# Documentation at http://www.scintilla.org/SciTEDoc.html

# Globals

# Masm files

file.patterns.asm=\*.asm;\*.s;\*.A51;\*.A66;\*.A251;\*.inc;\*.sct

filter.asm=Assembler (asm)|$(file.patterns.asm)|

lexer.$(file.patterns.asm)=asm

# c/cpp

file.patterns.hpp=\*.h;\*.hh;\*.hpp;\*.hxx

file.patterns.cpp=\*.c;\*.cc;\*.cpp;\*.cxx;\*.h;\*.hh;\*.hpp;\*.hxx;\*.ipp;\*.sma;\*.ini

file.patterns.cplusplus=\*.cc;\*.cpp;\*.cxx

file.patterns.java=\*.java;\*.jad;\*.pde

file.patterns.javafx=\*.fx

file.patterns.test=\*.pln;\*.inc;\*.t

file.patterns.js=\*.js;\*.es

file.patterns.cs=\*.cs

file.patterns.rc=\*.rc;\*.rc2;\*.dlg

file.patterns.idl=\*.idl;\*.odl

file.patterns.flash=\*.as;\*.asc;\*.jsfl

file.patterns.ch=\*.ch;\*.chf;\*.chs

file.patterns.vala=\*.vala

file.patterns.c.like=$(file.patterns.cpp);$(file.patterns.cs);$(file.patterns.idl);$(file.patterns.java);$(file.patterns.js);$(file.patterns.flash);$(file.patterns.ch);$(file.patterns.vala)

shbang.tcc=cpp

shbang.tinycc=cpp

filter.cpp=C/C++ (c cc cpp cxx cs vala h hh hxx hpp ipp dlg rc rc2 mak)|\

$(file.patterns.cpp);$(file.patterns.cs);$(file.patterns.rc);$(file.patterns.vala);\*.mak;make\*|

filter.java=Java (java)|\*.java|

filter.js=JavaScript (js es)|$(file.patterns.js)|

filter.idl=IDL (idl odl)|$(file.patterns.idl)|

filter.test=SilkTest (pln inc t)|$(file.patterns.test)|

filter.flash=Flash (as asc jsfl)|$(file.patterns.flash)|

filter.ch=Ch (ch chf chs)|$(file.patterns.ch)|

lexer.$(file.patterns.java)=cpp

lexer.$(file.patterns.javafx)=cpp

lexer.$(file.patterns.cpp)=cpp

lexer.$(file.patterns.rc)=cpp

lexer.$(file.patterns.idl)=cpp

lexer.$(file.patterns.cs)=cpp

lexer.$(file.patterns.js)=cpp

lexer.$(file.patterns.test)=cpp

lexer.$(file.patterns.flash)=cpp

lexer.$(file.patterns.ch)=cpp

lexer.$(file.patterns.vala)=cpp

# html

# tpl is X and Mail specific

file.patterns.web=\*.html;\*.htm;\*.asp;\*.shtml;\*.htd;\*.jsp

file.patterns.php=\*.php3;\*.phtml;\*.php

file.patterns.html=$(file.patterns.web);$(file.patterns.php);\*.htt;\*.cfm;\*.tpl;\*.dtd;\*.hta

file.patterns.vxml=\*.vxml

file.patterns.xml=\*.xml;\*.xsl;\*.svg;\*.xul;\*.xsd;\*.dtd;\*.xslt;\*.axl;\*.xrc;\*.rdf

file.patterns.docbook=\*.docbook

file.patterns.mako=\*.mak;\*.mako

lexer.$(file.patterns.html)=hypertext

lexer.$(file.patterns.vxml)=hypertext

lexer.$(file.patterns.xml)=xml

lexer.$(file.patterns.docbook)=hypertext

lexer.$(file.patterns.mako)=hypertext

filter.web=Web (html htm asp shtml css xml docbook jsp)|$(file.patterns.web);\*.css;\*.xml;\*.docbook|

