Security Interface Framework Reference

Security > Cocoa

2006-07-14

Apple Inc.
1 Infinite Loop
Cupertino, CA 95014
408-996-1010

Simultaneously published in the United States and Canada.

Contents

Figures

Introduction

Framework	/System/Library/Frameworks/SecurityInterface.framework
Header file directories	/System/Library/Frameworks/SecurityInterface.framework/Headers
Declared in	SFAuthorizationPluginView.h SFAuthorizationView.h SFCertificatePanel.h SFCertificateTrustPanel.h SFCertificateView.h SFChooseIdentityPanel.h SFKeychainSavePanel.h SFKeychainSettingsPanel.h

Note: This document was previously titled *Security Objective-C API*. The documentation for the SFAuthorization class is now in a separate document, *Security Foundation Framework Reference*.

The Security Interface framework is a set of Objective-C classes that provide user interface elements for programs that implement security features such as authorization, access to digital certificates, and access to items in keychains.

Introduction

Classes

SFAuthorizationPluginView Class Reference

Inherits from	NSObject
Conforms to	NSObject (NSObject)
Framework	/System/Library/Frameworks/SecurityInterface.framework
Declared in	SFAuthorizationPluginView.h
Availability	Available in Mac OS X v10.5 and later
Companion guide	Authorization Services Programming Guide
Related sample code	NameAndPassword

Overview

The `SFAuthorizationPluginView` class allows authorization plug-in developers to create a custom view their plug-in can display.

If you're developing an authorization plug-in, you can subclass the `SFAuthorizationPluginView` class to create views that provide a custom user interface for your plug-in. By subclassing the `SFAuthorizationPluginView` class, you avoid changing or duplicating the Apple-provided authentication or login window dialogs to display your custom view.

To instantiate your `SFAuthorizationPluginView` subclass, you need the callbacks structure containing entry points to the Security Server that you receive in your plug-in's `AuthorizationPluginCreate` function and the authorization engine handle you receive in your plug-in's `MechanismCreate` function.

Your custom subclass of `SFAuthorizationPluginView` must override the following methods:

`buttonPressed:` (page 13)
`viewForType:` (page 18)

Tasks

Initializing an SFAuthorizationPluginView Object

- `initWithCallbacks:andEngineRef` (page 16)
  Returns an `SFAuthorizationPluginView` object with the specified callbacks and authorization engine handle.

Getting Instance Information

- `callbacks` (page 14)
  Returns the `AuthorizationCallbacks` structure with which this instance was initialized.
- `engineRef` (page 15)
  Returns the authorization engine handle with which this instance was initialized.

Responding to User Actions

- `buttonPressed:` (page 13)
  Informs the `SFAuthorizationPluginView` instance when a user presses a button in the custom view.
- `viewForType:` (page 18)
  Returns the appropriate `NSView` object for the specified `SFViewType` (page 19).

Configuring the User Interface

- `didActivate` (page 14)
  Informs the `SFAuthorizationPluginView` instance when the authorization plug-in makes the instance's user interface active.
- `didDeactivate` (page 14)
  Informs the `SFAuthorizationPluginView` instance when the authorization plug-in deactivates its user interface.
- `willActivateWithUser:` (page 18)
  Informs the `SFAuthorizationPluginView` instance when its user interface is about to be made active by the Apple-provided Security Agent.

Setting Up the Keyboard Loop

- `firstKeyView` (page 15)
  Returns the first view in the keyboard loop of the view.
- `firstResponderView` (page 16)
  Returns the view that should get focus for keyboard events.

- `lastKeyView` (page 16)
  Returns the last view in the keyboard loop of the view.

Enabling and Disabling Controls

- `setEnabled:` (page 17)
  Enables or disables the controls in the `SFAuthorizationPluginView` instance's view.

Communicating with the Authorization Plug-in

- `displayView` (page 14)
  Displays the user interface provided by the `SFAuthorizationPluginView` subclass.
- `setButton:enabled:` (page 17)
  Enables or disables a button in the `SFAuthorizationPluginView` instance's user interface.
- `updateView` (page 17)
  Tells the authorization plug-in to get and display the appropriate view in the `SFAuthorizationPluginView` instance's user interface.

Instance Methods

buttonPressed:

Informs the `SFAuthorizationPluginView` instance when a user presses a button in the custom view.

```
- (void)buttonPressed:(SFButtonType)inButtonType
```

Parameters

inButtonType
  The type of button that was pressed.

Discussion

By default, `buttonPressed:` will set a result of Deny when the OK or Login buttons are pressed. An `SFAuthorizationPluginView` subclass needs to override this method to set the context values for the short name of the user so that user attributes can be looked up. To do this, use `kAuthorizationEnvironmentUsername` as the key. A subclass should also set any additional context values that are needed by the authorization plug-in to verify the user's credentials. To do this, use the appropriate function pointers you receive from `callbacks` (page 14).

When you override this method, do not call `[super buttonPressed]`.

Availability

Available in Mac OS X v10.5 and later.

Declared In

`SFAuthorizationPluginView.h`

callbacks

Returns the `AuthorizationCallbacks` structure with which this instance was initialized.

```
- (const AuthorizationCallbacks *)callbacks
```

Return Value
An object of type `AuthorizationCallbacks`.

Discussion
Use the `AuthorizationCallbacks` structure to get the function pointers to functions such as `SetResult` and `SetContextValue`.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`SFAuthorizationPluginView.h`

didActivate

Informs the `SFAuthorizationPluginView` instance when the authorization plug-in makes the instance's user interface active.

```
- (void)didActivate
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`SFAuthorizationPluginView.h`

didDeactivate

Informs the `SFAuthorizationPluginView` instance when the authorization plug-in deactivates its user interface.

```
- (void)didDeactivate
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`SFAuthorizationPluginView.h`

displayView

Displays the user interface provided by the `SFAuthorizationPluginView` subclass.

```
- (void)displayView
```

Discussion
It's not likely that you will want to override this method, but if you do, be sure to call `[super displayView]`. If you don't call `[super displayView]`, your custom view will not get displayed.

This method will raise an `SFDisplayViewException` exception if an error occurs while displaying the authorization dialog.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`SFAuthorizationPluginView.h`

engineRef

Returns the authorization engine handle with which this instance was initialized.

```
- (AuthorizationEngineRef)engineRef
```

Return Value
A handle of type `AuthorizationEngineRef`.

Discussion
Use the authorization engine handle when you call the functions in the `AuthorizationCallbacks` structure to set a result or a context value.

Availability
Available in Mac OS X v10.5 and later.

Related Sample Code
NameAndPassword

Declared In
`SFAuthorizationPluginView.h`

firstKeyView

Returns the first view in the keyboard loop of the view.

```
- (NSView *)firstKeyView
```

Discussion
The default return value of this method is `nil`. When the authorization plug-in calls this method, your subclass should return the first view in the keyboard loop of your custom `NSView` object.

Availability
Available in Mac OS X v10.5 and later.

Related Sample Code
NameAndPassword

Declared In
`SFAuthorizationPluginView.h`

firstResponderView

Returns the view that should get focus for keyboard events.

```
- (NSView *)firstResponderView
```

Discussion
The default return value of this method is `nil`. When the authorization plug-in calls this method, your subclass should return the view that should get the focus for keyboard events.

Availability
Available in Mac OS X v10.5 and later.

initWithCallbacks:andEngineRef

Returns an `SFAuthorizationPluginView` object with the specified callbacks and authorization engine handle.

```
- (id)initWithCallbacks:(const AuthorizationCallbacks *)callbacks
    andEngineRef:(AuthorizationEngineRef)engineRef
```

Parameters

`callbacks`

The structure of type `AuthorizationCallbacks` provided to the authorization plug-in in its `AuthorizationPluginCreate` function.

`engineRef`

The handle of type `AuthorizationEngineRef` provided to the authorization plug-in in its `MechanismCreate` function.

Return Value
An initialized `SFAuthorizationPluginView` instance.

Availability
Available in Mac OS X v10.5 and later.

lastKeyView

Returns the last view in the keyboard loop of the view.

```
- (NSView *)lastKeyView
```

Discussion
The default return value of this method is `nil`. When the authorization plug-in calls this method, your subclass should return the last view in the keyboard loop of your custom `NSView` object.

Availability
Available in Mac OS X v10.5 and later.

Related Sample Code
NameAndPassword

Declared In
`SFAuthorizationPluginView.h`

setButton:enabled:

Enables or disables a button in the `SFAuthorizationPluginView` instance's user interface.

```
- (void)setButton:(SFButtonType)inButtonType enabled:(BOOL)inEnabled
```

Parameters

`inButtonType`

The type of the button.

`inEnabled`

`YES` to enable the button, `NO` to disable the button.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`SFAuthorizationPluginView.h`

setEnabled:

Enables or disables the controls in the `SFAuthorizationPluginView` instance's view.

```
- (void)setEnabled:(BOOL)inEnabled
```

Parameters

`inEnabled`

The state the controls should be in.

Discussion
When the authorization plug-in calls this method, the subclass should call `setEnabled:` on the controls that are in its view.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`SFAuthorizationPluginView.h`

updateView

Tells the authorization plug-in to get and display the appropriate view in the `SFAuthorizationPluginView` instance's user interface.

```
- (void)updateView
```

Discussion
Your subclass of `SFAuthorizationPluginView` should call this method when a user clicks a button in your view that should result in a new view being displayed. Calling this method causes the authorization plug-in to get the new view and display it.

Availability
Available in Mac OS X v10.5 and later.

Declared In
SFAuthorizationPluginView.h

viewForType:

Returns the appropriate NSView object for the specified SFViewType (page 19).

```
- (NSView *)viewForType:(SFViewType)inType
```

Parameters

inType

The type of view being requested by the authorization plug-in.

Return Value
An NSView object representing either a credentials view or an identity and credentials view.

Discussion
When the authorization plug-in calls this method, the SFAuthorizationPluginView instance should return the NSView object that represents the view indicated by the specified SFViewType (page 19). The NSView object and its contents should have the autoresize flags set to allow the view to be resized.

Note that although a maximum width of 394 points is currently supported, this may change in the future. You should not assume that the width of the NSView object will never change.

Availability
Available in Mac OS X v10.5 and later.

Declared In
SFAuthorizationPluginView.h

willActivateWithUser:

Informs the SFAuthorizationPluginView instance when its user interface is about to be made active by the Apple-provided Security Agent.

```
- (void)willActivateWithUser:(NSDictionary *)inUserInformation
```

Parameters

inUserInformation

A dictionary that contains the following information:

kSFAuthorizationPluginViewUserNameKey
An NSString object containing the selected user's name

kSFAuthorizationPluginViewUserShortNameKey
An NSString object containing the selected user's short name

Note: inUserInformation may be nil.

Discussion
Your SFAuthorizationPluginView instance can use the user name to pre-populate a text field in the user interface.

