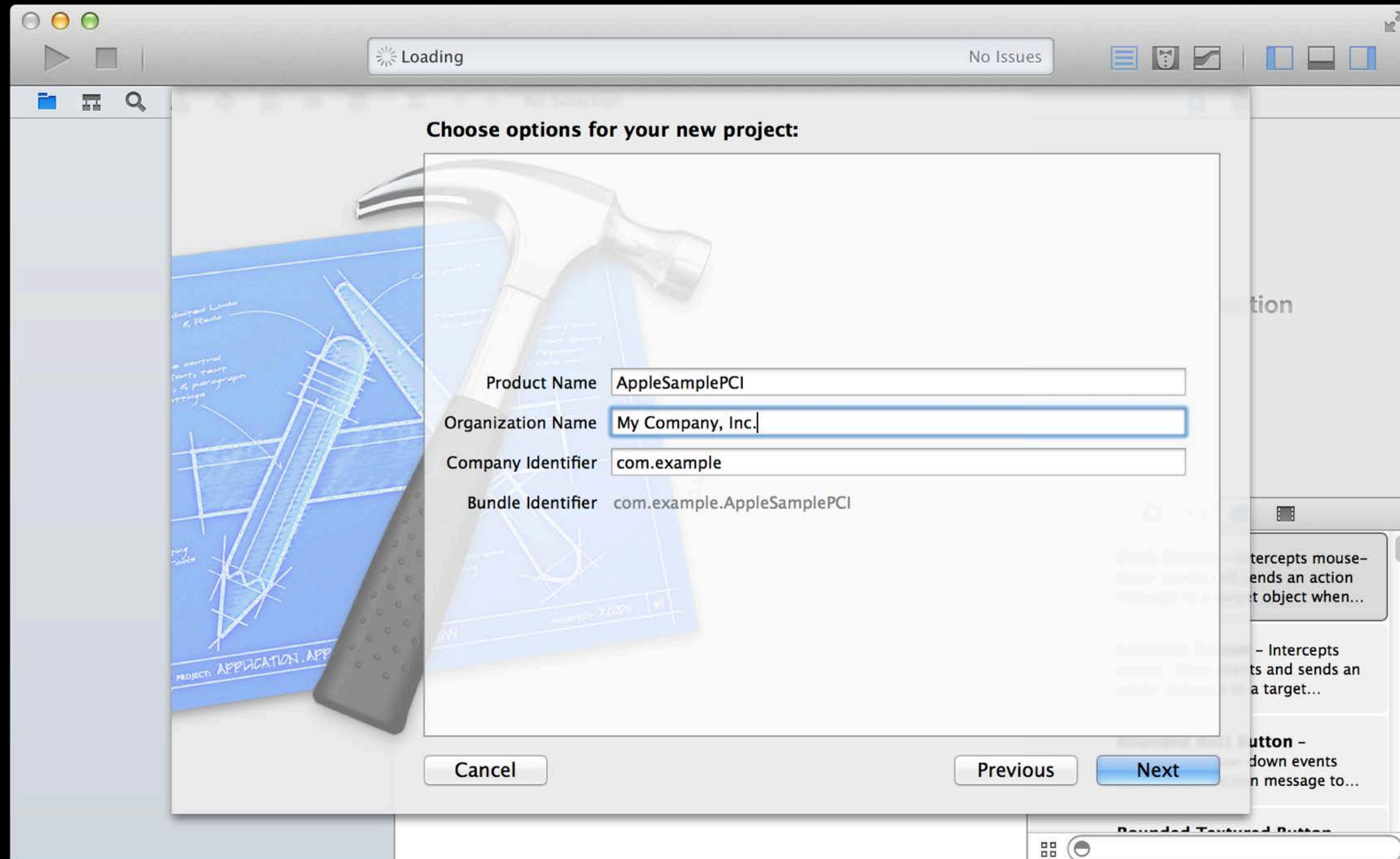
# Building a Signed Kext for OS X 10.9

## New Xcode project



# Building a Signed Kext for OS X 10.9

## New Xcode project





# Building a Signed Kext for OS X 10.9

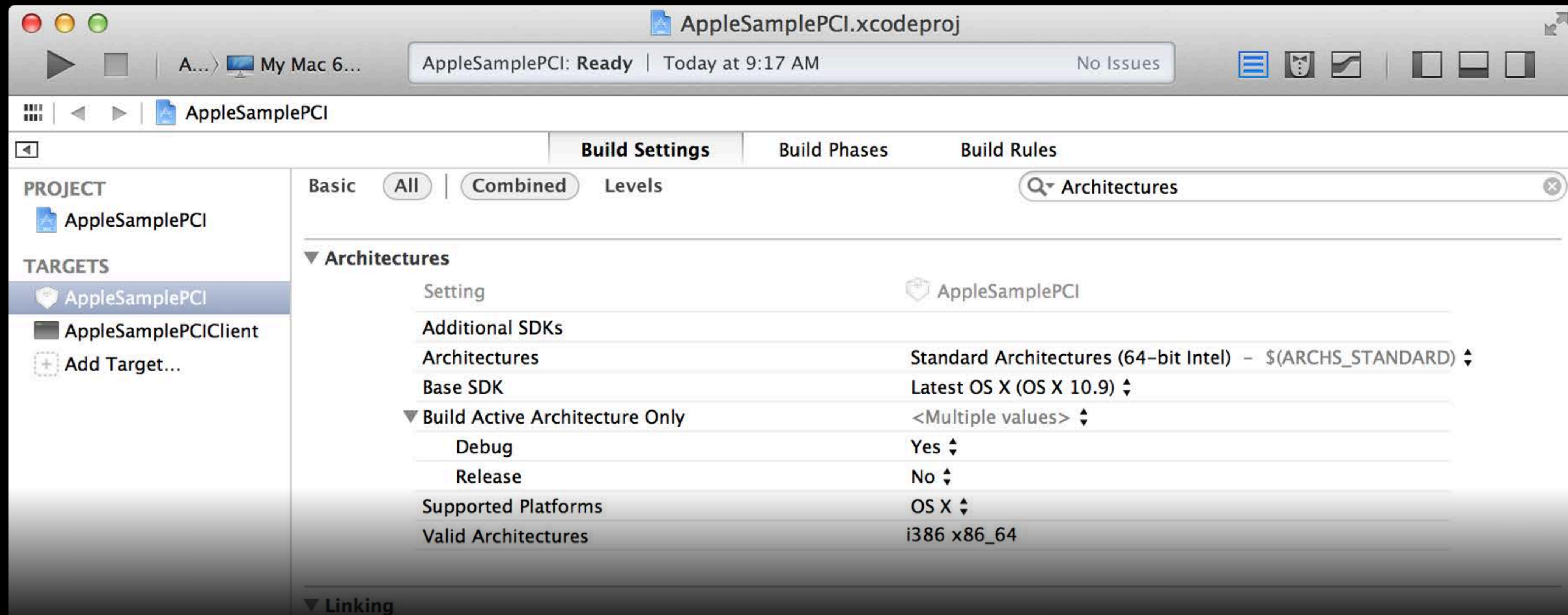
## Build settings

- Set SDK and Deployment for OS X 10.9
- Enable code signing
- Install kext in `/Library/Extensions`

# Building a Signed Kext for OS X 10.9

## Build settings

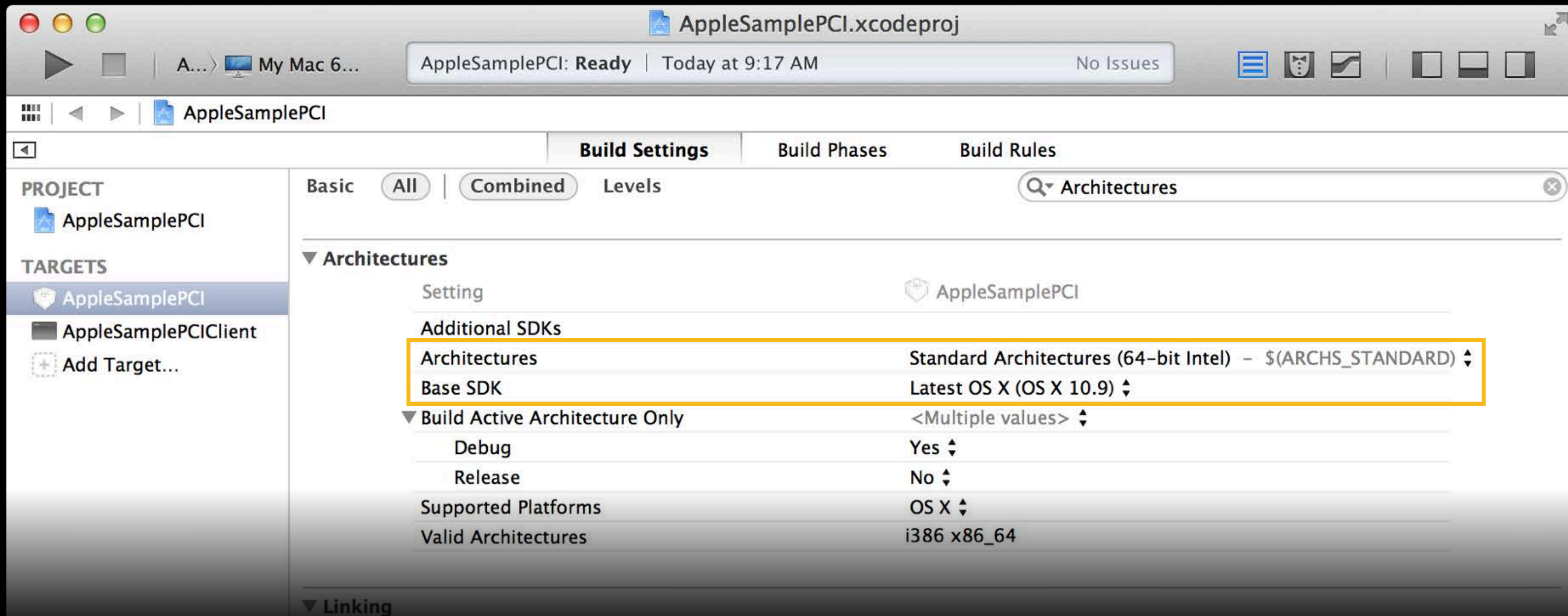
- In “Architectures” section
  - Set Architecture to “Standard Architectures”
  - Set Base SDK to Latest OS X (OS X 10.9)



# Building a Signed Kext for OS X 10.9

## Build settings

- In “Architectures” section
  - Set Architecture to “Standard Architectures”
  - Set Base SDK to Latest OS X (OS X 10.9)

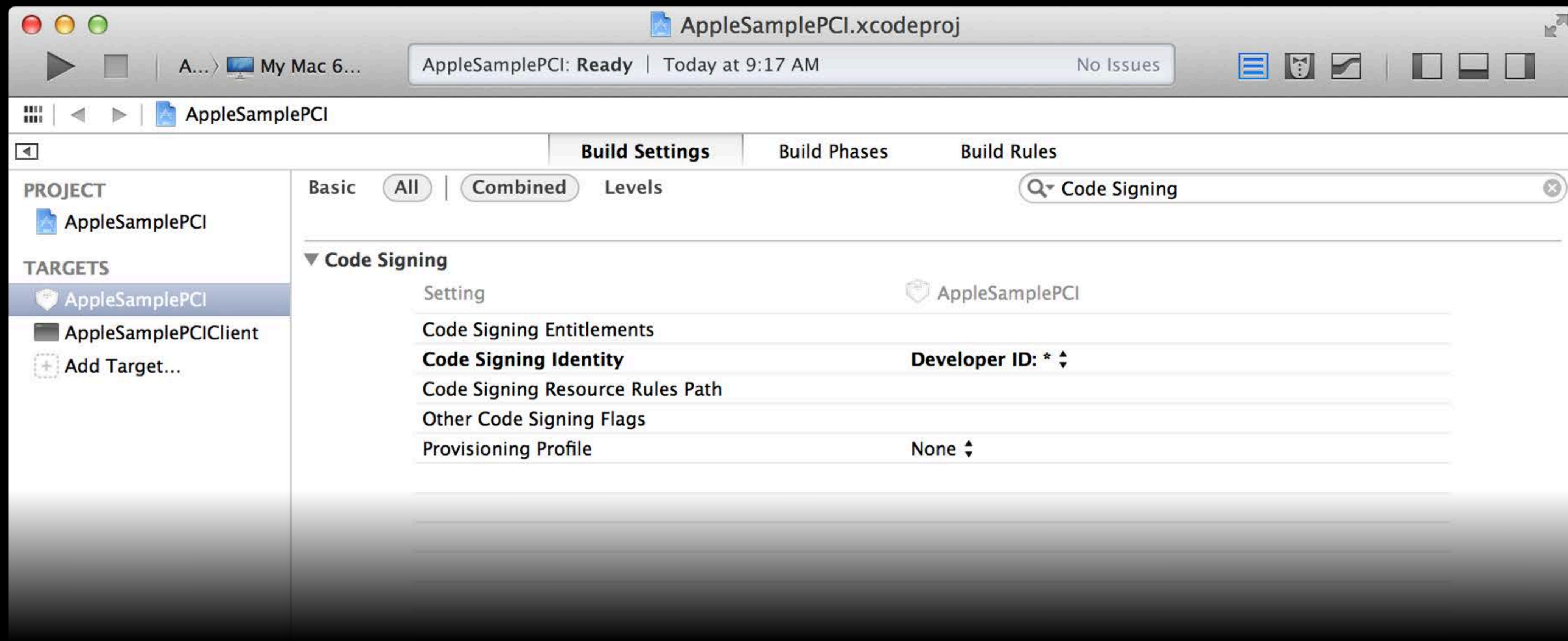




# Building a Signed Kext for OS X 10.9

## Build settings

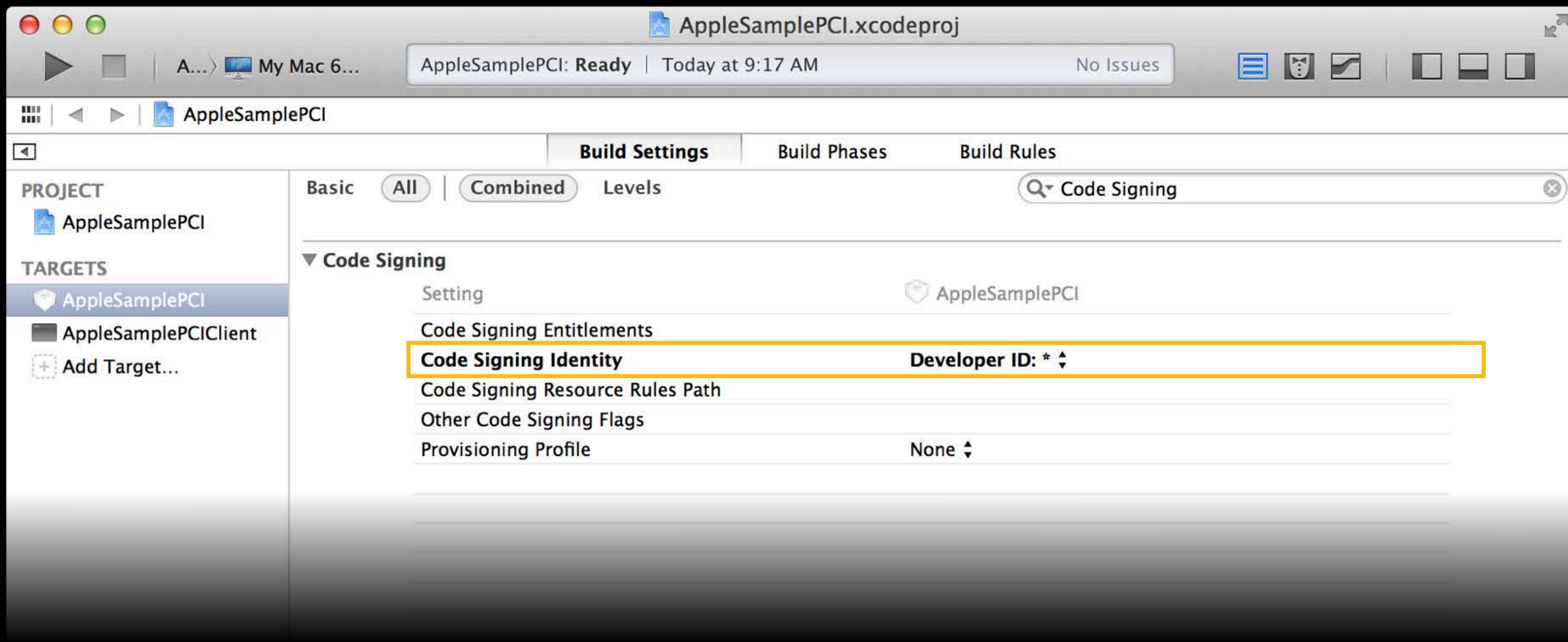
- In “Code Signing” section
  - Set “Code Signing Identity” to Developer ID: \*



