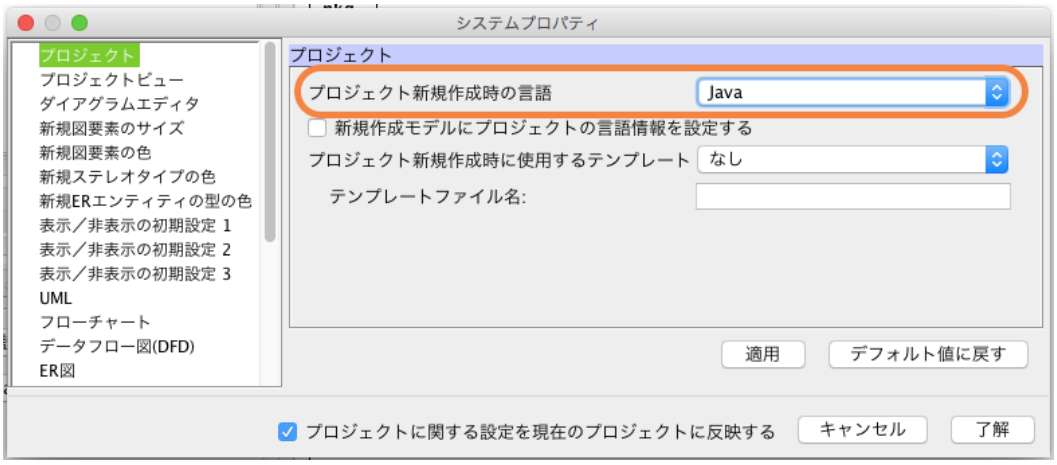
言語サポート機能を使ってみよう

astah*では、基本的によく使用される基本的な言語Java、C++、C#に対応しています。
これ以外の言語について、主に出力機能はプラグインで対応しています。プラグイン一覧もご参照ください。
プラグイン一覧：http://astah.change-vision.com/ja/feature/plugins-sample.html

JAVA

プロジェクトの言語設定は、プロパティビュー、またはシステムプロパティで設定します。
プロパティビュー：[プロジェクトの設定]タブ

システムプロパティ：[ツール] – [システムプロパティ] – [プロジェクト] – [プロジェクトの新規作成時の言語] – [Java]



JAVA 言語固有の設定をしてみよう

クラス、属性、操作の[言語]タブ(プロパティビュー)で、Java 言語固有の設定が可能です。

[クラス]

[属性]

[操作]

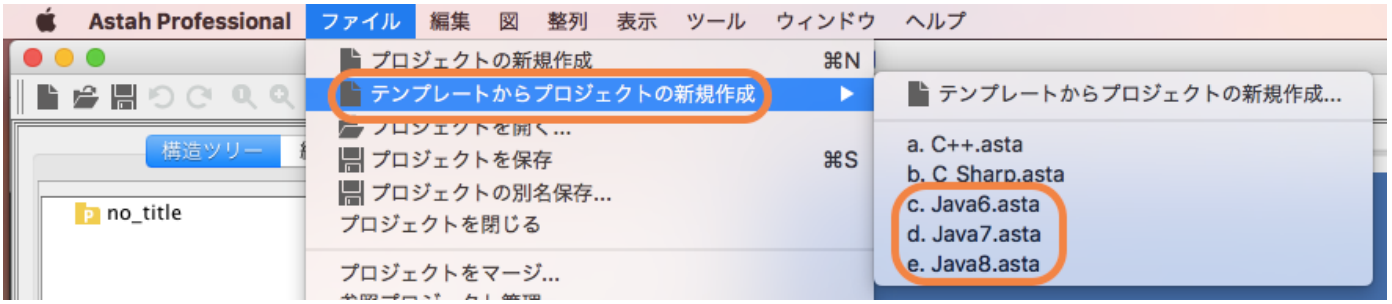
JAVA のデフォルトモデルを使ってみよう

astah* professional インストールフォルダ¥template¥project¥Java6.asta

astah* professional インストールフォルダ¥template¥project¥Java7.asta

astah* professional インストールフォルダ¥template¥project¥Java8.asta

デフォルトモデルは、[ファイル] – [テンプレートからプロジェクトの新規作成] – [Javax.asta]を選択して開けます。

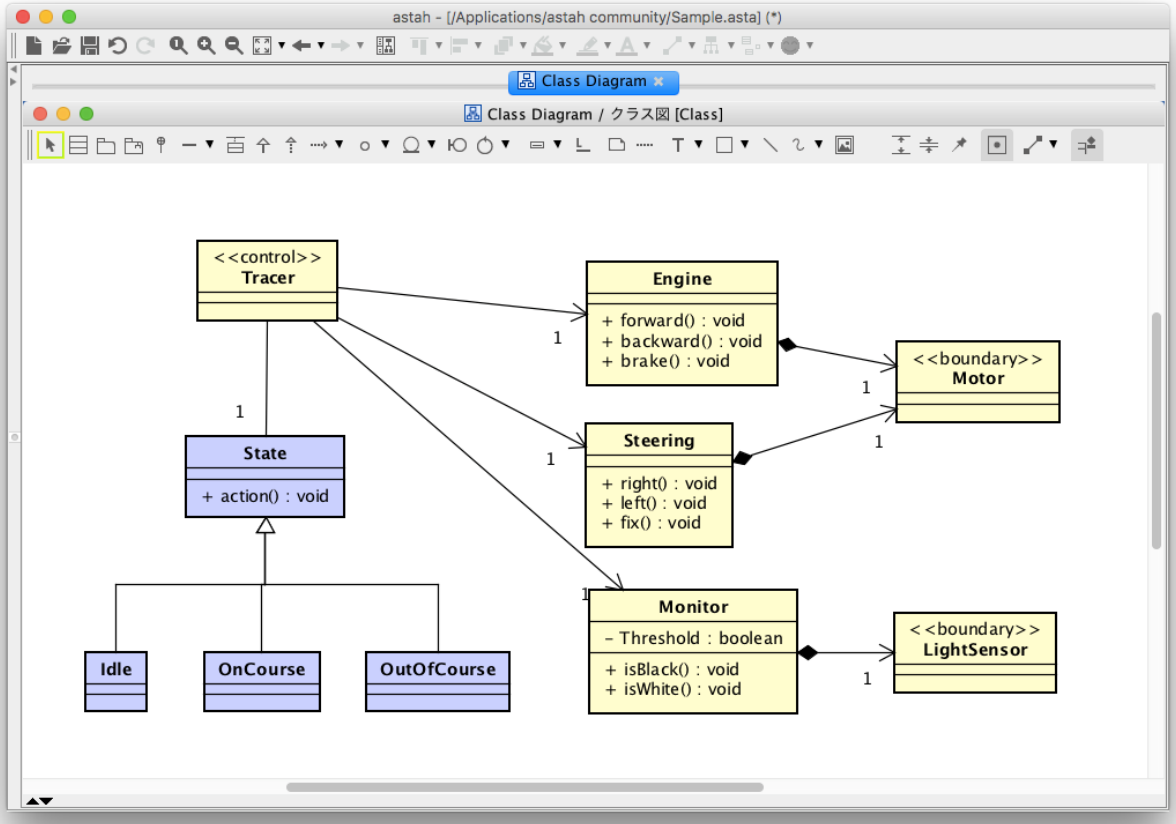


また、新規プロジェクト作成時に、該当のデフォルトモデルを開くよう指定することも可能です。
[ツール] – [システムプロパティ] – [プロジェクト]を開き、プロジェクト新規作成時に使用したいテンプレートの種類をドロップダウンリストから選び、その下にファイル名を入力します。

