Sequence Log

ver.1.0.1

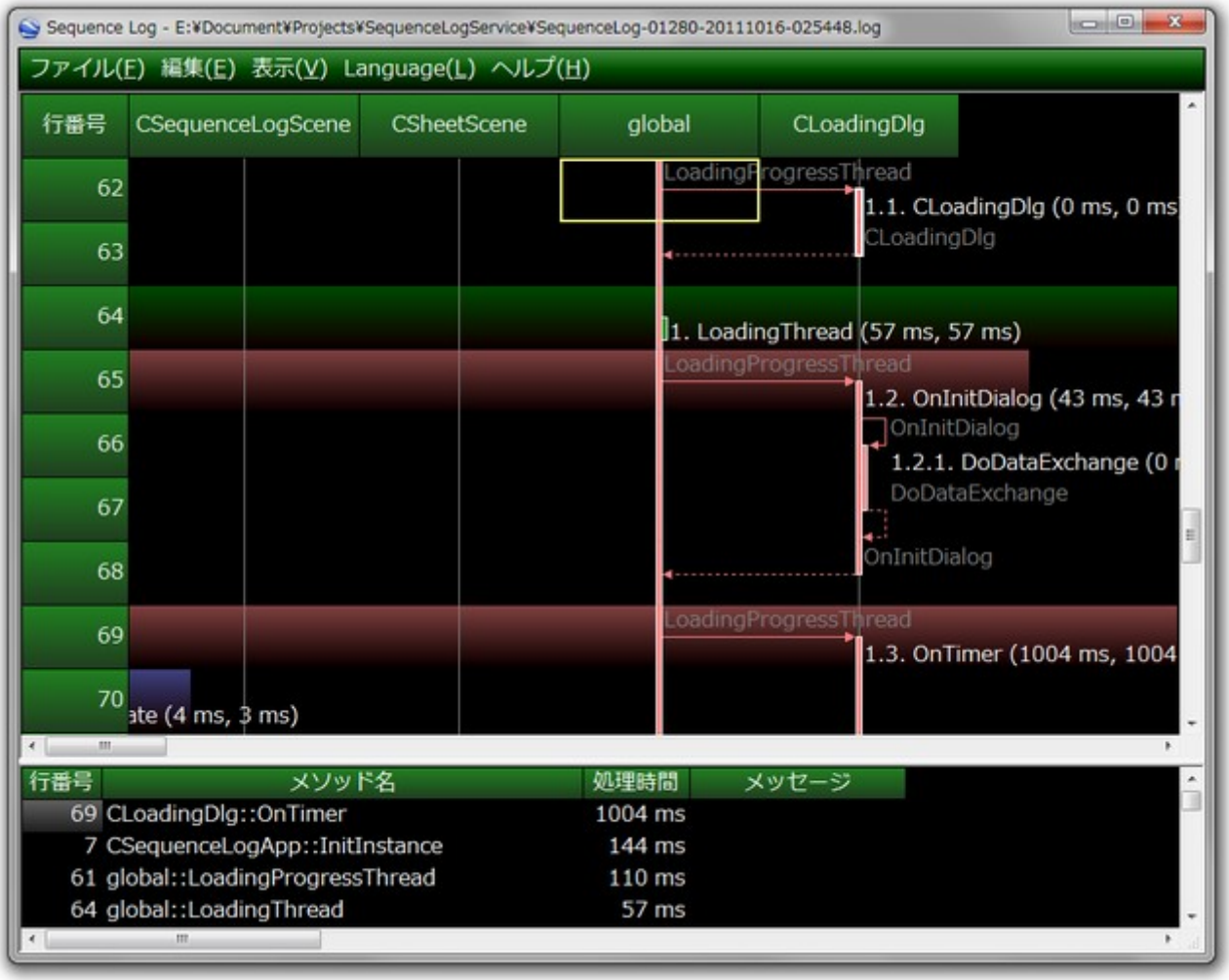
log-tools.net

2011/12/09

目次

1 はじめに

Sequence Log はソースコードにログを記述し、実行した結果をシーケンス図（風）として表示するソフトウェアです。また、シーケンスログを出力するために構成された他のソフトウェア（Sequence Log Service、Sequence Log Print）、及び各 OS 毎に用意された Sequence Log Library の総称です。



Sequence Log

2 特長

- ログの出力を別スレッドで行うため高速です。
- 実際の出力は Sequence Log Service が行うため、出力元のソフトウェアが何らかの理由で突然動作を停止しても、ログは失われません。
- 条件を満たした場合のみログを出力します。不要なログ出力を抑えることで解析を容易にします。
- スレッド毎に色付けがされて見やすい。
- コールスタック表示も可能です。デッドロック等で処理が止まっている箇所の発見に役立ちます。
- ログファイルのサイズやログファイル数の上限を設定できます。

3 詳細説明

シーケンスログでは、出力結果をシーケンス図（風、以下略）として表示するために、メソッドが呼ばれたところと抜けるところにログを記述します。完全なシーケンス図を得るためには全てのメソッドに記述する必要がありますが、ログのないメソッドは単にシーケンス図に現れないだけなので、初めは必要なところにだけ記述しても良いでしょう。

以下で詳細説明をするにあたり、実例はC++で示すこととしますのでご了承下さい。

3.1 出力フラグ

シーケンスログは内部で２つの出力バッファを持っています。１つは保管庫としてのバッファで、ここにあるログはまだ出力されるかどうか分からない状態のログです。もう１つは出力が確定したログを格納するためのバッファです。

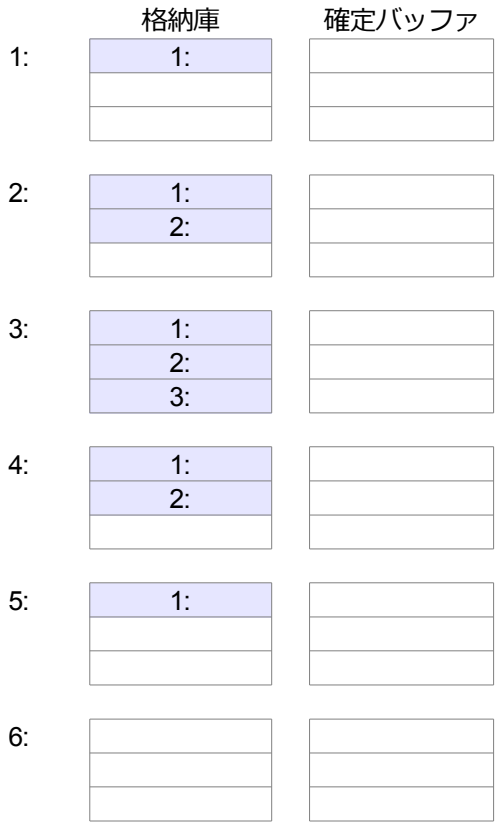
そして、それを制御するための出力フラグというものがあります。出力フラグはKEEP、OUTPUT_ALL、ALWAYS、ROOTの４つです。

3.1.1 KEEP

KEEP はもっとも良く使われるフラグで、通常はログを出力せずに破棄されます。

```
    void App::init()
    {
1:      SLOG(“App”, “init”, slog::KEEP);
        mWindow.init();
6:  }

    void Window::init()
    {
2:      SLOG(“Window”, “init”, slog::KEEP);
        mMenu.init();
5:  }

    void Menu::init()
    {
3:      SLOG(“Menu”, “init”, slog::KEEP);
4:  }
```

バッファの様子



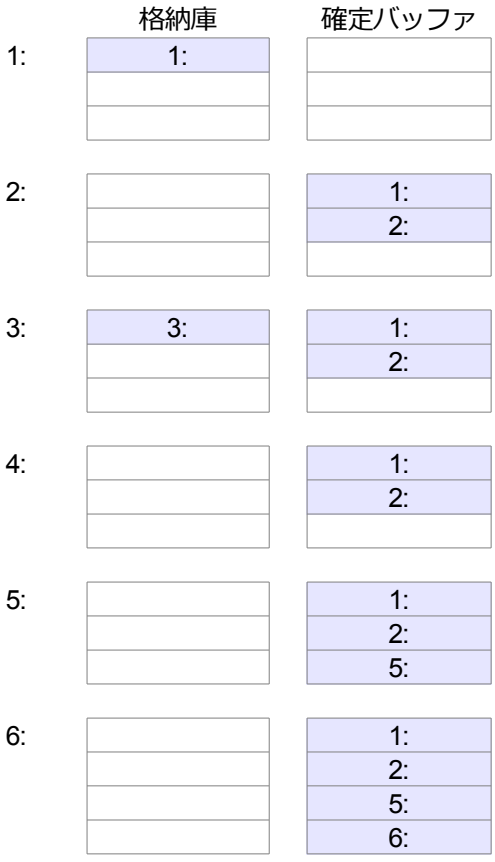
また、4～6 でもログを出力しようとしますが、格納庫にすら入ることなく破棄されます。

3.1.2 OUTPUT_ALL

OUTPUT_ALL は格納庫のログを確定バッファに移します。

```
    void App::init()
    {
1:      SLOG(“App”, “init”, slog::KEEP);
        mWindow.init();
6:  }

    void Window::init()
    {
2:      SLOG(“Window”, “init”, slog::OUTPUT_ALL);
        mMenu.init();
5:  }

    void Menu::init()
    {
3:      SLOG(“Menu”, “init”, slog::KEEP);
4:  }
```