Availability
Available in Mac OS X v10.5 and later.

Declared In
SFAuthorizationPluginView.h

Constants

SFButtonType

These constants define the button types used by authorization plug-ins.

```
typedef enum{
    SFButtonTypeCancel    = NSCancelButton,
    SFButtonTypeOK        = NSOKButton,
    SFButtonTypeBack      = SFButtonTypeCancel,
    SFButtonTypeLogin     = SFButtonTypeOK
} SFButtonType;
```

Constants

SFButtonTypeCancel
Indicates the Cancel button was pressed.
Available in Mac OS X v10.5 and later.
Declared in SFAuthorizationPluginView.h.

SFButtonTypeOK
Indicates the OK button was pressed.
Available in Mac OS X v10.5 and later.
Declared in SFAuthorizationPluginView.h.

SFButtonTypeBack
Indicates the Back button was pressed.
Available in Mac OS X v10.5 and later.
Declared in SFAuthorizationPluginView.h.

SFButtonTypeLogin
Indicates the Login button was pressed.
Available in Mac OS X v10.5 and later.
Declared in SFAuthorizationPluginView.h.

Availability
Available in Mac OS X v10.5 and later.

Declared In
SFAuthorizationPluginView.h

SFViewType

These constants define the view type requested by the authorization plug-in.

```
typedef enum {
    SFViewTypeIdentityAndCredentials,
    SFViewTypeCredentials
} SFViewType;
```

Constants

`SFViewTypeIdentityAndCredentials`

Indicates a view that contains controls for identity and credentials was requested by the authorization plug-in.

Available in Mac OS X v10.5 and later.

Declared in `SFAuthorizationPluginView.h`.

`SFViewTypeCredentials`

Indicates a view that contains controls for credentials was requested by the authorization plug-in.

Available in Mac OS X v10.5 and later.

Declared in `SFAuthorizationPluginView.h`.

Availability

Available in Mac OS X v10.5 and later.

Declared In

`SFAuthorizationPluginView.h`

SFAuthorizationView Class Reference

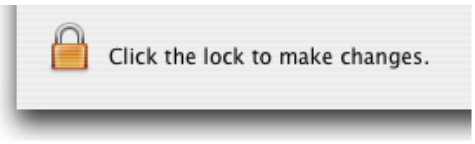
Inherits from	NSView : NSResponder : NSObject
Conforms to	NSAnimatablePropertyContainer (NSView) NSCoding (NSResponder) NSObject (NSObject)
Framework	/System/Library/Frameworks/SecurityInterface.framework
Declared in	SFAuthorizationView.h
Availability	Available in Mac OS X v10.3 and later
Companion guide	Authorization Services Programming Guide

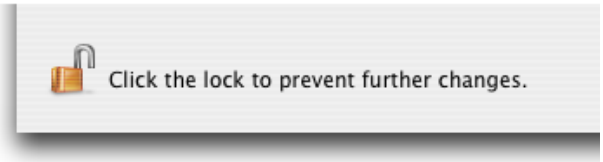
Overview

The `SFAuthorizationView` class displays a lock icon that can be used as a visual indication that a user interface has restricted access.

The lock appears locked when the user must be authorized and appears open when the user has been authorized. The closed and open lock icons of the authorization view are shown in the following figure.

Figure 2-1 Authorization view lock icon





When you add an authorization view as a custom view to a window or dialog box, you must initialize it before it displays correctly. To initialize the view, use the `setString:` (page 29) method to create a default rights structure (containing a prompt string) or the `setAuthorizationRights:` (page 26) method to specify a rights structure. You must also either specify automatic updates (`setAutoupdate:` (page 27) or `setAutoupdate:interval:` (page 27)) or perform a manual update (`updateStatus:` (page 29)) to set the lock icon to its initial state.

You can implement delegate methods that are invoked when the authorization view changes state. You can optionally implement the delegate methods to obtain the state of the authorization object when you are using an authorization view.

When the user clicks a locked authorization view icon, the Security Server displays an authentication dialog (to request a user name and password, for example). When the user provides the requested credentials, the lock icon unlocks and the user is considered preauthorized to perform the functions specified by the authorization rights structure. You can call the `updateStatus:` (page 29) method to determine whether the user has been preauthorized: this method returns `YES` if the view is in the unlocked state, `NO` otherwise. Before committing changes or performing actions that require authorization, you should check the user's authorization again, even if they are preauthorized.

The default behavior of this view is to preauthorize rights; if this is not possible it unlocks and waits for authorization to be checked when explicitly required.

Tasks

Setting Up the Authorization View

- `setString:` (page 29)
  Sets the requested-right string to use with the default authorization rights set.
- `setAuthorizationRights:` (page 26)
  Sets the authorization rights for this view.
- `setAutoupdate:` (page 27)
  Sets the authorization view to update itself automatically.
- `setAutoupdate:interval:` (page 27)
  Sets the authorization view to update itself at a specific interval.
- `setFlags:` (page 28)
  Sets the current authorization flags for the view.
- `setEnabled:` (page 28)
  Sets the current state of the authorization view.

Setting and Getting the Delegate for the View

- `setDelegate:` (page 28)
  Sets the delegate for this authorization view.
- `delegate` (page 25)
  Returns the delegate for this view.

Updating the View

- `updateStatus:` (page 29)
  Manually updates the authorization view.

Getting Information About the Authorization View

- `authorization` (page 24)
  Returns the authorization object associated with this view.
- `authorizationRights` (page 24)
  Returns the authorization rights for this view.
- `authorizationState` (page 24)
  Returns the current state of the authorization view.
- `isEnabled` (page 26)
  Indicates whether the authorization view is enabled (`YES`) or disabled (`NO`).

Setting the Authorization State

- `authorize:` (page 25)
  Attempts to unlock the lock icon in the view.
- `deauthorize:` (page 25)
  Sets the authorization state to unauthorized and locks the lock icon in the view.

Actions performed

- `authorizationViewCreatedAuthorization:` (page 30) *delegate method*
  Sent to the delegate to indicate the authorization object has been created or changed. If you have saved a copy of the authorization object for your own purposes, you should discard it and call `authorization` (page 24) for a new authorization object.
- `authorizationViewDidAuthorize:` (page 30) *delegate method*
  Sent to the delegate to indicate the user was authorized and the authorization view was changed to unlocked.
- `authorizationViewDidDeauthorize:` (page 31) *delegate method*
  Sent to the delegate to indicate the user was deauthorized and the authorization view was changed to locked.
- `authorizationViewReleasedAuthorization:` (page 31) *delegate method*
  Sent to the delegate to indicate that deauthorization is about to occur.

Action about to be performed

- `authorizationViewShouldDeauthorize:` (page 31) *delegate method*
  Sent to the delegate when a user clicks the open lock icon.

Instance Methods

authorization

Returns the authorization object associated with this view.

```
- (SFAuthorization *)authorization
```

Discussion
The authorization object is defined in *Security Foundation Framework Reference*.

Availability
Available in Mac OS X v10.3 and later.

Declared In
`SFAuthorizationView.h`

authorizationRights

Returns the authorization rights for this view.

```
- (AuthorizationRights *)authorizationRights
```

Availability
Available in Mac OS X v10.3 and later.

See Also
- `setAuthorizationRights:` (page 26)
- `setString:` (page 29)

Declared In
`SFAuthorizationView.h`

authorizationState

Returns the current state of the authorization view.

```
- (SFAuthorizationViewState)authorizationState
```

Availability
Available in Mac OS X v10.3 and later.

See Also
- `authorize:` (page 25)
- `deauthorize:` (page 25)

Declared In
`SFAuthorizationView.h`

authorize:

Attempts to unlock the lock icon in the view.

```
- (BOOL)authorize:(id)inSender
```

Parameters

inSender

The authorization view to unlock.

Discussion

This method has the same behavior as if the user clicked on the lock icon; if the user is authorized, the lock icon unlocks. If this method succeeds, it returns `YES`; if it fails, the lock icon remains locked and the method returns `NO`.

Availability

Available in Mac OS X v10.3 and later.

See Also

- `deauthorize:` (page 25)
- `authorizationState` (page 24)

Declared In

`SFAuthorizationView.h`

deauthorize:

Sets the authorization state to unauthorized and locks the lock icon in the view.

```
- (BOOL)deauthorize:(id)inSender
```

Parameters

inSender

The authorization view to lock.

Discussion

If this method succeeds, it returns `YES`; if it fails, the lock icon remains unlocked and the method returns `NO`.

Availability

Available in Mac OS X v10.3 and later.

See Also

- `authorize:` (page 25)

Declared In

`SFAuthorizationView.h`

delegate

Returns the delegate for this view.

```
- (id)delegate
```

Availability
Available in Mac OS X v10.3 and later.

See Also
- `setDelegate:` (page 28)

Declared In
`SFAuthorizationView.h`

isEnabled

Indicates whether the authorization view is enabled (`YES`) or disabled (`NO`).

```
- (BOOL)isEnabled
```

Availability
Available in Mac OS X v10.3 and later.

See Also
- `setEnabled:` (page 28)

Declared In
`SFAuthorizationView.h`

setAuthorizationRights:

Sets the authorization rights for this view.

```
- (void)setAuthorizationRights:(const AuthorizationRights *)authorizationRights
```

Parameters

`authorizationRights`

An authorization rights structure specifying the authorization rights represented by the authorization view.

Discussion
Either this method or the `setString:` method must be called before the view displays correctly.

The authorization rights structures are defined in `AuthorizationRights` in *Authorization Services C Reference*.

Availability
Available in Mac OS X v10.3 and later.

See Also
- `authorizationRights` (page 24)
- `setString:` (page 29)

Declared In
`SFAuthorizationView.h`

setAutoupdate:

Sets the authorization view to update itself automatically.

```
- (void)setAutoupdate:(BOOL)autoupdate
```

Parameters

autoupdate

Specifies whether the authorization view should update itself automatically. Set to `YES` to enable autoupdates.

Discussion

If autoupdates are enabled and the authorization times out (for example), the authorization view automatically relocks. If autoupdates are disabled, you have to call the `updateStatus:` (page 29) method to manually update the view if the status changes when the user has not clicked on the lock icon. Autoupdates are disabled by default. Because autoupdates poll, they can affect system performance.

Availability

Available in Mac OS X v10.3 and later.

See Also

- `setAutoupdate:interval:` (page 27)
- `updateStatus:` (page 29)

Declared In

`SFAuthorizationView.h`

setAutoupdate:interval:

Sets the authorization view to update itself at a specific interval.

```
- (void)setAutoupdate:(BOOL)autoupdate interval:(NSTimeInterval)interval
```

Parameters

autoupdate

Specifies whether the authorization view should update itself automatically. Set to `YES` to enable autoupdates.

interval

If `autoupdate` is `YES`, sets the interval at which updates take place, in seconds.