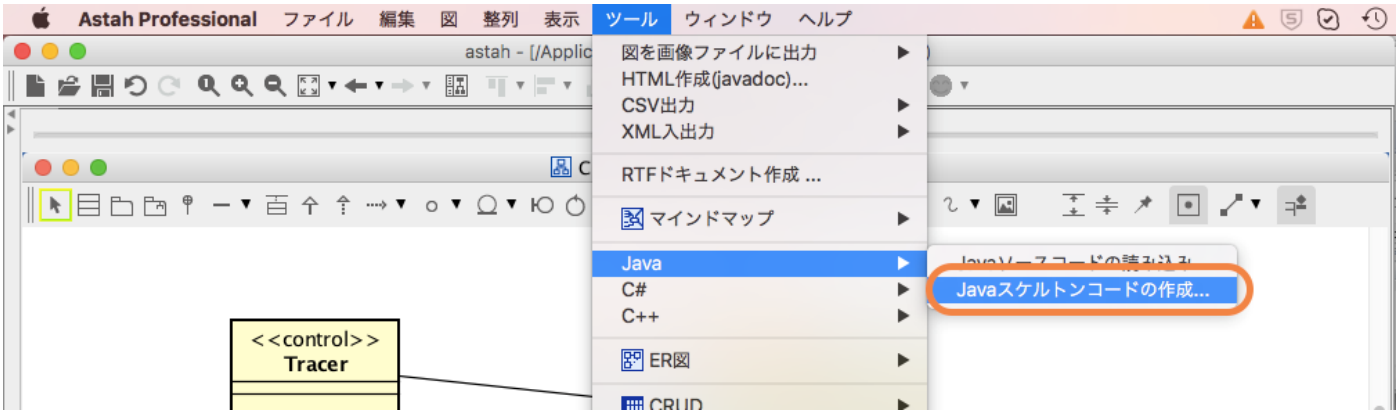
JAVA スケルトンコードを作成してみよう

作成したモデルから Java のスケルトンコードを出力できます。

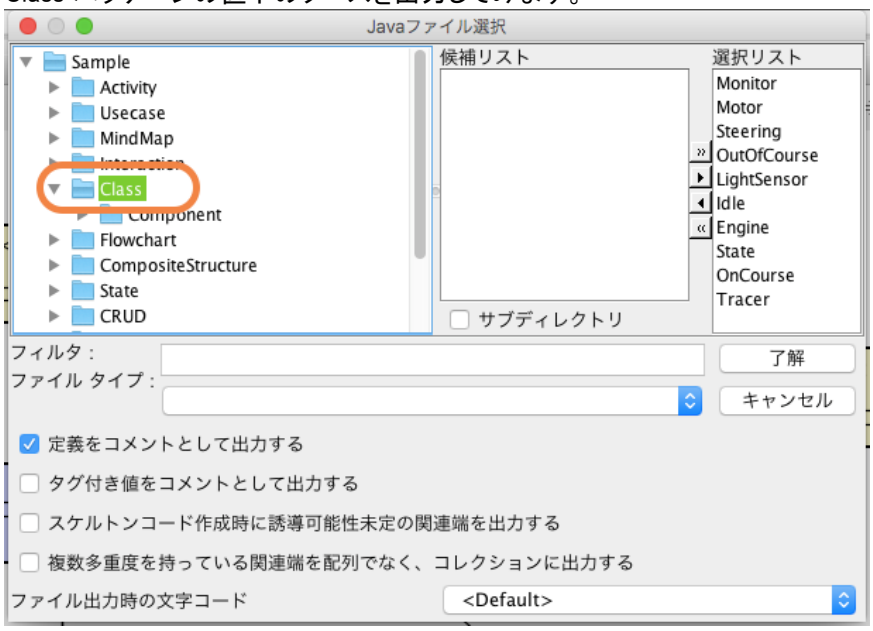
インストールフォルダ配下のSample.asta を開いて出力してみましょう。



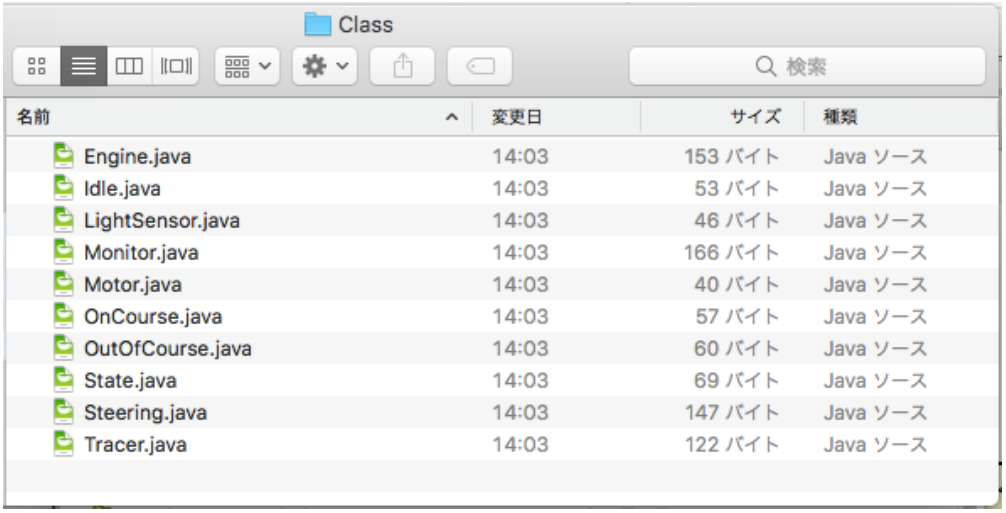
メインメニューの[ツール] – [Java] – [Java スケルトンコードの作成]を選択します。



