



SquidNet

Network Render Manager

SDK/API User's Guide

Version 1.0

Table of Contents

ABOUT THIS DOCUMENT:	3
1 PREREQUISITES	1
2 INTRODUCTION	1
3 WHERE IS IT?	1
4 HOW DO I BUILD MY APPLICATION?	1
5 API FUNCTIONS	2
6 SAMPLE APPLICATION	4

SquidNet API User's Guide

About this document:

This document will describe how to use SquidNet's C/C++ SDK/API.

It's assumed that the reader is familiar with the rendering and compositing applications that are installed on their system and, as such, this manual will not attempt to cover any material related to those applications.

For any suggestions or comments, feel free to contact us; contact information can be found in the Help menu in the SMC console and on our website at <http://www.squidnetsoftware.com/>.

Always check website <http://www.squidnetsoftware.com> for the latest documentation updates.

SquidNet API User's Guide

1 Prerequisites

Before reading this document, make sure that you have the following tasks completed:

1. The network is fully operational with all network shares (mapped drives, UNC paths, NAS servers, etc...) fully accessible by all nodes.
2. SquidNet is properly installed and configured on all relevant machines.

2 Introduction

SquidNet provides a C/C++ SDK/API to allow software developers to develop their own applications and plug-ins to submit jobs to the network render queue.

Perform the following actions in your code to submit a job:

1. Initialize SquidNet API interface:
 - a. Call **sqnInitialize()** before using any API functions.
2. Determine application template to use:
 - a. Include template header file in to your source code and call **sqnJobCreate()** function.
3. Set job request parameters:
 - a. Set template-specific options using **sqnJobValueSet()** function. Macro options are listed in template header file.
4. Submit request:
 - a. Call **sqnJobCommit()** to submit job request.
5. Check for status and results:
 - a. Check for SQN_STATUS_OK return status.

3 Where is it?

You can find the SquidNet API in the installation folder under the "sqnApi" folder. In the folder, you find the following subfolders:

- **include:** API header file and application specific header files.
- (WINDOWS) **win32:** Platform specific static library - - sqnApi.lib
- (LINUX) **linux32:** Platform specific static library - - libsqnApi.a
- (OSX X) **osx32:** Platform specific static library - - libsqnApi.a

4 How do I build my application?

Assuming your application is called test.cpp, you can compile/link your application as follows:

- **WINDOWS:** (Visual Studio):
 - Project dependencies are: ws2_32.lib pthreadVC2.lib IPHlpApi.Lib mpr.lib Psapi.lib
- **LINUX** (gcc/g++):
 - g++ test.cpp libsqnApi.a -lpthread -lncurses
- **OSX** (gcc/g++)

SquidNet API User's Guide

- g++ test.cpp libsqnApi.a -Incurses -m32 -arch i386 -framework CoreServices -framework CoreFoundation -framework SystemConfiguration -framework IOKit

5 API functions

Function	Description
sqnInitialize()	Initialize SquidNet API. Must be called prior to using any API functions. Returns: SQN_STATUS
sqnJobCreate()	Creates job context for the specified application template. Returns: SQN_STATUS
sqnJobValueSet()	Set value for a specific option. Returns: SQN_STATUS
sqnJobValueGet()	Return value for a specific option. Returns: Constant character string pointer to value or NULL if no value is present.
sqnJobCommit()	Send job to network queue. Returns: SQN_STATUS
sqnJobDestroy()	Free resources used up by SQN_JOB_CTX object. Returns: SQN_STATUS
sqnJobMandatoryList()	Call this function if SQN_STATUS_MISSING_MANDATORY_OPTIONS value is returned from sqnJobCommit() . A list of mandatory options missing from job request is returned. Returns: Constant character string pointer to list of mandory options or NULL if no values are present.
sqnJobUnknownName()	Call this function if SQN_STATUS_JOB_REQUEST_VALUE_INVALID value is returned from sqnJobCommit() . The name of the option that contains an invalid value is returned. Returns: Constant character string pointer to invalid option or NULL if no value is present.
sqnErrorStr()	Returns string associated with a SQN_STATUS code. Returns: Constant character string pointer to value or NULL if no value is present.
sqnCmdLineUserDefined()	Define user specific command line option. Returns: SQN_STATUS
sqnCmdLineIsPassed()	Determine if a specific option was passed from command line interface. Returns: Boolean value.
sqnCmdLineGet()	Get value of specific option from command line interface.

SquidNet API User's Guide

	Returns: Constant character string pointer to value or NULL if no value is present.
sqnCmdLineSet()	Set value for specific option in command line interface Returns: SQN_STATUS

Notes:

1. All values set with `sqnCmdLineSet()` are automatically copied to all **SQN_JOB_CTX** contexts returned from the **sqnJobCreate()** function.
2. All values are passed and returned from job context are constant character strings (const char *). **DO NOT free these pointers with free() or delete.**
3. If not using command line arguments, use "**sqnInitialize(0, NULL)**" to initialize API.