バッファの様子



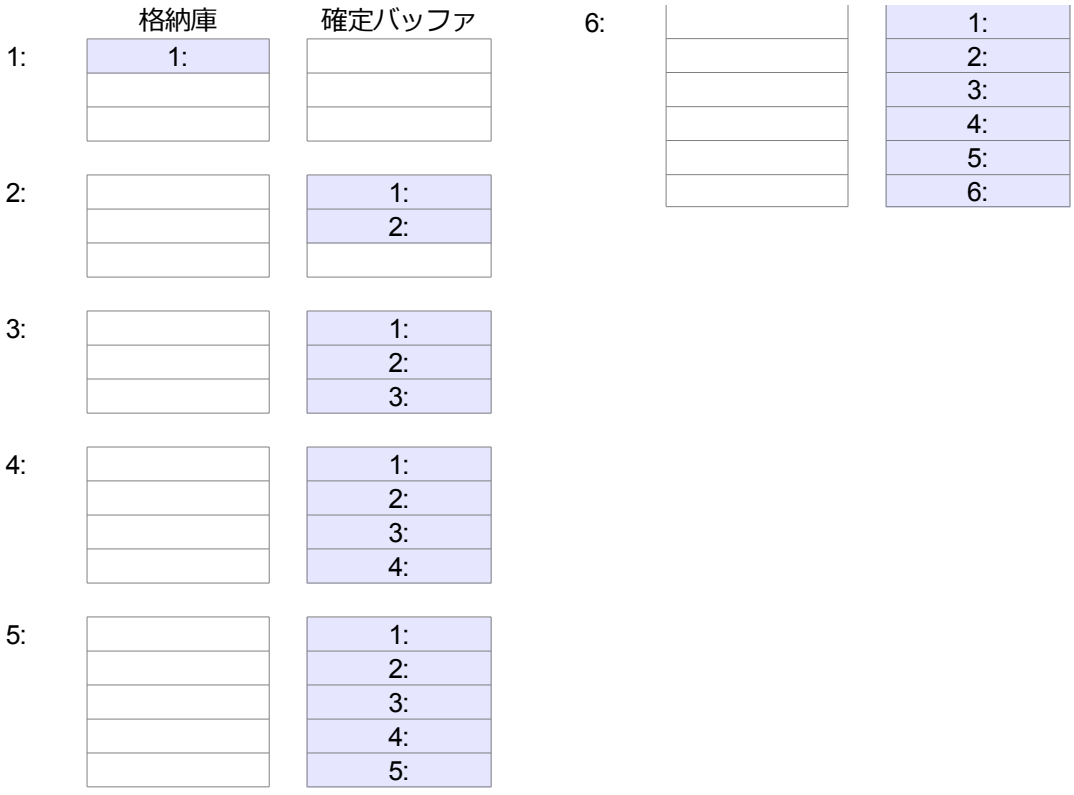
本来なら 6 は破棄されるところですが、1 が確定バッファにあるので（というより格納庫にないので）出力されます。

3.1.3 ALWAYS

ALWAYS は格納庫のログを確定バッファに移し、配下メソッドでのログは出力フラグによらず即確定バッファに移します。

```
    void App::init()
    {
1:      SLOG(“App”, “init”, slog::KEEP);
        mWindow.init();
6:  }

    void Window::init()
    {
2:      SLOG(“Window”, “init”, slog::ALWAYS);
        mMenu.init();
5:  }

    void Menu::init()
    {
3:      SLOG(“Menu”, “init”, slog::KEEP);
4:  }
```

バッファの様子



3.1.4 ROOT

ROOT は、ROOT を ALWAYS として扱うようにシーケンスログサービスで設定されている場合は ALWAYS として、そうでなければ KEEP として動作します。例えば onCreate()や onDestroy()といったイベントハンドラで ROOT を使用して、開発段階では全てのログを、リリース版では必要最小限のログを出力するといった使い方が出来ます。

3.1.5 応用

下記は出力フラグを応用した少し複雑な例です。

```
    void App::init()
    {
1:      SLOG(“App”, “init”, slog::KEEP);
        mWindow.init();
        mWindow.move();
        mWindow.show();
10: }

    void Window::init()
    {
2:      SLOG(“Window”, “init”, slog::ALWAYS);
        mMenu.init();
5:  }

    void Menu::init()
    {
3:      SLOG(“Menu”, “init”, slog::KEEP);
4:  }

    void Window::move()
    {
6:      SLOG(“Window”, “move”, slog::KEEP);
7:  }

    void Window::show()
    {
8:      SLOG(“Window”, “show”, slog::OUTPUT_ALL);
9:  }
```

バッファの様子

	格納庫	確定バッファ
1:	1:	
2:		1:
		2:
3:		1:
		2:
		3:
4:		1:
		2:
		3:
		4:
5:		1:
		2:
		3:
		4:
		5:
6:	6:	1:
		2:
		3:
		4:
		5:
7:		1:
		2:
		3:
		4:
		5:
8:		1:
		2:
		3:
		4:
		5:
		8:
9:		1:
		2:
		3:
		4:
		5:
		8:
		9:
10:		1:
		2:
		3:
		4:
		5:
		8:
		9:
		10:

4 ソフトウェア構成

	Windows	Linux	Android
Sequence Log	SequenceLog.exe (*1)	—	—
Sequence Log Service	SequenceLogService.exe (*2)	slogsvc	SequenceLogService.apk libslogsvc.so
Sequence Log Print	SequenceLogPrint.exe	slogprint	—
Sequence Log Library (C/C++)	SequenceLog.lib SequenceLogd.lib SequenceLogMD.lib SequenceLogMDd.lib	libslog.so	libslog.so libslog.a
Sequence Log Library (C#)	slogcs.dll	—	—
Sequence Log Library (Java)	slog.jar slog.dll	slog.jar libslog.so	slog.jar libslog.so libslog.a
Sequence Log Library (PHP)	–	slog.so	–
Sequence Log Id	SequenceLogId.exe	—	—

(*1) 販売価格¥10,500（税込）。ライセンスがない場合は一部の機能がご使用できません。使用期間は無制限です。

(*1)(*2)http://www.log-tools.net/からダンロードして下さい。

4.1 Sequence Log 機能一覧

機能	ライセンスあり	ライセンスなし
シーケンス図表示	✓	✓
コールスタック表示	✓	✓
メソッドコールの戻りを表示	✓	✓
戻り先メソッド名表示	✓	✓
処理時間表示	✓	✓
処理時間バー表示	✓	
実行時間表示	✓	✓
ログを行頭に表示	✓	
図を非表示	✓	
呼び出し元メソッド名表示	✓	✓
呼び出し№表示	✓	
ハイライト	✓	✓
ピックアップ	✓	✓
ログファイルマージ表示	✓	

詳細は 10 Sequence Log 参照。

5 フォルダ構成

slog	ルートフォルダ
bin Android Java Windows x64 x86	バイナリフォルダ
doc	ドキュメントフォルダ
doxygen SequenceLogLib SequenceLogService	doxygen フォルダ
include slog	外部公開用インクルードフォルダ
src include SequenceLogLib SequenceLogPrint SequenceLogService	ソースファイルフォルダ

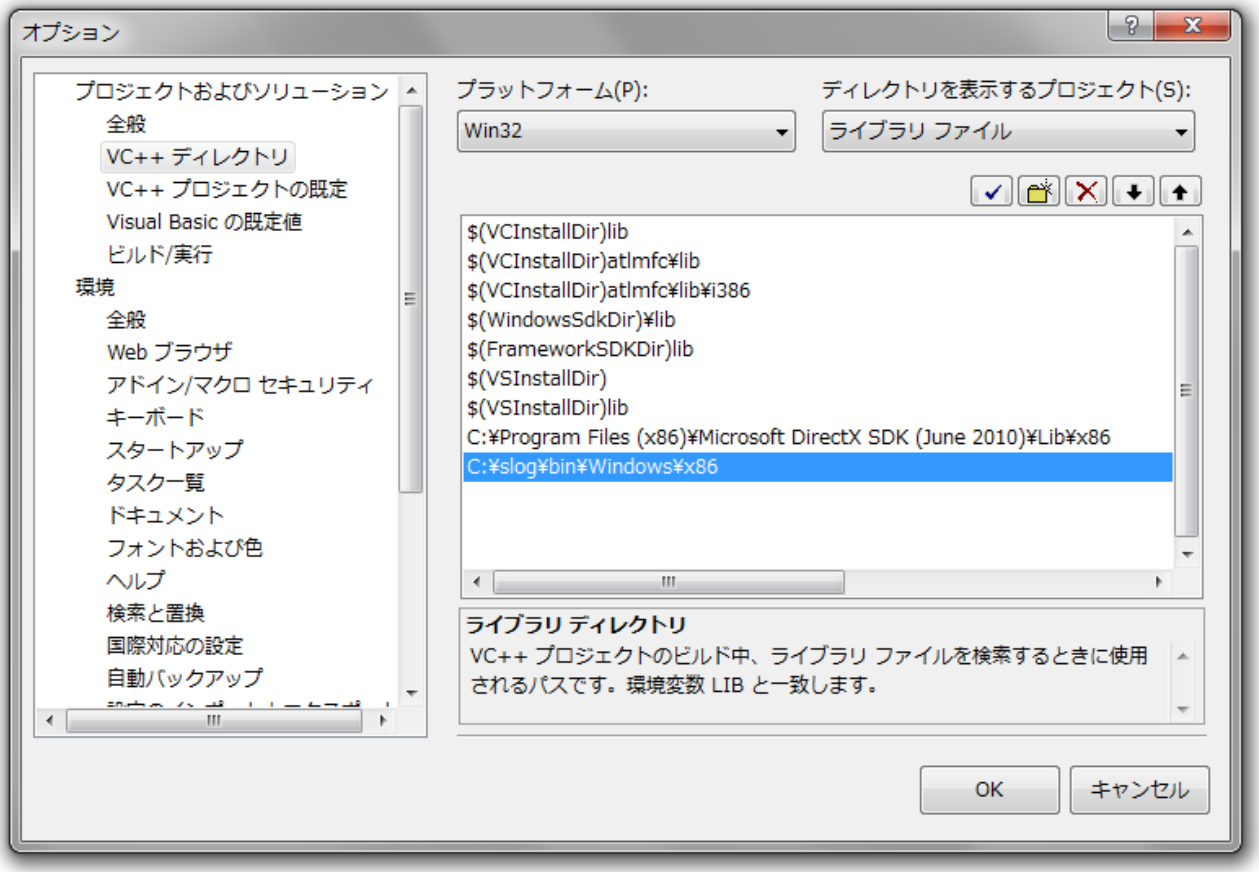
本ドキュメントでパスを表記する場合、slog/（ルートフォルダ）から始まるパスで示しておりますので、実環境に合わせて置き換えて下さい。特に注意の必要な箇所では念のため赤字で記述しています。