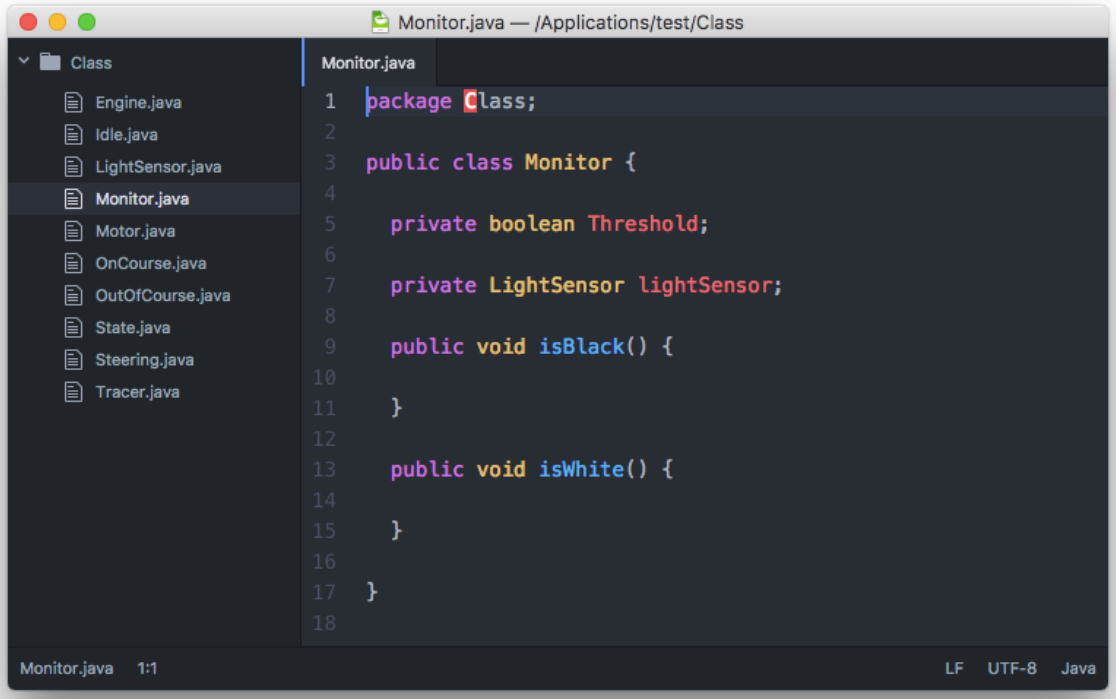
Class パッケージの直下のソースを出力してみます。



以下のように出力されましたね。



Monitor.java を開いてみます。



正しく出力されていることがわかりますね。

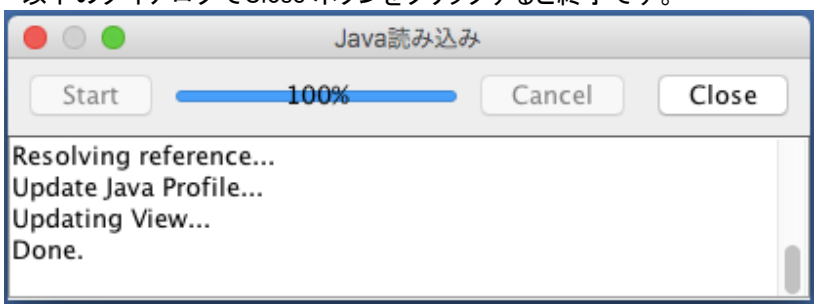
JAVA ソースコードの読み込みをしてみよう

今度は、プロジェクトを新規作成して、前章で出力したソースをリバースしてみます。

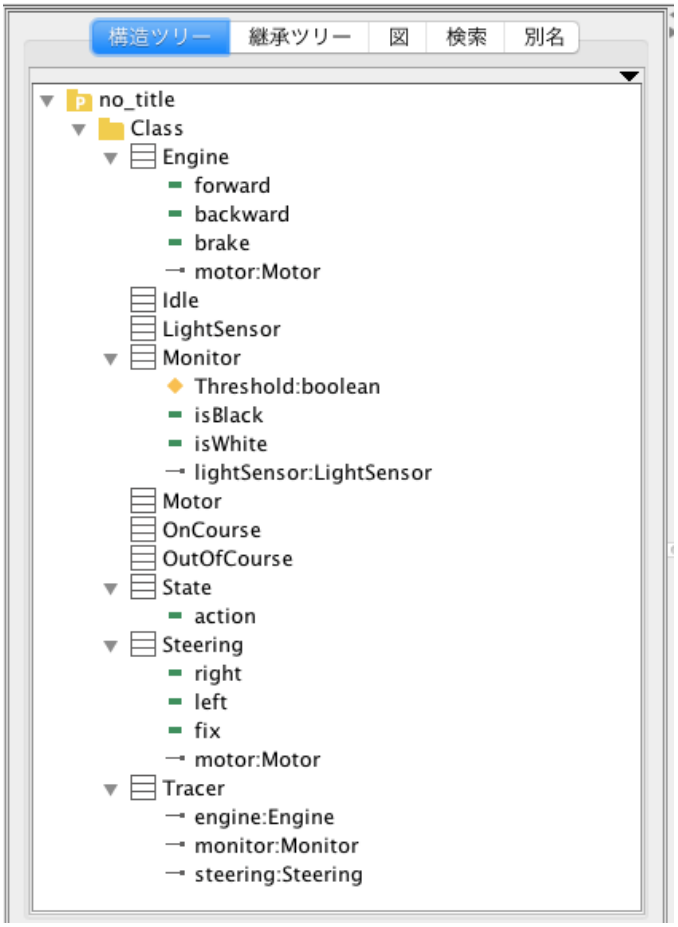
1. プロジェクトを 新規作成し、**[ツール] – [Java ソースコードの読み込み]**を選択します。
2. Java ファイルを選択するダイアログが開きます。先ほど出力した Java ファイルを選択します。
3. Java のソースコードより**パッケージ**や**クラス**のモデルを生成します。

   下のダイアログで、関連にしたい属性を指定することもできます

以下のダイアログでClose ボタンをクリックすると終了です。

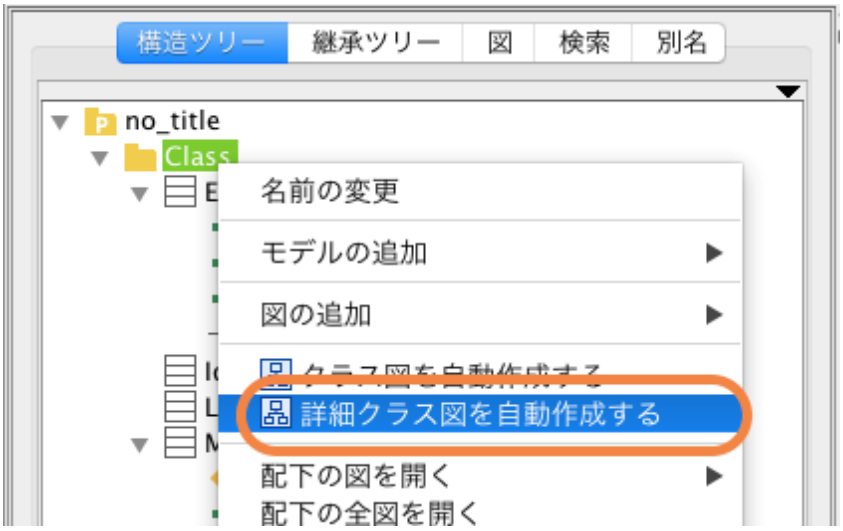


以下のようにリバースされました。

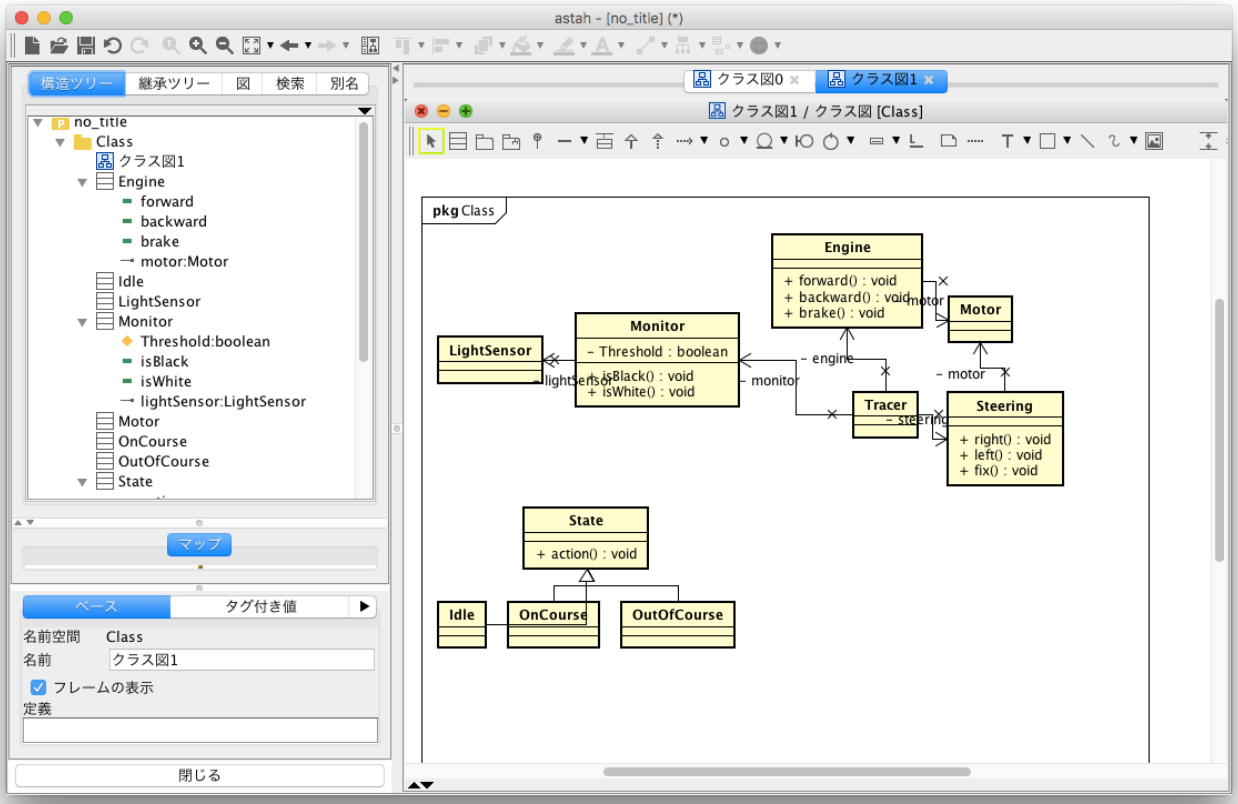


リバースしただけではクラス図は作成されません。読み込んだモデルからクラス図を自動作成するには、構造ツリーのプロジェクトのポップアップメニューより以下のメニューを選択します。