# Building a Signed Kext for OS X 10.9

## Build settings

- In “Code Signing” section
  - Set “Code Signing Identity” to Developer ID: \*

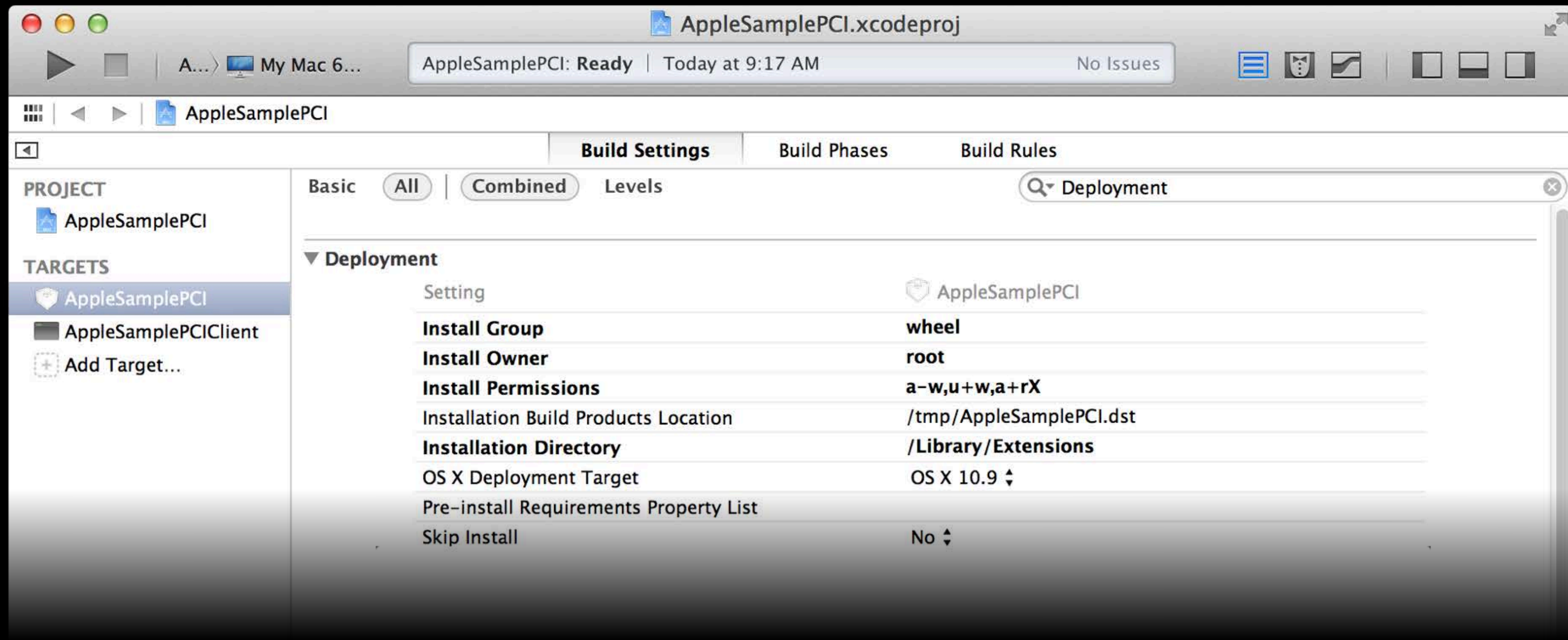




# Building a Signed Kext for OS X 10.9

## Build settings

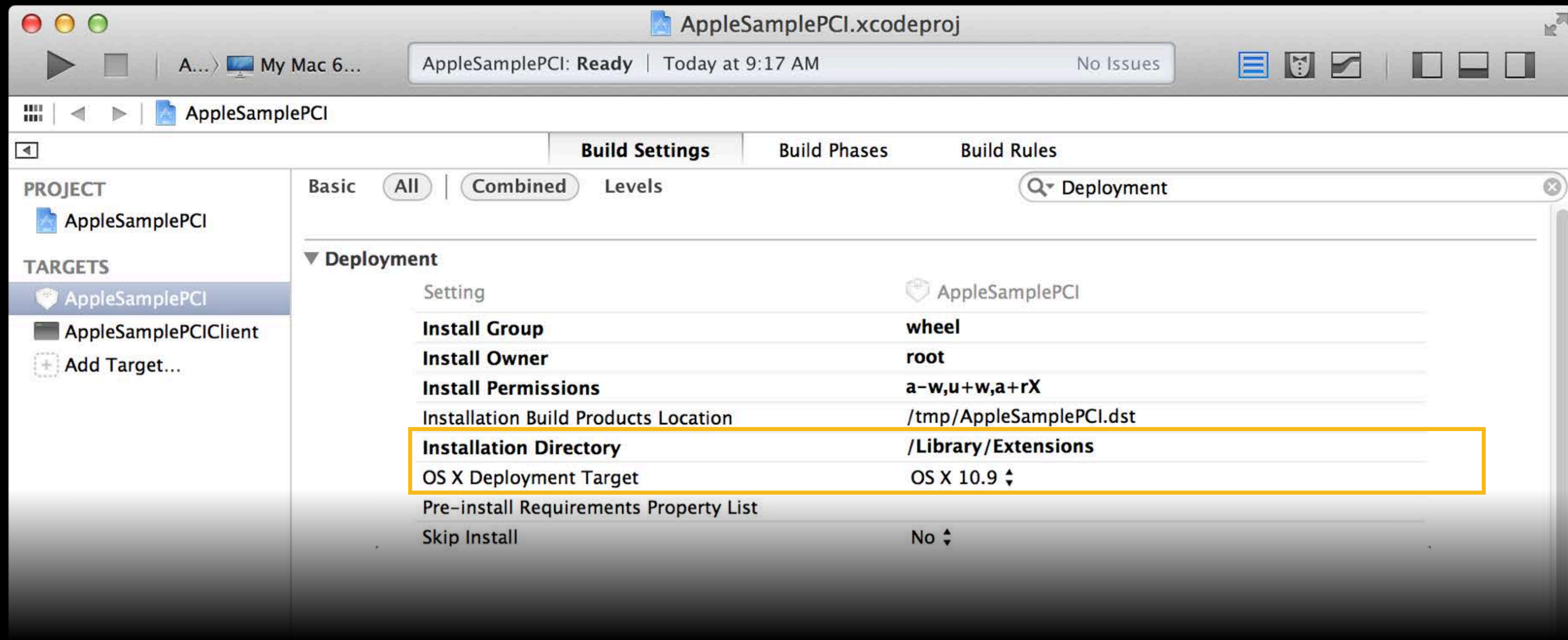
- In “Deployment” section
  - Set Installation Directory to “/Library/Extensions”
  - Set OS X Deployment Target to OS X 10.9



# Building a Signed Kext for OS X 10.9

## Build settings

- In “Deployment” section
  - Set Installation Directory to “/Library/Extensions”
  - Set OS X Deployment Target to OS X 10.9



# Building a Signed Kext for OS X 10.9

Coding

# Building a Signed Kext for OS X 10.9

## Coding

- Kexts may **only** use Kernel.framework
  - Applications may **not** use Kernel.framework



# Building a Signed Kext for OS X 10.9

## Coding

- Kexts may **only** use Kernel.framework
  - Applications may **not** use Kernel.framework
- OSObject (sub)classes must use **OSDeclare\*** and **OSDefine\*** macros
  - Required for release-to-release binary compatibility
  - Without, kext will not load on future releases
  - See OSMetaclass.h



# Building a Signed Kext for OS X 10.9

## Coding

- Kexts may **only** use Kernel.framework
  - Applications may **not** use Kernel.framework
- OSObject (sub)classes must use **OSDeclare\*** and **OSDefine\*** macros
  - Required for release-to-release binary compatibility
  - Without, kext will not load on future releases
  - See OSMetaclass.h
- Must not strip necessary symbols from kext binary
  - Safe to use **strip -S -x**

# Building a Signed Kext for OS X 10.9

Build, test, and deploy

# Building a Signed Kext for OS X 10.9

Build, test, and deploy

- Builds one kext

# Building a Signed Kext for OS X 10.9

Build, test, and deploy

- Builds one kext
- Kext should load on OS X 10.9 without warnings or alerts

# Building a Signed Kext for OS X 10.9

## Build, test, and deploy

- Builds one kext
- Kext should load on OS X 10.9 without warnings or alerts
- Use lldb for kernel debugging
  - Requires OS X 10.9 on both host and system under test

# Building a Signed Kext for OS X 10.9

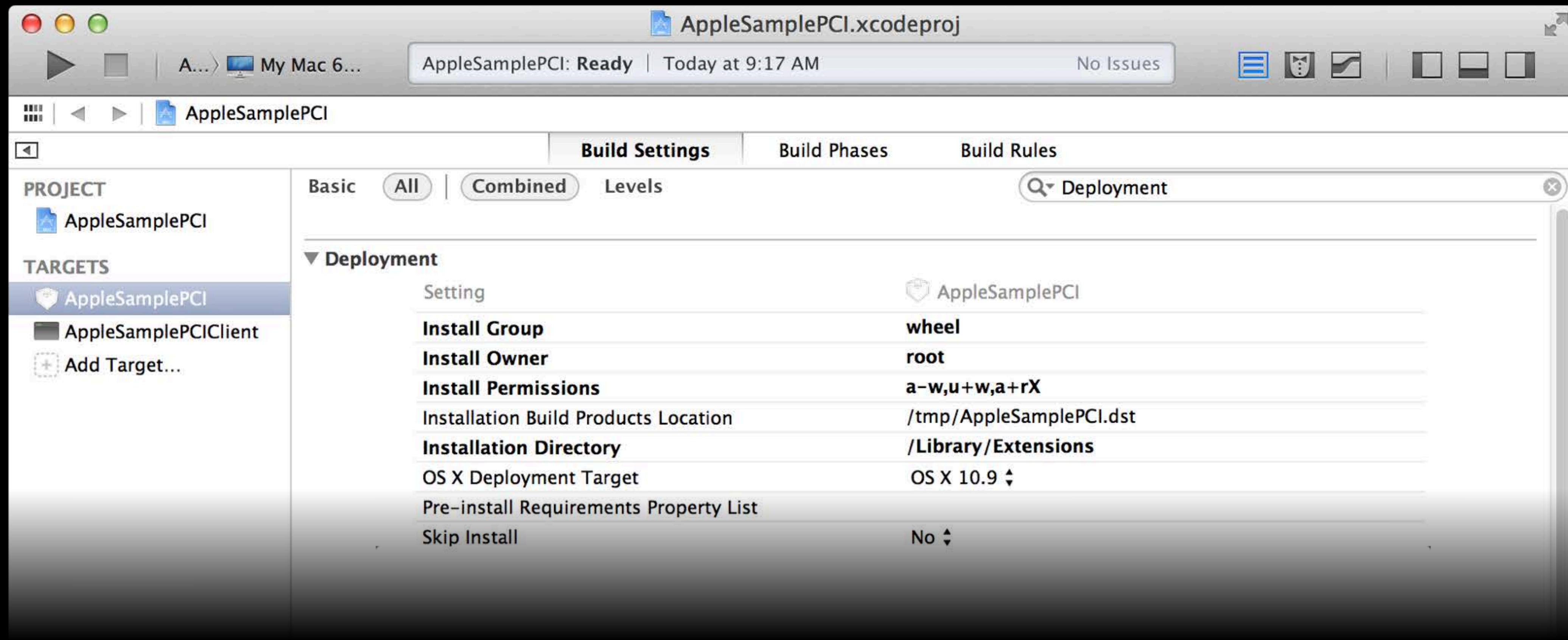
## Build, test, and deploy

- Builds one kext
- Kext should load on OS X 10.9 without warnings or alerts
- Use lldb for kernel debugging
  - Requires OS X 10.9 on both host and system under test
- After packaging, verify installed into in /Library/Extensions

# Building a Kext for OS X 10.8 and Later

## Add a build target

- Start with OS X 10.9 kext example
- Add a 2nd kext target for OS X 10.8 kext
- Same values used for initial project setup

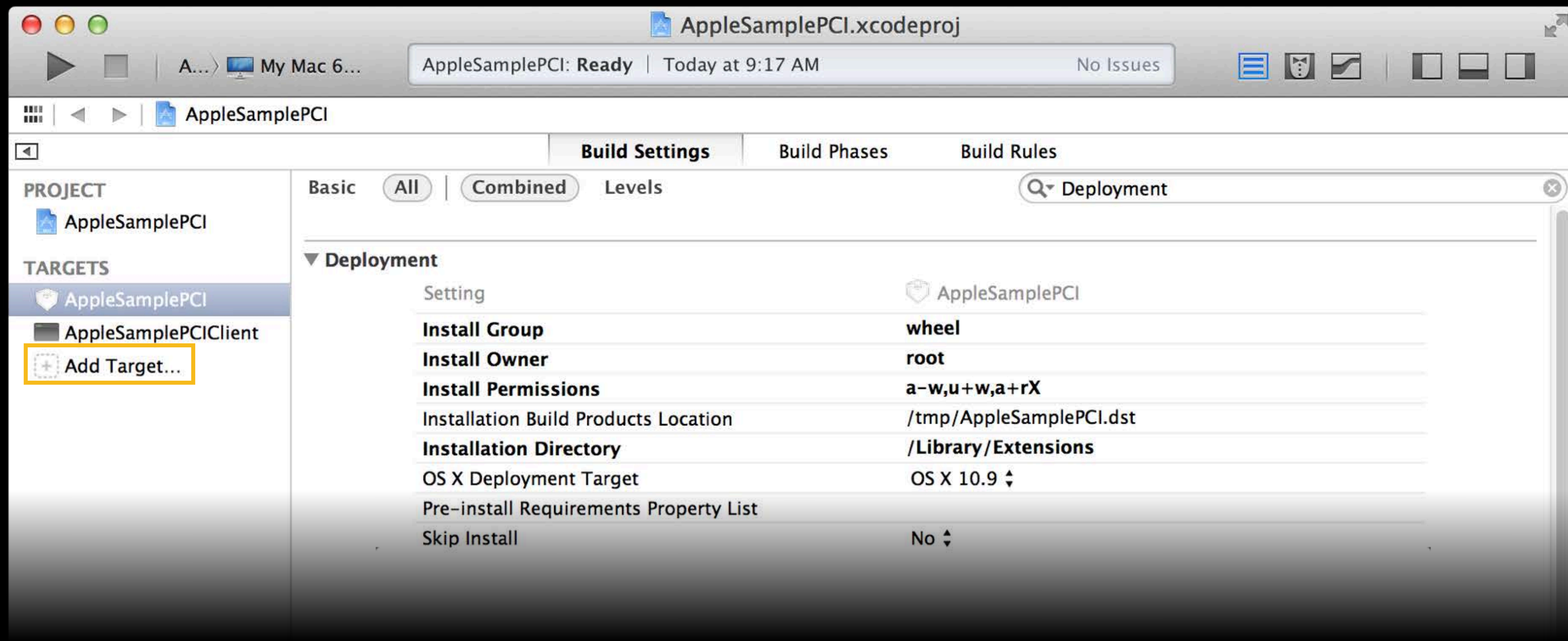




# Building a Kext for OS X 10.8 and Later

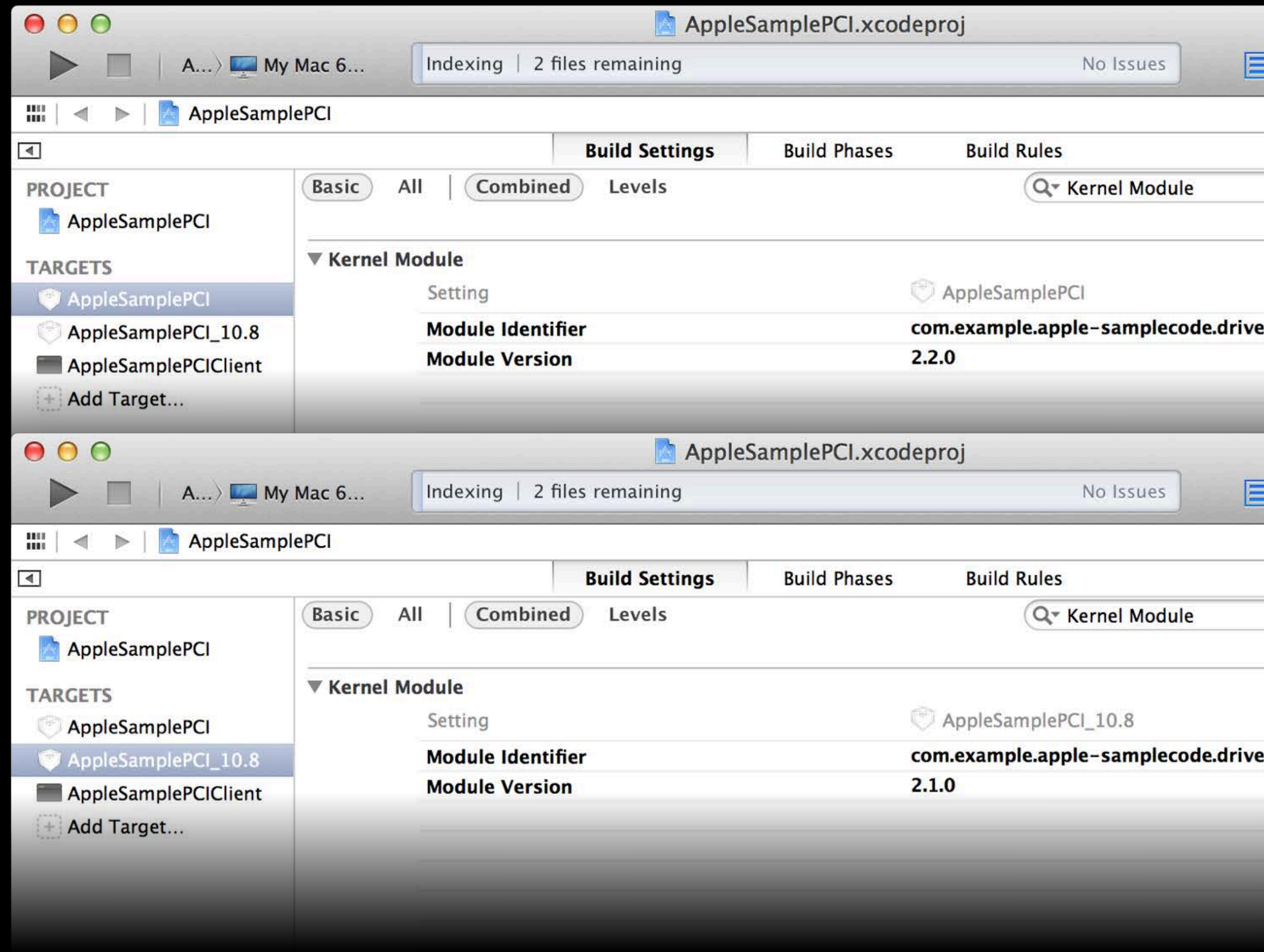
## Add a build target

- Start with OS X 10.9 kext example
- Add a 2nd kext target for OS X 10.8 kext
- Same values used for initial project setup