Discussion

If autoupdates are enabled and the authorization times out (for example), the authorization view automatically relocks. If autoupdates are disabled, you have to call the `updateStatus:` (page 29) method to manually update the view if the status changes when the user has not clicked on the lock icon. Autoupdates are disabled by default. Because autoupdates poll, they can affect system performance. For that reason, you might want to set a time interval so that the polling does not take place as often.

Availability

Available in Mac OS X v10.3 and later.

See Also

- `setAutoupdate:` (page 27)

Declared In
SFAuthorizationView.h

setDelegate:

Sets the delegate for this authorization view.

```
- (void)setDelegate:(id)delegate
```

Parameters

delegate
The object to which messages about the state of the authorization object should be sent.

Discussion
If you want to be notified of state changes (for example, when the user clicks the button), set a delegate and implement the delegate methods described in the delegate methods section.

Availability
Available in Mac OS X v10.3 and later.

See Also
- delegate (page 25).

Declared In
SFAuthorizationView.h

setEnabled:

Sets the current state of the authorization view.

```
- (void)setEnabled:(BOOL)enabled
```

Parameters

enabled
Specifies whether the authorization view should be enabled (YES) or disabled (NO).

Discussion
A disabled view is visible but dimmed.

Availability
Available in Mac OS X v10.3 and later.

See Also
- authorizationState (page 24)

Declared In
SFAuthorizationView.h

setFlags:

Sets the current authorization flags for the view.

```
- (void)setFlags:(AuthorizationFlags)flags
```

Parameters

flags

The authorization flags to set for this view.

Discussion

You can use this method to change the authorization flag settings made with the `setAuthorizationRights:` method or to specify flags other than the default (`kAuthorizationFlagDefaults`) used by the `setString:` method.

The authorization flags are described in Authorization Options in *Authorization Services C Reference*.

Availability

Available in Mac OS X v10.3 and later.

See Also

- `setAuthorizationRights:` (page 26)
- `setString:` (page 29)

Declared In

`SFAuthorizationView.h`

setString:

Sets the requested-right string to use with the default authorization rights set.

```
- (void)setString:(AuthorizationString)authorizationString
```

Parameters

authorizationString

The string to be displayed.

Discussion

This is a convenience method that creates an authorization rights set when you specify only the name of the requested right. The requested-right string is displayed in the Details pane of the user authentication dialog box. Either this method or the `setAuthorizationRights:` method must be called before the view displays correctly.

Availability

Available in Mac OS X v10.3 and later.

See Also

- `setAuthorizationRights:` (page 26)
- `authorizationRights` (page 24)

Declared In

`SFAuthorizationView.h`

updateStatus:

Manually updates the authorization view.

```
- (BOOL)updateStatus:(id)inSender
```

Parameters

inSender

The authorization view to update.

Discussion

Calls to `updateStatus:` return `YES` if in the unlocked state, `NO` otherwise.

If autoupdates have not been set, you must call `updateStatus` for the authorization view's initial state to display correctly. The Security Framework calls this method for you when you change the state of the lock (by calling `deauthorize:` (page 25), for example.

Availability

Available in Mac OS X v10.3 and later.

See Also

- `setAutoupdate:` (page 27)
- `setAutoupdate:interval:` (page 27).

Declared In

`SFAuthorizationView.h`

Delegate Methods

authorizationViewCreatedAuthorization:

Sent to the delegate to indicate the authorization object has been created or changed. If you have saved a copy of the authorization object for your own purposes, you should discard it and call `authorization` (page 24) for a new authorization object.

```
- (void)authorizationViewCreatedAuthorization:(SFAuthorizationView *)view
```

Availability

Available in Mac OS X v10.3 and later.

Declared In

`SFAuthorizationView.h`

authorizationViewDidAuthorize:

Sent to the delegate to indicate the user was authorized and the authorization view was changed to unlocked.

```
- (void)authorizationViewDidAuthorize:(SFAuthorizationView *)view
```

Availability

Available in Mac OS X v10.3 and later.

Declared In

`SFAuthorizationView.h`

authorizationViewDidDeauthorize:

Sent to the delegate to indicate the user was deauthorized and the authorization view was changed to locked.

```
- (void)authorizationViewDidDeauthorize:(SFAuthorizationView *)view
```

Availability
Available in Mac OS X v10.3 or later.

Declared In
SFAuthorizationView.h

authorizationViewReleasedAuthorization:

Sent to the delegate to indicate that deauthorization is about to occur.

```
- (void)authorizationViewReleasedAuthorization:(SFAuthorizationView *)view
```

Discussion
This method is called after deauthorization has been approved (either you called the `deauthorize:` method, or the user clicked an open lock icon and the `authorizationViewShouldDeauthorize:` delegate method did not cancel the operation), and before the user is deauthorized (that is, before the `authorizationViewDidDeauthorize:` delegate method is called).

Availability
Available in Mac OS X v10.3 and later.

See Also
- `deauthorize:` (page 25)
- `authorizationViewShouldDeauthorize:` (page 31)
- `authorizationViewDidDeauthorize:` (page 31)

Declared In
SFAuthorizationView.h

authorizationViewShouldDeauthorize:

Sent to the delegate when a user clicks the open lock icon.

```
- (BOOL)authorizationViewShouldDeauthorize:(SFAuthorizationView *)view
```

Discussion
The delegate can react to this before deauthorization happens and avoid it by returning `NO`. This delegate method is not called when you call the `deauthorize:` method.

Availability
Available in Mac OS X v10.3 and later.

See Also
- `deauthorize:` (page 25)

Declared In
SFAuthorizationView.h

Constants

These constants are of type `SFAuthorizationViewState` and define the current state of the authorization view.

Constant	Description
`SFAuthorizationStartupState`	Indicates the state is starting up. Available in Mac OS X v10.3 and later. Declared in `SFAuthorizationView.h`.
`SFAuthorizationViewLockedState`	Indicates the state is locked. Available in Mac OS X v10.3 and later. Declared in `SFAuthorizationView.h`.
`SFAuthorizationViewInProgressState`	Indicates the state is in progress. Available in Mac OS X v10.3 and later. Declared in `SFAuthorizationView.h`.
`SFAuthorizationViewUnlockedState`	Indicates the state is unlocked. Available in Mac OS X v10.3 and later. Declared in `SFAuthorizationView.h`.

SFCertificatePanel Class Reference

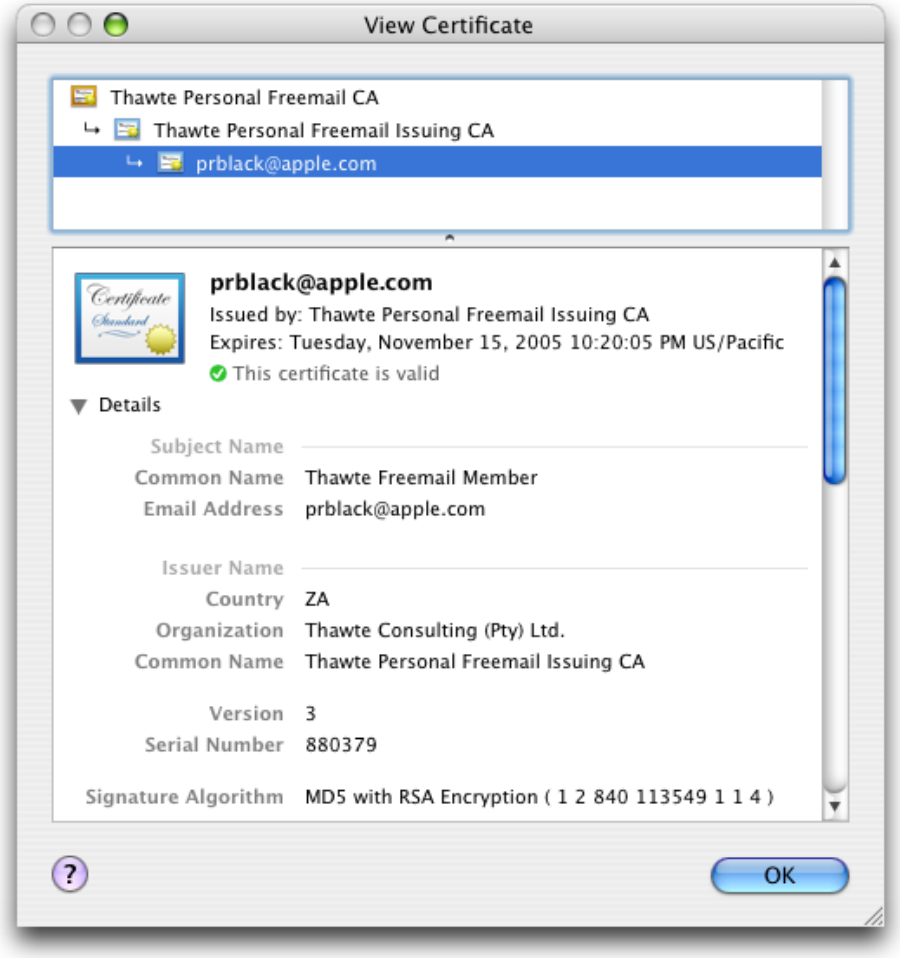
Inherits from	NSPanel : NSWindow : NSResponder : NSObject
Conforms to	NSUserInterfaceValidations (NSWindow) NSAnimatablePropertyContainer (NSWindow) NSCoding (NSResponder) NSObject (NSObject)
Framework	/System/Library/Frameworks/SecurityInterface.framework
Declared in	SFCertificatePanel.h
Availability	Available in Mac OS X v10.3 and later
Companion guide	Certificate, Key, and Trust Services Programming Guide

Overview

The `SFCertificatePanel` class displays one or more certificates in a panel or sheet. It can optionally display all of the certificates in a certificate chain.

The following figure shows an example of a certificate panel.

Figure 3-1 Certificate panel



This class displays certificate details, but not trust settings. To display a certificate with editable trust settings in a panel or sheet, use the `SFCertificateTrustPanel` class (SFCertificateTrustPanel). To display certificates in a custom view, use the `SFCertificateView` class (SFCertificateView).

Note that for Mac OS X v10.4 and later, this class displays the evaluation status for each certificate. You can modify how the certificates are evaluated by calling the `setPolicies:` (page 40) method.

Tasks

Returning a Shared Certificate Panel Object

`+ sharedCertificatePanel` (page 35)

Returns a shared certificate panel object. If the object has not already been created, this method allocates and initializes the object first.

Providing Help

- `setHelpAnchor:` (page 40)
  Sets the help anchor string for the sheet or modal panel.
- `setShowsHelp:` (page 41)
  Displays a Help button in the sheet or panel.
- `helpAnchor` (page 37)
  Returns the current help anchor string for the sheet or panel.
- `showsHelp` (page 41)
  Indicates whether the help button is currently set to be displayed.