- **[クラス図を自動作成する]**
- **[詳細クラス図を自動作成する]**



以下のようにクラス図が自動生成されました。



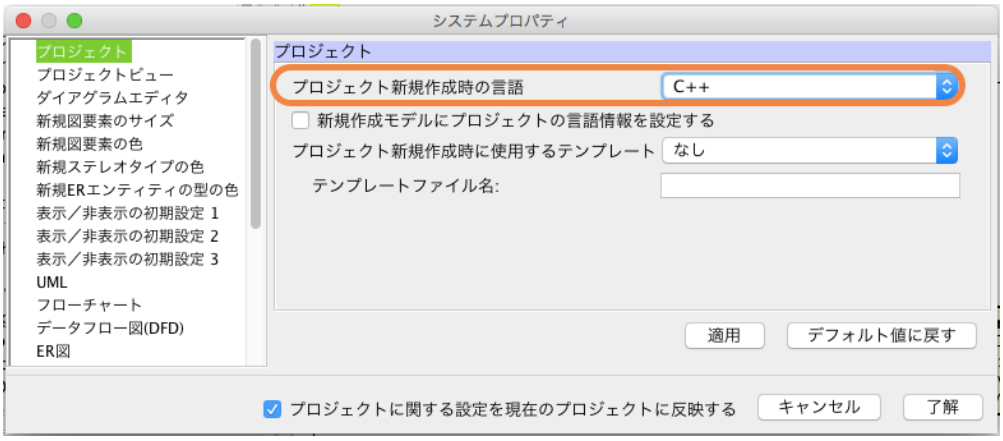
図の自動生成後は、[整列] – [全図要素の自動レイアウト]機能などを使って、図を整えてください。

C++

プロジェクトの言語設定は、プロパティビュー、または、システムプロパティで設定します。

プロパティビュー:

システムプロパティ:[ツール] – [システムプロパティ] – [プロジェクト] – [プロジェクトの新規作成時の言語] – [C++]

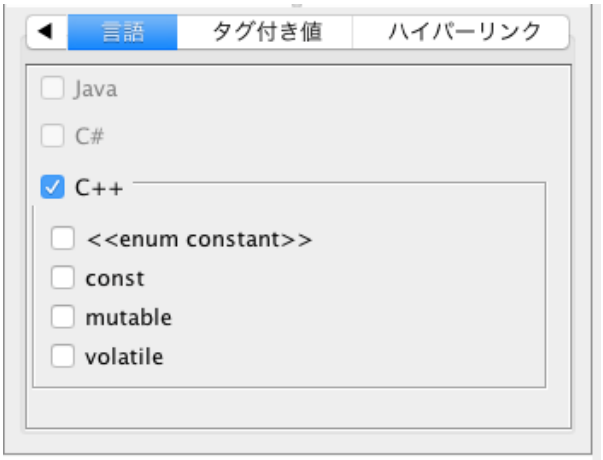


C++言語固有の設定をしよう

クラス、属性、操作の[言語]タブ（プロパティビュー）で、C++言語固有の設定が可能です。

[クラス]

[属性]

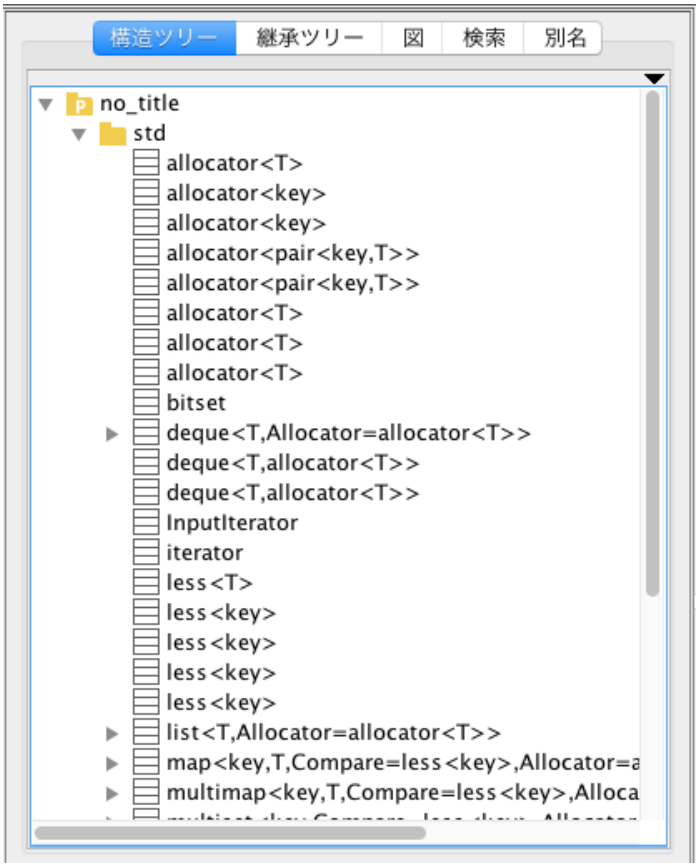


[操作]

C++用のデフォルトモデルを使ってみよう

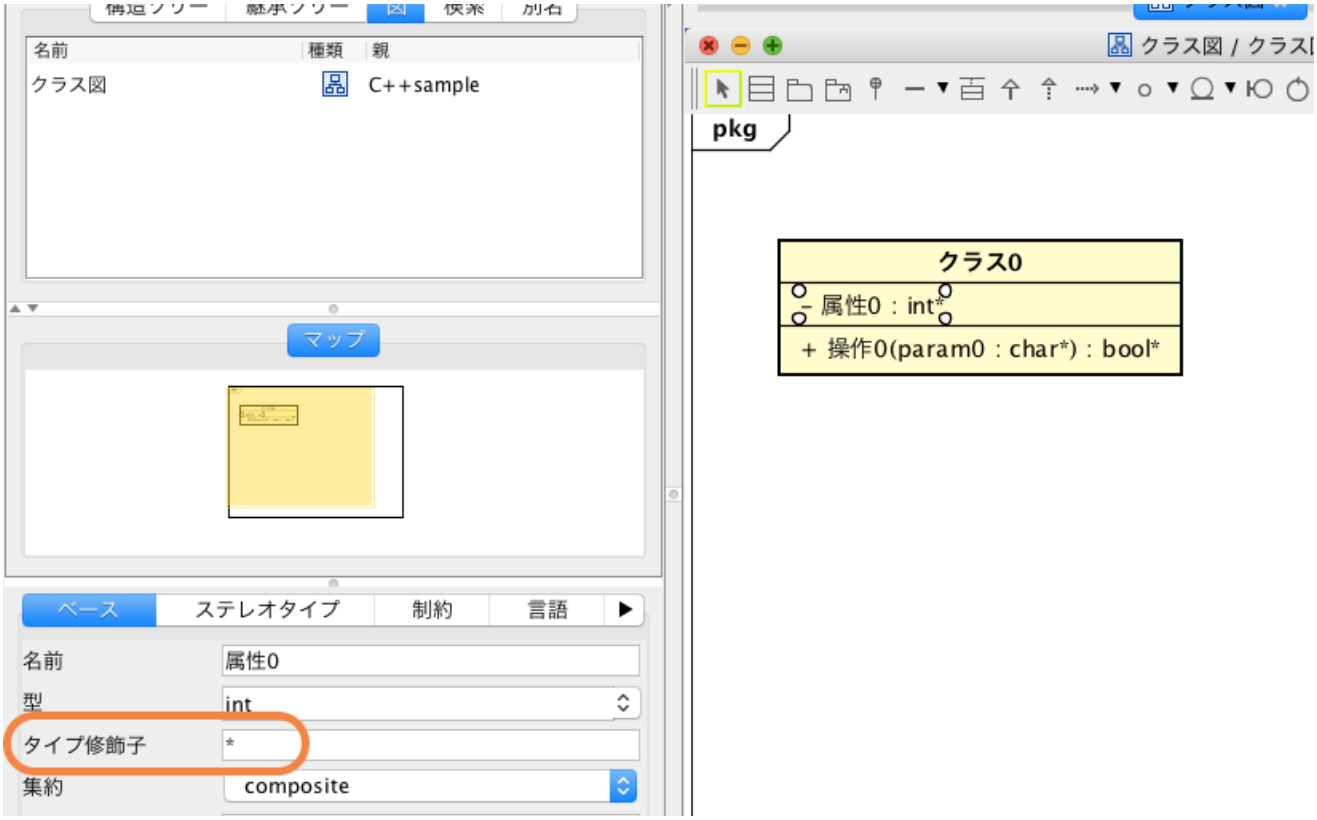
C++用のデフォルトモデルは、[ファイル] – [テンプレートからプロジェクトの新規作成] – [C++.asta]を選択して開けます。又、新規プロジェクト作成時に、常にこのモデルを使いたい場合は、[ツール] – [システムプロパティ] – [プロジェクト] を開き、下記のように設定します。