# Building a Kext for OS X 10.8 and Later

## Build settings

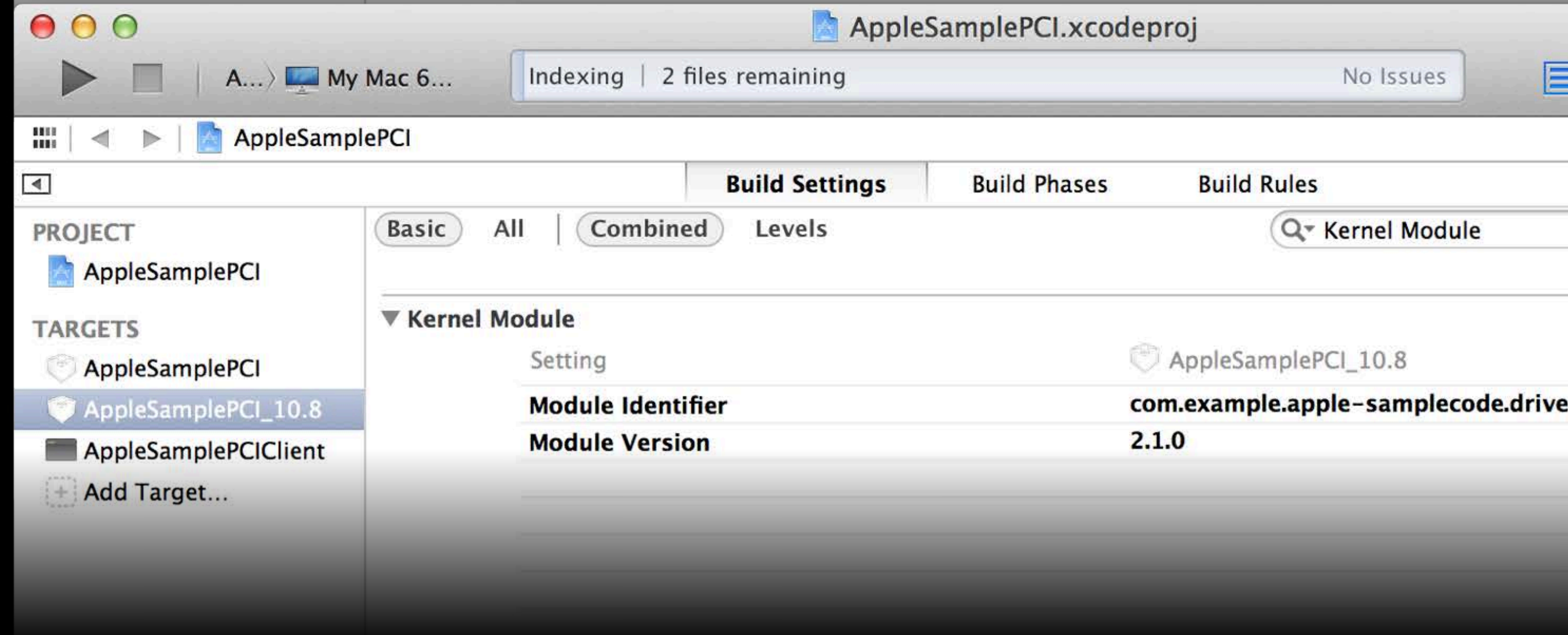
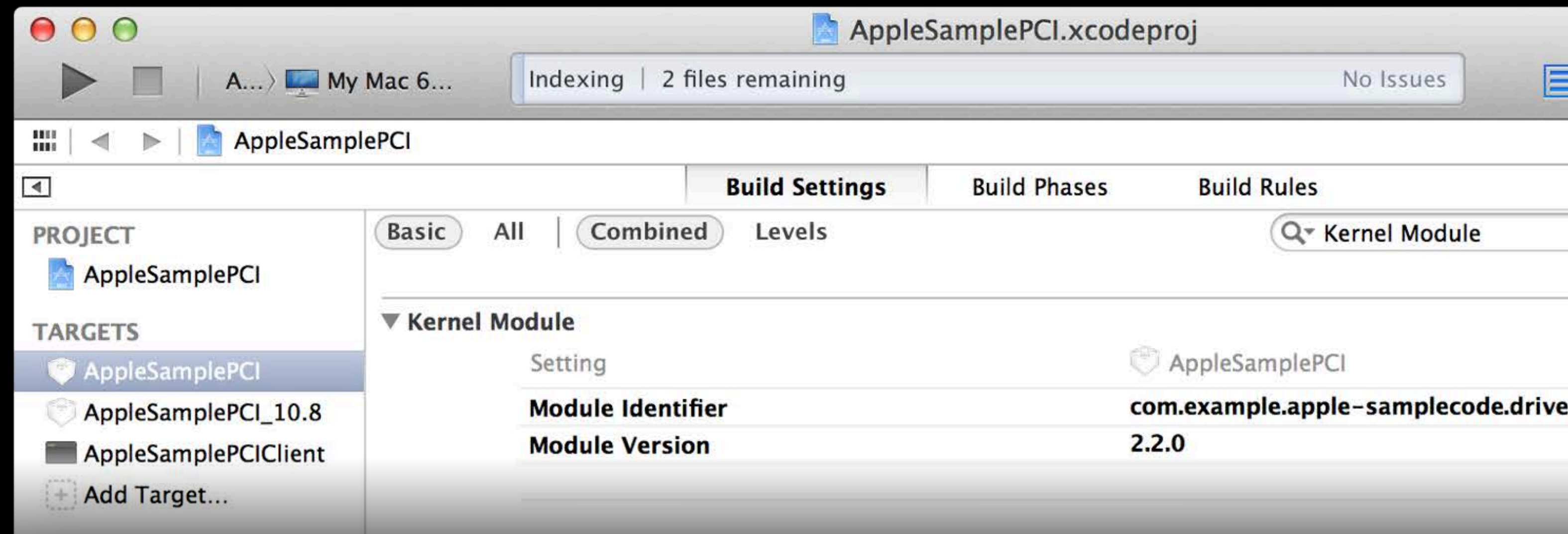




# Building a Kext for OS X 10.8 and Later

## Build settings

- Under “Kernel Module”

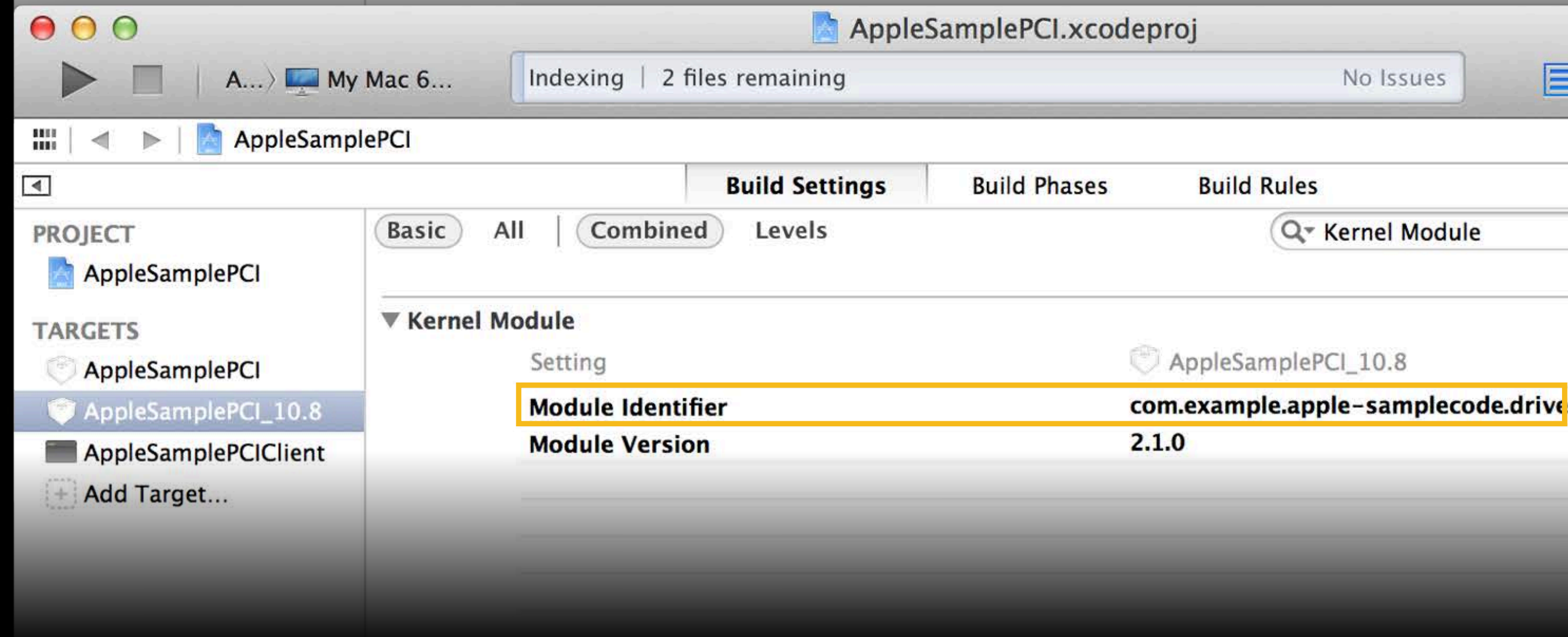
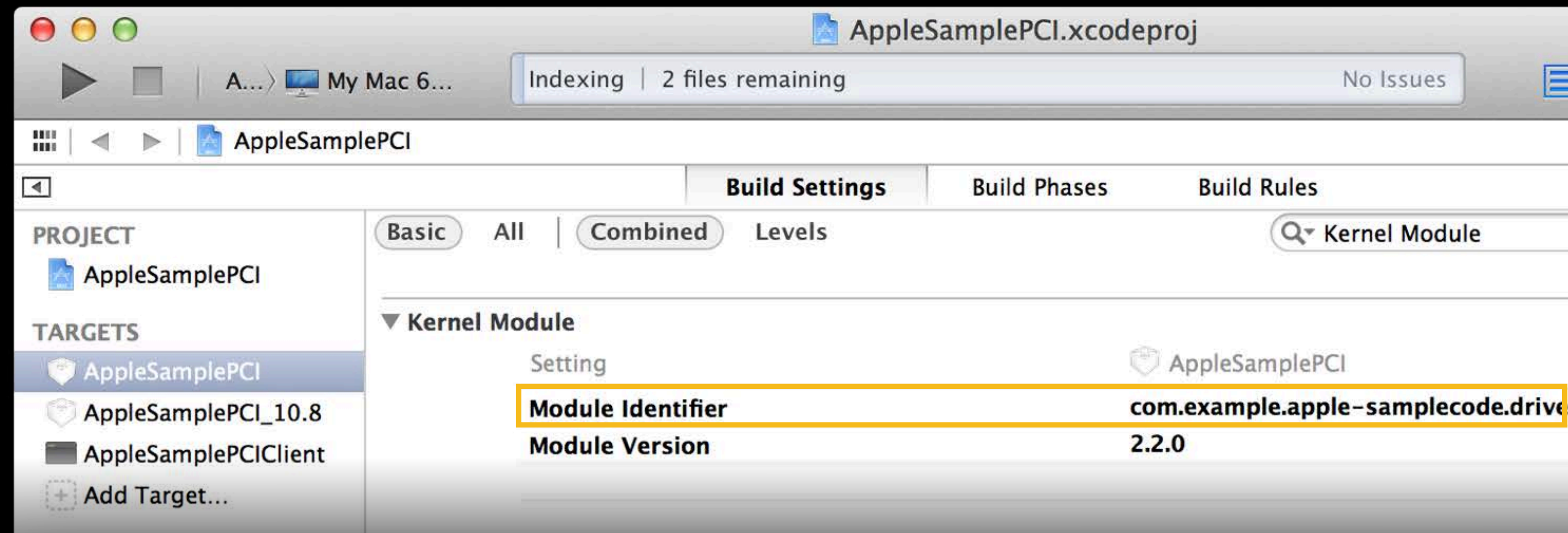




# Building a Kext for OS X 10.8 and Later

## Build settings

- Under “Kernel Module”
  - “Module Identifier” must be same

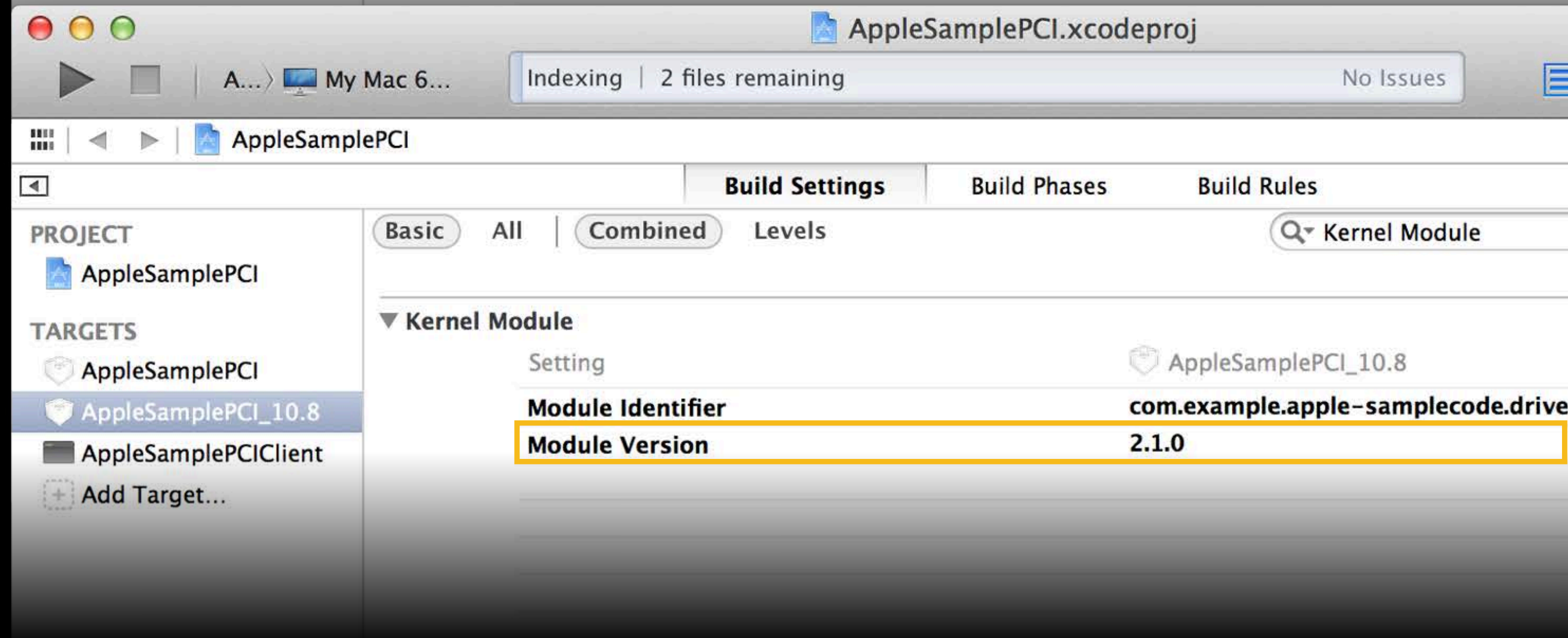
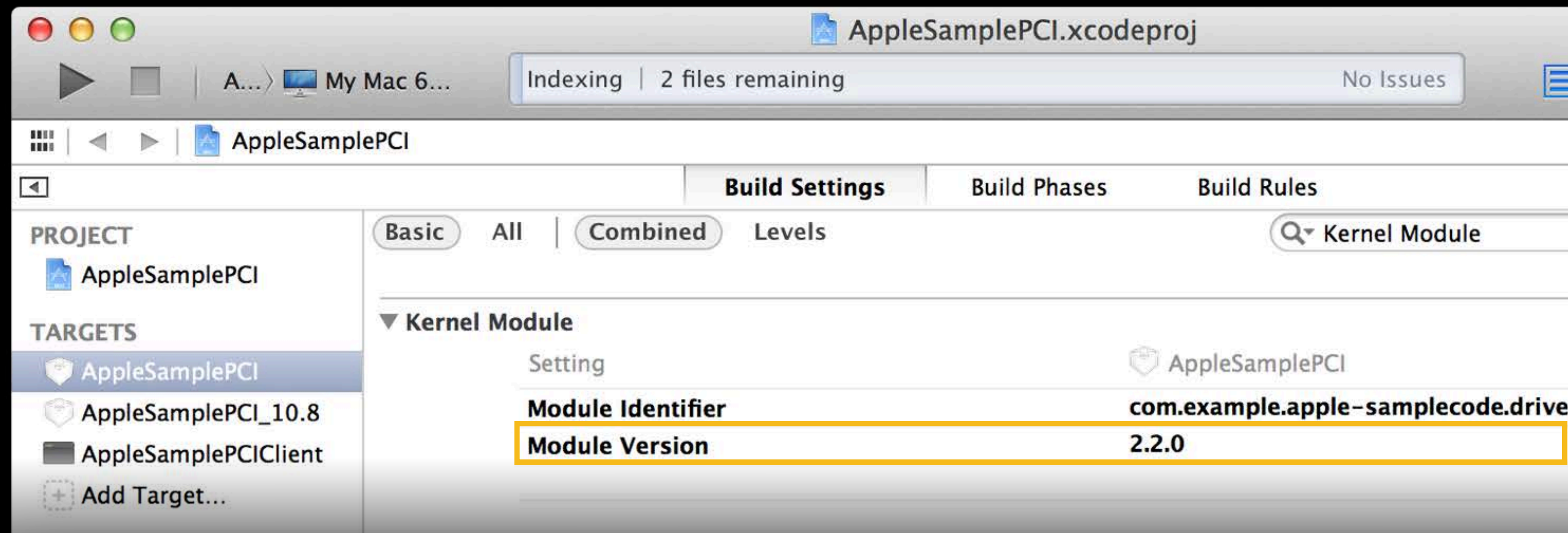




# Building a Kext for OS X 10.8 and Later

## Build settings

- Under “Kernel Module”
  - “Module Identifier” must be same
  - “Module Version” must be lower for OS X 10.8 target