Customizing the Appearance of the Sheet or Panel

- `setAlternateButtonTitle:` (page 39)
  Customizes the title of the alternate button.
- `setDefaultButtonTitle:` (page 39)
  Customizes the title of the default button.
- `setPolicies:` (page 40)
  Specifies one or more policies that apply to the displayed certificates.
- `policies` (page 38)
  Returns an array of policies used to evaluate the status of the displayed certificates.

Displaying a Sheet or Panel

- `beginSheetForWindow:modalDelegate:didEndSelector:contextInfo:certificates:showGroup:` (page 36)
  Displays one or more certificates in a modal sheet.
- `runModalForCertificates:showGroup:` (page 38)
  Displays one or more specified certificates in a modal panel.

Providing help

- `certificatePanelShowHelp:` (page 42) *delegate method*
  Implements custom help behavior for the modal panel.

Class Methods

sharedCertificatePanel

Returns a shared certificate panel object. If the object has not already been created, this method allocates and initializes the object first.

```
+ (SFCertificatePanel *)sharedCertificatePanel
```

Discussion
Use this method if your application displays a single certificate panel or sheet at a time. If your application can display multiple certificate panels or sheets at once, you must allocate separate object instances (using the `alloc` class method inherited from `NSObject`) and initialize them (using the `init` instance method, also inherited from `NSObject`) instead of using this class method.

Availability
Available in Mac OS X v10.3 and later.

See Also
`+alloc` (`NSObject`)
`-init` (`NSObject`)

Declared In
`SFCertificatePanel.h`

Instance Methods

beginSheetForWindow:modalDelegate:didEndSelector:contextInfo:certificates:showGroup:

Displays one or more certificates in a modal sheet.

```
- (void)beginSheetForWindow:(NSWindow *)docWindow modalDelegate:(id)delegate
    didEndSelector:(SEL)didEndSelector contextInfo:(void *)contextInfo
    certificates:(NSArray *)certificates showGroup:(BOOL)showGroup
```

Parameters

docWindow
The parent window to which the sheet is attached.

delegate
The delegate object in which the method specified in the `didEndSelector` parameter is implemented.

didEndSelector
A method selector for a delegate method called when the sheet has been dismissed. Implementation of this delegate method is optional.

contextInfo
A pointer to data that is passed to the delegate method. You can use this data pointer for any purpose you wish.

certificates
The certificates to display. Pass an NSArray containing one or more objects of type `SecCertificateRef` in this parameter. The first certificate in the array must be the leaf certificate. The other certificates (if any) can be included in any order.

showGroup
Specifies whether additional certificates (other than the leaf certificate) are displayed.

Discussion
The behavior of this method is somewhat different in Mac OS X v10.4 and later versus Mac OS X v10.3. In Mac OS X v10.3, the sheet displays whatever certificates you pass in the *certificates* parameter (provided the *showGroup* parameter is set to YES). Starting with Mac OS X v10.4, the sheet displays the leaf certificate (that is, the first certificate in the array you pass) plus any other certificates in the certificate chain that the Security Server can find. If you include all of the certificates in the chain in the *certificates* parameter, you can ensure that the same certificates are displayed whatever the version of the operating system, and may decrease the time required to find and display the certificates in Mac OS X v10.4 and later.

The delegate method has the following signature:

```
-(void)createPanelDidEnd:(NSWindow *)sheet
       returnCode:(int)returnCode
       contextInfo:(void *)contextInfo
```

The parameters for the delegate method are:

sheet
The window to which the sheet was attached.

returnCode
The result code indicating which button the user clicked: either NSFileHandlingPanelOKButton or NSFileHandlingPanelCancelButton.

contextInfo
Client-defined contextual data that is passed in the contextInfo parameter of the beginSheetForDirectory:... method.

The delegate method may dismiss the keychain settings sheet itself; if it does not, the sheet is dismissed on return from the beginSheetForDirectory:... method.

Availability
Available in Mac OS X v10.3 and later.

See Also
- runModalForCertificates:showGroup: (page 38)

Declared In
SFCertificatePanel.h

helpAnchor

Returns the current help anchor string for the sheet or panel.

```
- (NSString *)helpAnchor
```

Availability
Mac OS X v10.4

See Also
- setHelpAnchor: (page 40)

Declared In
SFCertificatePanel.h

policies

Returns an array of policies used to evaluate the status of the displayed certificates.

```
- (NSArray *)policies
```

Discussion
This method returns an autoreleased NSArray containing one or more objects of type `SecPolicyRef`, as set by a previous `setPolicies:` call, or the Apple X.509 Basic Policy if `setPolicies:` has not been called. See "AppleX509TP Trust Policies" in *Certificate, Key, and Trust Services Reference* for a list of policies and object identifiers provided by the AppleX509TP module.

Availability
Mac OS X v10.4

See Also
`- setPolicies:` (page 40)

Declared In
`SFCertificatePanel.h`

runModalForCertificates:showGroup:

Displays one or more specified certificates in a modal panel.

```
- (NSInt)runModalForCertificates:(NSArray *)certificates showGroup:(BOOL)showGroup
```

Parameters

certificates
The certificates to display. Pass an NSArray containing one or more objects of type `SecCertificateRef` in this parameter. The first certificate in the array must be the leaf certificate. The other certificates (if any) can be included in any order.

showGroup
Specifies whether additional certificates (other than the leaf certificate) are displayed. To show only a single certificate, specify only one `SecCertificateRef` in the array and set `showGroup` to `NO`.

Discussion
This method returns the integer constant `NSOKButton` when dismissed.

The behavior of this method is somewhat different in Mac OS X v10.4 and later versus Mac OS X v10.3. In Mac OS X v10.3, the panel displays whatever certificates you pass in the *certificates* parameter (provided the *showGroup* parameter is set to `YES`). Starting with Mac OS X v10.4, the panel displays the leaf certificate (that is, the first certificate in the array you pass) plus any other certificates in the certificate chain that the Security Server can find. If you include all of the certificates in the chain in the *certificates* parameter, you can ensure that the same certificates are displayed whatever the version of the operating system, and may decrease the time required to find and display the certificates in Mac OS X v10.4 and later.

Availability
Available in Mac OS X v10.3 and later.

See Also
`- beginSheetForWindow:modalDelegate:didEndSelector:contextInfo:certificates:showGroup:` (page 36)

Declared In
SFCertificatePanel.h

setAlternateButtonTitle:

Customizes the title of the alternate button.

```
- (void)setAlternateButtonTitle:(NSString *)title
```

Parameters

title

The new title for the alternate button. If this method is not called, or if `title` is set to `nil`, the button is not shown.

Discussion
The alternate button is typically labelled "Cancel". The alternate button dismisses the sheet or panel and returns a value of `NSCancelButton`.

Availability
Mac OS X v10.4

See Also
- `setDefaultButtonTitle:` (page 39)
- `runModalForTrust:message:` (page 47)

Declared In
SFCertificatePanel.h

setDefaultButtonTitle:

Customizes the title of the default button.

```
- (void)setDefaultButtonTitle:(NSString *)title
```

Parameters

title

The new title for the default button. The default title for this button is "OK".

Discussion
The default button dismisses the sheet or panel and returns a value of `NSOKButton`.

Availability
Mac OS X v10.4

See Also
- `setAlternateButtonTitle:` (page 39)

Declared In
SFCertificatePanel.h

setHelpAnchor:

Sets the help anchor string for the sheet or modal panel.

```
- (void)setHelpAnchor:(NSString *)anchor
```

Parameters

anchor

The new help anchor string.

Discussion

You may call this function to set a help anchor string if you display a help button in the sheet or modal panel and do not implement the delegate method `certificatePanelShowHelp:`, or if the delegate method returns `NO`. If you display a help button, do not set a help anchor string, and do not implement a delegate, the certificate panel displays a default help page ("Why isn't a certificate being accepted?").

Availability

Mac OS X v10.4

See Also

- `setShowsHelp:` (page 41)
- `certificatePanelShowHelp:` (page 42)
- `helpAnchor` (page 37)

Declared In

`SFCertificatePanel.h`

setPolicies:

Specifies one or more policies that apply to the displayed certificates.

```
- (void)setPolicies:(id)policies
```

Parameters

policies

The policies to use when evaluating the certificates' status. You can pass either a `SecPolicyRef` object or an NSArray (containing one or more `SecPolicyRef` instances) in this parameter. If `policies` is set to `nil`, the Apple X.509 Basic Policy is used.

Discussion

Applications typically display a certificate panel in the context of a specific use, such as SSL or S/MIME. You should set only the policy references that apply to your intended use. See "AppleX509TP Trust Policies" for a list of policies and object identifiers provided by the AppleX509TP module.

Availability

Mac OS X v10.4

See Also

- `policies` (page 38)

Declared In

`SFCertificatePanel.h`

setShowsHelp:

Displays a Help button in the sheet or panel.

```
- (void)setShowsHelp:(BOOL)showsHelp
```

Parameters

showsHelp

Set to YES to display the help button. The help button is hidden by default.

Discussion

When a user clicks the help button, the certificate panel first checks the delegate for a certificatePanelShowHelp: method. If the delegate does not implement such a method, or the delegate method returns NO, then the NSHelpManager method openHelpAnchor:inBook: is called with a nil book and the anchor specified by the setHelpAnchor: method. An exception is raised if the delegate returns NO and there is no help anchor set.

Availability

Mac OS X v10.4

See Also

- certificatePanelShowHelp: (page 42)
- setHelpAnchor: (page 40)
- openHelpAnchor:inBook:(NSHelpManager)
- showsHelp (page 41)

Declared In

SFCertificatePanel.h

showsHelp

Indicates whether the help button is currently set to be displayed.

```
- (BOOL)showsHelp
```

Discussion

This method returns YES if the help button is currently set to be displayed.

Availability

Mac OS X v10.4

See Also

- setShowsHelp: (page 41)

Declared In

SFCertificatePanel.h

Delegate Methods

certificatePanelShowHelp:

Implements custom help behavior for the modal panel.

```
- (BOOL)certificatePanelShowHelp:(SFCertificatePanel *)sender
```

Parameters

sender

The certificate panel for which to implement custom help.

Discussion

You can use this delegate method to implement custom help if you call the `setShowsHelp:` method to display a help button in the sheet or panel. If you are not implementing custom help, do not implement this method.