デフォルトモデルは、以下のモデルを含みます。



型修飾子（*、&等、ポインタ情報）を使ってみよう

プロパティビューより型修飾子を入力できます。型修飾子は、図上に表示され、HTML 出力やRTF 出力にも対応しています。



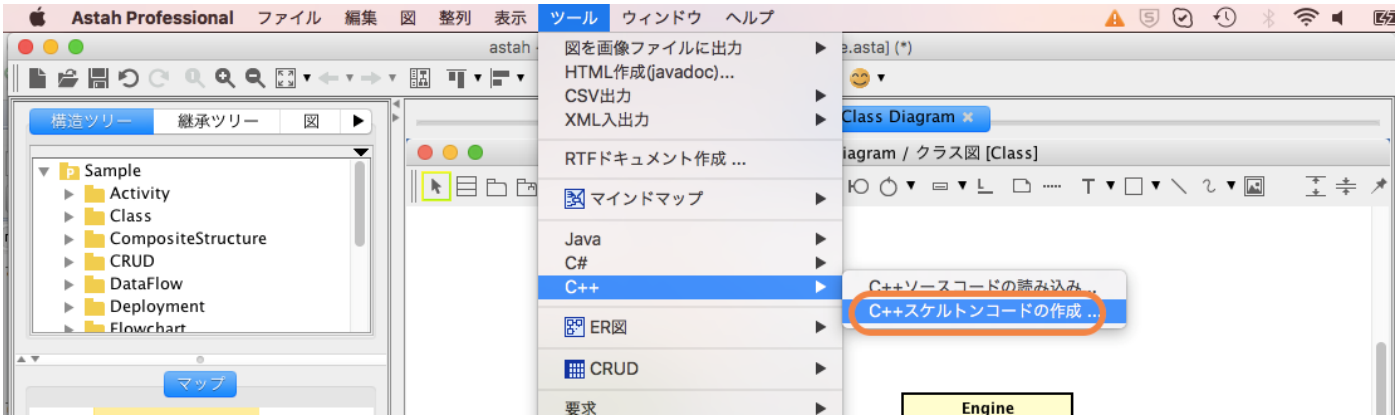
C++プリミティブ型の対応

各プリミティブ型に対応しています。

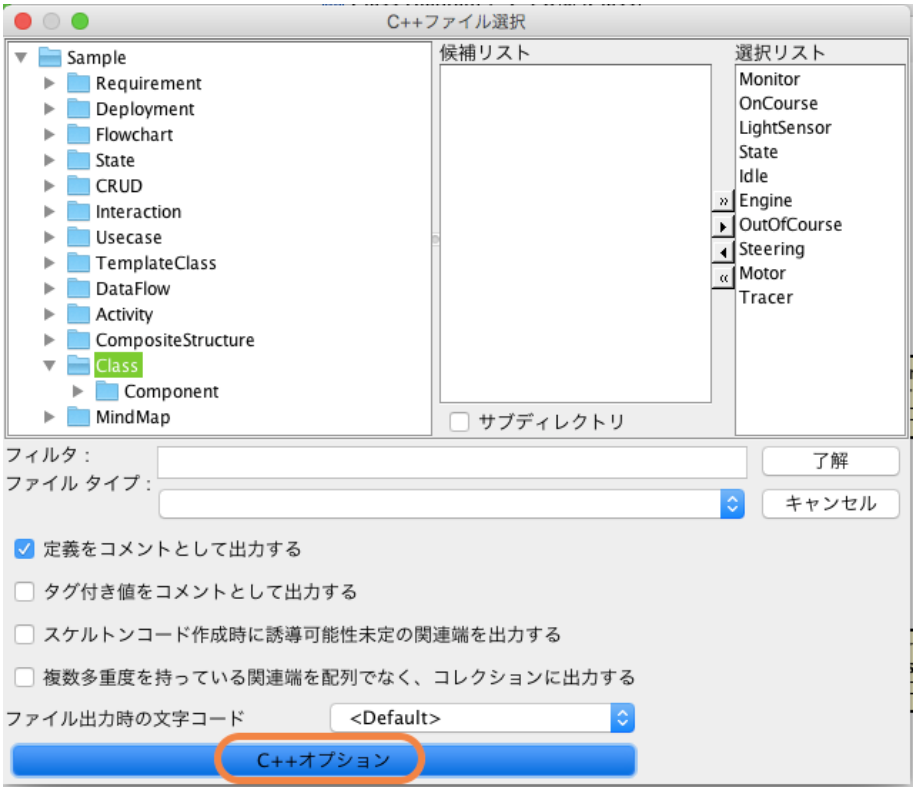
bool / char / signed char / unsigned char / short / unsigned short / short int / signed short int / unsigned short int / int / signed int / unsigned int / long / unsigned long / long int / signed long int / unsigned long int / float / double / long double / wchar_t