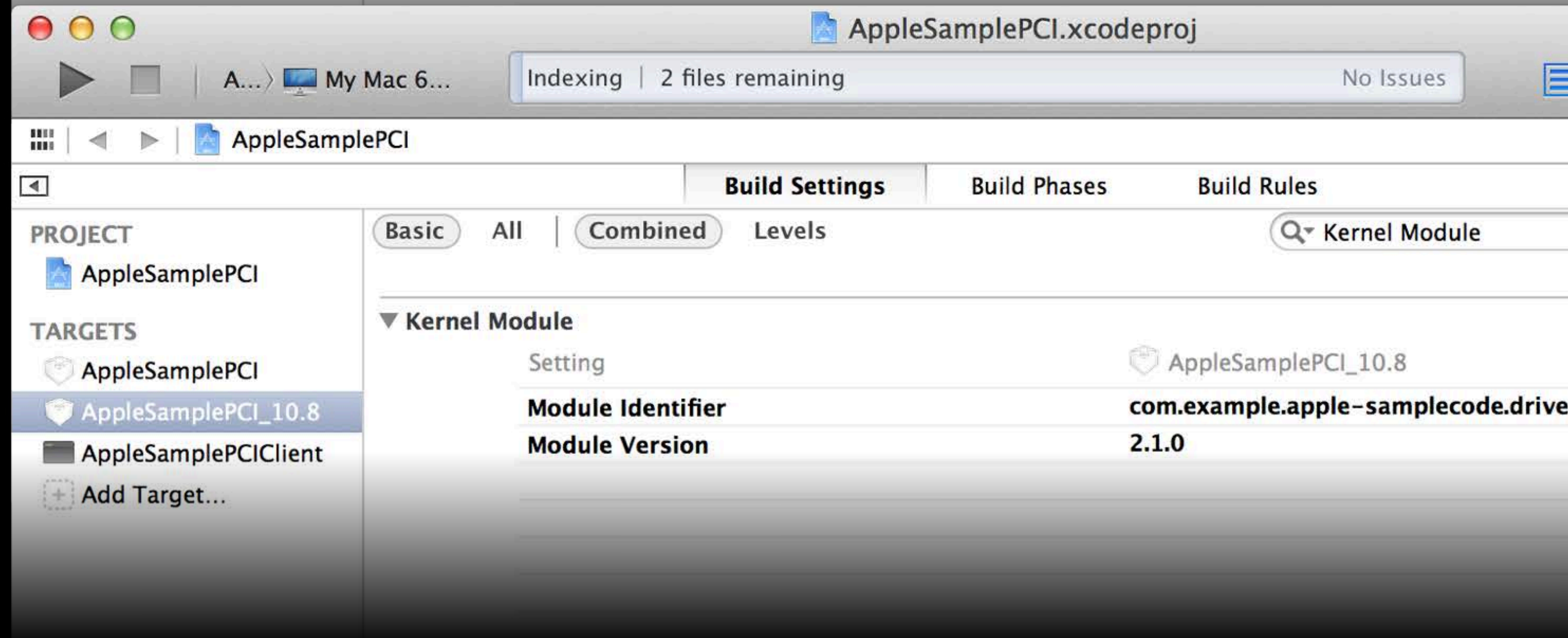
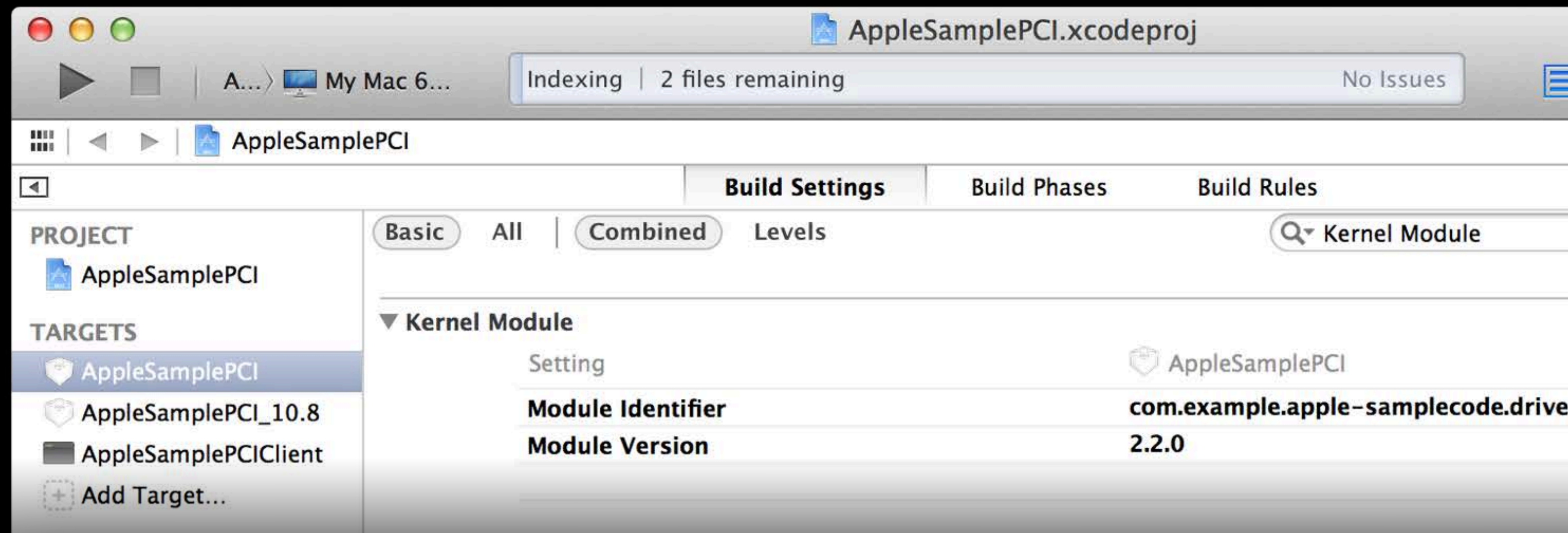




# Building a Kext for OS X 10.8 and Later

## Build settings

- Under “Kernel Module”
  - “Module Identifier” must be same
  - “Module Version” must be lower for OS X 10.8 target
  - Necessary for OS to select correct kext





# Building a Kext for OS X 10.8 and Later

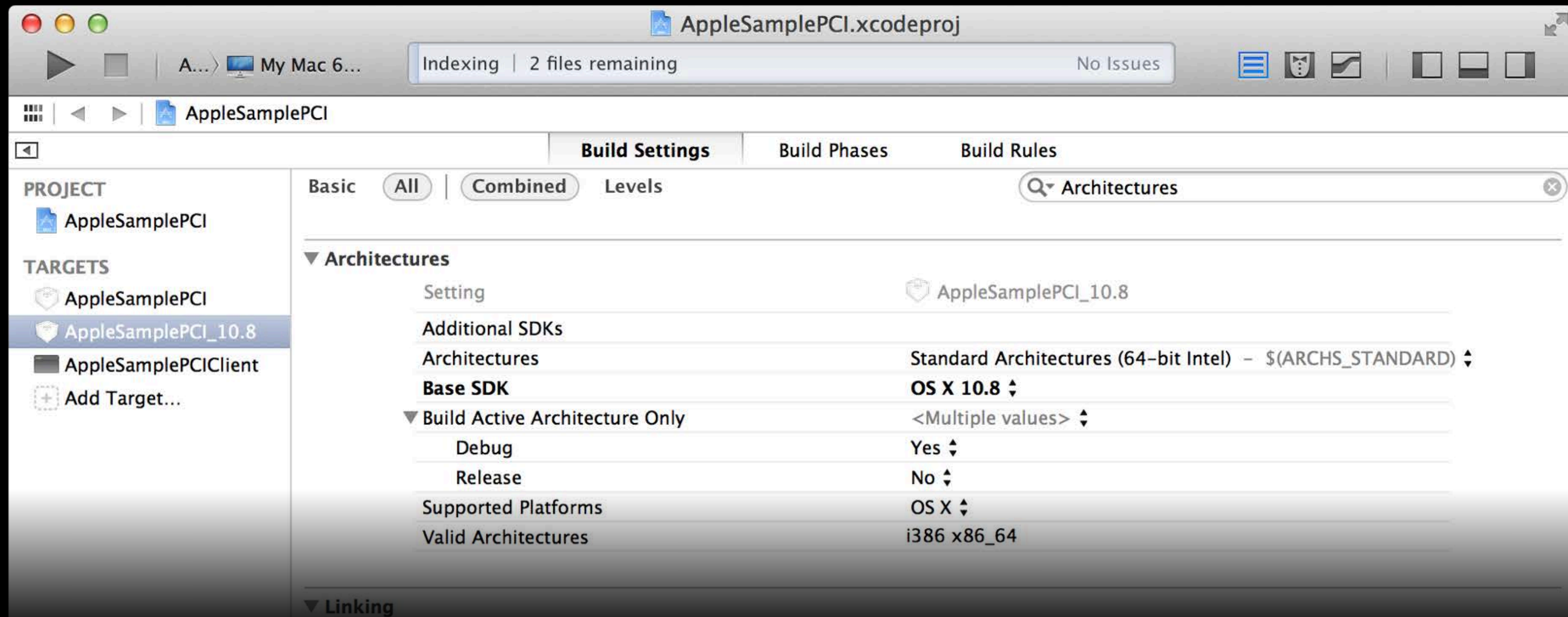
## Build settings (OS X 10.8 Target)

- Set SDK and Deployment for OS X 10.8
- Disable code signing
- Install kext in `/System/Library/Extensions`

# Building a Kext for OS X 10.8 and Later

## Build settings (OS X 10.8 Target)

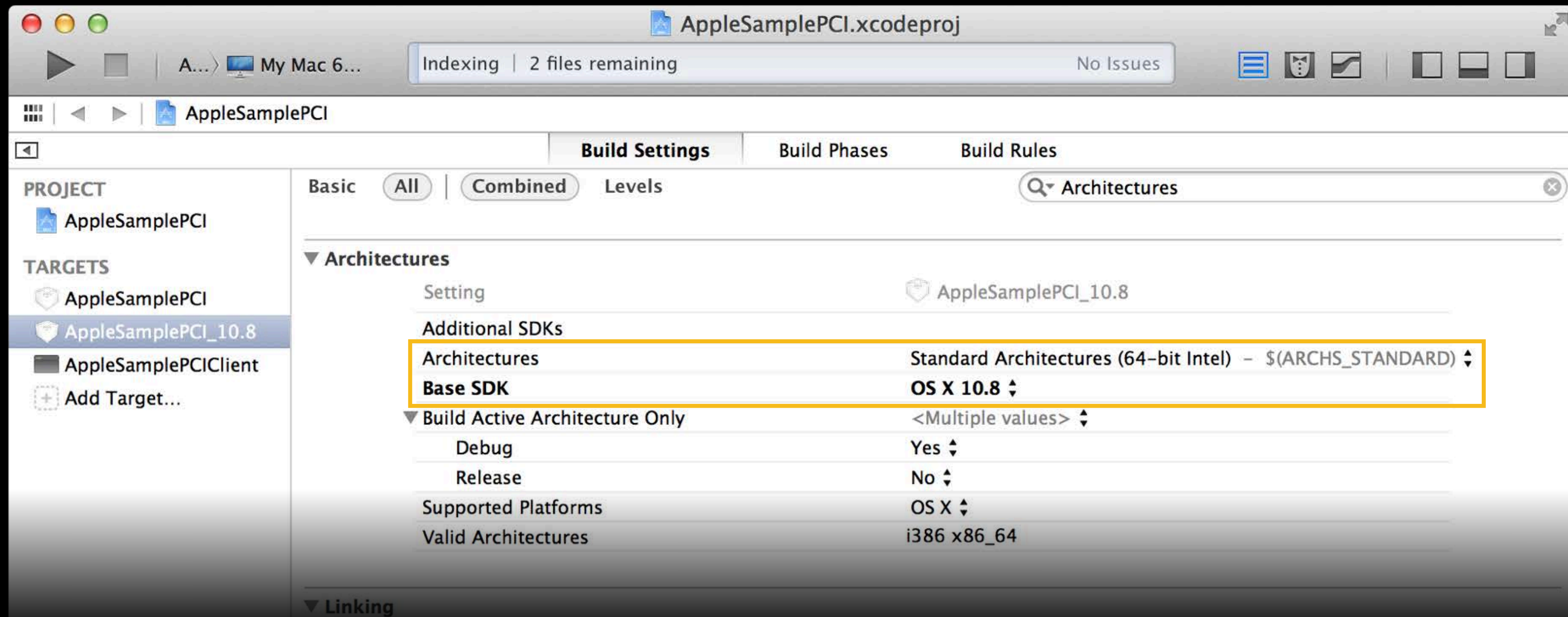
- In “Architectures” section
  - Set Architecture to “Standard Architectures”
  - Set Base SDK to OS X 10.8



# Building a Kext for OS X 10.8 and Later

## Build settings (OS X 10.8 Target)

- In “Architectures” section
  - Set Architecture to “Standard Architectures”
  - Set Base SDK to OS X 10.8

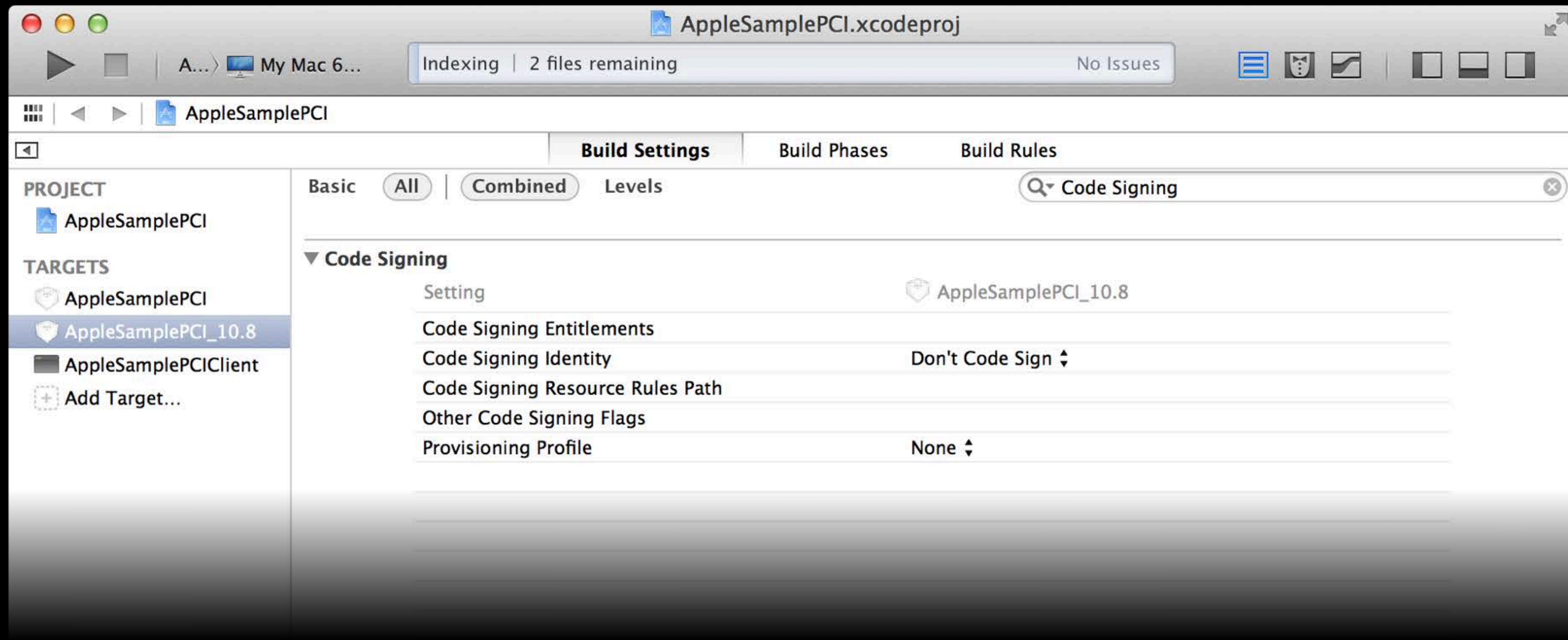




# Building a Kext for OS X 10.8 and Later

## Build settings (OS X 10.8 Target)

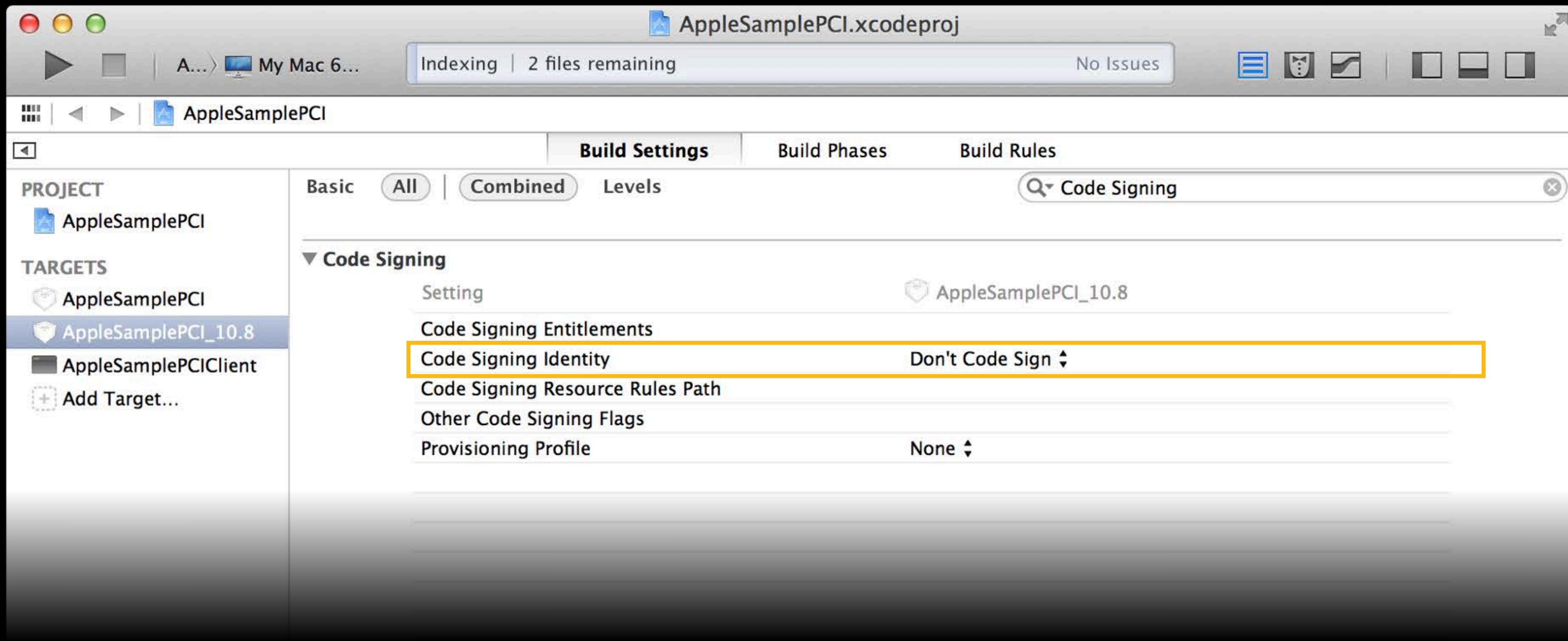
- In “Code Signing” section
  - Set “Code Signing Identity” to “Don’t Code Sign”



# Building a Kext for OS X 10.8 and Later

## Build settings (OS X 10.8 Target)

- In “Code Signing” section
  - Set “Code Signing Identity” to “Don’t Code Sign”