Availability

Mac OS X v10.4

See Also

`- setShowsHelp:` (page 41)

`-setDelegate:` (`NSWindow`)

Declared In

`SFCertificatePanel.h`

SFCertificateTrustPanel Class Reference

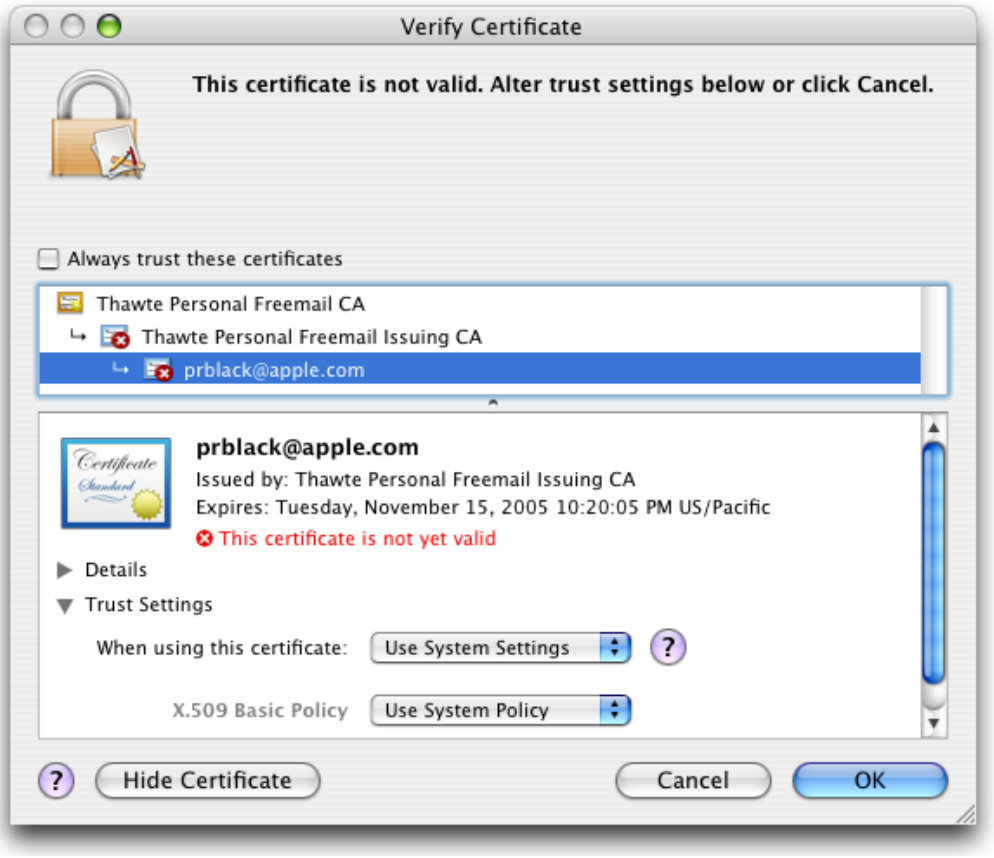
Inherits from	SFCertificatePanel : NSPanel : NSWindow : NSResponder : NSObject
Conforms to	NSUserInterfaceValidations (NSWindow) NSAnimatablePropertyContainer (NSWindow) NSCoding (NSResponder) NSObject (NSObject)
Framework	/System/Library/Frameworks/SecurityInterface.framework
Declared in	SFCertificateTrustPanel.h
Availability	Available in Mac OS X v10.3 and later
Companion guide	Certificate, Key, and Trust Services Programming Guide

Overview

The `SFCertificateTrustPanel` class opens a panel or sheet that lets the user edit the trust settings in any of the certificates in a certificate chain.

The following figure shows an example of a certificate trust panel.

Figure 4-1 Certificate trust panel



You can use this class to enable a user to make trust decisions when one or more certificates required for an operation are invalid or cannot be verified.

To display a certificate in a panel or sheet without editable trust settings, use the `SFCertificatePanel` class. To display certificates in a custom view, use the `SFCertificateView` class.

Tasks

Returning a Shared Certificate Trust Panel Object

`+ sharedCertificateTrustPanel` (page 45)

Returns a shared certificate trust panel object. If the object has not already been created, this method allocates and initializes the object first.

Displaying a Sheet or Panel

`- beginSheetForWindow:modalDelegate:didEndSelector:contextInfo:trust:message:` (page 45)

Displays a modal sheet that shows the results of a certificate trust evaluation and that allows the user to edit trust settings.

`- runModalForTrust:message:` (page 47)

Displays a modal panel that shows the results of a certificate trust evaluation and that allows the user to edit trust settings.

Class Methods

sharedCertificateTrustPanel

Returns a shared certificate trust panel object. If the object has not already been created, this method allocates and initializes the object first.

```
+ (SFCertificateTrustPanel *)sharedCertificateTrustPanel
```

Discussion
Use this method if your application displays a single certificate trust panel or sheet at a time. If your application can display multiple certificate trust panels or sheets at once, you must allocate separate object instances (using the `alloc` class method inherited from `NSObject`) and initialize (using the `init` instance method, also inherited from `NSObject`) instead of using this class method.

Availability
Available in Mac OS X v10.3 and later.

See Also
`+alloc(NSObject)`
`-init(NSObject)`

Declared In
`SFCertificateTrustPanel.h`

Instance Methods

beginSheetForWindow:modalDelegate:didEndSelector:contextInfo:trust:message:

Displays a modal sheet that shows the results of a certificate trust evaluation and that allows the user to edit trust settings.

```
- (void)beginSheetForWindow:(NSWindow *)docWindow modalDelegate:(id)delegate
    didEndSelector:(SEL)didEndSelector contextInfo:(void *)contextInfo
    trust:(SecTrustRef)trust message:(NSString *)message
```

Parameters

docWindow

The parent window to which the sheet is attached.

delegate

The delegate object in which the method specified in the `didEndSelector` parameter is implemented.

didEndSelector

A method selector for a delegate method called when the sheet has been dismissed. Implementation of this delegate method is optional.

contextInfo

A pointer to data that is passed to the delegate method. You can use this data pointer for any purpose you wish.

trust

A trust management object. Use the `SecTrustCreateWithCertificates` function (in Security/SecTrust.h) to create the trust management object.

message

A message string to display in the sheet.

Discussion

The delegate method has the following signature:

```
-(void)createPanelDidEnd:(NSWindow *)sheet
       returnCode:(int)returnCode
       contextInfo:(void *)contextInfo
```

The parameters for the delegate method are:

sheet

The window to which the sheet was attached.

returnCode

The result code indicating which button the user clicked: either `NSFileHandlingPanelOKButton` or `NSFileHandlingPanelCancelButton`.

contextInfo

Client-defined contextual data that is passed in the `contextInfo` parameter of the `beginSheetForDirectory:...` method.

The delegate method may dismiss the keychain settings sheet itself; if it does not, the sheet is dismissed on return from the `beginSheetForDirectory:...` method.

Availability

Available in Mac OS X v10.3 and later.

See Also

`SecTrustCreateWithCertificates` (Security/SecTrust.h)

`SecTrustGetUserTrust` (Security/SecTrust.h)

`SecTrustGetResult`

`- runModalForTrust:message:` (page 47)

Declared In

`SFCertificateTrustPanel.h`

runModalForTrust:message:

Displays a modal panel that shows the results of a certificate trust evaluation and that allows the user to edit trust settings.

```
- (NSInt)runModalForTrust:(SecTrustRef)trust message:(NSString *)message
```

Parameters

trust

A trust management object. Use the `SecTrustCreateWithCertificates` function (in Security/SecTrust.h) to create the trust management object.

message

A message string to display in the panel.

Discussion

This method returns `NSOKButton` if the default button is clicked, or `NSCancelButton` if the alternate button is clicked.

The user can use this panel to edit trust decisions for the specified certificate or for any of the certificates in the certificate chain. The trust settings are saved when the user clicks the default button. Call `SecTrustGetUserTrust` to o btain the user's trust settings.

Note that changing the user trust settings does not affect the results of a trust evaluation. Therefore, the trust evaluation shown in the panel (such as "This certificate is not yet valid") does not change, nor does the result of a call to `SecTrustGetResult`. It is up to your application to determine how to handle the user's trust decision.

Availability

Available in Mac OS X v10.3 and later.

See Also

`SecTrustCreateWithCertificates` (Security/SecTrust.h)

`SecTrustGetUserTrust` (Security/SecTrust.h)

`SecTrustGetResult` (Security/SecTrust.h)

`- beginSheetForWindow:modalDelegate:didEndSelector:contextInfo:trust:message:` (page 45)

Declared In

`SFCertificateTrustPanel.h`

SFCertificateView Class Reference

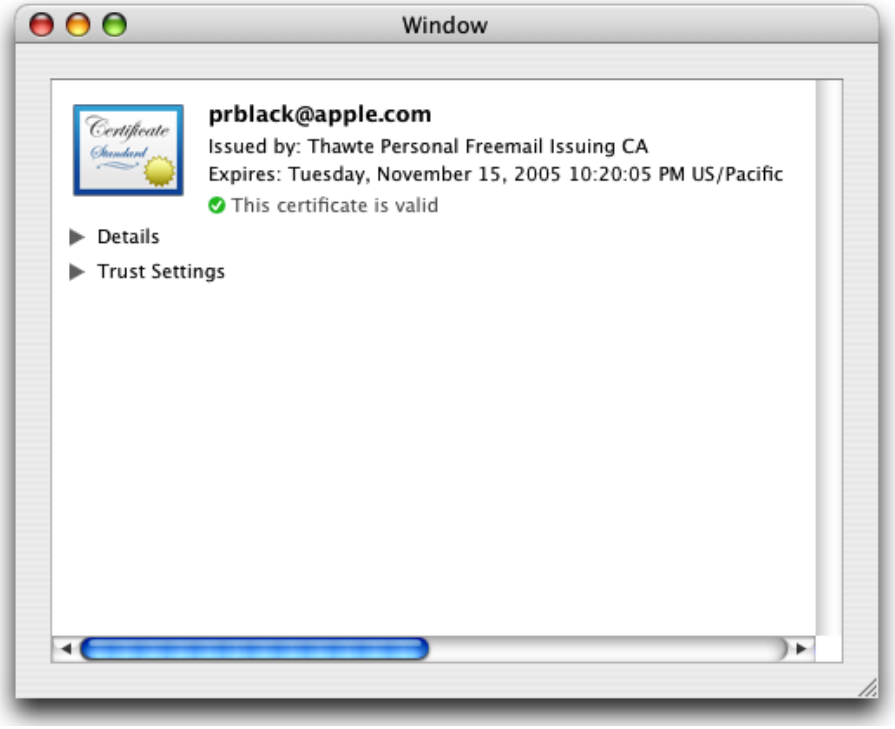
Inherits from	NSView : NSResponder : NSObject
Conforms to	NSAnimatablePropertyContainer (NSView) NSCoding (NSResponder) NSObject (NSObject)
Framework	/System/Library/Frameworks/SecurityInterface.framework
Declared in	SFCertificateView.h
Availability	Available in Mac OS X v10.3 and later
Companion guide	Certificate, Key, and Trust Services Programming Guide

Overview

The `SFCertificateView` class displays the contents of a certificate. It includes options to display certificate details, display trust settings, and allow users to edit a certificate's trust settings.

The following figure shows a certificate view that includes editable trust settings and certificate details.