C++スケルトンコードを作成してみよう

インストールフォルダ配下のSample.asta を開いて[ツール]-[C++]-[C++スケルトンコードの作成]を選択します。



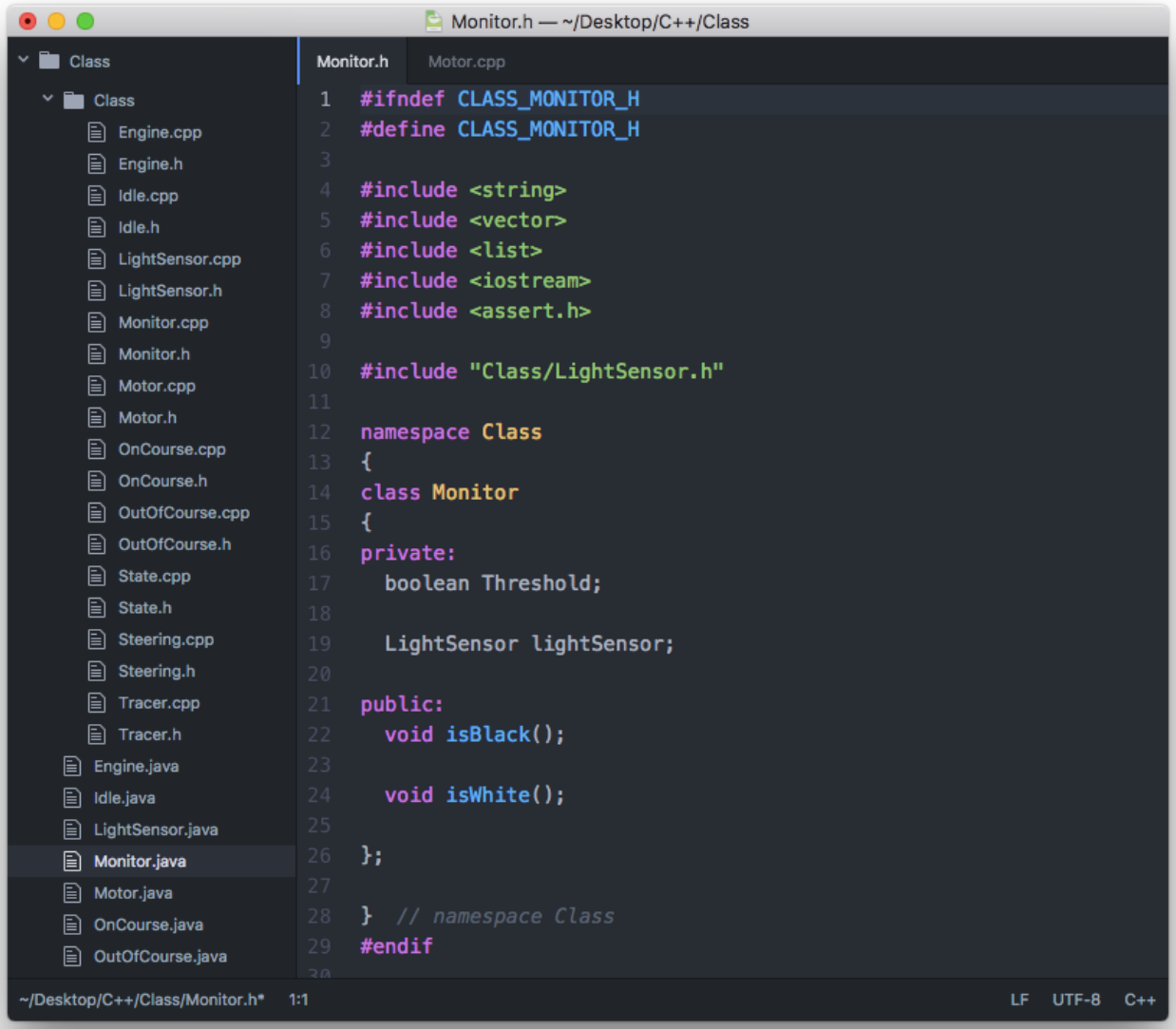
Class パッケージ直下のソースを出力しましょう。又、[C++出力オプション]を押すと、ヘッダーソースファイルなどの出力方法などを設定できます。



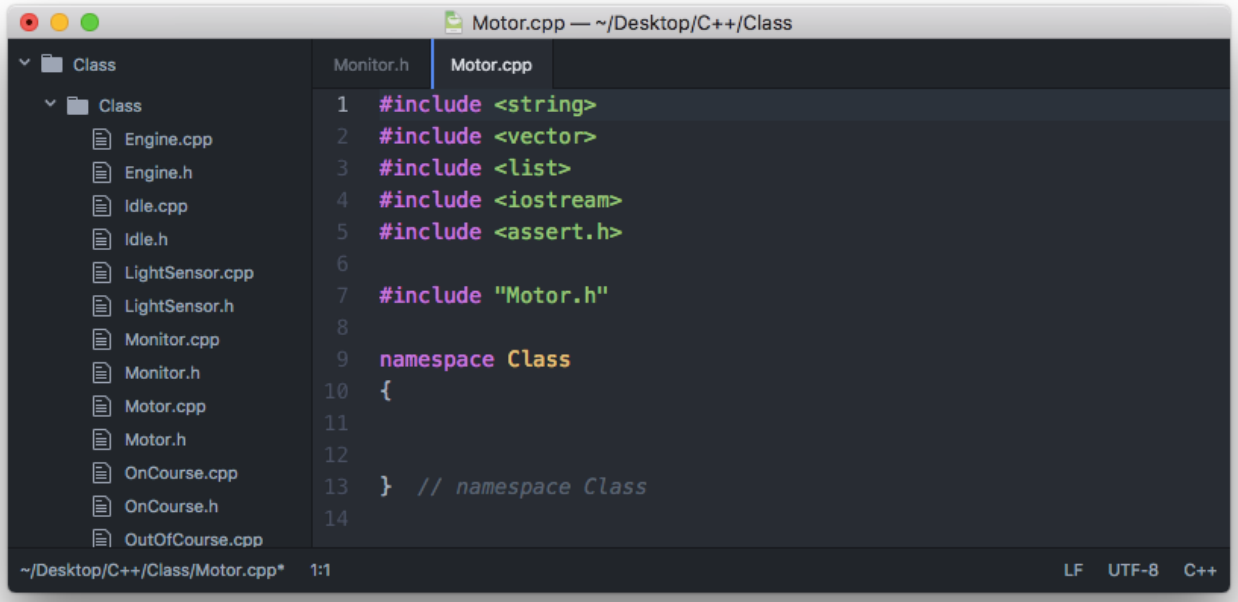
以下のようにヘッダーファイル、ソースファイル共に出力されましたね。

名前	変更日	サイズ	種類
Engine.cpp	14:42	237 バイト	C++ ソース
Engine.h	14:42	317 バイト	C ヘッダ
Idle.cpp	14:42	154 バイト	C++ ソース
Idle.h	14:42	241 バイト	C ヘッダ
LightSensor.cpp	14:42	161 バイト	C++ ソース
LightSensor.h	14:42	223 バイト	C ヘッダ
Monitor.cpp	14:42	213 バイト	C++ ソース
Monitor.h	14:42	342 バイト	C ヘッダ
Motor.cpp	14:42	155 バイト	C++ ソース
Motor.h	14:42	203 バイト	C ヘッダ
OnCourse.cpp	14:42	158 バイト	C++ ソース
OnCourse.h	14:42	255 バイト	C ヘッダ
OutOfCourse.cpp	14:42	161 バイト	C++ ソース
OutOfCourse.h	14:42	266 バイト	C ヘッダ
State.cpp	14:42	179 バイト	C++ ソース
State.h	14:42	228 バイト	C ヘッダ
Steering.cpp	14:42	237 バイト	C++ ソース
Steering.h	14:42	315 バイト	C ヘッダ
Tracer.cpp	14:42	156 バイト	C++ ソース
Tracer.h	14:42	351 バイト	C ヘッダ

