AudioXplorer SDK 1.1 Developer's Guide

May 28, 2003

Table of content

Introduction

AudioXplorer can be extended through the use of plug-ins. Plug-ins are currently Cocoa bundle only: they can be written in a few minutes and are very easy to understand.

Plug-ins kind

AudioXplorer currently accepts two kind of plug-ins: single method and multiple methods. Here are the differences:

o Single method: the plug-in contains only one effect (also called method) that can be called to process the audio data.
o Multiple methods: the plug-in contains more than one effect (also called method) that can be called to process the audio data. It is a convenient way to write several effects within one plug-in rather to have one plug-in per effect.

Both kind of plug-in can display dialog to allow the user to modify the plug-in parameters.

What are inside the SDK ?

You will find two Project Builder projects:

o SinglePlugIn: this project can be used to create single method plug-ins
o MultiplePlugIn: this project can be used to create multiple methods plug-ins. It shows also how to create a dialog to allow the user to change parameters.

How to use the SDK ?

o Copy one of the project examples and customize it to implement your process.
o Walk through the code of SinglePlugIn.h/m or MultiplePlugIn.h/m and read the comments to better understand how the plug-in is working.
o The important thing is always to implement the AXPlugInSingleProtocol or AXPlugInMultipleProtocol protocol: these protocols provide the required methods to be loaded by AudioXplorer.

How to write a single method plug-in ?

To write a single method plug-in, they are two options:

- o Copy the SinglePlugIn project and customize it to your needs.

or

- o Create a plug-in from scratch using PB (create a Cocoa bundle and add the AXPlugInHeader.h file) .
- o Create your own Cocoa class and implement either AXPlugInSingleProtocol or AXPlugInMultipleProtocol.
- o Write the plug-in code inside your class (see below for more information on which method to use for single method plug-in).
- o Don't forget to set the main class of your bundle to the name of your class (the one derived from AXPlugInBase)).

How to write a multiple methods plug-in ?

To write a multiple methods plug-in, they are two options:

- o Copy the MultiplePlugIn project and customize it to your needs.

or

- o Create a plug-in from scratch using PB (create a Cocoa bundle and add the AXPlugInHeader.h file) .
- o Create your own Cocoa class and implement either AXPlugInSingleProtocol or AXPlugInMultipleProtocol.
- o Write the plug-in code inside your class (see below for more information on which method to use for multiple methods plug-in).
- o Don't forget to set the main class of your bundle to the name of your class (the one derived from AXPlugInBase)).

Plug-in method description

The following lines describe the method used in both type of plug-in.

Common plug-in method description

Single or multiple methods plug-ins MUST implement the following methods:

`- (long)supportAudioXplorerVersion`

Returns which version of AudioXplorer the plug-in supports. The following constant is currently defined:

AXPLUGIN_AX_VERSION_100	Version 1.00 of AudioXplorer

`- (long)versionID`

Returns the version ID of the plug-in (can be any number).

`- (long)plugInType`

Returns the type of the plug-in. The following constants are defined:

AXPLUGIN_TYPE_SINGLE	Single method plug-in
AXPLUGIN_TYPE_MULTIPLE	Multiple method plug-in

Use one of the above constant to define what kind of plug-in you are writing.

Single method plug-in methods description

`- (NSString*)methodTitle`

Returns the title of the method (will be display in the Effects menu in AudioXplorer).

`- (NSString*)methodName`

Returns the name of the method to be called. This is the name used to call your method (i.e. "myProcess:").

- (NSString*)aboutString

Returns the about string (will be displayed in the about box of the plug-in in AudioXplorer).

- (NSString*)versionString

Returns the version string (will be displayed in the about box of the plug-in in AudioXplorer).

- (NSString*)authorsString

Returns the authors string (will be displayed in the about box of the plug-in in AudioXplorer).

- (NSString*)descriptionString

Returns the description string (will be displayed in the about box of the plug-in in AudioXplorer).

- (BOOL)plugInWillBeCalled

AudioXplorer calls this method before it will call the process method to perform the job on the audio data. This allows the plug-in to perform any setup operation it needs (i.e. display a dialog box to query parameters from the user, etc.).

Returns YES if AudioXplorer can continue the operation or NO to cancel the operation.

- (void)myProcess:(id<AXPlugInParametersProtocol>)sender

AudioXplorer will call this method, after calling plugInWillBeCalled, to process sound data. The name of this method is the name you have provided in the methodName method (see above).
The sender parameter has the AXPlugInParametersProtocol protocol described later in this document. Look at the example projects to see how they are getting information from this parameter.

Multiple methods plug-in methods description

- (NSArray*)methodTitles

Returns an array containing the titles of all available methods (they will be display in the Effects menu in AudioXplorer).

- (NSArray*)methodNames

Returns an array containing the names of the all method to be called. These names are the names used to call your methods (i.e. "myProcess:").

- (NSString*)aboutStringOfMethod:(NSString*)methodName

Returns the about string (will be displayed in the about box of the plug-in in AudioXplorer) of the method.

- (NSString*)versionStringOfMethod:(NSString*)methodName

Returns the version string (will be displayed in the about box of the plug-in in AudioXplorer) of the method.

- (NSString*)authorsStringOfMethod:(NSString*)methodName

Returns the authors string (will be displayed in the about box of the plug-in in AudioXplorer) of the method.

- (NSString*)descriptionStringOfMethod:(NSString*)methodName

Returns the description string (will be displayed in the about box of the plug-in in AudioXplorer) of the method.

- (BOOL)plugInWillBeCalledWithMethod:(NSString*)methodName

AudioXplorer calls this method before it will call the process method (passed in parameter) to perform the job on the audio data. This allows the plug-in to perform any setup operation it needs (i.e. display a dialog box to query parameters from the user, etc.).

Returns YES if AudioXplorer can continue the operation or NO to cancel the operation.

`- (void)myProcess:(id<AXPlugInParametersProtocol>)sender`

AudioXplorer will call this method, after calling plugInWillBeCalledWithMethod, to process sound data. The name of this method is the name you have provided in the methodNames method (see above).
The sender parameter has the AXPlugInParametersProtocol protocol described later in this document. Look at the example projects to see how they are getting information from this parameter.

Parameter object

The parameter object passed to the process method of any plug-in has the AXPlugInParametersProtocol protocol. The following methods are currently available:

`- (unsigned short)channelCount`

Returns the number of available channel.

`- (float*)dataPointerOfChannel:(unsigned short)channel`

Returns a pointer to the sound data buffer of the specified channel. The sound data buffer is a 32 bits floating buffer: the value are in the range [-1; 1].

`- (unsigned long)dataCountOfChannel:(unsigned short)channel`

Returns the number of points in the data buffer of the specified channel.

`- (void)setError:(long)error`

Set the error, if any, that occurred during the process of the audio data. Use a value of 0 when no error occurred.

`- (void)setCancelled:(BOOL)flag`

Use this method to cancel the operation. Currently not used.

`- (BOOL)firstTime`

Returns if the method is called for the first time during a process.

Localized plug-in

Plug-in can be localized in the same way as standard application. However, the NSLocalizedString() function seems to consider only the Localizable.strings file in AudioXplorer, not in the plug-in. The current work-around is to use the AXLocalized() macro defined in the AXPlugInHeader.h: this macro will use the Localizable.strings file located in the plug-in bundle.

Example:

myString = AXLocalized(@"Reverse")

Contact

If you have any question, comments or suggestions, don't hesitate to send an e-mail to info@arizona-software.ch.