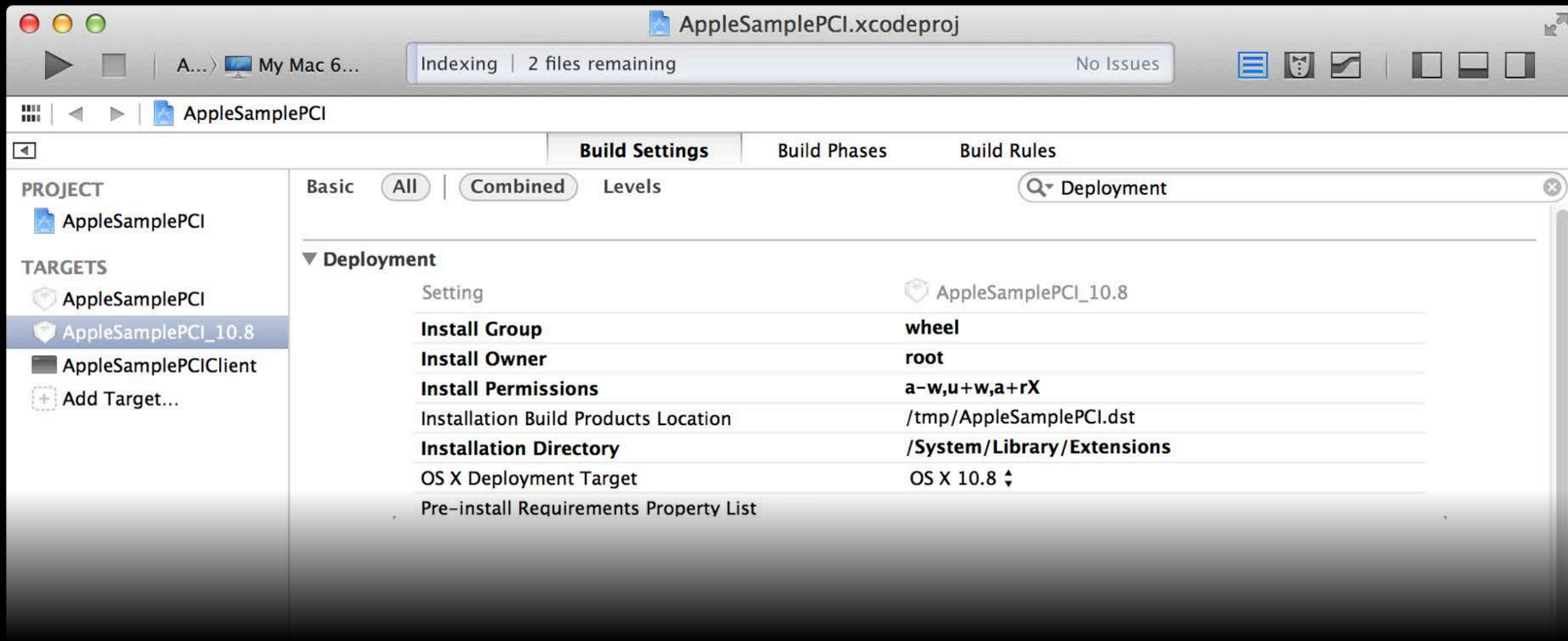




# Building a Kext for OS X 10.8 and Later

## Build settings (OS X 10.8 Target)

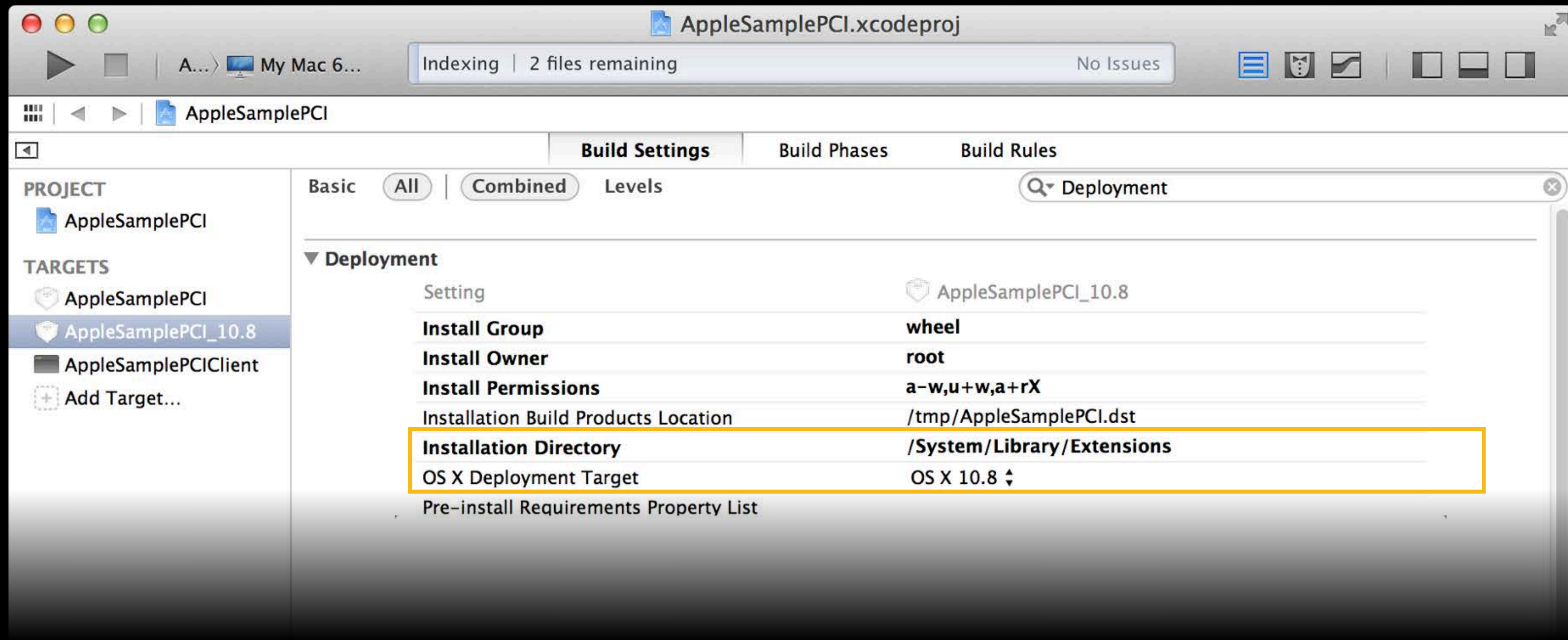
- In “Deployment” section
  - Set Installation Directory to “/System/Library/Extensions”
  - Set OS X Deployment Target to OS X 10.8



# Building a Kext for OS X 10.8 and Later

## Build settings (OS X 10.8 Target)

- In “Deployment” section
  - Set Installation Directory to “/System/Library/Extensions”
  - Set OS X Deployment Target to OS X 10.8



# Building a Kext for OS X 10.8 and Later

Build, test, and deploy

- Builds two kexts



# Building a Kext for OS X 10.8 and Later

## Build, test, and deploy

- Builds two kexts
- Signed kext
  - Should load on OS X 10.9 without warnings or alerts
  - Use lldb to debug
  - Verify installed into /Library/Extensions

# Building a Kext for OS X 10.8 and Later

## Build, test, and deploy

- Builds two kexts
- Signed kext
  - Should load on OS X 10.9 without warnings or alerts
  - Use lldb to debug
  - Verify installed into /Library/Extensions
- Unsigned kext
  - Should load on OS X 10.8
  - Use gdb to debug
  - Verify installed into /System/Library/Extensions



# Building a Kext for OS X 10.8 and Later

## Build, test, and deploy

- Builds two kexts
- Signed kext
  - Should load on OS X 10.9 without warnings or alerts
  - Use lldb to debug
  - Verify installed into /Library/Extensions
- Unsigned kext
  - Should load on OS X 10.8
  - Use gdb to debug
  - Verify installed into /System/Library/Extensions
- Consider Installer script to avoid installing OS X 10.8 kext on OS X 10.9

# Building a Kext for OS X 10.7 and Later

## The tools

- Use OS X 10.8.3 or later
  - Required for correct signing

# Building a Kext for OS X 10.7 and Later

## The tools

- Use OS X 10.8.3 or later
  - Required for correct signing
- Use Xcode 4.6—not Xcode 5.0
  - Required for OS X 10.7 SDK

# Building a Kext for OS X 10.7 and Later

## The tools

- Use OS X 10.8.3 or later
  - Required for correct signing
- Use Xcode 4.6—not Xcode 5.0
  - Required for OS X 10.7 SDK
- Requires Developer ID certificate for Applications and Kernel Extensions
  - Acquire from Apple Developer website
  - Install in your keychain

# Building a Kext for OS X 10.7 and Later

## Add a build target

- Same two-target approach as previously shown
  - OS X 10.9 target identical



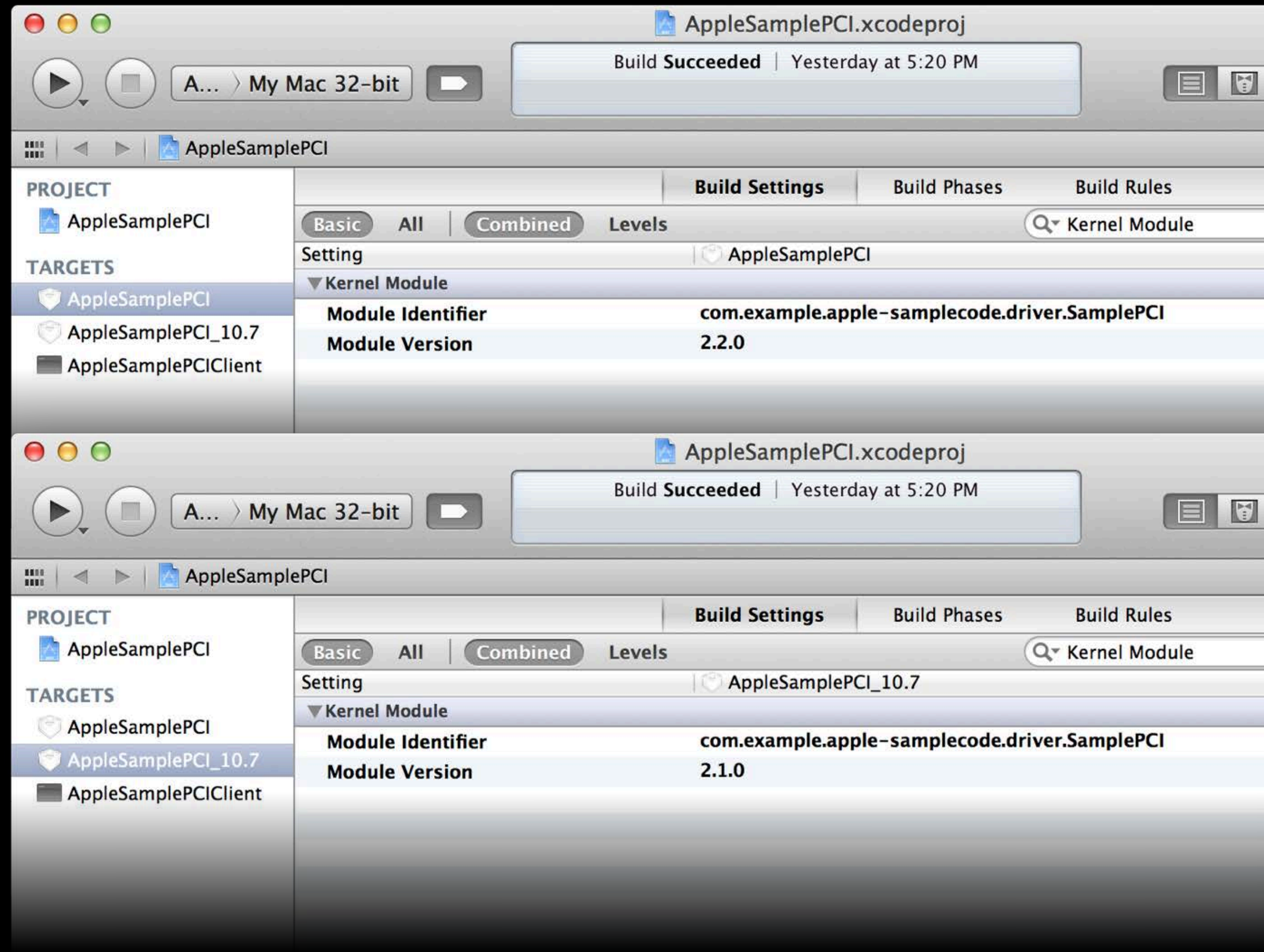
# Building a Kext for OS X 10.7 and Later

## Add a build target

- Same two-target approach as previously shown
  - OS X 10.9 target identical
- 2nd target has slightly different settings
  - OS X 10.7 SDK and deployment target
  - Need to add 32-bit architecture
  - Resulting kext will work on OS X 10.7 and OS X 10.8

# Building a Kext for OS X 10.7 and Later

## Build settings

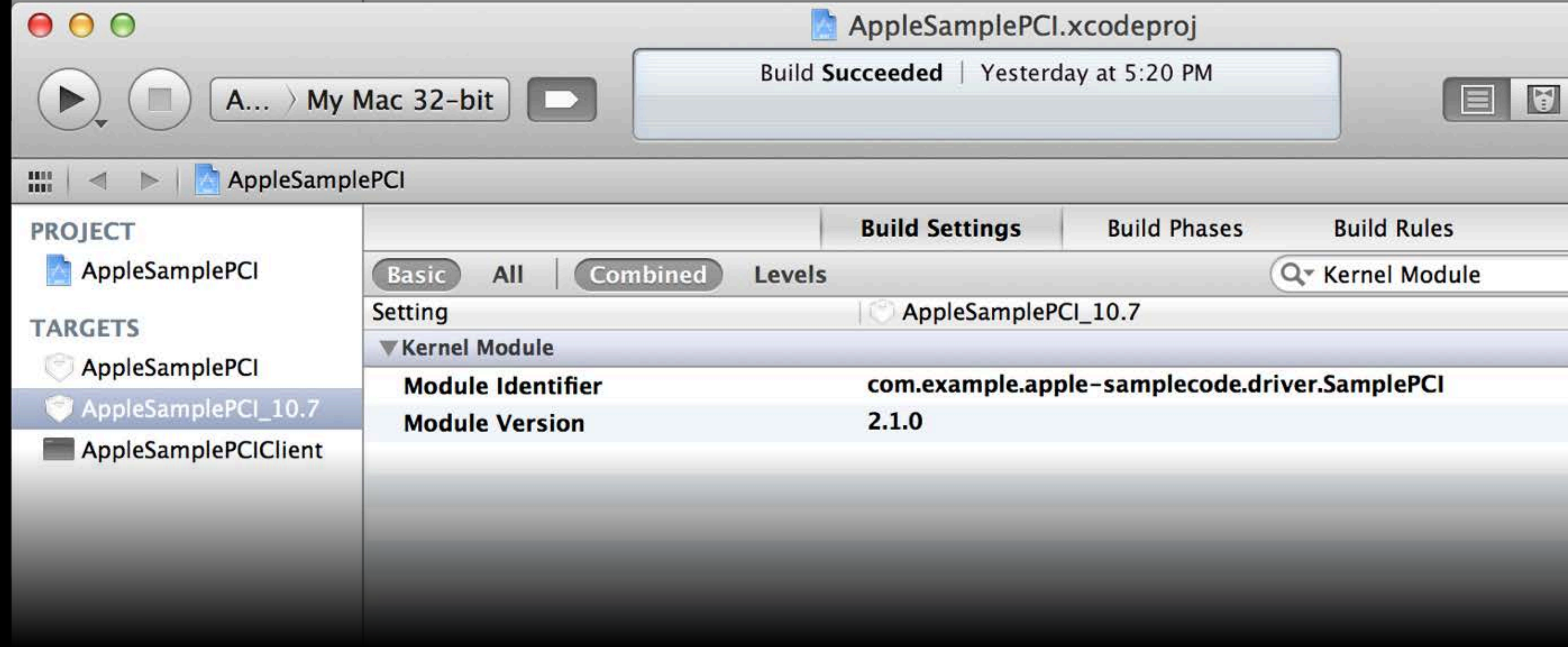
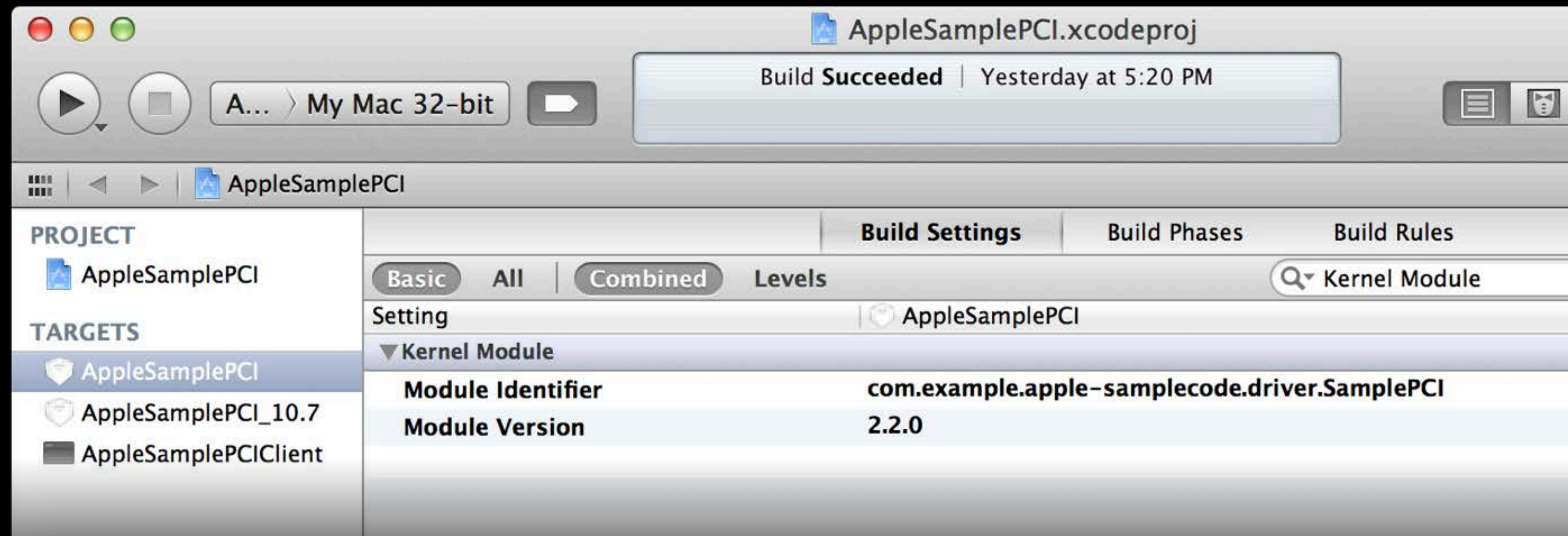




# Building a Kext for OS X 10.7 and Later

## Build settings

- Under “Kernel Module”