Monitor.h を開いてみます。



Monitor.cpp を開いてみます。



正しく出力されていることがわかりますね。

C++リバースをしてみよう

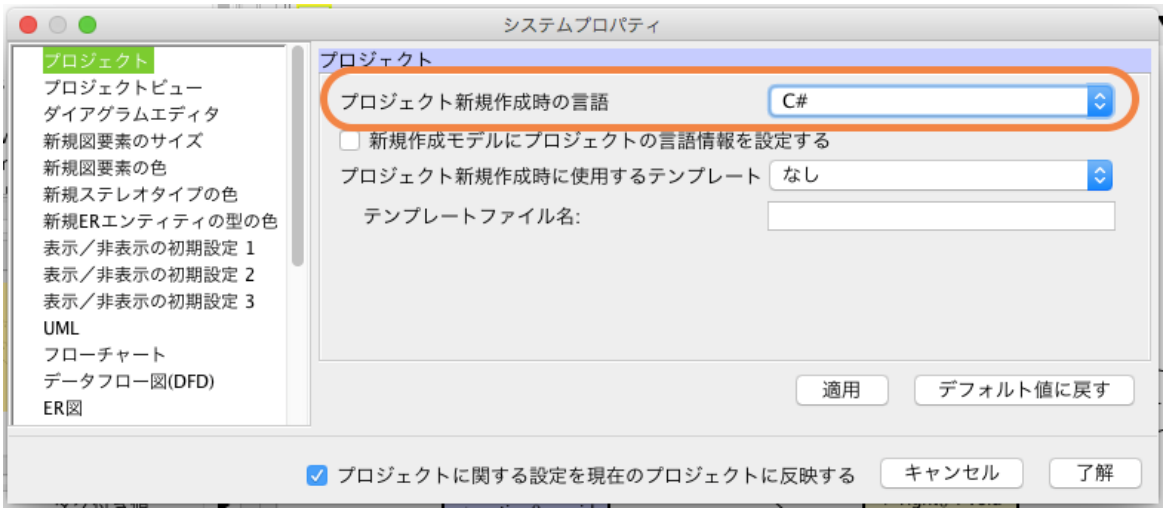
C++リバースは、プラグインのインストールが必要です。詳しい仕様方法は、プラグインページをご参照ください。
詳しい使用法はastah* C++プラグインページをご参照ください。
C++リバースプラグイン：http://astah.change-vision.com/ja/feature/cpp-reverse-plugin.html

C#

プロジェクトの C#言語設定は、プロパティビュー、または、システムプロパティで設定します。

プロパティビュー：

システムプロパティ：[ツール] – [システムプロパティ] – [プロジェクト] – [プロジェクトの新規作成時の言語] – [C#]



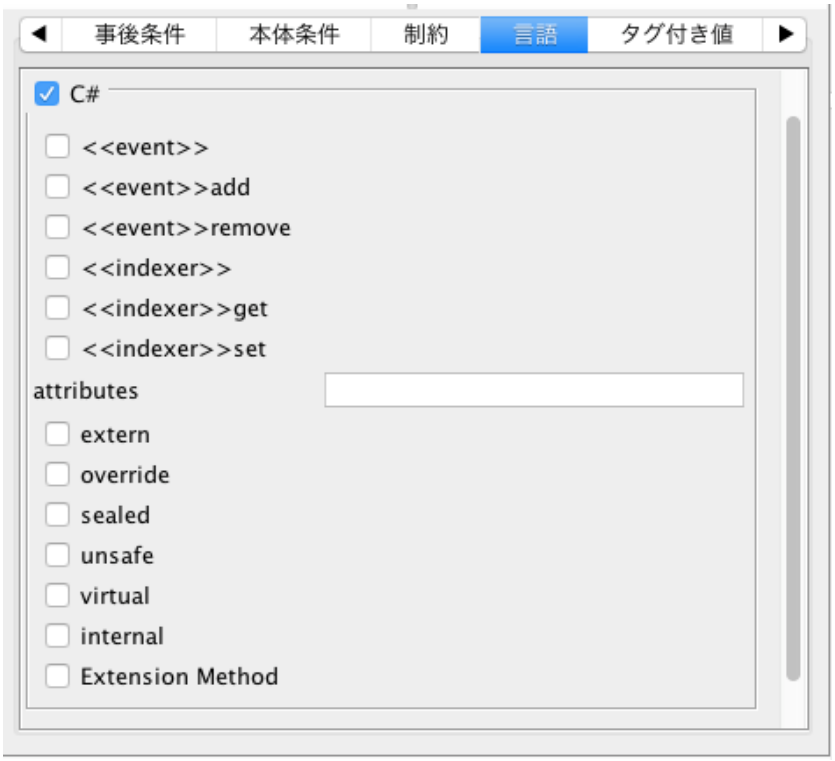
C#言語固有の設定をしてみよう

クラス、属性、操作の[言語]タブ（プロパティビュー）で、C#言語固有の設定が可能です。

［クラス］

［属性］

［操作］

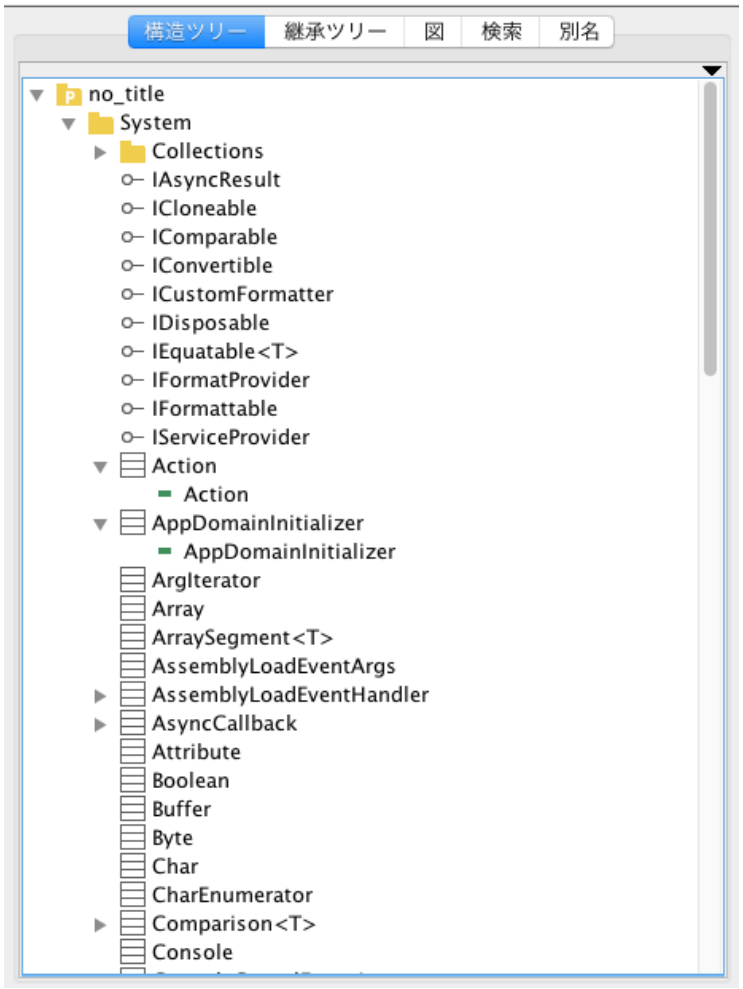


C#用のデフォルトモデルを使ってみよう

astah* professional インストールフォルダ¥ template¥project¥ C_Sharp.asta

C#用のデフォルトモデルは、[ファイル] – [テンプレートからプロジェクトの新規作成] – [C_Sharp.asta]を選択して開けます。又、新規プロジェクト作成時に、常にこのモデルを使いたい場合は、[ツール] – [システムプロパティ] – [プロジェクト] を開き、下記のように設定します。