Figure 5-1 Certificate view



Tasks

Specifying the Certificate to Display

`- setCertificate:` (page 53)
Specifies the certificate that's displayed in the view.

Customizing the Appearance and Behavior of the View

`- setDisplayDetails:` (page 53)
Specifies whether the user can see the certificate details.

`- setDisplayTrust:` (page 54)
Specifies whether the user can see the certificate's trust settings.

`- setEditableTrust:` (page 54)
Specifies whether the user can edit the certificate's trust settings.

`- setPolicies:` (page 55)
Specifies the policies to use when evaluating this certificate's status.

Getting Information About the View

- `certificate` (page 51)
  Returns the certificate currently displayed in the view.
- `detailsDisplayed` (page 51)
  Indicates if the view currently shows the certificate's details.
- `isTrustDisplayed` (page 52)
  Indicates if the view currently shows the certificate's trust settings.
- `isEditable` (page 52)
  Indicates if the view allows the user to edit the certificate's trust.
- `policies` (page 52)
  Returns an array of policies used to evaluate the status of the displayed certificate.

Saving User Trust Settings

- `saveTrustSettings` (page 53)
  Saves the user's current trust settings for the displayed certificate.

Instance Methods

certificate

Returns the certificate currently displayed in the view.

```
- (SecCertificateRef)certificate
```

Availability
Available in Mac OS X v10.3 and later.

See Also
- `setCertificate:` (page 53)

Declared In
`SFCertificateView.h`

detailsDisplayed

Indicates if the view currently shows the certificate's details.

```
- (BOOL)detailsDisplayed
```

Availability
Available in Mac OS X v10.4 and later

See Also
- `setDisplayDetails:` (page 53)

Declared In
SFCertificateView.h

isEditable

Indicates if the view allows the user to edit the certificate's trust.

```
- (BOOL)isEditable
```

Availability
Available in Mac OS X v10.3 and later.

See Also
- setEditableTrust: (page 54)

Declared In
SFCertificateView.h

isTrustDisplayed

Indicates if the view currently shows the certificate's trust settings.

```
- (BOOL)isTrustDisplayed
```

Availability
Available in Mac OS X v10.3 and later.

See Also
- setDisplayTrust: (page 54)

Declared In
SFCertificateView.h

policies

Returns an array of policies used to evaluate the status of the displayed certificate.

```
- (NSArray *)policies
```

Discussion
This method returns an autoreleased NSArray containing one or more instances of SecPolicyRef. The array always contains at least one item (the Apple X.509 Basic policy, if you have never called the setPolicies: method).

Availability
Available in Mac OS X v10.4 and later

See Also
- setPolicies: (page 55)

Declared In
SFCertificateView.h

saveTrustSettings

Saves the user's current trust settings for the displayed certificate.

```
- (void)saveTrustSettings
```

Discussion
If trust settings are not editable, this method effectively does nothing. You can use `SecTrustGetUserTrust` to subsequently retrieve the trust settings.

Availability
Available in Mac OS X v10.3 and later.

See Also
`- setEditableTrust:` (page 54)

Declared In
`SFCertificateView.h`

setCertificate:

Specifies the certificate that's displayed in the view.

```
- (void)setCertificate:(SecCertificateRef)certificate
```

Parameters
certificate
The new certificate for the view.

Availability
Available in Mac OS X v10.3 and later.

See Also
`- certificate` (page 51)

Declared In
`SFCertificateView.h`

setDisplayDetails:

Specifies whether the user can see the certificate details.

```
- (void)setDisplayDetails:(BOOL)display
```

Parameters
display
Pass `YES` to display the certificate details, or `NO` to hide them.

Discussion
For behavioral compatibility with Mac OS X v10.3, certificate details are displayed by default. To hide the details of a certificate, you must explicitly set the display value to `NO`.

Availability
Available in Mac OS X v10.4 and later

See Also
- `detailsDisplayed` (page 51)

Declared In
`SFCertificateView.h`

setDisplayTrust:

Specifies whether the user can see the certificate's trust settings.

```
- (void)setDisplayTrust:(BOOL)display
```

Parameters

display

Pass YES to display the trust settings, or `NO` to hide them.

Discussion
Certificate trust settings are not displayed by default. To show the certificate's trust settings, you must explicitly set the display value to `YES`. with either this method or the `setEditableTrust` method.

Availability
Available in Mac OS X v10.3 and later.

See Also
- `setEditableTrust:` (page 54)
- `isTrustDisplayed` (page 52)

Declared In
`SFCertificateView.h`

setEditableTrust:

Specifies whether the user can edit the certificate's trust settings.

```
- (void)setEditableTrust:(BOOL)editable
```

Parameters

editable

Pass `YES` if the trust settings should be editable.

Discussion
For behavioral compatibility with Mac OS X v10.3, this method causes the certificate trust settings to be displayed if they are not currently visible (that is, if `setDisplayTrust:` is set to `NO`).

Availability
Available in Mac OS X v10.3 and later.

See Also
- `setDisplayTrust:` (page 54)
- `isEditable` (page 52)

Declared In
`SFCertificateView.h`

setPolicies:

Specifies the policies to use when evaluating this certificate's status.

```
- (void)setPolicies:(id)policies
```

Parameters

policies

The policy or policies to use. You can pass either a `SecPolicyRef` object or an NSArray (containing one or more objects of type `SecPolicyRef`) in this parameter. If `policies` is set to nil, the Apple X.509 Basic Policy is used. See "AppleX509TP Trust Policies" for a list of policies and object identifiers provided by the AppleX509TP module.

Discussion

Applications typically display a certificate view in the context of a specific use, such as SSL or S/MIME. You should set only the policy references that apply to your intended use.

Availability

Available in Mac OS X v10.4 and later.

See Also

- `policies` (page 52)

Declared In

`SFCertificateView.h`

SFChooseIdentityPanel Class Reference

Inherits from	NSPanel : NSWindow : NSResponder : NSObject
Conforms to	NSUserInterfaceValidations (NSWindow) NSAnimatablePropertyContainer (NSWindow) NSCoding (NSResponder) NSObject (NSObject)
Framework	/System/Library/Frameworks/SecurityInterface.framework
Declared in	SFChooseIdentityPanel.h
Availability	Available in Mac OS X v10.3 and later
Companion guide	Certificate, Key, and Trust Services Programming Guide

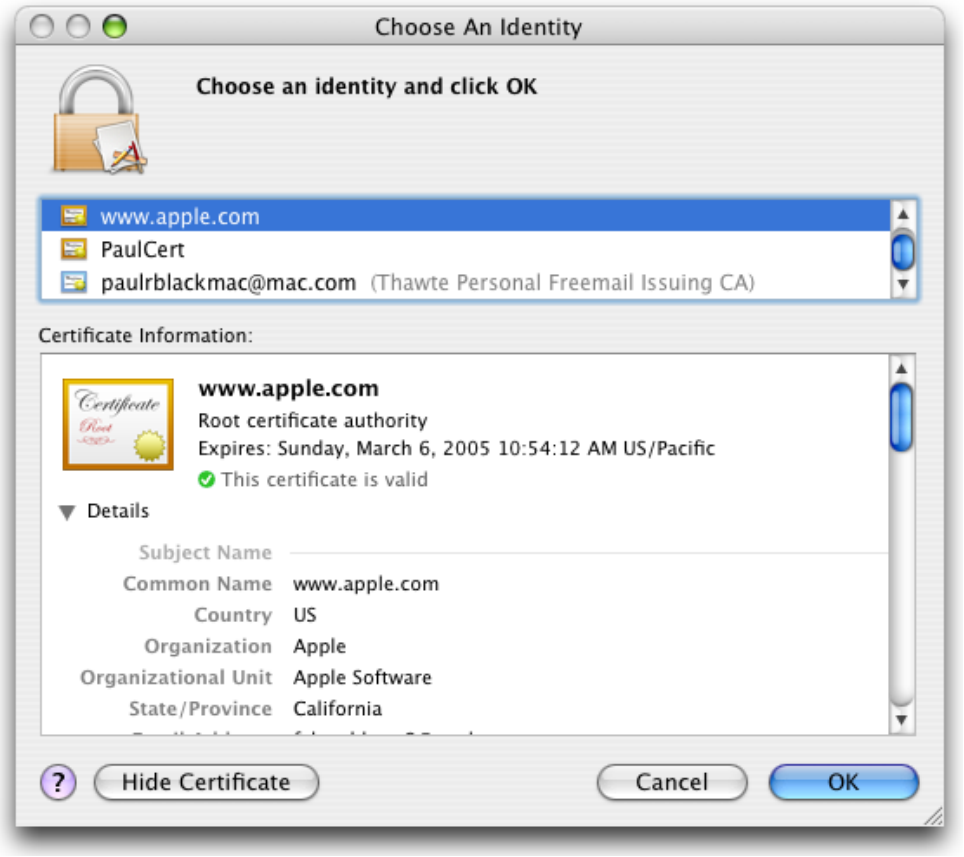
Overview

The `SFChooseIdentityPanel` class displays a sheet or panel containing a list of identities and allows a user to select an identity from the list.

An identity is a digital certificate together with its associated private key. This class also allows the user to display the contents of any certificate in the list.

The following figure shows an example of a choose identity panel.

Figure 6-1 Choose identity panel



Tasks

Returning a Shared Certificate Panel Object

`+ sharedChooseIdentityPanel` (page 60)
Returns a shared choose identity panel object. If the object has not already been created, this method allocates and initializes the object first.

Providing Help

`- setHelpAnchor:` (page 64)
Sets the help anchor string for the sheet or modal panel.

`- setShowsHelp:` (page 65)
Displays a Help button in the sheet or panel.

`- helpAnchor` (page 61)
Returns the current help anchor string for the sheet or panel.

- `showsHelp` (page 65)
  Indicates whether the help button is currently set to be displayed.

Customizing the Appearance of the Sheet or Panel

- `setAlternateButtonTitle:` (page 63)
  Customizes the title of the alternate button.
- `setDefaultButtonTitle:` (page 63)
  Customizes the title of the default button.
- `setPolicies:` (page 64)
  Specifies one or more policies that apply to the displayed certificates.
- `policies` (page 62)
  Returns an array of policies used to evaluate the status of the displayed certificates.

Displaying a Sheet or Panel

- `beginSheetForWindow:modalDelegate:didEndSelector:contextInfo:identities:message:` (page 60)
  Displays a list of identities in a modal sheet from which the user can select an identity.
- `runModalForIdentities:message:` (page 62)
  Displays a list of identities in a modal panel.

Getting Identity Information from a Sheet or Panel

- `identity` (page 62)
  Returns the identity that the user chose in the panel or sheet.

Providing help

- `chooseIdentityPanelShowHelp:` (page 66) *delegate method*
  Implements custom help behavior for the modal panel.

Getting identity information from a sheet or panel

- `chooseIdentitySheetDidEnd:returnCode:contextInfo:` (page 66) *delegate method*
  The delegate method called when the sheet is closed.

