SFAuthorizationPluginView Class 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

SFAuthorizationPluginView Class Reference

Inherits from	NSObject
Conforms to	NSObject (NSObject)
Framework	/System/Library/Frameworks/SecurityInterface.framework
Availability	Available in Mac OS X v10.5 and later
Companion guide	Authorization Services Programming Guide
Declared in	SFAuthorizationPluginView.h
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 7)
`viewForType:` (page 12)

Tasks

Initializing an SFAuthorizationPluginView Object

- `initWithCallbacks:andEngineRef` (page 10)
  Returns an `SFAuthorizationPluginView` object with the specified callbacks and authorization engine handle.

Getting Instance Information

- `callbacks` (page 8)
  Returns the `AuthorizationCallbacks` structure with which this instance was initialized.
- `engineRef` (page 9)
  Returns the authorization engine handle with which this instance was initialized.

Responding to User Actions

- `buttonPressed:` (page 7)
  Informs the `SFAuthorizationPluginView` instance when a user presses a button in the custom view.
- `viewForType:` (page 12)
  Returns the appropriate `NSView` object for the specified `SFViewType` (page 13).

Configuring the User Interface

- `didActivate` (page 8)
  Informs the `SFAuthorizationPluginView` instance when the authorization plug-in makes the instance's user interface active.
- `didDeactivate` (page 8)
  Informs the `SFAuthorizationPluginView` instance when the authorization plug-in deactivates its user interface.
- `willActivateWithUser:` (page 12)
  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 9)
  Returns the first view in the keyboard loop of the view.
- `firstResponderView` (page 10)
  Returns the view that should get focus for keyboard events.

- `lastKeyView` (page 10)
  Returns the last view in the keyboard loop of the view.

Enabling and Disabling Controls

- `setEnabled:` (page 11)
  Enables or disables the controls in the `SFAuthorizationPluginView` instance's view.

Communicating with the Authorization Plug-in

- `displayView` (page 8)
  Displays the user interface provided by the `SFAuthorizationPluginView` subclass.
- `setButton:enabled:` (page 11)
  Enables or disables a button in the `SFAuthorizationPluginView` instance's user interface.
- `updateView` (page 11)
  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 8).

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 13).

```
- (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 13). 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`

Document Revision History

This table describes the changes to *SFAuthorizationPluginView Class Reference*.

Date	Notes
2006-07-14	New document that describes the class an authorization plug-in uses to display a custom view within the Apple-supplied authorization views.

Document Revision History

Index