6 導入方法

6.1 Windows (Visual Studio 2008)

6.1.1 C/C++

1. [ツール] - [オプション]でオプションダイアログを開きます。



2. 左側の一覧から[プロジェクトおよびソリューション] – [VC++ ディレクトリ]を選択します。

3. [プラットフォーム]で Win32 を、[ディレクトリを表示するプロジェクト]でライブラリ ファイルを選択します。

4. ディレクトリ一覧に slog¥bin¥Windows¥x86 を追加します。

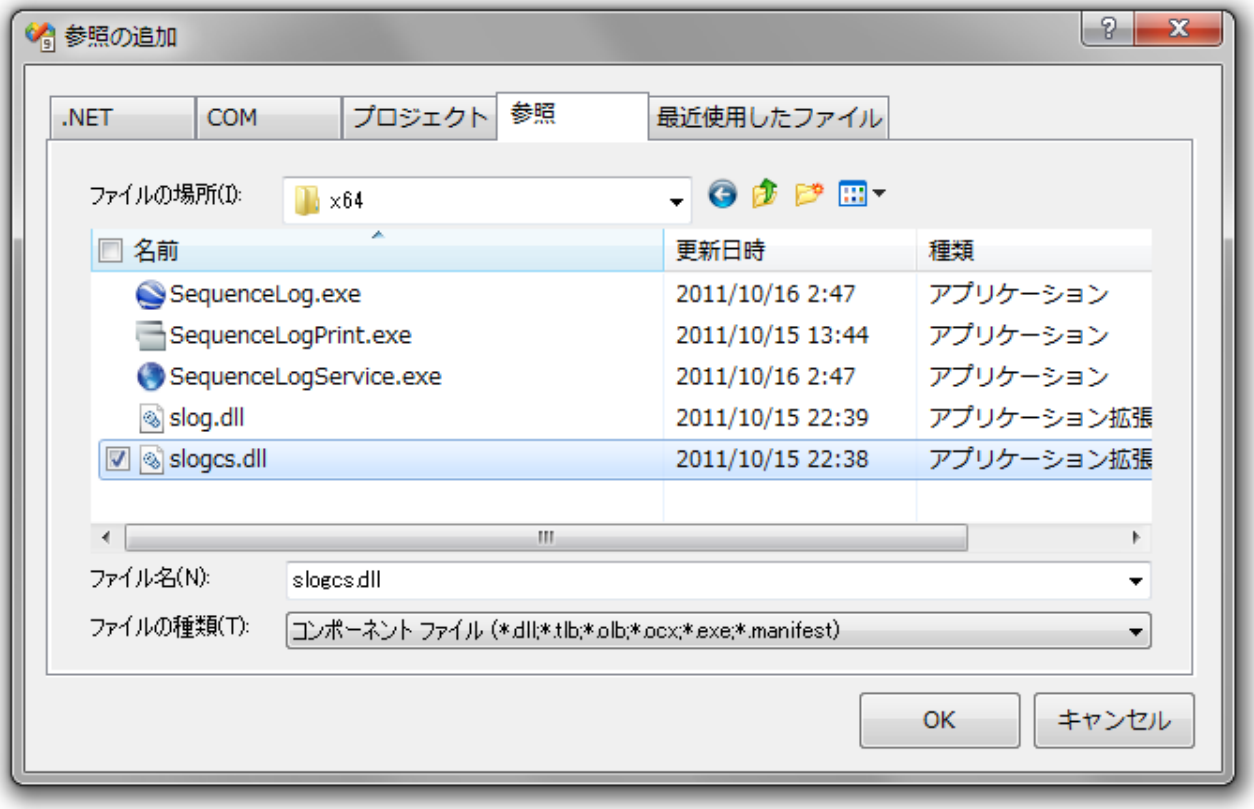
   Win64 の場合は[プラットフォーム]で x64 を選択し、ディレクトリ一覧に slog¥bin¥Windows¥x64 を追加します。

5. 同じように[ディレクトリを表示するプロジェクト]でインクルード ファイルを選択し、ディレクトリ一覧に slog¥include を追加します。

6.1.2 C#

1. C#のプロジェクトを開きます。

2. [プロジェクト] - [参照の追加]で参照の追加ダイアログを開きます。



3. 32 ビット環境では slog¥bin¥Windows¥x86¥slogcs.dll を、64 ビット環境では slog¥bin¥Windows¥x64¥slogcs.dll を選択します。間違えると System.BadImageFormatException が発生するので注意して下さい。

6.1.3 Java

1. 環境変数、またはコマンドプロンプトでクラスパスを設定します。コマンドプロンプトで設定する場合は以下のようにして下さい。

   set CLASSPATH=%CLASSPATH%;slog/bin/Java/slog.jar

2. 実行時は slog.dll のパスを java コマンドの-D オプションで指定します。32 ビット環境では以下のように、

   java -Djava.library.path=slog/bin/Windows/x86 [yourApp]

   64 ビット環境では以下のように指定します。

   java -Djava.library.path=slog/bin/Windows/x64 [yourApp]

6.2 Linux

6.2.1 C/C++

1. libslog.so をビルドするために、カレントディレクトリを変更します。

   cd slog/src/SequenceLogLib/src

2. ソースファイルの依存関係を調べます。

   make depend

3. ビルドを行います。

   make

4. インストールします。

   sudo make install

   sudo cp -rf ../../../include/slog /usr/include/

5. slogsvc と slogprint も同様にビルド＆インストールします。

   cd ../../SequenceLogService/src

   make depend

   make

   sudo make install

   cd ../../SequenceLogPrint/src

   make depend

   make

   sudo make install

6.2.2 Java

1. libslog.so をビルドします（6.2.1 参照）。

2. クラスパスを設定します。

   export CLASSPATH=$CLASSPATH:slog/bin/Java/slog.jar

6.2.3 PHP

1. libslog.so をビルドします（6.2.1 参照）。

2. slog.so をビルド＆インストールします。

   cd slog/src/SequenceLogLib

   pecl-gen slog.xml

   cd slog

   phpize

   ./configure

   make

   sudo make install

6.3 Android

6.3.1 C/C++

ここでの作業環境はLinuxを前提としています。

1. libslog.soとlibslog.aをビルドするために、カレントディレクトリを変更します。

   cd slog/src/SequenceLogLib/jni

2. ビルドとインストールを行います。

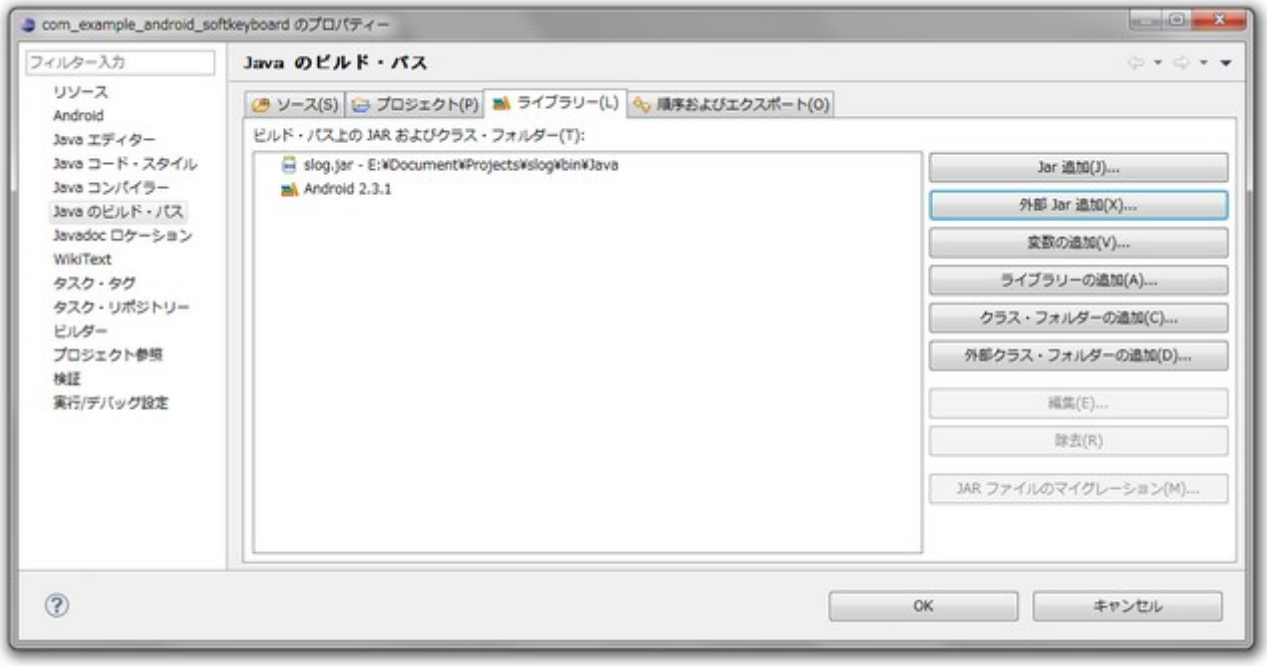
   ./make.sh

6.3.2 Java (Eclipse)

1. libslog.so をアプリケーションのプロジェクトフォルダにコピーします。

   cp slog/bin/Android/libslog.so [yourAppProj]/libs/armeabi/

2. Eclipse の[ファイル] – [プロパティー]でプロパティーダイアログを開きます。



3. 左側の一覧から[Java のビルド・パス]を選択し、[ライブラリー]タブを選択します。

4. [外部 Jar 追加]を選択し、slog/bin/Java/slog.jar を追加します。

5. Sequence Log Service とログ出力元はソケットを通じて情報の受け渡しを行います。また、ログの出力においては共有メモリ（mmap）を使用するため、AndroidManifest.xml に以下の設定を追加します。

```
<uses-permission android:name="android.permission.INTERNET">
</uses-permission>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE">
</uses-permission>
```

7 リファレンス

7.1 シーケンスログファイル名について

シーケンスログファイル名は以下のフォーマットで作成されます。

[任意]-[プロセス ID]-yyyymmdd-hhmiss-msec.拡張子

プロセス ID と作成日時は自動的に付加しますので、後述の setSequenceLogFileName() や Log.SetFileName()等でファイル名を指定する際には[任意].拡張子で指定して下さい。