Class Methods

sharedChooseIdentityPanel

Returns a shared choose identity panel object. If the object has not already been created, this method allocates and initializes the object first.

```
+ (SFChooseIdentityPanel *)sharedChooseIdentityPanel
```

Discussion
Use this method if your application displays a single choose identity panel or sheet at a time. If your application can display multiple choose identity panels or sheets at once, you must allocate separate object instances (using the `alloc` class method inherited from `NSObject`) and initialize them (using the `init` instance method, also inherited from `NSObject`) instead of using this class method.

Availability
Available in Mac OS X v10.3 and later.

See Also
`+alloc(NSObject)`
`-init(NSObject)`

Declared In
`SFChooseIdentityPanel.h`

Instance Methods

beginSheetForWindow:modalDelegate:didEndSelector:contextInfo:identities:message:

Displays a list of identities in a modal sheet from which the user can select an identity.

```
- (void)beginSheetForWindow:(NSWindow *)docWindow modalDelegate:(id)delegate
    didEndSelector:(SEL)didEndSelector contextInfo:(void *)contextInfo
    identities:(NSArray *)identities message:(NSString *)message
```

Parameters

docWindow
The parent window to which the sheet is attached.

delegate
The delegate object in which the method specified in the `didEndSelector` parameter is implemented.

didEndSelector
A method selector for a delegate method called when the sheet has been dismissed. Implementation of this delegate method is optional.

contextInfo
A pointer to data that is passed to the delegate method. You can use this data pointer for any purpose you wish.

identities
An array of identity objects (objects of type `SecIdentityRef`). Use the `SecIdentitySearchCopyNext` function (in Security/SecIdentitySearch.h) to find identity objects.

message
A message string to display in the sheet.

Discussion
Use the `identity` method to obtain the identity chosen by the user.

The delegate method has the following signature:

```
-(void)createPanelDidEnd:(NSWindow *)sheet
       returnCode:(int)returnCode
       contextInfo:(void *)contextInfo
```

The parameters for the delegate method are:

sheet
The window to which the sheet was attached.

returnCode
The result code indicating which button the user clicked: either `NSFileHandlingPanelOKButton` or `NSFileHandlingPanelCancelButton`.

contextInfo
Client-defined contextual data that is passed in the `contextInfo` parameter of the `beginSheetForDirectory:...` method.

The delegate method may dismiss the keychain settings sheet itself; if it does not, the sheet is dismissed on return from the `beginSheetForDirectory:...` method.

Availability
Available in Mac OS X v10.3 and later.

See Also
`SecIdentitySearchCopyNext` (Security/SecIdentitySearch.h)
`- identity` (page 62)
`- runModalForIdentities:message:` (page 62)

Declared In
`SFChooseIdentityPanel.h`

helpAnchor

Returns the current help anchor string for the sheet or panel.

```
- (NSString *)helpAnchor
```

Availability
Available in Mac OS X v10.4 and later.

See Also
`- setHelpAnchor:` (page 64)

Declared In
SFChooseIdentityPanel.h

identity

Returns the identity that the user chose in the panel or sheet.

```
- (SecIdentityRef)identity
```

Availability
Available in Mac OS X v10.3 and later.

Declared In
SFChooseIdentityPanel.h

policies

Returns an array of policies used to evaluate the status of the displayed certificates.

```
- (NSArray *)policies
```

Discussion
This method returns an autoreleased NSArray containing one or more objects of type `SecPolicyRef`, as set by a previous `setPolicies:` call, or the Apple X.509 Basic Policy if `setPolicies:` has not been called. See "AppleX509TP Trust Policies" in *Certificate, Key, and Trust Services Reference* for a list of policies and object identifiers provided by the AppleX509TP module.

Availability
Available in Mac OS X v10.4 and later.

See Also
`- setPolicies:` (page 64)

Declared In
SFChooseIdentityPanel.h

runModalForIdentities:message:

Displays a list of identities in a modal panel.

```
- (NSInt)runModalForIdentities:(NSArray *)identities message:(NSString *)message
```

Parameters

identities
An array of identity objects (objects of type `SecIdentityRef`. Use the `SecIdentitySearchCopyNext` function (in Security/SecIdentitySearch.h) to find identity objects.

message
A message string to display in the panel.

Discussion
This method returns `NSOKButton` if the default button is clicked, or `NSCancelButton` if the alternate button is clicked.

Use the `identity` method to obtain the identity chosen by the user.

Availability
Available in Mac OS X v10.3 and later.

See Also
`SecIdentitySearchCopyNext` (Security/SecIdentitySearch.h)
`- identity` (page 62)
`- beginSheetForWindow:modalDelegate:didEndSelector:contextInfo:identities:message:` (page 60)

Declared In
`SFChooseIdentityPanel.h`

setAlternateButtonTitle:

Customizes the title of the alternate button.

```
- (void)setAlternateButtonTitle:(NSString *)title
```

Parameters
title
The new title for the alternate button. If this method is not called, or if `title` is set to `nil`, the button is not shown.

Discussion
The alternate button is typically labelled "Cancel". The alternate button dismisses the sheet or panel and returns a value of `NSCancelButton`.

Availability
Available in Mac OS X v10.4 and later.

See Also
`- setDefaultButtonTitle:` (page 63)

Declared In
`SFChooseIdentityPanel.h`

setDefaultButtonTitle:

Customizes the title of the default button.

```
- (void)setDefaultButtonTitle:(NSString *)title
```

Parameters
title
The new title for the default button. The default title for this button is "OK".

Discussion
The default button dismisses the sheet or panel and returns a value of `NSOKButton`.

Availability
Available in Mac OS X v10.4 and later.

See Also
- `setAlternateButtonTitle:` (page 63)

Declared In
`SFChooseIdentityPanel.h`

setHelpAnchor:

Sets the help anchor string for the sheet or modal panel.

```
- (void)setHelpAnchor:(NSString *)anchor
```

Parameters

anchor

The new help anchor string.

Discussion
You may call this function to set a help anchor string if you display a help button in the sheet or modal panel and do not implement the delegate method `certificatePanelShowHelp:`, or if the delegate method returns `NO`. If you display a help button, do not set a help anchor string, and do not implement a delegate, the certificate panel displays a default help page ("What is a digital identity?").

Availability
Available in Mac OS X v10.4 and later.

See Also
- `setShowsHelp:` (page 65)
- `chooseIdentityPanelShowHelp:` (page 66)
- `helpAnchor` (page 61)

Declared In
`SFChooseIdentityPanel.h`

setPolicies:

Specifies one or more policies that apply to the displayed certificates.

```
- (void)setPolicies:(id)policies
```

Parameters

policies

The policies to use when evaluating the certificates' status. You can pass either a `SecPolicyRef` object or an NSArray (containing one or more `SecPolicyRef` instances) in this parameter. If `policies` is set to `nil`, the Apple X.509 Basic Policy is used.

Discussion
The `SFChooseIdentityPanel` class evaluates trust for the certificates it displays. Applications typically display certificates in the context of a specific use, such as SSL or S/MIME. You should set only the policy references that apply to your intended use. See "AppleX509TP Trust Policies" for a list of policies and object identifiers provided by the AppleX509TP module.

Availability
Available in Mac OS X v10.4 and later.

See Also
- policies (page 62)

Declared In
SFChooseIdentityPanel.h

setShowsHelp:

Displays a Help button in the sheet or panel.

```
- (void)setShowsHelp:(BOOL)showsHelp
```

Parameters

showsHelp
Set to YES to display the help button. The help button is hidden by default.

Discussion
When a user clicks the help button, the choose identity panel first checks the delegate for a certificatePanelShowHelp: method. If the delegate does not implement such a method, or the delegate method returns NO, then the NSHelpManager method openHelpAnchor:inBook: is called with a nil book and the anchor specified by the setHelpAnchor: method. An exception is raised if the delegate returns NO and there is no help anchor set.

Availability
Available in Mac OS X v10.4 and later.

See Also
- chooseIdentityPanelShowHelp: (page 66)
- setHelpAnchor: (page 64)
-openHelpAnchor:inBook: (NSHelpManager)
- showsHelp (page 65)

Declared In
SFChooseIdentityPanel.h

showsHelp

Indicates whether the help button is currently set to be displayed.

```
- (BOOL)showsHelp
```

Discussion
This method returns YES if the help button is currently set to be displayed.

Availability
Available in Mac OS X v10.4 and later.

See Also
- setShowsHelp: (page 65)

Declared In
SFChooseIdentityPanel.h

Delegate Methods

chooseIdentityPanelShowHelp:

Implements custom help behavior for the modal panel.

```
- (BOOL)chooseIdentityPanelShowHelp:(SFChooseIdentityPanel *)sender
```

Parameters

sender

The choose identity panel for which to implement custom help.

Discussion

You can use this delegate method to implement custom help if you call the `setShowsHelp:` method to display a help button in the sheet or panel. If you are not implementing custom help, do not implement this method.

Availability

Available in Mac OS X v10.4 and later.

See Also

`- setShowsHelp:` (page 65)

`-setDelegate:` (`NSWindow`)

Declared In

`SFChooseIdentityPanel.h`

chooseIdentitySheetDidEnd:returnCode:contextInfo:

The delegate method called when the sheet is closed.

```
- *)sheet
    returnCode:(int)returnCode  contextInfo:(void *)contextInfo
```

Parameters

sheet

The parent window to which the sheet is attached.

returnCode

An indication of which button the user clicked: either `NSOKButton` or `NSCancelButton`.

contextInfo

A pointer to data that is passed in the `contextInfo` parameter of the `beginSheetForWindow:modalDelegate:didEndSelector:contextInfo:identities:message:` method. You can use this data pointer for any purpose you wish.

Discussion

Modal delegates in sheets are temporary and the relationship only lasts until the sheet is dismissed. The `SFChooseIdentityPanel` object does not retain the modal delegate.

Availability

Available in Mac OS X v10.3 and later.

See Also
`- beginSheetForWindow:modalDelegate:didEndSelector:contextInfo:identities:message:` (page 60)

SFKeychainSavePanel Class Reference

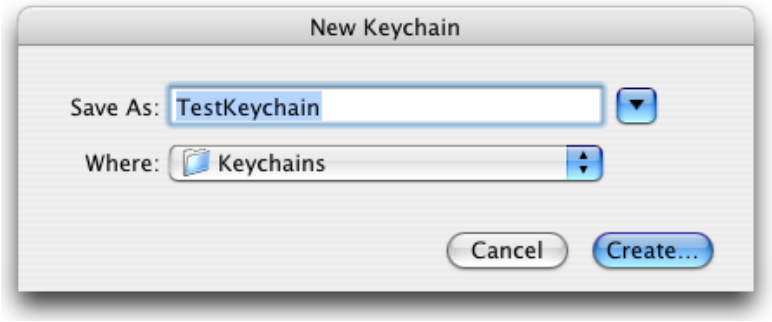
Inherits from	NSSavePanel : NSPanel : NSWindow : NSResponder : NSObject
Conforms to	NSUserInterfaceValidations (NSWindow) NSAnimatablePropertyContainer (NSWindow) NSCoding (NSResponder) NSObject (NSObject)
Framework	/System/Library/Frameworks/SecurityInterface.framework
Declared in	SFKeychainSavePanel.h
Availability	Available in Mac OS X v10.3 and later
Companion guide	Keychain Services Programming Guide

Overview

The `SFKeychainSavePanel` class displays a sheet or panel that allows the user to create a keychain.

