

Getting Started with the API

The Surround SCM API provides developers with a dynamic library that can be used to create custom tools that leverage the Surround SCM Server. The Surround SCM API is available as a C library, a Java library, and a .NET assembly (Windows only). This guide provides a basic orientation to the Surround SCM API and the key concepts that are necessary for using it.

The Surround SCM API documentation, which includes all available packages, classes, data structures, and examples, is installed along with the API library files in your Surround SCM application directory in the following locations:

- C API documentation—API/docs/C/index.html
- Java API documentation—API/docs/Java/index.html
- .NET API documentation—API/docs/DotNet/index.html

Note: The APIs are not thread safe. Limit the use of any Surround SCM API to a single thread.

Avoiding conflicts with the Surround SCM Client and CLI

The Surround SCM API check out commands do not use working directories, which allows more flexibility when integrating with other tools. When using the API you should not check out files to a working directory because this can cause conflicts with the Surround SCM Client and CLI. To avoid these conflicts, you should use separate user accounts for the API.

Using the C API

The C API is separated into a set of data structures and functions. When working with the C API, the only header that needs to be included in your code is the `sscapi.h` file. Keep the following in mind:

- If you are working on Windows, you need to include the `sscapi.dll` and `sscapi.lib` files in your Visual Studio project.
- If you are working on Mac OS X, you need to link against the `libsscapi.dylib` file.
- If you are working on Linux, you need to link against the `libsscapi.so` file.
- The C API uses UTF-8 encoding for all string parameters and return values. If you have strings in other Unicode encodings, you must convert them to UTF-8 before passing them to Surround SCM API functions.

Source code examples

Source code examples are included in the C API documentation for every function that is used to interact with the Surround SCM Server. Refer to these examples if you want to perform a specific operation. The examples are located in the `API/docs/C/examples` directory in the Surround SCM application directory.

Key concepts

All C API calls use an `SSCMContext` structure as the container that holds data for the currently connected session. The `pMainline` and `pBranch` members are used by the other functions to identify the mainline and branch to perform operations against. The `cookie` member is initially populated inside the `sscm_connect` function. The `statusCallback` function pointer is optional, but if it is set the C API uses it to return string-based status messages about long-running operations.

You typically begin using the C API with a call to `sscm_connect`, which establishes the initial `SSCMContext`. You then set the `SSCMContext` mainline and branch fields to those currently being used. After all calls are completed, make a final call to `sscm_disconnect` to close the session.

Note: Pay close attention to the `sscm_free_xxxx` calls (e.g., `sscm_free_branch`, `sscm_free_file_props`, etc.) to ensure that memory is properly being released. Refer to the examples in the Surround SCM API Documentation for each call to determine when these memory management calls are needed.

About the `SSCMResult` result

All C API calls also return an `SSCMResult` result. This result code indicates the status of the last call made and several enumerated values are defined. You can use the `sscm_get_last_error` function to return a string based on the error code along with the error generated by the Surround SCM Server. Only the server error string for the last error is stored by the library. The call string is cleared on the next successful function call.

Using the .NET API

The .NET API is separated into a set of classes. Keep the following in mind:

- You need to add the `sscmapi.dll` file as a reference to your .NET project in Visual Studio before you can use the API. This file is located in the `\API\lib` folder in the Surround SCM directory.
- All timestamps received from the .NET API are in GMT. Make sure all timestamps sent to the .NET API are also in GMT.

Key concepts

All .NET API calls use an `SSCMContext` class as the container that holds data for the currently connected session. The `SetBranch()` and `SetMainline()` methods are used by the other functions to identify the mainline and branch to perform operations against.

You typically begin using the .NET API with a call to `SSCMAPI.Connect`, which establishes the initial context by taking an `SSCMContext` parameter as an out variable. You then set the `SSCMContext` mainline and branch fields to those currently being used. After all calls are completed, make a final call to `SSCMAPI.Disconnect` to close the session.

About the SSCMResult class

All .NET API calls also return an SSCMResult class. This result code indicates the status of the last call made and several methods are provided to discover which error was returned. You can use the SSCMAPI.GetLastError static method to return a string based on the error code along with the error generated by the Surround SCM Server. Only the server error string for the last error is stored by the library. The call string is cleared on the next successful function call.

Using the Java API

The Java API is separated into a set of packages and classes. Keep the following in mind:

- The main class is SSCMAPI, which includes methods for all the primary actions that can be performed against the Surround SCM Server. This class is located in the com.seapine.surroundscm.api package.
- The Surround SCM API library file must be in the application path for the solution to run.

Key concepts

All Java API calls use an SSCMContext class as the container that holds data for the currently connected session. The Mainline and Branch members are used by other functions to identify the mainline and branch to perform operations against. The Cookie member is initially populated inside the connect function.

You typically begin using the Java API with a call to SSCMAPI.Connect, which establishes the initial SSCMContext. You then set the context to the mainline and repository being used as needed. After all calls are completed, make a final call to SSCMAPI.Disconnect to close the session.

About the SSCMResult class

All calls also return an SSCMResult class or one of its subclasses. This result code indicates the status of the last call made and several enumerated values are defined. You can use the SSCMAPI.getLastError static method to return a string based on the error code along with the error generated by the Surround SCM Server. Only the server error string for the last error is stored by the API. The call string is cleared on the next successful function call.

Using the Java API on Mac OS X

The surroundscm-api.jar looks for the native library with the file name libsscmapapi.jnilib. The native library included with the Java API is named in the format libsscmapapi.x.y.z.dylib where x, y, and z are the major, minor, and point release version numbers respectively. To use the Java API on Mac OS X, create a symlink called libsscmapapi.jnilib that points to the native library.

Using the Java API on Linux

The `surroundscm-api.jar` looks for the native library with the file name `libsscmapapi.so`. The native library included with the Java API is named in the format `libsscmapapi.so.x.y.z` where `x`, `y`, and `z` are the major, minor, and point release version numbers respectively. To use the Java API on Linux, create a symlink called `libsscmapapi.so` that points to the native library.