拡張子が.slog の場合はバイナリ、それ以外はテキストでログを出力します。

7.2 ID について

クラス名やメソッド名、メッセージは、文字列以外に ID を使用することも出来ます。ID を使用すると文字列操作が不要なためログ出力時の負荷を下げることができ、さらにログファイルサイズの節約にも繋がります。また、ログファイルだけではプログラムの詳細な情報を取得しにくくなりますので、リリースバージョンにログ出力を含めたままでも情報を隠蔽できます（すべてのメソッドを ID 化することは現実的ではないかもしれませんが）。

SequenceLog.exe でログを参照する際には、ID とクラス名を関連付ける ID 定義ファイル（拡張子.sid）をログファイルと共に使用して下さい。

ID 定義ファイルは、”ID（数値）,文字列”で記述されたファイルです。
詳細は 9 Sequence Log Id 参照。

7.3 文字コードについて

SJIS、またはUTF-8が使用出来ます。

7.4 ログレベル

DEBUG	デバッグ
INFO	情報
WARN	警告
ERROR	エラー

7.5 C / C++

slog/SequenceLog.h をインクルードし、__SLOG__マクロを定義してシーケンスログを有効化する必要があります。

7.5.1 getSequenceLogFileName()

シグネチャ	const char* getSequenceLogFileName()		
	型	引数名	説明
引数	なし		
戻り値	const char*		シーケンスログファイル名

説明

ユーザー定義関数です。C/C++では main()より前に関数を呼べるので、その場合でもログ出力ができるようにこの関数を定義する必要があります。そして最初のログ出力が行われるまでに setSequenceLogFileName()を呼び出していなければ、Sequence Log Library からこの関数が呼び出されます。

例：

```
const char* getSequenceLogFileName()
{
    static const char* fileName = "sampleA.log";
    return fileName;
}

class Sample
{
public:
    Sample() {SLOG(“Sample”, “Sample”, slog::ROOT);}
}

static Sample sample;  // (*1)

int main()
{
    setSequenceLogFileName(“sampleB.log”);
    SLOG(“sample.cpp”, “main”, slog::ROOT);
    return 0;
}
```

この例では main()より前に Sample のコンストラクタが呼ばれ、そこでログ出力をしているので sampleA.log が作成されます。(*1)をコメントアウトした場合は sampleB.log が作成されます。

7.5.2 setSequenceLogFileName(const char*)

シグネチャ	void setSequenceLogFileName()		
	型	引数名	説明
引数	const char*	name	シーケンスログファイル名
戻り値	なし		
説明			
7.1 参照。 C/C++では getSequenceLogFileName()を定義しなければならないので、この関数を使用する機会はないでしょう。			

7.5.3 setRootFlag(int)

シグネチャ	static void SetRootFlag()		
	型	引数名	説明
引数	int32_t	outputFlag	出力フラグ
戻り値	なし		
説明			
ROOT の既定値を設定します。シーケンスログサービスでの設定より優先されます。			

7.6 C++

7.6.1 SLOG(const char*, const char*, SequenceLogOutputFlag)

マクロ名	SLOG		
	型	引数名	説明
引数	const char*	className	クラス名
	const char*	funcName	メソッド名
	SequenceLogOutputFlag	outputFlag	出力フラグ

説明

各メソッドの先頭にこのマクロを記述することで、メソッド呼び出しのログを出力します。また、メソッドから戻る時にはリターンログを出力します。

className

クラス名。最大 255 バイト。

funcName

メソッド名。最大 255 バイト。

outputFlag

出力フラグ（3.1 参照）。デフォルトは KEEP。

7.6.2 SLOG(uint32_t, const char*, SequenceLogOutputFlag)

マクロ名	SLOG		
	型	引数名	説明
引数	uint32_t	classID	クラス ID
	const char*	funcName	メソッド名
	SequenceLogOutputFlag	outputFlag	出力フラグ
説明			
7.6.1 参照。			

7.6.3 SLOG(uint32_t, uint32_t, SequenceLogOutputFlag)

マクロ名	SLOG		
	型	引数名	説明
引数	uint32_t	classID	クラス ID
	uint32_t	funcID	メソッド ID
	SequenceLogOutputFlag	outputFlag	出力フラグ
説明			
7.6.1 参照。			

7.6.4 SMSG(SequenceLogLevel, const char*, ...)

マクロ名	SMSG		
	型	引数名	説明
引数	SequenceLogLevel	level	ログレベル
	const char*	format	フォーマット
	…	...	
説明			
メッセージを出力します。 format 書式フォーマット。printf と同様です。最大 255 バイト。			

7.6.5 SMSG(SequenceLogLevel, uint32_t)

マクロ名	SMSG		
	型	引数名	説明
引数	SequenceLogLevel	level	ログレベル
	uint32_t	messageID	メッセージ ID
説明			
メッセージを出力します。			

7.7 C

7.7.1 SLOG_STEPIN(const char*, const char*, int32_t)

マクロ名	SLOG_STEPIN		
	型	引数名	説明
引数	const char*	className	クラス名
	const char*	funcName	メソッド名
	int32_t	outputFlag	出力フラグ

説明

各メソッドの先頭にこのマクロを記述することで、メソッド呼び出しのログを出力します。

className

クラス名。最大 255 バイト。

funcName

メソッド名。最大 255 バイト。

outputFlag

出力フラグ（3.1 参照）。デフォルトはありません。

7.7.2 SLOG_STEPIN2(uint32_t, const char*, int32_t)

マクロ名	SLOG_STEPIN2		
	型	引数名	説明
引数	uint32_t	classID	クラス ID
	const char*	funcName	メソッド名
	int32_t	outputFlag	出力フラグ
説明			
7.7.1 参照。			

7.7.3 SLOG_STEPIN3(uint32_t, uint32_t, int32_t)

マクロ名	SLOG_STEPIN3		
	型	引数名	説明
引数	uint32_t	classID	クラス ID
	uint32_t	funcID	メソッド ID
	int32_t	outputFlag	出力フラグ
説明			
7.7.1 参照。			

7.7.4 SLOG_STEPOUT()

マクロ名	SLOG_STEPOUT		
	型	引数名	説明
引数	なし		
説明			
リターンログを出力します。			

7.7.5 SMSGC(SequenceLogLevel, const char*, ...)

マクロ名	SMSGC		
	型	引数名	説明
引数	SequenceLogLevel	level	ログレベル
	const char*	format	フォーマット
	…	...	
説明			
メッセージを出力します。 format 書式フォーマット。printf と同様です。最大 255 バイト。			

7.7.6 SMSGC2(SequenceLogLevel, uint32_t)

マクロ名	SMSGC2		
	型	引数名	説明
引数	SequenceLogLevel	level	ログレベル
	uint32_t	messageID	メッセージ ID
説明			
メッセージを出力します。			

7.8 C#

7.8.1 SetFileName(String)

シグネチャ	static void setFileName()		
	型	引数名	説明
引数	String	aName	シーケンスログファイル名
戻り値	なし		
説明			
7.1 参照。			

7.8.2 SetRootFlag(int)

シグネチャ	static void SetRootFlag()		
	型	引数名	説明
引数	int	outputFlag	出力フラグ
戻り値	なし		
説明			
ROOT の既定値を設定します。シーケンスログサービスでの設定より優先されます。			

7.8.3 StepIn(String, String, int)

シグネチャ	static long StepIn()		
	型	引数名	説明
引数	String	className	クラス名
	String	funcName	メソッド名
	int	outputFlag	出力フラグ
戻り値	long		シーケンスログ ID
説明			
7.7.1 参照。 戻り値のシーケンスログ ID は StepOut()、D()等のメソッドで使用します。			

7.8.4 StepIn(String, String)

シグネチャ	static long StepIn()		
	型	引数名	説明
引数	String	className	クラス名
	String	funcName	メソッド名
戻り値	long		シーケンスログ ID
説明			
7.8.3 参照。 StepIn(String, String, Log.KEEP)と同等です。			

7.8.5 StepIn(int, String, int)

シグネチャ	static long StepIn()		
	型	引数名	説明
引数	int	classID	クラス ID
	String	funcName	メソッド名
	int	outputFlag	出力フラグ
戻り値	long		シーケンスログ ID
説明			
7.8.3 参照。			

7.8.6 StepIn(int, String)

シグネチャ	static long StepIn()		
	型	引数名	説明
引数	int	classID	クラス ID
	String	funcName	メソッド名
戻り値	long		シーケンスログ ID
説明			
7.8.3 参照。 StepIn(int, String, Log.KEEP)と同等です。			

7.8.7 StepIn(int, int, int)

シグネチャ	static long StepIn()		
	型	引数名	説明
引数	int	classID	クラス ID
	int	funcID	メソッド ID
	int	outputFlag	出力フラグ
戻り値	long		シーケンスログ ID
説明			
7.8.3 参照。			

7.8.8 StepIn(int, int)

シグネチャ	static long StepIn()		
	型	引数名	説明
引数	int	classID	クラス ID
	int	funcID	メソッド ID
戻り値	long		シーケンスログ ID
説明			
7.8.3 参照。 StepIn(int, int, Log.KEEP)と同等です。			

7.8.9 StepOut(long)

シグネチャ	static void StepOut()		
	型	引数名	説明
引数	long	slog	シーケンスログ ID
戻り値	なし		
説明			
リターンログを出力します。			

7.8.10 D(long, String)

シグネチャ	static void D()		
	型	引数名	説明
引数	long	slog	シーケンスログ ID
	String	message	メッセージ
戻り値	なし		
説明			
メッセージを出力します。 I()、W()、E()も同様です。V()は機能的にはD()と全く同じです。			

7.8.11 D(long, int)

シグネチャ	static void D()		
	型	引数名	説明
引数	long	slog	シーケンスログ ID
	int	messageID	メッセージ ID
戻り値	なし		
説明			
メッセージを出力します。			

7.9 Java

パッケージ名　　net.log_tools.slog

クラス名　　Log

7.9.1　　setFileName(String)

シグネチャ	public static void setFileName()		
	型	引数名	説明
引数	String	name	シーケンスログファイル名
戻り値	なし		
説明			
7.1 参照。			

7.9.2　　setRootFlag(int)

シグネチャ	public static void setRootFlag()		
	型	引数名	説明
引数	int	outputFlag	出力フラグ
戻り値	なし		
説明			
ROOT の既定値を設定します。シーケンスログサービスでの設定より優先されます。			