The following figure shows an example of a keychain save panel.

Figure 7-1 Keychain save panel



Tasks

Returning a Shared Keychain Save Panel Object

`+ sharedKeychainSavePanel` (page 70)
Returns a shared keychain save panel object. If the object has not already been created, this method allocates and initializes the object first.

Displaying a Sheet or Panel

`- setPassword:` (page 73)
Specifies the password for the keychain that will be created.

`- beginSheetForDirectory:file:modalForWindow:modalDelegate:didEndSelector:contextInfo:` (page 71)
Displays a sheet that allows a user to create a new keychain.

`- runModalForDirectory:file:` (page 72)
Displays a panel that allows a user to create a new keychain.

Returning Information from the Sheet or Panel

`- keychain` (page 72)
Returns the keychain created by the keychain save panel.

Class Methods

sharedKeychainSavePanel

Returns a shared keychain save panel object. If the object has not already been created, this method allocates and initializes the object first.

```
+ (SFKeychainSavePanel *)sharedKeychainSavePanel
```

Availability
Available in Mac OS X v10.3 and later.

Declared In
`SFKeychainSavePanel.h`

Instance Methods

beginSheetForDirectory:file:modalForWindow:modalDelegate:didEndSelector:contextInfo:

Displays a sheet that allows a user to create a new keychain.

```
- (void)beginSheetForDirectory:(NSString *)path file:(NSString *)name
    modalForWindow:(NSWindow *)docWindow modalDelegate:(id)delegate
    didEndSelector:(SEL)didEndSelector contextInfo:(void *)contextInfo
```

Parameters

path

The path to the folder where the keychain is created. Specify `nil` for `~/Library/Keychains`.

name

The keychain name to be automatically displayed in the Save As field of the sheet.

docWindow

The parent window to which the sheet is attached. If this parameter is `nil`, the behavior defaults to a standalone modal window.

delegate

The delegate object in which the method specified in the `didEndSelector` parameter is implemented.

didEndSelector

A method selector for a delegate method called after the modal session has ended, but before the sheet has been dismissed. Implementation of this delegate method is optional.

contextInfo

A pointer to data that is passed to the delegate method. You can use this data pointer for any purpose you wish.

Discussion

The delegate method has the following signature:

```
-(void)createPanelDidEnd:(NSWindow *)sheet
       returnCode:(int)returnCode
       contextInfo:(void *)contextInfo
```

The parameters for the delegate method are:

sheet

The window to which the sheet was attached.

returnCode

The result code indicating which button the user clicked: either `NSFileHandlingPanelOKButton` or `NSFileHandlingPanelCancelButton`.

contextInfo

Client-defined contextual data that is passed in the `contextInfo` parameter of the `beginSheetForDirectory:file:modalForWindow:modalDelegate:didEndSelector:contextInfo:` method.

The delegate method may dismiss the keychain settings sheet itself; if it does not, the sheet is dismissed on return from the `beginSheetForDirectory:...` method.

Use the `keychain` method to obtain the keychain created by the user.

Availability
Available in Mac OS X v10.3 and later.

See Also
- `keychain` (page 72)
- `runModalForDirectory:file:` (page 72)

Declared In
`SFKeychainSavePanel.h`

keychain

Returns the keychain created by the keychain save panel.

```
- (SecKeychainRef)keychain
```

Availability
Available in Mac OS X v10.3 and later.

See Also
- `beginSheetForDirectory:file:modalForWindow:modalDelegate:didEndSelector:contextInfo:` (page 71)
- `runModalForDirectory:file:` (page 72)

Declared In
`SFKeychainSavePanel.h`

runModalForDirectory:file:

Displays a panel that allows a user to create a new keychain.

```
- (NSInt)runModalForDirectory:(NSString *)path file:(NSString *)name
```

Parameters

path
The path to the folder where the keychain is created. Specify `nil` for `~/Library/Keychains`.

name
The keychain name to be automatically displayed in the Save As field of the panel.

Discussion
This method returns a result code from the `runModalForDirectory:file:` method of the `NSSavePanel` class: `NSFileHandlingPanelOKButton` if the user clicks the OK button or `NSFileHandlingPanelCancelButton` if the user clicks the Cancel button.

Use the `keychain` method to obtain the keychain created by the user.

Availability
Available in Mac OS X v10.3 and later.

See Also
`-runModalForDirectory:file:` (`NSSavePanel`)
`- keychain` (page 72)
`- beginSheetForDirectory:file:modalForWindow:modalDelegate:didEndSelector:contextInfo:` (page 71)

Declared In
`SFKeychainSavePanel.h`

setPassword:

Specifies the password for the keychain that will be created.

```
- (void)setPassword:(NSString *)password
```

Parameters
password
The password to be used for the new keychain.

Discussion
This method is optional. If you don't call this method, the keychain save panel displays a password-entry dialog.

Availability
Available in Mac OS X v10.3 and later.

Declared In
`SFKeychainSavePanel.h`

SFKeychainSettingsPanel Class Reference

Inherits from	NSPanel : NSWindow : NSResponder : NSObject
Conforms to	NSUserInterfaceValidations (NSWindow) NSAnimatablePropertyContainer (NSWindow) NSCoding (NSResponder) NSObject (NSObject)
Framework	/System/Library/Frameworks/SecurityInterface.framework
Declared in	SFKeychainSettingsPanel.h
Availability	Available in Mac OS X v10.3 and later
Companion guide	Keychain Services Programming Guide

Overview

The `SFKeychainSettingsPanel` class displays a panel or sheet that allows users to change their keychain settings.

Keychain settings include:

- Lock after a set period of inactivity
- Lock on sleep
- Synchronize using .Mac

The following figure shows an example of a keychain settings panel.

Figure 8-1 Keychain settings panel

Tasks

Returning a Shared Keychain Save Panel Object

`+ sharedKeychainSettingsPanel` (page 76)
Returns a shared keychain settings panel object. If the object has not already been created, this method allocates and initializes the object first.

Displaying a Sheet or Panel

`- beginSheetForWindow:modalDelegate:didEndSelector:contextInfo:settings:keychain:` (page 77)
Displays a sheet that allows users to change keychain settings.

`- runModalForSettings:keychain:` (page 78)
Displays a panel that allows users to change keychain settings.

Class Methods

sharedKeychainSettingsPanel

Returns a shared keychain settings panel object. If the object has not already been created, this method allocates and initializes the object first.

```
+ (SFKeychainSettingsPanel *)sharedKeychainSettingsPanel
```

Availability
Available in Mac OS X v10.3 and later.

Declared In
SFKeychainSettingsPanel.h

Instance Methods

beginSheetForWindow:modalDelegate:didEndSelector:contextInfo:settings:keychain:

Displays a sheet that allows users to change keychain settings.

```
- (void)beginSheetForWindow:(NSWindow *)docWindow modalDelegate:(id)delegate
    didEndSelector:(SEL)didEndSelector contextInfo:(void *)contextInfo
    settings:(SecKeychainSettings *)settings keychain:(SecKeychainRef)keychain
```

Parameters

docWindow
The parent window to which the sheet is attached. If this parameter is `nil`, the behavior defaults to a standalone modal window.

delegate
The delegate object in which the method specified in the `didEndSelector` parameter is implemented.

didEndSelector
A method selector for a delegate method called after the modal session has ended, but before the sheet has been dismissed. Implementation of this delegate method is optional.

contextInfo
A pointer to data that is passed to the delegate method. You can use this data pointer for any purpose you wish.

settings
A pointer to a keychain settings structure. Because this structure is versioned, you must preallocate it and fill in the version of the structure.

keychain
The keychain whose settings you wish to have the user change.

Discussion
The delegate method has the following signature:

```
-(void)createPanelDidEnd:(NSWindow *)sheet
       returnCode:(int)returnCode
       contextInfo:(void *)contextInfo
```

The parameters for the delegate method are:

sheet
The window to which the sheet was attached.

returnCode
The result code indicating which button the user clicked: either `NSFileHandlingPanelOKButton` or `NSFileHandlingPanelCancelButton`.

contextInfo
Client-defined contextual data that is passed in the `contextInfo` parameter of the `beginSheetForDirectory:...` method.

The delegate method may dismiss the keychain settings sheet itself; if it does not, the sheet is dismissed on return from the `beginSheetForDirectory:...` method.

Availability
Available in Mac OS X v10.3 and later.

See Also
`- runModalForSettings:keychain:` (page 78)

Declared In
`SFKeychainSettingsPanel.h`

runModalForSettings:keychain:

Displays a panel that allows users to change keychain settings.

```
- (NSInt)runModalForSettings:(SecKeychainSettings *)settings
    keychain:(SecKeychainRef)keychain
```

Parameters

settings

A pointer to a keychain settngs structure. Because this structure is versioned, you must preallocate it and fill in the version of the structure.

keychain

The keychain whose settings you wish to have the user change.

Discussion
The method result indicates which button the user clicks: `NSOKButton` or `NSCancelButton`.

If the user attempts to chanage the settings of a locked keychain, the unlock authorization dialog appears.

Availability
Available in Mac OS X v10.3 and later.

See Also
`- beginSheetForWindow:modalDelegate:didEndSelector:contextInfo:settings:keychain:` (page 77)

Declared In
`SFKeychainSettingsPanel.h`

Data Types

SecurityInterface Data Types Reference

Framework: SecurityInterface/SFAuthorizationView.h

Declared in SFAuthorizationView.h

Overview

This document describes the data types found in the Security Interface framework.

Data Types

SFAuthorizationViewState

Defines the current state of the authorization view.

```
typedef enum{
    SFAuthorizationStartupState,
    SFAuthorizationViewLockedState,
    SFAuthorizationViewInProgressState,
    SFAuthorizationViewUnlockedState
} SFAuthorizationViewState;
```

Discussion
These constants are described in "Constants" (page 32) in SFAuthorizationView.

Availability
Available in Mac OS X v10.3 and later.

Declared In
`SFAuthorizationView.h`

Document Revision History

This table describes the changes to *Security Interface Framework Reference*.

Date	Notes
2006-07-14	Added documentation for the SFAuthorizationPluginView class.
2006-05-23	Republication of this content as a collection of separate documents.

Document Revision History

Index

I

K

L

P

R

S

U

V

W