6 Sample application

```

#include "../sqnApi/include/sqn.h"
#include "../sqnApi/include/sqn_LightWave.h"
#include "stdio.h"

int main(int argc, char* argv[])
{
    int retCode = 0;

    //
    // --- Initialize SquidNet API
    //
    SQN_STATUS st = sqnInitialize( 0, NULL );
    if ( st == SQN_STATUS_OK )
    {
        SQN_JOB_CTX ctx;

        //
        // --- Create SquidNet job request context ...
        //
        st = sqnJobCreate( &ctx, SQN_LIGHTWAVE_TEMPLATE );
        if ( st == SQN_STATUS_OK )
        {
            //
            // --- Set job request parameters...
            //
            sqnJobValueSet( ctx, SQN_ARG_JOB_IDENTIFIER, "Lightwave Scene 001" );
            sqnJobValueSet( ctx, SQN_ARG_JOB_START_FRAME, "1" );
            sqnJobValueSet( ctx, SQN_ARG_JOB_END_FRAME, "5" );
            sqnJobValueSet( ctx, SQN_ARG_JOB_FRAMES_PER_SLICE, "1" );
            sqnJobValueSet( ctx, SQN_ARG_JOB_PRIORITY, "19" );
            sqnJobValueSet( ctx, SQN_LIGHTWAVE_CONFIG_FOLDER, "\\raid-server00\\Volume_1\\SquidNet\\LightWaveProjects\\LW10\\FromRalph" );
        };
        sqnJobValueSet( ctx, SQN_LIGHTWAVE_CONTENT_FOLDER, "\\raid-server00\\Volume_1\\SquidNet\\LightWaveProjects\\Except_GiChapel" );
    };
    sqnJobValueSet( ctx, SQN_ARG_JOB_SCENE_FILE, "\\raid-server00\\Volume_1\\SquidNet\\LightWaveProjects\\Except_GiChapel\\Scenes\\Chapel_v002.lws" );
    sqnJobValueSet( ctx, SQN_ARG_JOB_USERNAME, "MyFirstName" );
    sqnJobValueSet( ctx, SQN_ARG_JOB_COMMENT, "Test scene: Lightwave Scene 001" );
    sqnJobValueSet( ctx, SQN_ARG_JOB_NOTES, "Some notes for this job" );
    sqnJobValueSet( ctx, SQN_ARG_JOB_APP_PATH, "$APP(LIGHTWAVE)" );
    sqnJobValueSet( ctx, SQN_ARG_JOB_ERROR_BEFORE_STOP, "20" );
    sqnJobValueSet( ctx, SQN_ARG_JOB_OUTPUT_DIR, "\\RAID-SERVER00\\Volume_1\\SquidNet\\LightWaveProjects\\Except_GiChapel\\Images" );
    sqnJobValueSet( ctx, SQN_ARG_JOB_IMAGE_PREFIX, "chapel " );
    sqnJobValueSet( ctx, SQN_ARG_JOB_IMAGE_SIZE, "SVGA (800 x 600)" );
    sqnJobValueSet( ctx, SQN_ARG_JOB_IMAGE_FORMAT, "LW_PICT32(.pct)" );
    sqnJobValueSet( ctx, SQN_ARG_JOB_PROCESSING_NODES, "RENDERNODE11,RENDERNODE12" );
    sqnJobValueSet( ctx, SQN_ARG_JOB_COMPLETION_ACTION, NULL );
    sqnJobValueSet( ctx, SQN_ARG_DISPLAY_REQUEST, NULL );

    //
    // --- Submit job request...
    //
    st = sqnJobCommit( ctx );
    if ( st != SQN_STATUS_OK )
    {
        if ( st == SQN_STATUS_MISSING_MANDATORY_OPTIONS )
            printf("%s \n(Missing options:%s)\n", sqnErrorStr( st ), sqnJobMandatoryList( ctx ));
        else if ( st == SQN_STATUS_JOB_REQUEST_VALUE_INVALID )
            printf("\n%s \n\nValue is not valid: %s\n", sqnErrorStr( st ), sqnJobUnknownName( ctx ));
        else
            printf("%s\n", sqnErrorStr( st ));
        retCode = 1;
    }

    //
    // --- Always destroy context when no longer using...
    //
    sqnJobDestroy( ctx );
}
else
    printf("%s\n", sqnErrorStr( st ));
}
else
    printf("sqnInitialize() returned %u:%s\n", st, sqnErrorStr( st ));

return( retCode );
}

```