7.9.3 stepIn(String, String, int)

シグネチャ	public static long stepIn()		
	型	引数名	説明
引数	String	className	クラス名
	String	funcName	メソッド名
	int	outputFlag	出力フラグ
戻り値	long		シーケンスログ ID
説明			
7.7.1 参照。 戻り値のシーケンスログ ID は stepOut()、d()等のメソッドで使用します。			

7.9.4 stepIn(String, String)

シグネチャ	public static long stepIn()		
	型	引数名	説明
引数	String	className	クラス名
	String	funcName	メソッド名
戻り値	long		シーケンスログ ID
説明			
7.9.3 参照。 stepIn(String, String, Log.KEEP)と同等です。			

7.9.5 stepIn(int, String, int)

シグネチャ	public static long stepIn()		
	型	引数名	説明
引数	int	classID	クラス ID
	String	funcName	メソッド名
	int	outputFlag	出力フラグ
戻り値	long		シーケンスログ ID
説明			
7.9.3 参照。			

7.9.6 stepIn(int, String)

シグネチャ	public static long stepIn()		
	型	引数名	説明
引数	int	classID	クラス ID
	String	funcName	メソッド名
戻り値	long		シーケンスログ ID
説明			
7.9.3 参照。 stepIn(int, String, Log.KEEP)と同等です。			

7.9.7 stepIn(int, int, int)

シグネチャ	public static long stepIn()		
	型	引数名	説明
引数	int	classID	クラス ID
	int	funcID	メソッド ID
	int	outputFlag	出力フラグ
戻り値	long		シーケンスログ ID
説明			
7.9.3 参照。			

7.9.8 stepIn(int, int)

シグネチャ	public static long stepIn()		
	型	引数名	説明
引数	int	classID	クラス ID
	int	funcID	メソッド ID
戻り値	long		シーケンスログ ID
説明			
7.9.3 参照。 stepIn(int, int, Log.KEEP)と同等です。			

7.9.9 stepOut(long)

シグネチャ	public static void stepOut()		
	型	引数名	説明
引数	long	slog	シーケンスログ ID
戻り値	なし		
説明			
リターンログを出力します。			

7.9.10 d(long, String)

シグネチャ	public static void d()		
	型	引数名	説明
引数	long	slog	シーケンスログ ID
	String	message	メッセージ
戻り値	なし		
説明			
メッセージを出力します。 i()、w()、e()も同様です。v()は既存のコードを修正せずに済ませるためのもので、機能的にはd()と全く同じです。			

7.9.11 d(long, int)

シグネチャ	public static void d()		
	型	引数名	説明
引数	long	slog	シーケンスログ ID
	int	messageID	メッセージ ID
戻り値	なし		
説明			
メッセージを出力します。			

7.10 PHP

7.10.1 slogSetFileName(string)

シグネチャ	void slogSetFileName()		
	型	引数名	説明
引数	string	name	シーケンスログファイル名
戻り値	なし		
説明			
7.1 参照。			

7.10.2 slogSetRootFlag(int)

シグネチャ	void slogSetRootFlag()		
	型	引数名	説明
引数	int	outputFlag	出力フラグ
戻り値	なし		
説明			
ROOT の既定値を設定します。シーケンスログサービスでの設定より優先されます。			

7.10.3 slogStepIn(string, string, int)

シグネチャ	int slogStepIn()		
	型	引数名	説明
引数	string	className	クラス名
	string	funcName	メソッド名
	int	outputFlag	出力フラグ
戻り値	int		シーケンスログ ID
説明			
7.6.1 参照。 戻り値のシーケンスログ ID は slogStepOut()、slogD()等のメソッドで使用します。クラス ID やメソッド ID には対応していません。			

7.10.4 slogStepOut(int)

シグネチャ	void slogStepOut()		
	型	引数名	説明
引数	int	slog	シーケンスログ ID
戻り値	なし		
説明			
リターンログを出力します。			

7.10.5 slogD(int, string)

シグネチャ	void slogD()		
	型	引数名	説明
引数	int	slog	シーケンスログ ID
	string	message	メッセージ
戻り値	なし		
説明			
メッセージを出力します。 slogI()、slogW()、slogE()も同様です。メッセージ ID には対応していません。			

8 Sequence Log Service

シーケンスログサービスはログ出力を管理するソフトウェアです。サービスの開始／終了、シーケンスログプリントとの接続／切断、ログファイルの出力先、サイズやログファイル数の上限の設定を行います。

8.1.1 ログファイルサイズ

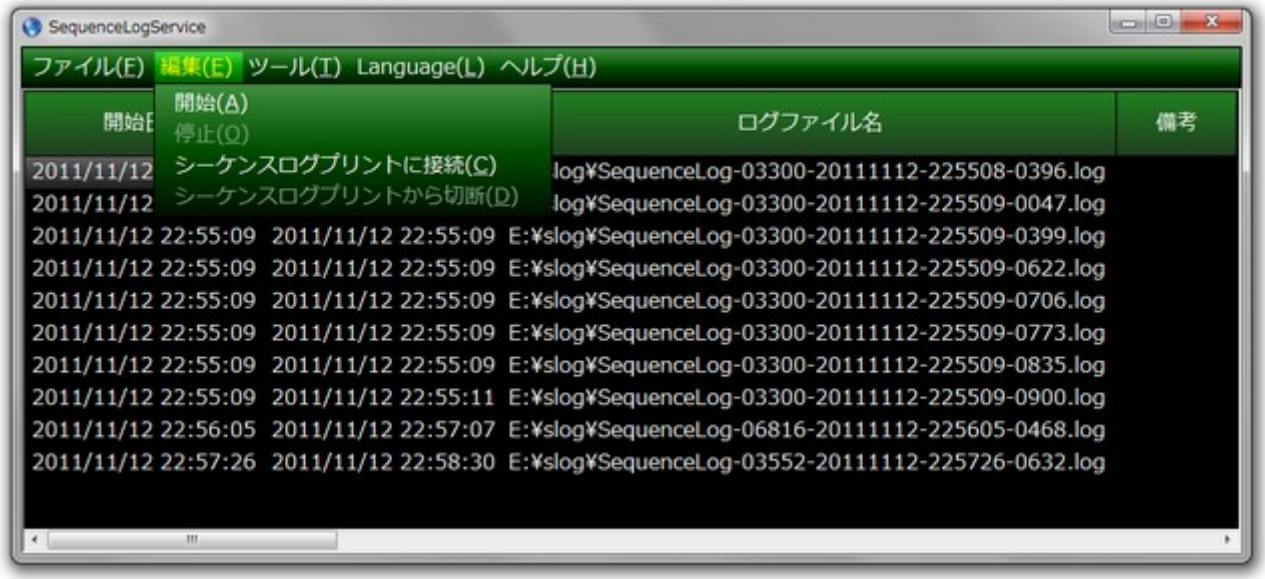
ログファイルサイズを 0 以外に設定した場合、その上限に達すると新たなログファイルを作成します。0 に設定した場合は上限無制限（デフォルト）となります。

8.1.2 ログファイル数

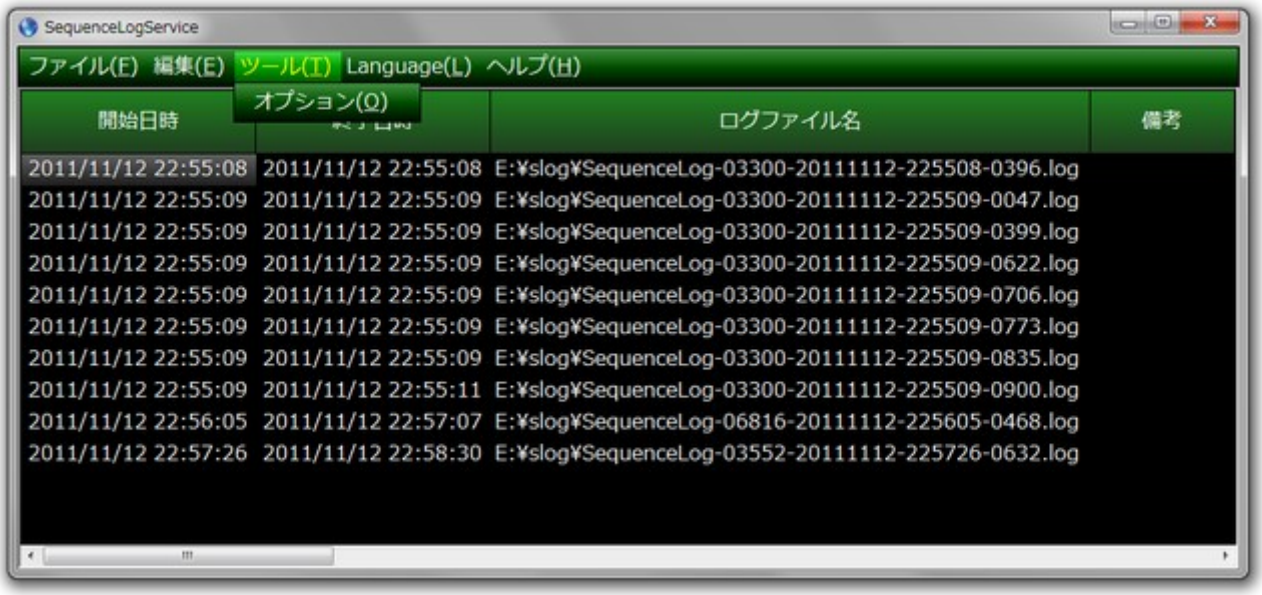
ログファイル数を 0 以外に設定した場合、その上限に達すると最も古いファイルから削除していきます。0 に設定した場合は上限無制限（デフォルト）となります。