C#用のデフォルトモデルは、下記のようなモデルを含みます。



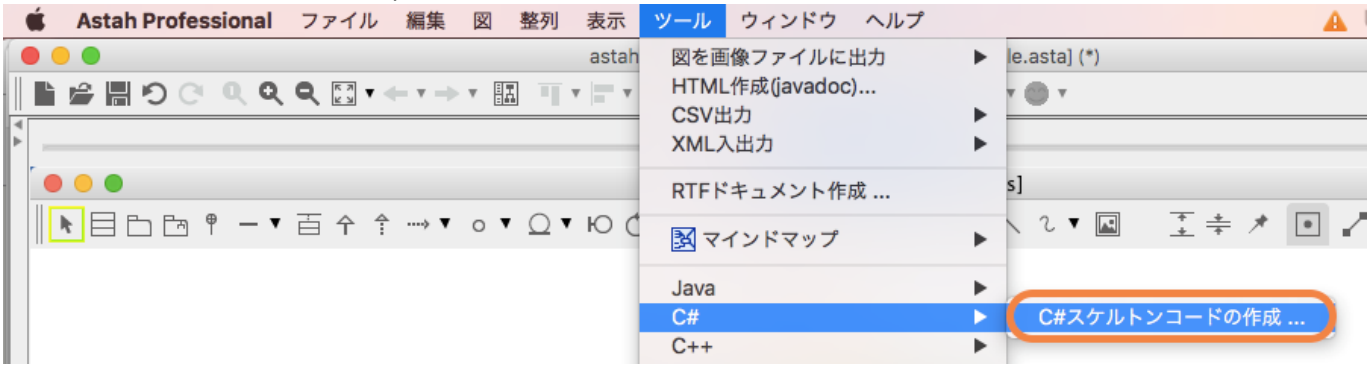
C#プリミティブ型の対応

各プリミティブ型に対応しています。

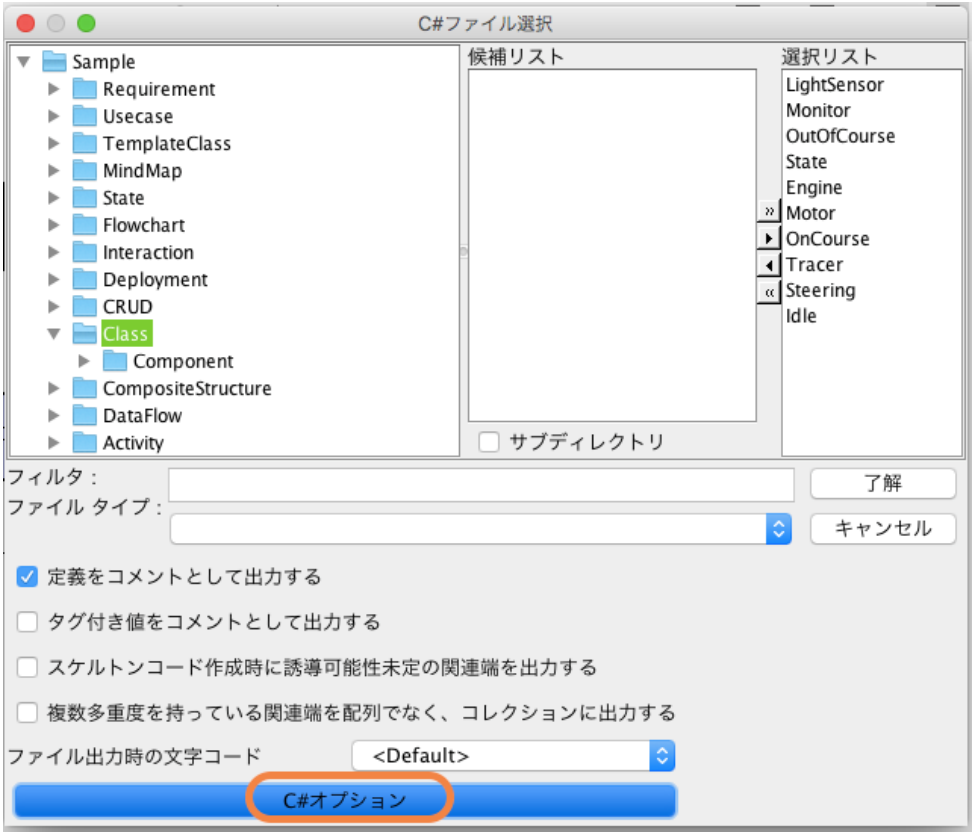
bool / byte / char / decimal / double / float / int / long / object / sbyte / short / string / uint / ulong / ushort

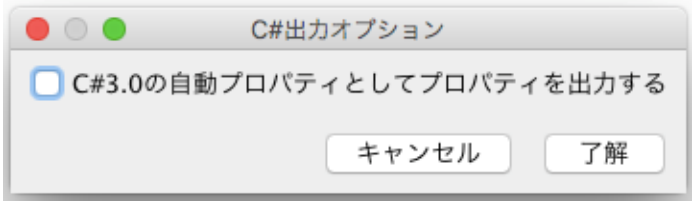
C#スケルトンコードを作成してみよう

インストールフォルダ配下のSample.asta を開いて[ツール]-[C#]-[C#スケルトンコードの作成]を選択します。



Class パッケージ直下のソースを出力しましょう。又、[C#出力オプション]を押すとプロパティ出力を設定できます。

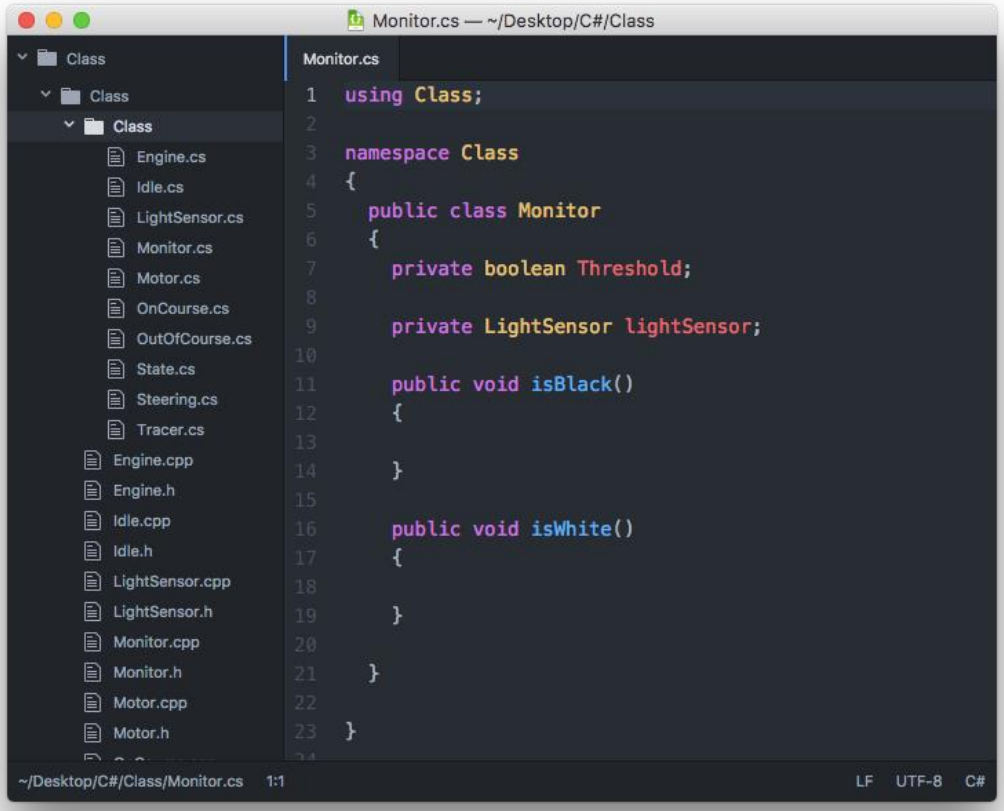




以下のように出力されましたね。

名前	変更日	サイズ	種類
Engine.cs	15:05	188 バイト	C# ソース
Idle.cs	15:05	69 バイト	C# ソース
LightSensor.cs	15:05	54 バイト	C# ソース
Monitor.cs	15:05	198 バイト	C# ソース
Motor.cs	15:05	48 バイト	C# ソース
OnCourse.cs	15:05	73 バイト	C# ソース
OutOfCourse.cs	15:05	76 バイト	C# ソース
State.cs	15:05	81 バイト	C# ソース
Steering.cs	15:05	182 バイト	C# ソース
Tracer.cs	15:05	147 バイト	C# ソース

Monitor.cs を開いてみます。



正しく出力されていることがわかりますね。

C#リバースをしてみよう

C++リバースは、プラグインのインストールが必要です。
詳しい使用方は、プラグインページをご参照ください。
C#リバースプラグイン：http://astah.change-vision.com/ja/feature/csharp-reverse-plugin.html