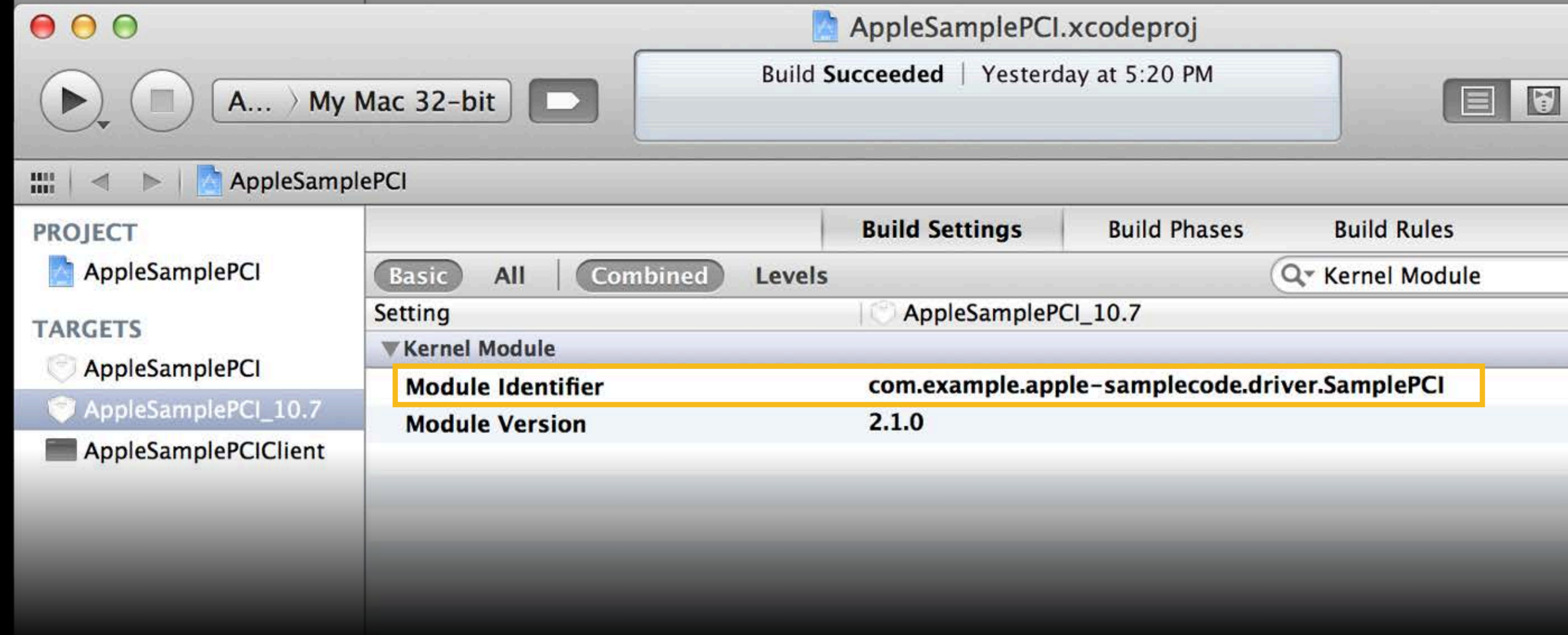
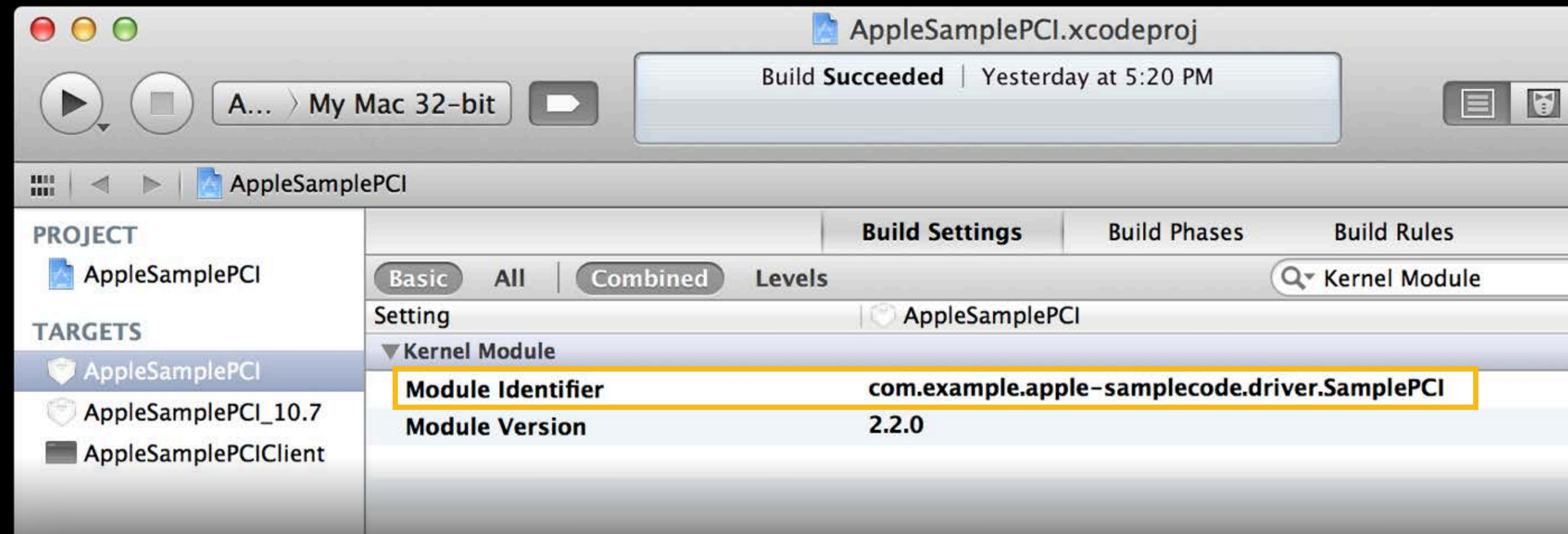




# Building a Kext for OS X 10.7 and Later

## Build settings

- Under “Kernel Module”
  - “Module Identifier” must be same

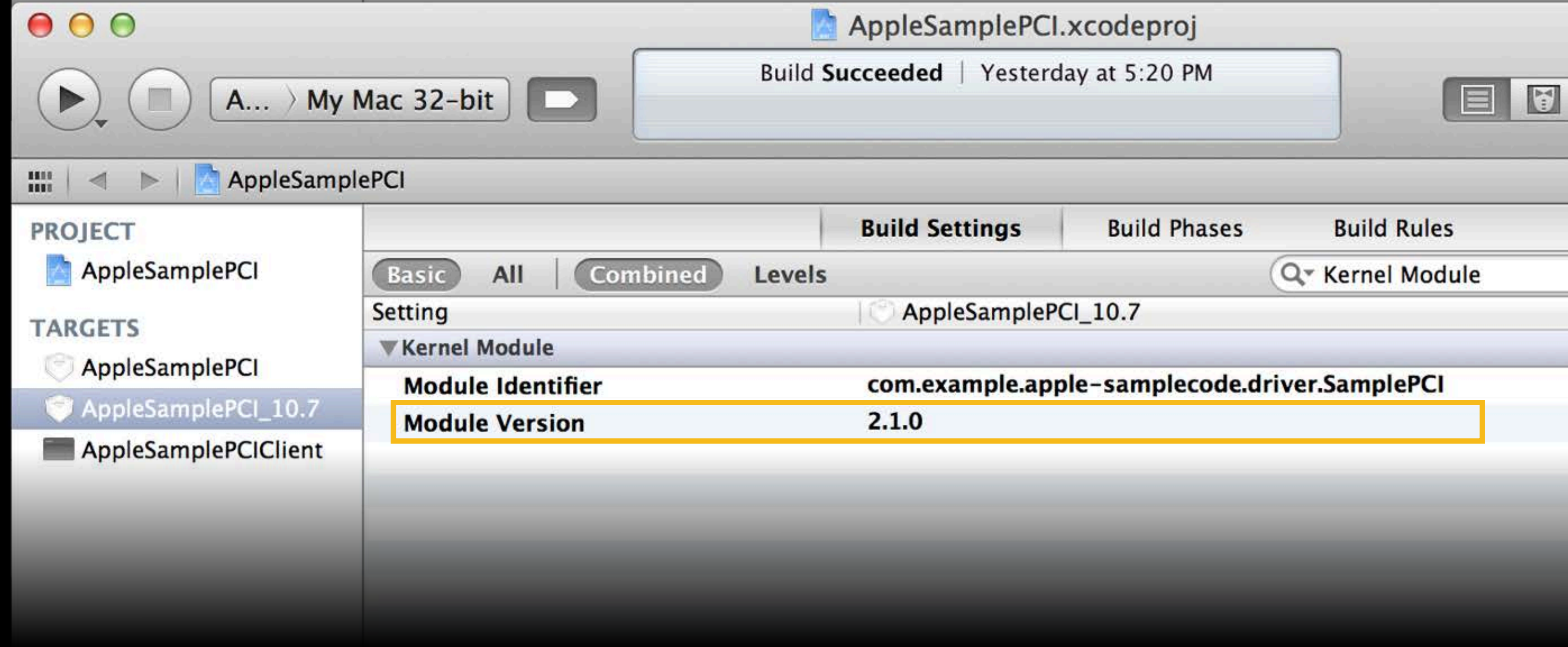
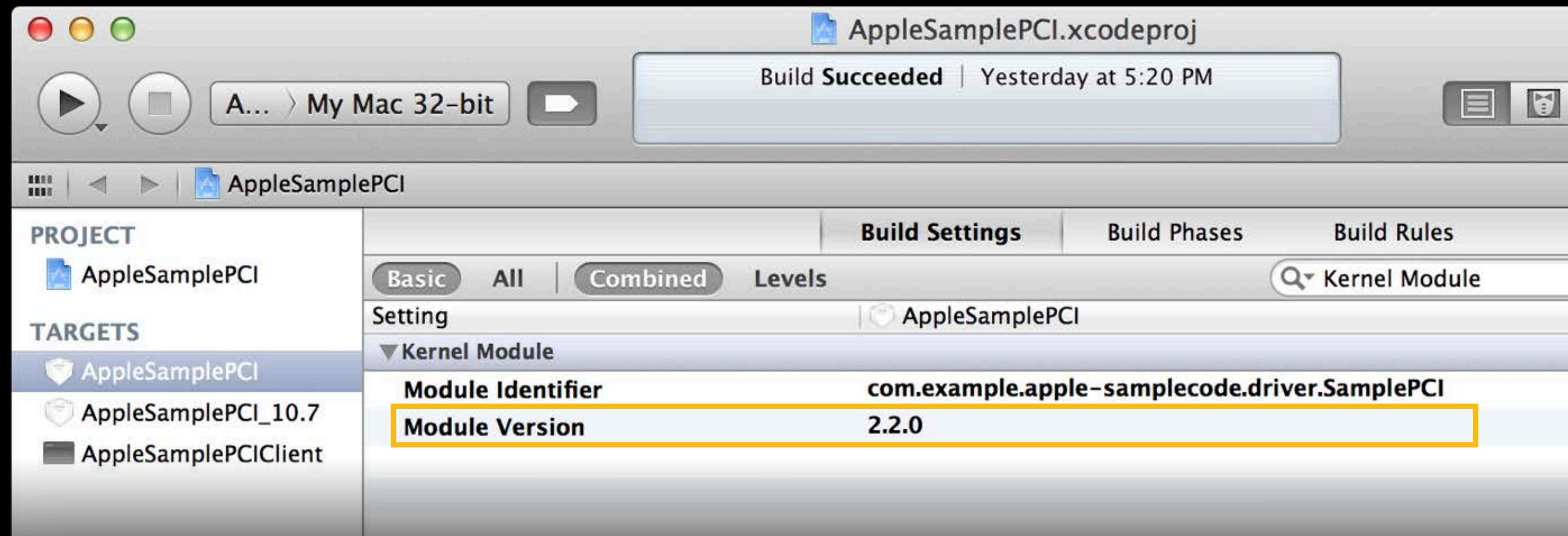




# Building a Kext for OS X 10.7 and Later

## Build settings

- Under “Kernel Module”
  - “Module Identifier” must be same
  - “Module Version” must be lower for OS X 10.7 target

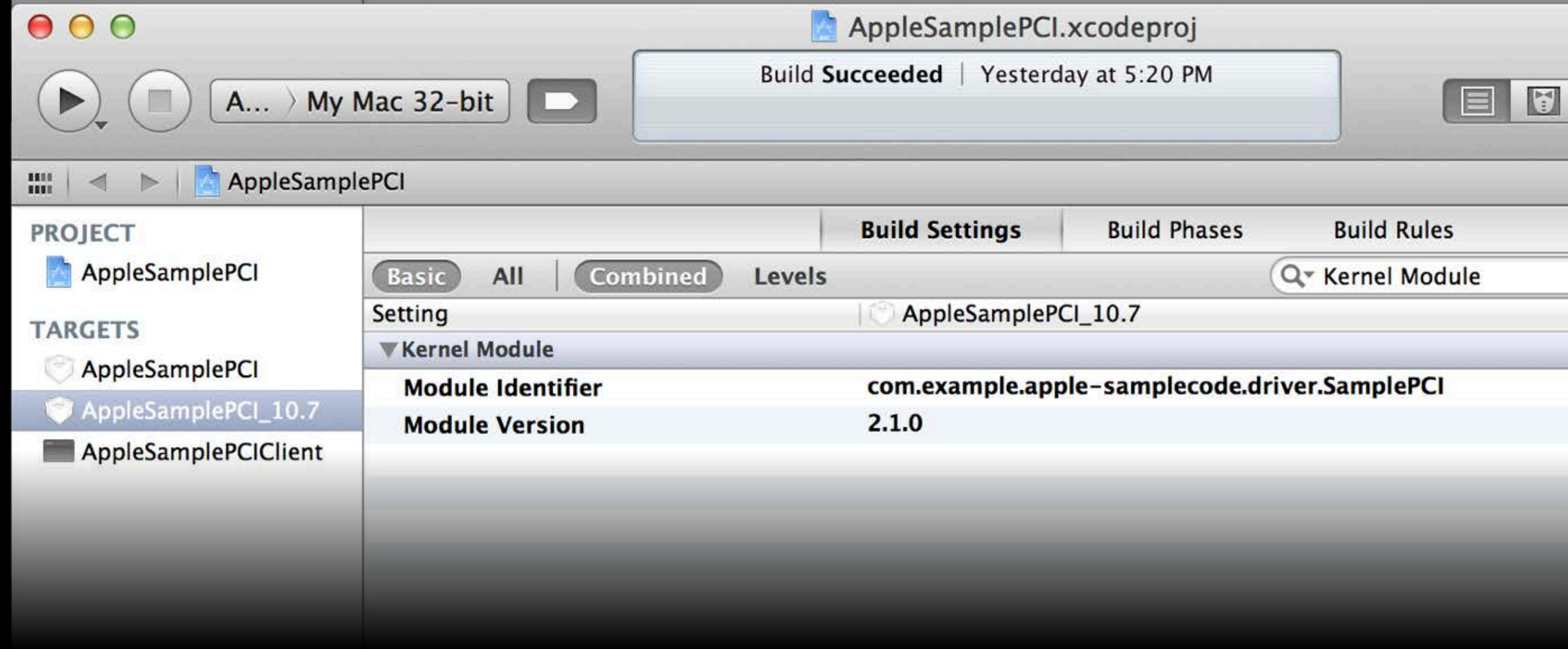
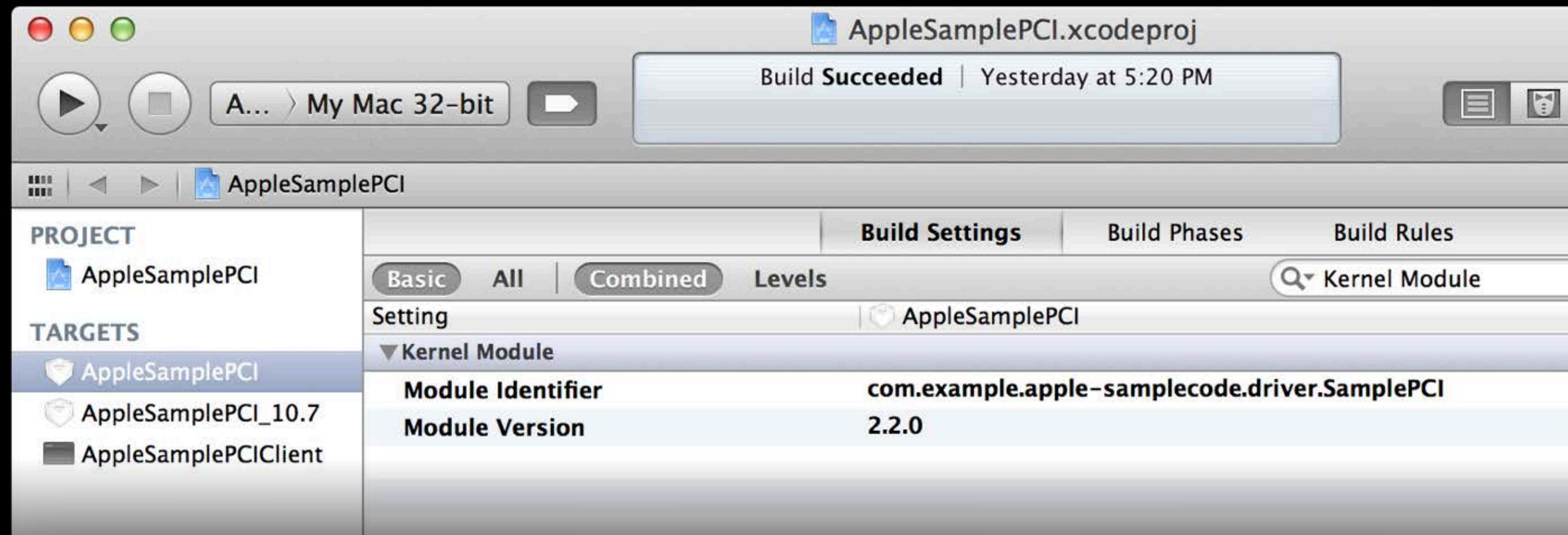




# Building a Kext for OS X 10.7 and Later

## Build settings

- Under “Kernel Module”
  - “Module Identifier” must be same
  - “Module Version” must be lower for OS X 10.7 target
  - Necessary for OS to select correct kext