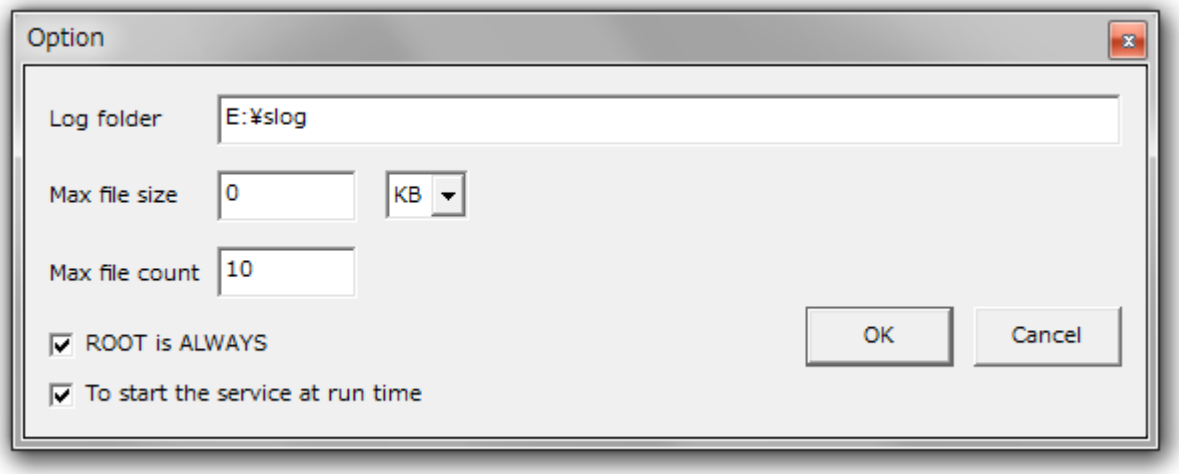
8.2 Windows

サービスの開始／停止、 シーケンスログプリントとの接続／切断は「編集」メニューで行います。



ログファイルの各種設定は「ツール」－「オプション」により行います。





8.3 Linux

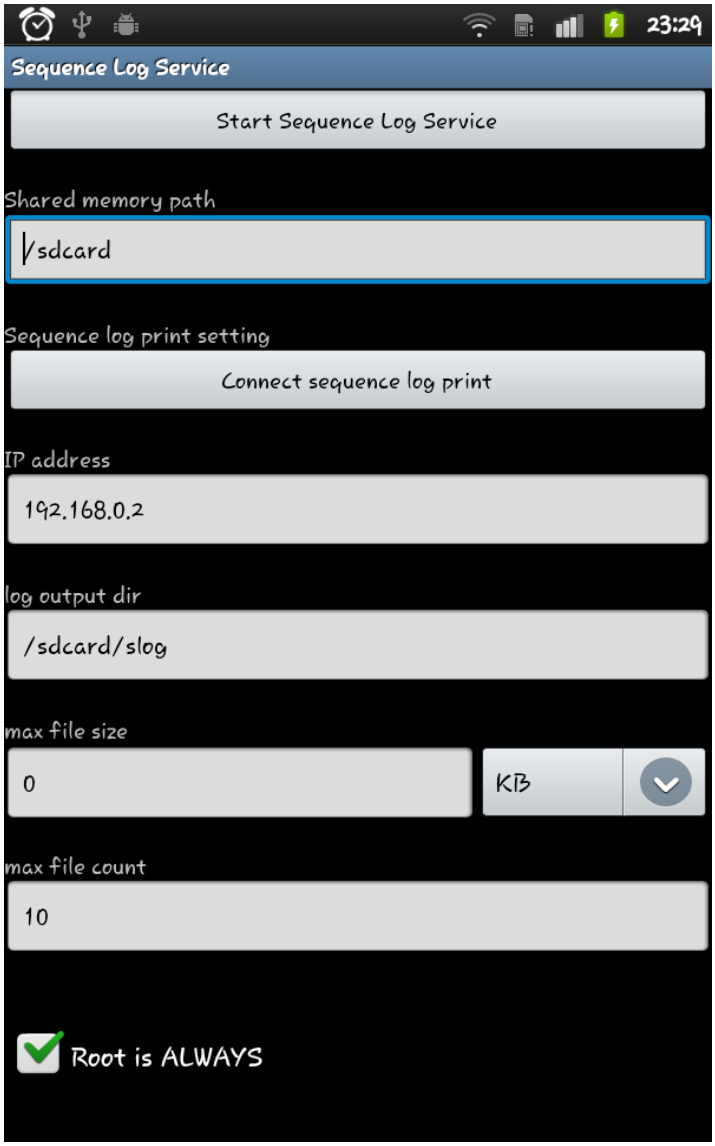
設定ファイルに設定を記述します。slogsvc はデフォルトでは/etc/slog.conf を、”slogsvc -f 設定ファイル名”で任意の設定ファイルを読み込みます。

8.3.1 slog.conf 設定例

```
# 共有メモリ用ディレクトリ
SHARED_MEMORY_DIR /tmp

# シーケンスログプリント IP
LOG_PRINT_IP      127.0.0.1

# シーケンスログ出力ディレクトリ
LOG_OUTPUT_DIR    /var/log/slog

# 最大ファイルサイズ
MAX_FILE_SIZE     0 MB

# 最大ファイル数
MAX_FILE_COUNT    10

# ROOT を ALWAYS とするかどうか
ROOT_ALWAYS       true
```

8.4 Android

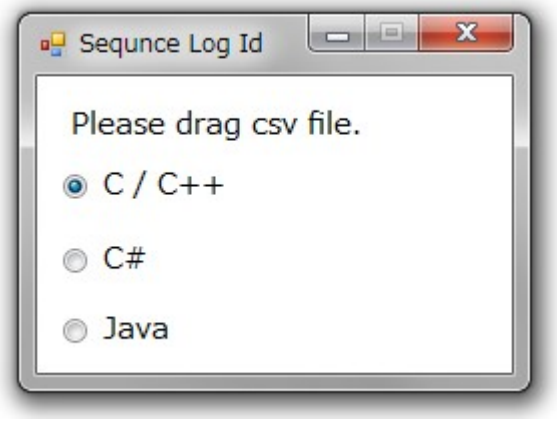
Android 用の Sequence Log Service 設定画面です。



9 Sequence Log Id

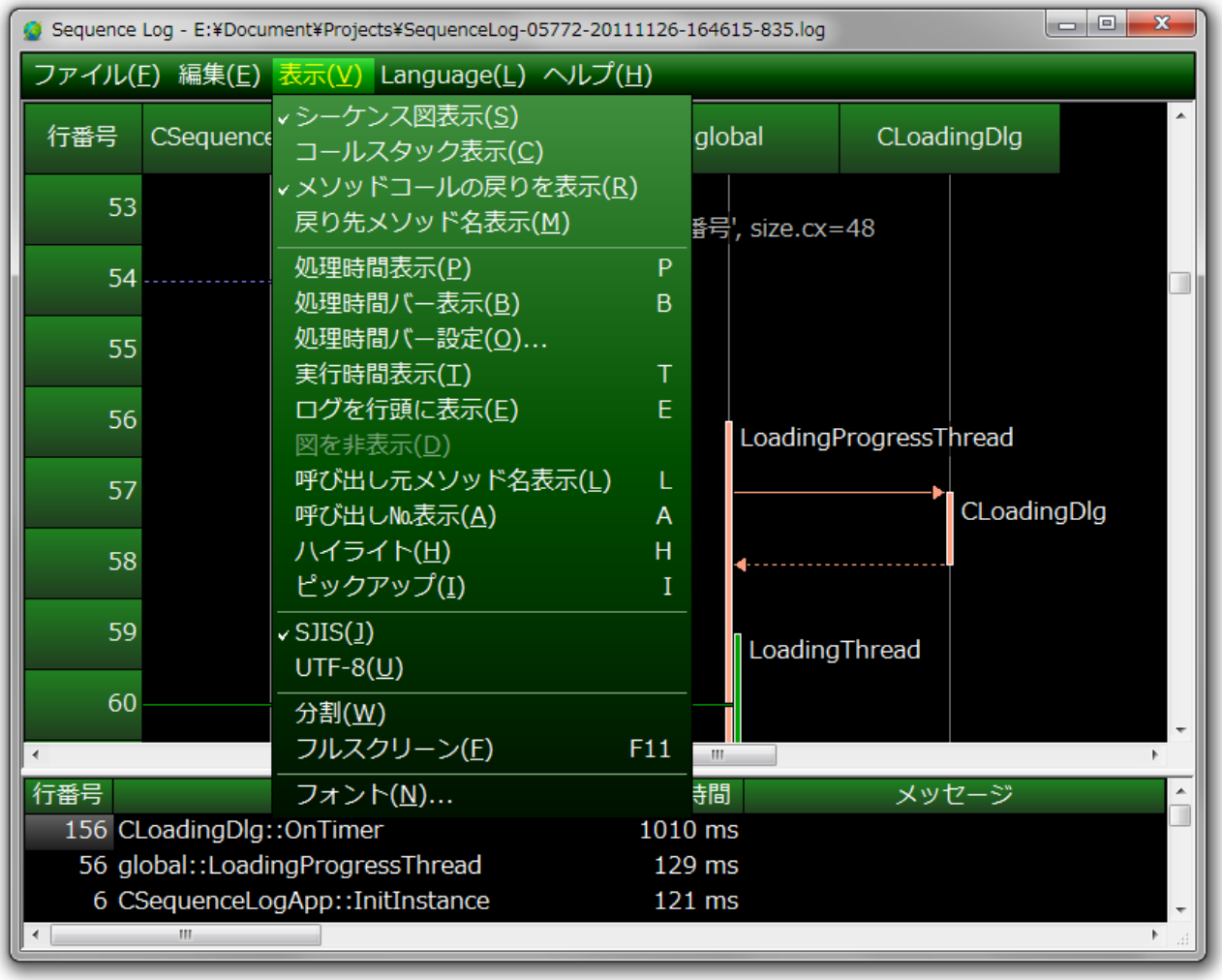
csv ファイルから、SequenceLog.exe で使用する ID 定義ファイルと各開発言語用の ID 定義ファイルを出力します。

csv ファイルを”ID,クラス名（またはメソッド名、メッセージ）”の形式で作成し、SequenceLogId.exe にドラッグして下さい。

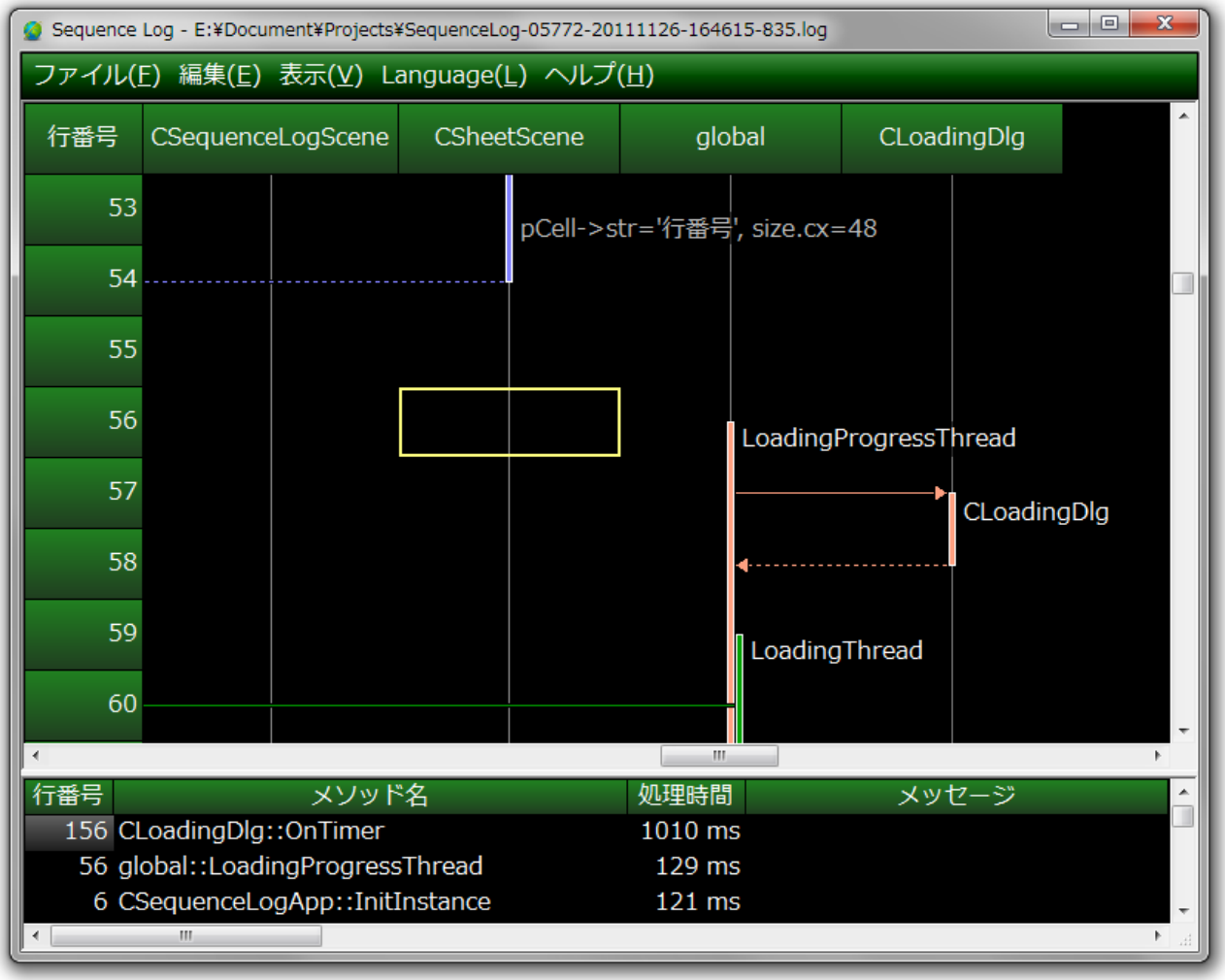


10 Sequence Log

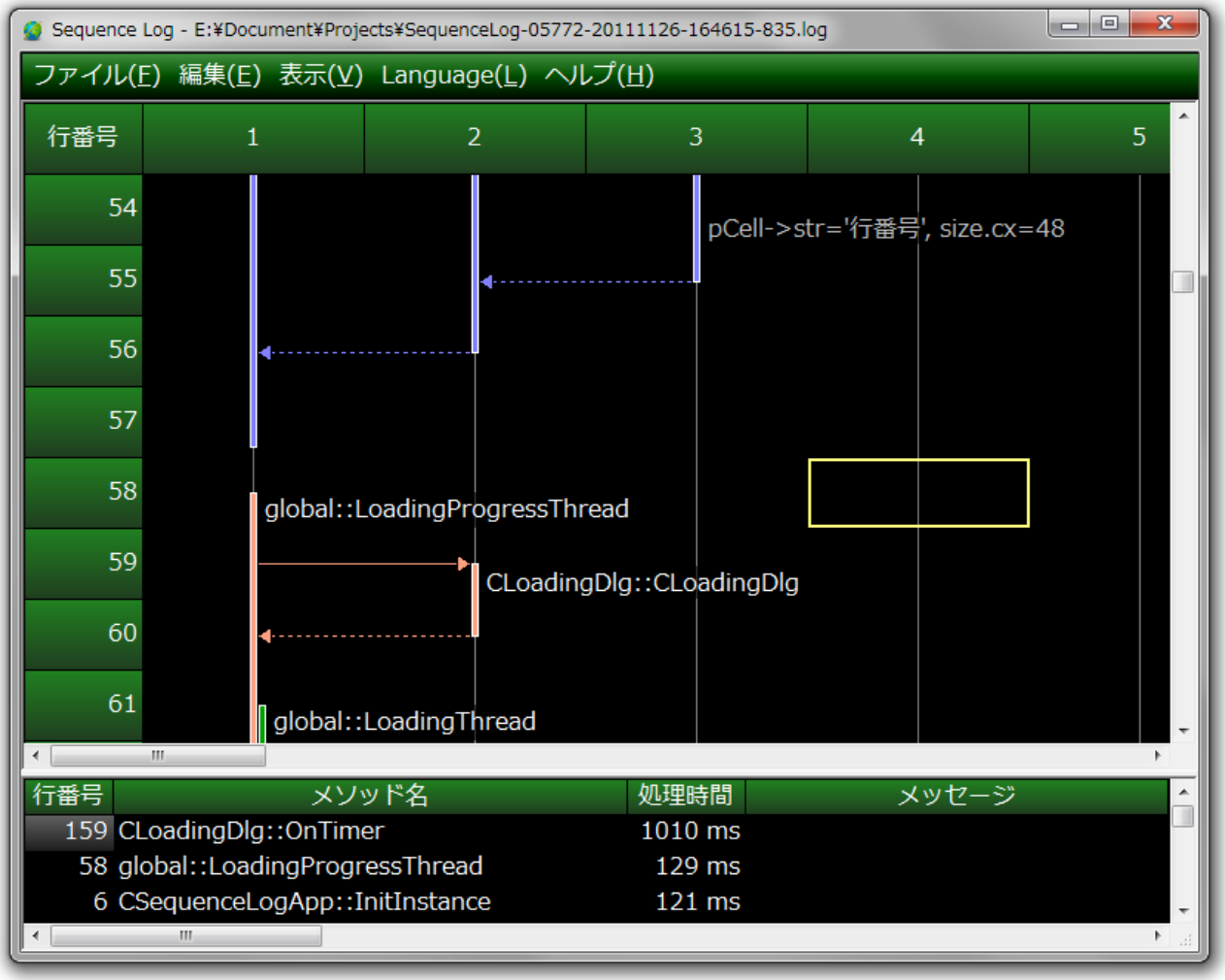
シーケンスログファイルを「ファイル」ー「開く」、またはドラッグで開きます。ID 定義ファイルを使用する場合は同時に指定して下さい。一度読み込んだ ID 定義は新たに ID 定義ファイルが指定されるまで有効です。シーケンスログファイルを複数指定するとマージして表示します（ライセンスがある場合のみ）。