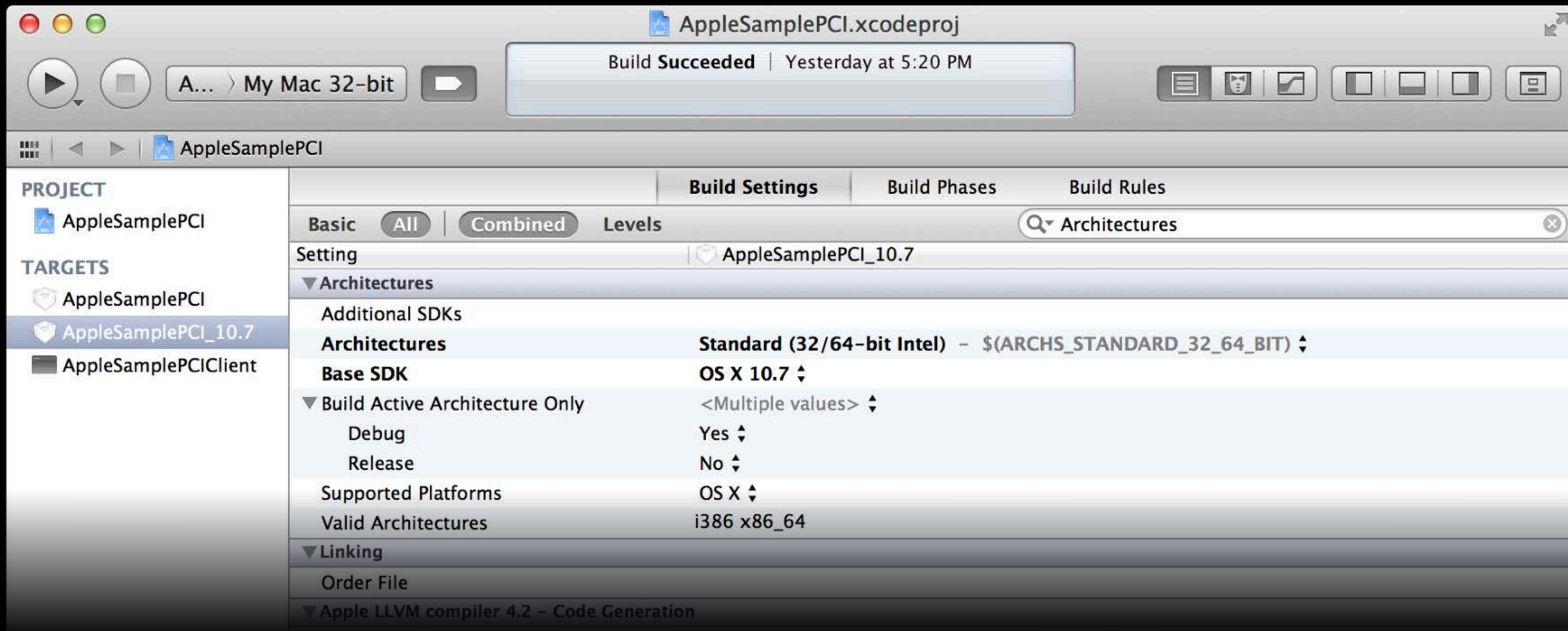




# Building a Kext for OS X 10.7 and Later

## Build settings (OS X 10.7 Target)

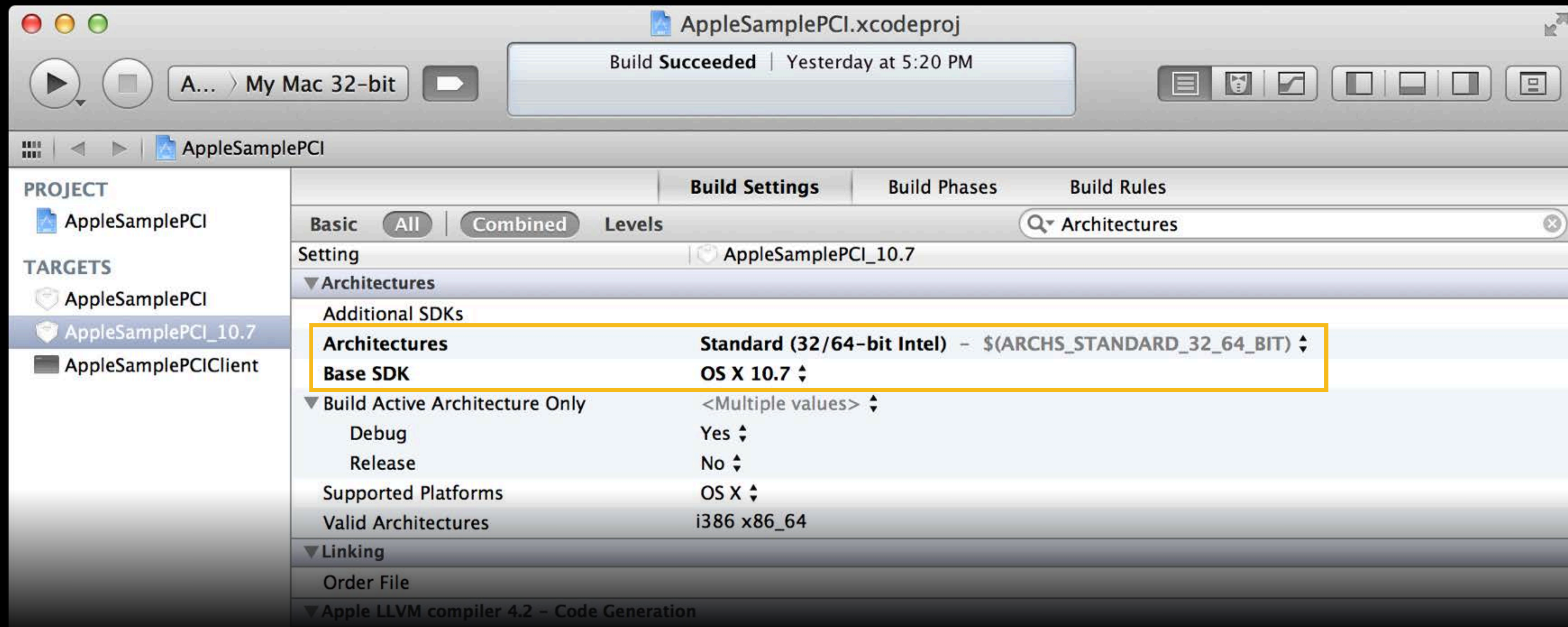
- In “Architectures” section
  - Set Architecture to “Standard (32/64-bit Intel)”
  - Set Base SDK to OS X 10.7



# Building a Kext for OS X 10.7 and Later

## Build settings (OS X 10.7 Target)

- In “Architectures” section
  - Set Architecture to “Standard (32/64-bit Intel)”
  - Set Base SDK to OS X 10.7

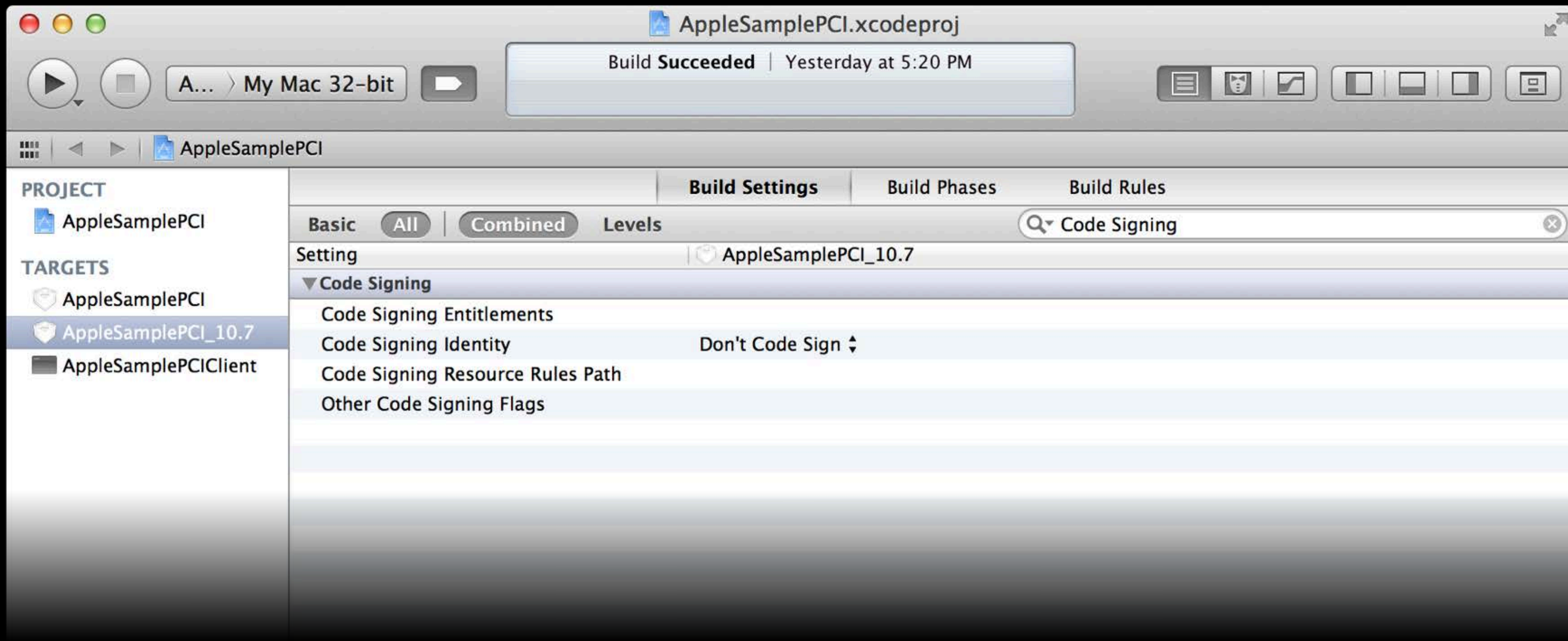




# Building a Kext for OS X 10.7 and Later

## Build settings (OS X 10.7 Target)

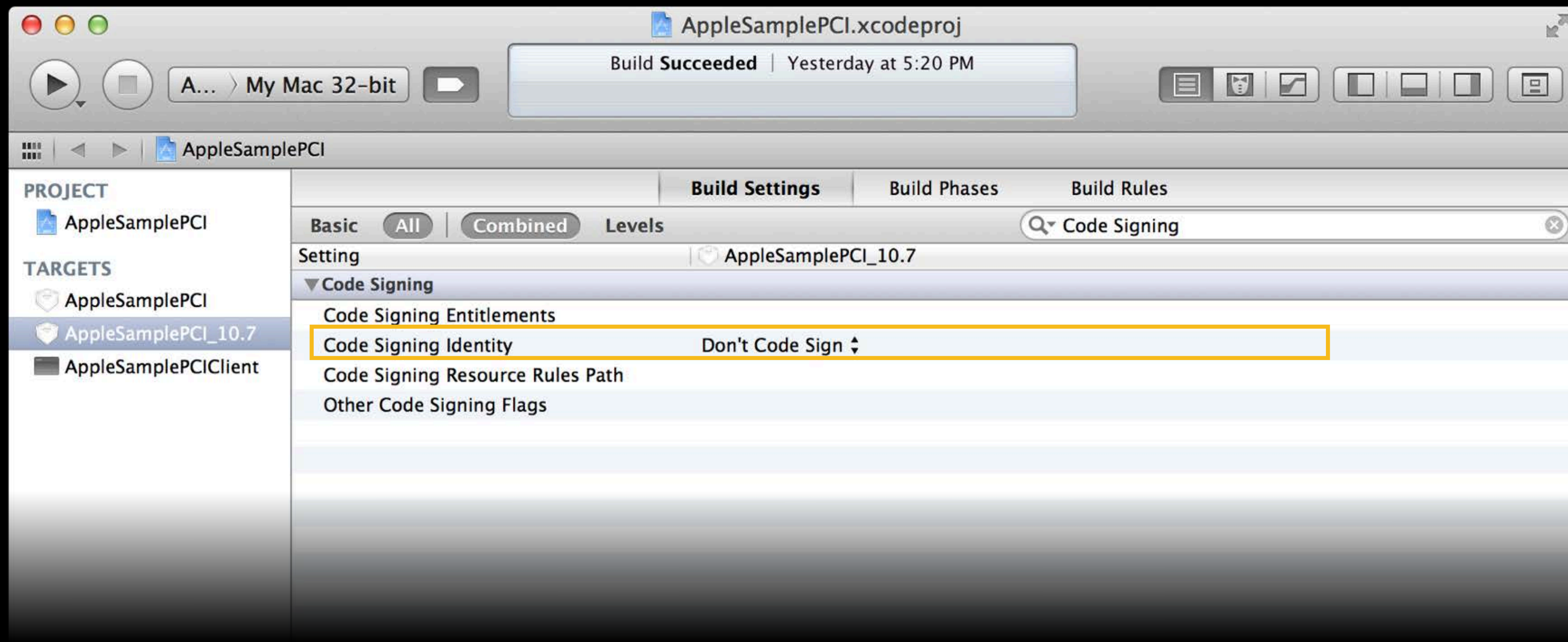
- In “Code Signing” section
  - Set “Code Signing Identity” to “Don’t Code Sign”



# Building a Kext for OS X 10.7 and Later

## Build settings (OS X 10.7 Target)

- In “Code Signing” section
  - Set “Code Signing Identity” to “Don’t Code Sign”

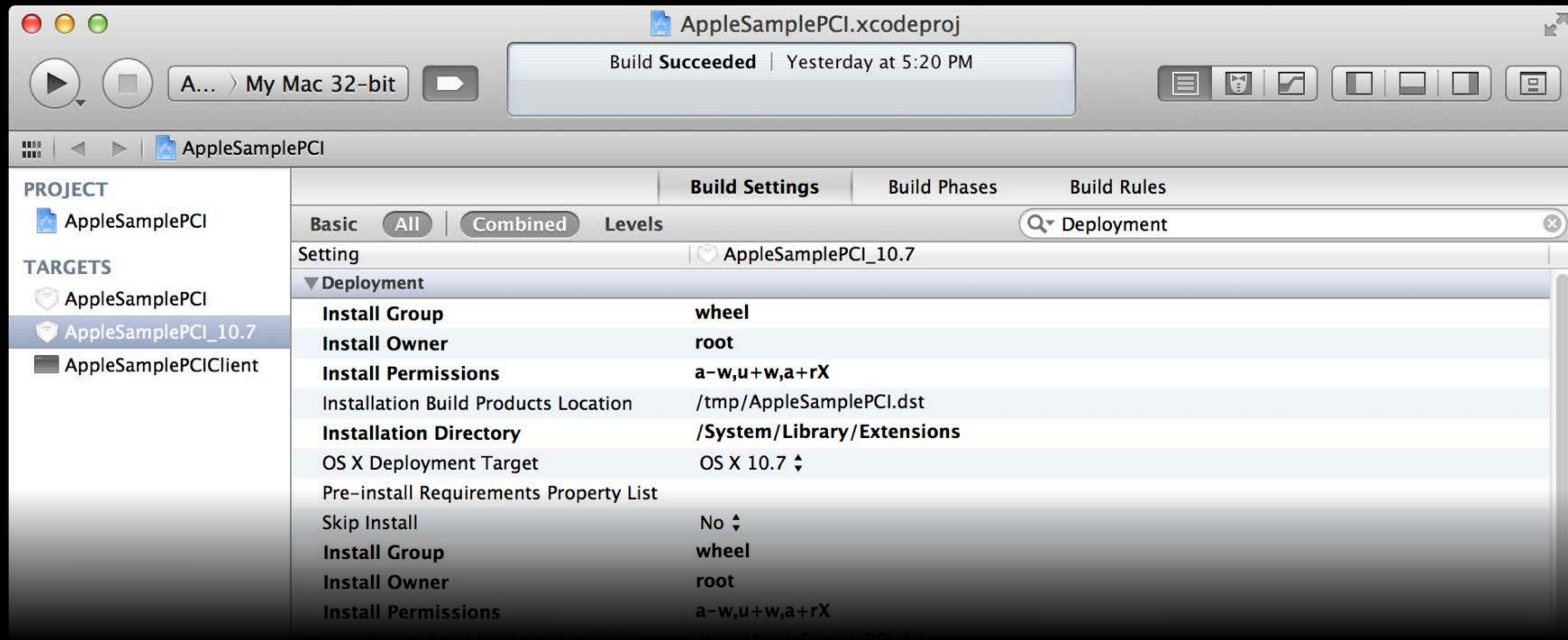




# Building a Kext for OS X 10.7 and Later

## Build settings (OS X 10.7 Target)

- In “Deployment” section
  - Set Installation Directory to “/System/Library/Extensions”
  - Set OS X Deployment Target to OS X 10.7

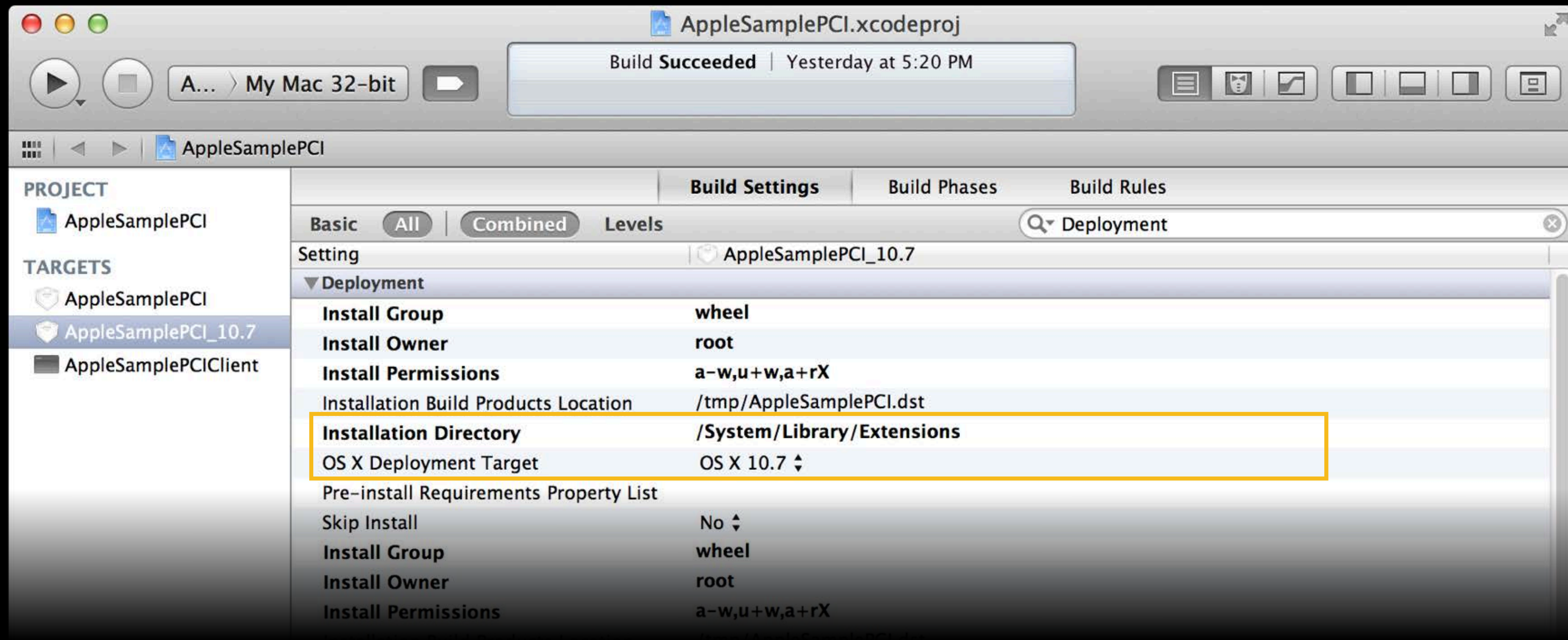




# Building a Kext for OS X 10.7 and Later

## Build settings (OS X 10.7 Target)

- In “Deployment” section
  - Set Installation Directory to “/System/Library/Extensions”
  - Set OS X Deployment Target to OS X 10.7