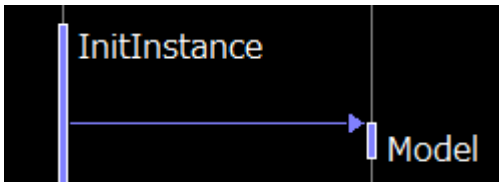
10.1 シーケンス図表示

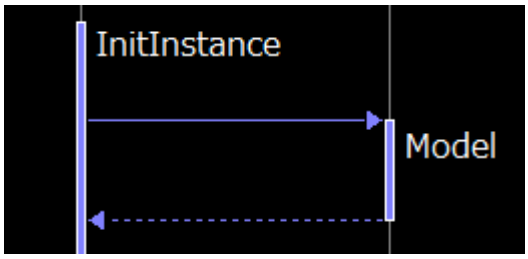


10.2 コールスタック表示

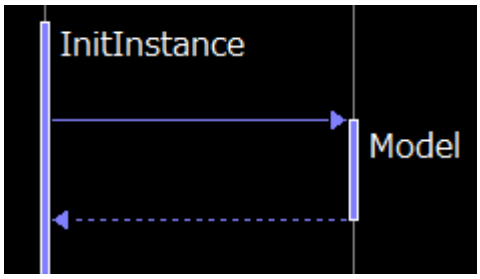


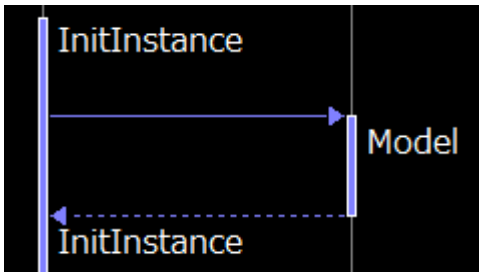
10.3　メソッドコールの戻りを表示



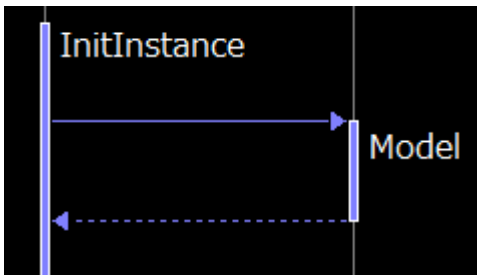


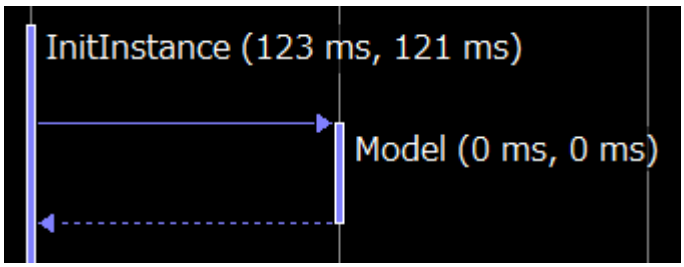
10.4　戻り先メソッド名表示





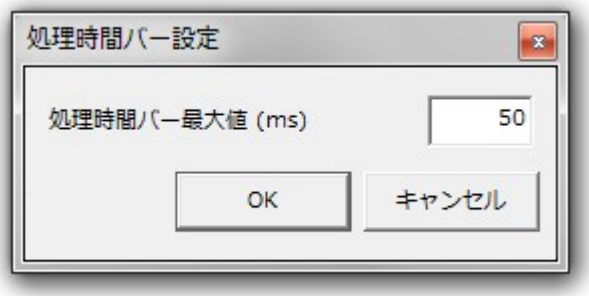
10.5　処理時間表示



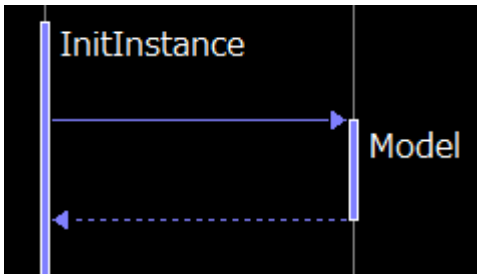


10.6　処理時間バー表示

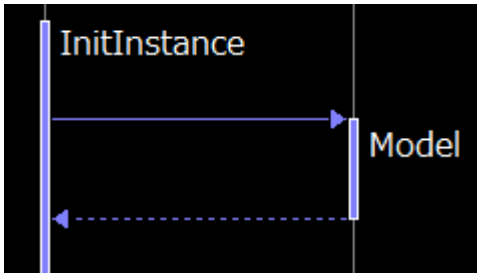
10.7 処理時間バー設定

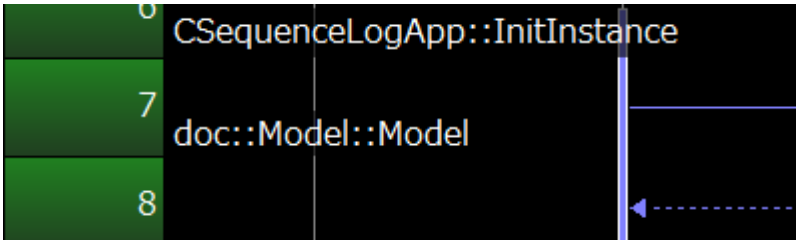


10.8 実行時間表示

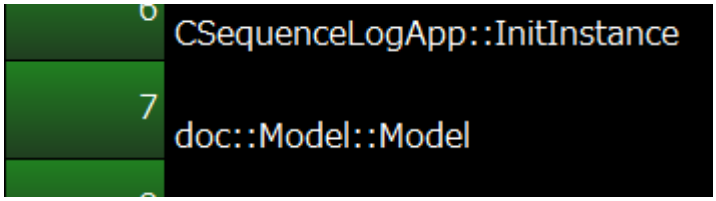


10.9 ログを行頭に表示

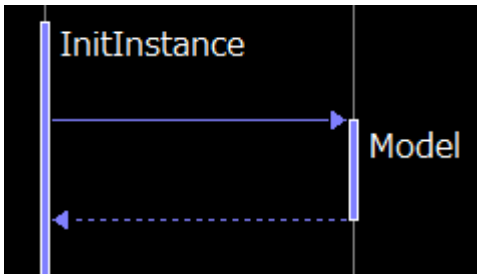


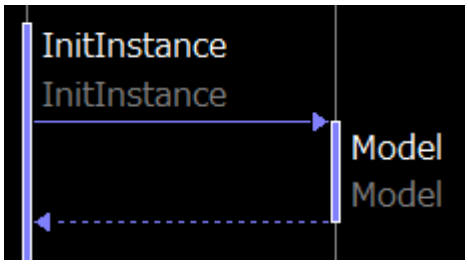


10.10 図を非表示

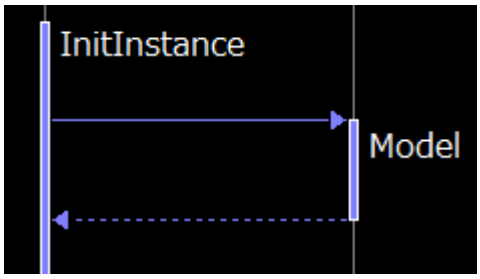


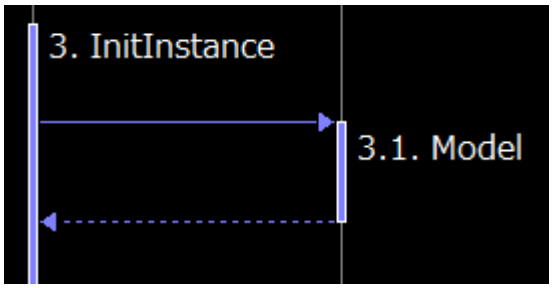
10.11　呼び出し元メソッド名表示



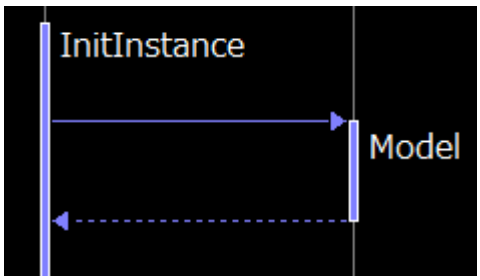


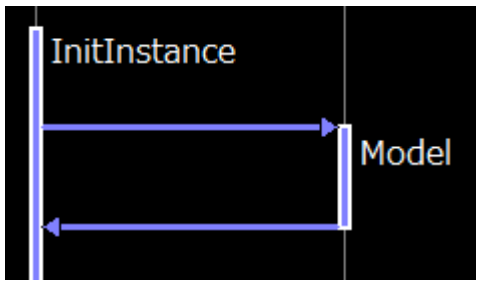
10.12　呼び出しNo.表示



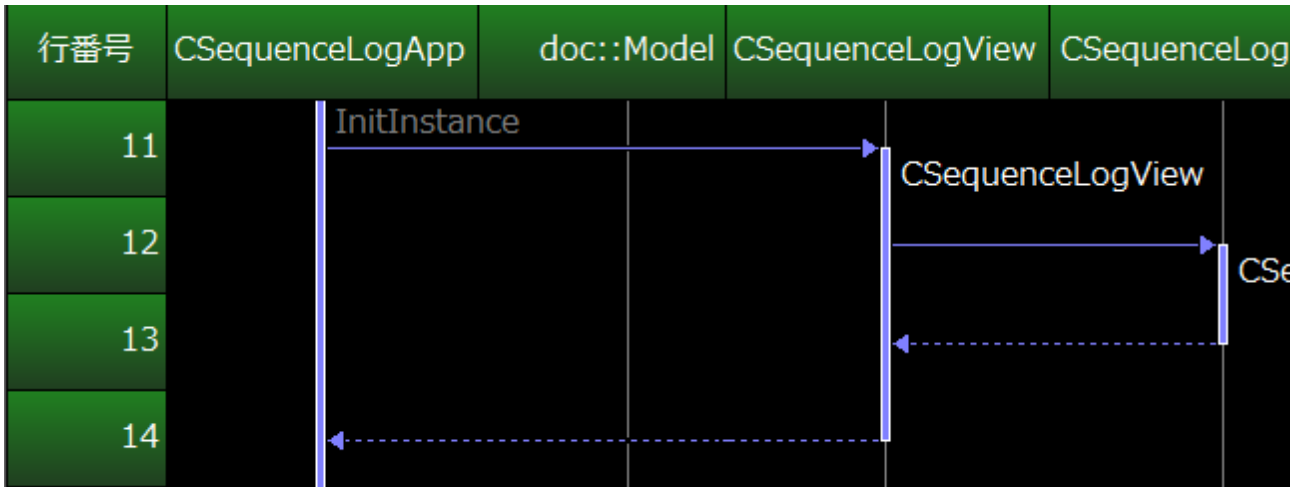


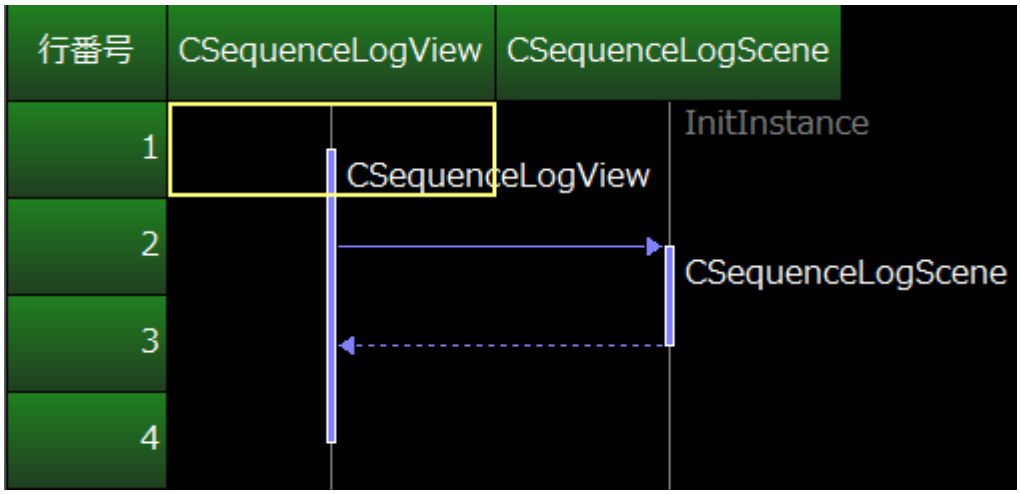
10.13　ハイライト





10.14 ピックアップ


行番号
CSequenceLogApp
doc::Model
CSequenceLogView
CSequenceLog
11
12
13
14
InitInstance
CSequenceLogView
CSe


行番号
CSequenceLogView
CSequenceLogScene
1
2
3
4
InitInstance
CSequenceLogView
CSequenceLogScene