# Building a Kext for OS X 10.7 and Later

Build, test, and deploy

# Building a Kext for OS X 10.7 and Later

Build, test, and deploy

- Builds two kexts

# Building a Kext for OS X 10.7 and Later

## Build, test, and deploy

- Builds two kexts
- Signed kext
  - Should load on OS X 10.9 without warnings or alerts
  - Use lldb to debug
  - Verify installed into /Library/Extensions



# Building a Kext for OS X 10.7 and Later

## Build, test, and deploy

- Builds two kexts
- Signed kext
  - Should load on OS X 10.9 without warnings or alerts
  - Use lldb to debug
  - Verify installed into /Library/Extensions
- Unsigned kext
  - Should load on OS X 10.7 and OS X 10.8
  - Use gdb to debug
  - Verify installed into /System/Library/Extensions

# Building a Kext for OS X 10.7 and Later

## Build, test, and deploy

- Builds two kexts
- Signed kext
  - Should load on OS X 10.9 without warnings or alerts
  - Use lldb to debug
  - Verify installed into /Library/Extensions
- Unsigned kext
  - Should load on OS X 10.7 and OS X 10.8
  - Use gdb to debug
  - Verify installed into /System/Library/Extensions
- Consider Installer script to avoid installing OS X 10.8 kext on OS X 10.9

# Building Kexts for Older Releases

## Two projects

- If you need to target releases prior to OS X 10.7
  - Create two or more kext projects
  - Must use prior versions of Xcode for older SDKs
  - May need to build on multiple OS releases

# Building Kexts for Older Releases

## Two projects

- If you need to target releases prior to OS X 10.7
  - Create two or more kext projects
  - Must use prior versions of Xcode for older SDKs
  - May need to build on multiple OS releases
- OS X 10.9 kext project installs in `/Library/Extensions`

# Building Kexts for Older Releases

## Two projects

- If you need to target releases prior to OS X 10.7
  - Create two or more kext projects
  - Must use prior versions of Xcode for older SDKs
  - May need to build on multiple OS releases
- OS X 10.9 kext project installs in /Library/Extensions
- Target other OS releases with additional project(s)
  - Follow compatibility guidance from those releases
  - Install unsigned kext(s) in /System/Library/Extensions



# Building Kexts for Older Releases

## Two projects

- If you need to target releases prior to OS X 10.7
  - Create two or more kext projects
  - Must use prior versions of Xcode for older SDKs
  - May need to build on multiple OS releases
- OS X 10.9 kext project installs in /Library/Extensions
- Target other OS releases with additional project(s)
  - Follow compatibility guidance from those releases
  - Install unsigned kext(s) in /System/Library/Extensions
- Installer script can skip kexts for releases older than target OS volume

# Compatibility Summary

	To Support	OS X 10.7+	OS X 10.8+	OS X 10.9
	Development System	OS X 10.8.3+		
		Xcode 4.6	Xcode 4.6 or 5.0	
OS X 10.9 Target	SDK and Deployment	OS X 10.9		
	Code Sign	Developer ID		
	Install Path	/Library/Extensions		
Legacy Target	SDK and Deployment	OS X 10.7	OS X 10.8	n/a
	Code Sign	Don't Code Sign		n/a
	Install Path	/System/Library/Extensions		n/a

# lldb Kernel Debugging

Or, how I learned to love debugging and ignore the bomb

Session 707

**Brian Bechtel**

Core OS Panic Triage

These are confidential sessions—please refrain from streaming, blogging, or taking pictures



**Your first 10,000 panics  
are the hardest to debug.**

After that, it gets somewhat easier.



# lldb—A New, Powerful Debugger



- Integration with llvm and the rest of the development tool chain
  - More knowledge shared between compiler and debugger
- Much faster
- Better supported and actively being developed

# lldb—A New, Powerful Debugger



- Extremely powerful Python scripting language
  - Lets you do anything Python can do
- gdb to lldb command map  
<http://lldb.llvm.org/lldb-gdb.html>
- The debugger for OS X 10.9 and beyond

# gdb is Dead



- gdb is gone in Xcode 5.0
  - No gdb binaries
  - No kgmacros file in Kernel Debug Kit for OS X 10.9 or later
  - No plan to fix any issues remaining
- Still the debugger of choice for OS X 10.8 and earlier
  - Available as part of Xcode 4.6

# *Demo*

zprint in gdb versus zprint in lldb





```
(gdb) zprint  
  
I  
  
0.00
```

```
(lldb) zprint  
  
0.00
```

# Standard kgmacros Are Still There

Minor improvements in output format

paniclog

backtrace

showcurrentstacks

showalltasks

showallstacks

zombproc

zombstacks

zprint

showallkmods

showallvmstats

memstats

systemlog

showbootargs

# Adding Kext Symbols

- On startup lldb will scan kexts installed in
  - /System/Library/Extensions
  - /Library/Extensions
  - /Volumes/KernelDebugKit
  - Additional locations can be specified in ~/.lldbinit

```
settings set platform.plugin.darwin-kernel.kext-directories /tmp /a /b
```
- Add additional kext symbols using the `addkext` macro

# Watch Out For

- “backtick” expression for evaluation

```
x/i `myRoutine+0x18`
```

- lldb is pedantic about “->” versus “.”

```
p myStruct->myField
```

- May need to cast a variable very explicitly to display correctly

```
p *(struct myStruct*)myVariable
```

- File bugs

- <http://bugreporter.apple.com>

# Mixed Mode Disassembly

```
disassemble -f -m
```

- The entire current frame

```
disassemble -l -m
```

- The current line

```
disassemble -s {address} -e {address} -m
```

- From start address to end address
- In lldb minimum abbreviations are allowed



# Mixed Mode Disassembly

```
(lldb) di -n fsevents_f_write -m
mach_kernel`fsevents_f_write at vfs_fsevents.c:1537
 1536     __unused int flags, __unused vfs_context_t ctx)
 1537 {
 1538     return EIO;
0xffffffff800ae04810:  pushq   %rbp
0xffffffff800ae04811:  movq    %rsp, %rbp
0xffffffff800ae04814:  movl    $5, %eax
mach_kernel`fsevents_f_write + 9 at vfs_fsevents.c:1538
 1537 {
 1538     return EIO;
 1539 }
0xffffffff800ae04819:  popq    %rbp
0xffffffff800ae0481a:  ret
```

# Mixed Mode Disassembly

```
(lldb) di -n fsevents_write -m
```

```
mach_kernel`fsevents_write at vfs_fsevents.c:1537
```

```
1536     __unused int flags, __unused vfs_context_t ctx)
```

```
1537 {
```

```
1538     return EIO;
```

```
0xffffffff800ae04810: pushq  %rbp
```

```
0xffffffff800ae04811: movq   %rsp, %rbp
```

```
0xffffffff800ae04814: movl   $5, %eax
```

```
mach_kernel`fsevents_write + 9 at vfs_fsevents.c:1538
```

```
1537 {
```

```
1538     return EIO;
```

```
1539 }
```

```
0xffffffff800ae04819: popq   %rbp
```

```
0xffffffff800ae0481a: ret
```

# Mixed Mode Disassembly

```
(lldb) di -n fsevents_write -m
```

```
mach_kernel`fsevents_write at vfs_fsevents.c:1537
```

```
1536     __unused int flags, __unused vfs_context_t ctx)
```

```
1537 {
```

```
1538     return EIO;
```

```
0xffffffff800ae04810: pushq %rbp
```

```
0xffffffff800ae04811: movq %rsp, %rbp
```

```
0xffffffff800ae04814: movl $5, %eax
```

```
mach_kernel`fsevents_write + 9 at vfs_fsevents.c:1538
```

```
1537 {
```

```
1538     return EIO;
```

```
1539 }
```

```
0xffffffff800ae04819: popq %rbp
```

```
0xffffffff800ae0481a: ret
```

# Mixed Mode Disassembly

```
(lldb) di -n fsevents_write -m
```

```
mach_kernel`fsevents_write at vfs_fsevents.c:1537
```

```
1536     __unused int flags, __unused vfs_context_t ctx)
```

```
1537 {
```

```
1538     return EIO;
```

```
0xffffffff800ae04810:  pushq  %rbp
```

```
0xffffffff800ae04811:  movq   %rsp, %rbp
```

```
0xffffffff800ae04814:  movl   $5, %eax
```

```
mach_kernel`fsevents_write + 9 at vfs_fsevents.c:1538
```

```
1537 {
```

```
1538     return EIO;
```

```
1539 }
```

```
0xffffffff800ae04819:  popq   %rbp
```

```
0xffffffff800ae0481a:  ret
```

# Mixed Mode Disassembly

```
(lldb) di -n fsevents_write -m
mach_kernel`fsevents_write at vfs_fsevents.c:1537
 1536     __unused int flags, __unused vfs_context_t ctx)
 1537 {
 1538     return EIO;
0xffffffff800ae04810:  pushq   %rbp
0xffffffff800ae04811:  movq    %rsp, %rbp
0xffffffff800ae04814:  movl    $5, %eax
mach_kernel`fsevents_write + 9 at vfs_fsevents.c:1538
 1537 {
 1538     return EIO;
 1539 }
0xffffffff800ae04819:  popq    %rbp
0xffffffff800ae0481a:  ret
```



# lldb kgmacros Additional Arguments

```
-o /tmp/foo          # output to file /tmp/foo  
-s search_string    # grep for search_string in output  
-v                  # sets verbosity +1  
-v -v -v            # increases verbosity +3
```

- Example, to find load information for just “AppleSamplePCI.kext”

```
showallkmods -s AppleSamplePCI
```

# lldbmacros Precautions

- lldbmacros do not automatically load in lldb
  - prevents you from debugging a malicious application and getting evil lldb macros from a hidden symbol file
- To overcome this, add to your ~/.lldbinit file

```
settings set target.load-script-from-symbol-file true
```
- Add custom python to Resources/Python in your kext bundle

```
AppleSamplePCI.kext.dSYM/Contents/Resources/Python/
```

# Faster Loading, No Symbols

- To speed startup and not load all kext symbols, use  
`settings set plugin.dynamic-loader.darwin-kernel.load-kexts false`
- To load your kext symbols after connecting, the commands are  
`addkext <uuid> : Load one kext based on uuid`  
`addkext -N <name> : Load one kext that matches the name provided`  
`addkext -F <abs/path/to/executable> <address>`  
`addkext all : Will load all the kext symbols - SLOW`

# Use nvram to Enable Debugging

- With no debug boot-args set, system automatically reboots on panic
  - OS X 10.8 or later
- Set boot-args using the nvram command line tool as root
  - Must reboot to let settings take effect
- **debug** flags are a bit map of behaviors
  - See table 20-1 of “Kernel Programming Guide”
  - E.g. 0x4 is **DB\_NMI** (drop into debugger on NMI)

# Ways to Connect

- Built-in Ethernet
- Built-in Firewire (e.g. iMac)
- Thunderbolt->Ethernet adapter
- Thunderbolt->Firewire adapter
- Apple Thunderbolt Display -> Ethernet
- Apple Thunderbolt Display -> Firewire
- No wireless, no USB



# kdp\_match\_name

- Two machine debugging assumes “network” port of en0
- If not, add “**kdp\_match\_name**=`{something}`” to boot-args
  - `{something}` depends upon your connection
- **ifconfig** will tell you appropriate ports

# Two Machine Debugging

- Use `ifconfig` to find appropriate port name

```
en1: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
options=2b<RXCSUM,TXCSUM,VLAN_HWTAGGING,TS04>
ether 3c:07:54:27:8d:0b
inet6 fe80::3e07:54ff:fe27:8d0b%en1 prefixlen 64 scopeid 0x4
inet6 2620:149:4:202:3e07:54ff:fe27:8d0b prefixlen 64 autoconf
inet6 2620:149:4:202:68ca:1665:6165:9944 prefixlen 64 autoconf temporary
inet 10.0.40.2 netmask 0xfffffc00 broadcast 10.0.255.255
media: autoselect (1000baseT <full-duplex,flow-control>)
status: active
```

# Two Machine Debugging

- Built-in Ethernet

```
sudo nvram boot-args="debug=0x144"
```

# Two Machine Debugging

- Thunderbolt -> Ethernet

```
sudo nvram boot-args="debug=0x144 kdp_match_name=en2"
```

- Apple Thunderbolt Display -> Ethernet

```
sudo nvram boot-args="debug=0x144 kdp_match_name=en4"
```

- Use port from `ifconfig` instead of en4 or en2

# Two Machine Debugging

- Native Firewire

```
sudo nvram boot-args="debug=0x144 kdp_match_name=firewire"
```

- Thunderbolt -> Firewire

```
sudo nvram boot-args="debug=0x144 kdp_match_name=firewire fwkdp=0x8000"
```

- Apple Thunderbolt Display -> Firewire

```
sudo nvram boot-args="debug=0x144 kdp_match_name=firewire fwkdp=0x8000"
```

- Always reboot to let settings take effect



# Firewire Debugging

- In one terminal window

```
fwkdp
```

- In a second terminal window

```
xcrun lldb /Volumes/KernelDebugKit/mach_kernel  
kdp-remote localhost
```

# Connecting

```
xcrun lldb /Volumes/KernelDebugKit/mach_kernel
```

- Just invoking lldb might grab some horrible out of date monstrosity
- Avoid confusion; use xcrun

## • Live

```
kdp-remote {name or ip}
```

```
kdp-remote 10.0.1.25
```

```
kdp-remote mycomputer.local
```

```
kdp-remote mycomputer.mydomain.com
```

## • CoreDump

```
file --core {/path/to/file}
```

```
file -c /PanicDumps/core-xnu-2050.22.13-10.0.1.25-47b8de62
```

# Miscellaneous Notes

- We've changed NMI
- Core dumps how-to
- Zone corruption boot-args
- Console output and graphics

# We've Changed NMI in OS X 10.9

- With boot-args set so that debug has the 0x4 bit set, e.g.  
`sudo nvram boot-args="debug=0x144"`
- In OS X 10.8 or earlier, NMI was Power key alone
- Now it is  
`Left-⌘ + Right-⌘ + Power`
- You can revert to the old behavior by or'ing in the 0x8000 bit  
`sudo nvram boot-args="debug=0x8144"`

# Automatically Save CoreDumps

- Host (e.g. 10.0.40.2)

```
sudo mkdir /PanicDumps  
sudo chown root:wheel /PanicDumps  
sudo chmod 1777 /PanicDumps  
sudo launchctl load -w /System/Library/LaunchDaemons/com.apple.kdumpd.plist
```

- Client (panicking machine)

```
sudo nvram boot-args="debug=0xd44 _panicd_ip=10.0.40.2"  
sudo reboot
```



# Zone Protection and Corruption Checks

- Buffer overruns, use-after-free errors, zone-mismatched frees, and timing induced races

-zc

- zalloc code checks the free list pointers of all zones for correctness

-zp

- zfree routine overwrites freed memory with 0xdeadbeefdeadbeef
  - Catches attempts to execute or access freed data

```
sudo nvram boot-args="debug=0x144 -zp -zc"
```

# Guard Mode Kernel Zone Allocator

`gmalloc_size=<size>`

- Target all zones with elements of `<size>` bytes

`gmalloc_min=<size>`    `gmalloc_max=<size>`

- Target zones with elements  $\geq$  size or  $\leq$  size

`-gmalloc_wp`

- Write protect, rather than unmap, freed allocations
- `-gmalloc_wp` requires `gmalloc_size` or `gmalloc_min/max`

# Performance Considerations

- -zp and -zc add additional runtime checks
  - Little additional memory usage
- -gmalloc\_wp gmalloc\_size={size} takes much more memory
  - Each allocation is a 4k page with additional guard page
  - Measurably slower, but catches more bugs

# Console Output and Graphics

- Graphics changed to support multiple buffers
  - OS X 10.8 and later
  - Draw to a buffer instead of directly to the screen
  - Enhancement for retina display machines
- Unreliable console output to the screen if you panic

# More Information

## Paul Danbold

Core OS Evangelist  
danbold@apple.com

## Documentation

Understanding and debugging kernel panics

[http://developer.apple.com/library/mac/#technotes/tn2063/\\_index.html](http://developer.apple.com/library/mac/#technotes/tn2063/_index.html)

Kernel Core Dumps

<http://developer.apple.com/library/mac/#technotes/tn2004/tn2118.html>

Generating a Non-Maskable Interrupt (NMI)

[http://developer.apple.com/library/mac/#qa/qa1264/\\_index.html](http://developer.apple.com/library/mac/#qa/qa1264/_index.html)

WWDC 2012 Session 415: "Debugging with lldb"

## Apple Developer Forums

<http://devforums.apple.com>



# Related Sessions

Advanced Debugging with LLDB

Pacific Heights  
Friday 9:00AM



# Labs

OS X Kernel Lab	Core OS Lab B Wednesday 2:00PM	
OS X Kernel Lab	Core OS Lab B Friday 10:15AM	

# Summary

- Sign your kext in OS X 10.9
- lldb is faster and more powerful
  - Still have the kgmacros you know and love
- New ways of two machine debugging via Thunderbolt adapters
- New boot-args to track down bugs

 WWDC2013