filter.php=PHP (php php3 phtml)|$(file.patterns.php)|

# other files

file.patterns.props=\*.properties;\*.session;\*.ini;\*.inf;\*.reg;\*.url;\*.cfg;\*.cnf;\*.aut

file.patterns.text=\*.txt;\*.log;\*.lst;\*.doc;\*.diz;\*.nfo

file.patterns.batch=\*.bat;\*.cmd;\*.nt

file.patterns.diff=\*.diff;\*.patch

file.patterns.make=makefile;Makefile;\*.mak;configure

filter.properties=Properties (ini inf reg url cfg cnf)|$(file.patterns.props)|

filter.text=Text (txt log lst doc diz nfo)|$(file.patterns.text);make\*|

filter.batch=Batch (bat cmd nt)|$(file.patterns.batch)|

filter.diff=Difference (diff patch)|$(file.patterns.diff)|

lexer.$(file.patterns.props)=props

lexer.$(file.patterns.batch)=batch

lexer.\*.err=errorlist

lexer.$(file.patterns.make)=makefile

lexer.\*.iface=makefile

lexer.$(file.patterns.diff)=diff

configclass.cpp.title=C/CPP Editor files

configclass.asm.title=Asm Editor files

# specification for configurable styles

configclass.type=cpp;asm;txt

configclass.cpp.title=C/C++ Editor files

# configurable C/CPP styles, caution! delimiter: : and ;

# style Nr 32 (STYLE\_DEFAULT) must be specified first

configclass.cpp.element=32:Text;4:Number;10:Operator, Block, Brackets;1:Block Comment;\

2:Line Comment;5:Keyword;6:String;11:Identifier;9:Preprocessor;7:Character;34:Matching Braces;\

35:Mismatched Braces;16:User Keyword / Label;12:Incomplete String

# configurable Asm styles, caution! delimiter: : and ;

configclass.asm.title=Asm Editor files

configclass.asm.element=32:Default;1:Comment;2:Number;3:String;4:Operator;5:Identifier;\

6:CPU Instruction;7:FPU Instruction;9:Assembler Directive;10:Assembler Directive Operand;\

11:Comment Block (e.g. GNU);12:Character/String (single quote);8:Register / User Keywords / Label

# configurable other styles (\*.txt), caution! delimiter: : and ;

configclass.txt.title=Editor Text files

configclass.txt.element=32:Default

font.monospace.txt=Courier New

style.txt.32=font:Courier New,size:10,fore:#000000,back:#FFFFFF

# templates specification

template.cpp="Header","// Header:\r\n// File Name: |\r\n// Author:\r\n// Date:\r\n";"for","for(|;;)\r\n{\r\n}";\\

"while","while (|)\r\n{\r\n}";"continue","continue;";"if","if (|)";"function","void function(|)\r\n{\r\n\r\n}\r\n";\\

"struct","struct | \r\n{\r\n\r\n};\r\n";"fpointer\_type","typedef int (\* |F)();\r\n";\\

"do","do\r\n{\r\n\t// TODO: enter the block content here\r\n\t\r\n\t|\r\n} while ();\r\n";\\

"enum","enum |\r\n{\r\n\t\r\n};\r\n";"void","void | ();\r\n";"#if","#if |\r\n\r\n#endif";\\

"#define","#define |";"switch","switch (|)\r\n{\r\n\tcase:\r\n\t\tbreak;\r\n\tcase:\r\n\t\tbreak;\r\n\tdefault:\r\n\t\tbreak;\r\n}";\\

"ifelse","if (|)\r\n{\r\n}\r\nelse\r\n{\r\n}";"#include","#include "

# Sizes and visibility in edit pane

line.margin.visible=1

line.margin.width=4

margin.width=16

#fold.margin.width=16

#fold.margin.colour=#FF0000

#fold.margin.highlight.colour=#0000FF

#blank.margin.left=4

#blank.margin.right=4

buffered.draw=1

#two.phase.draw=0

use.palette=0

highlight.matchingbraces=1

# Element styles

#view.eol=1

#control.char.symbol=.

caret.period=500

view.whitespace=0

view.indentation.whitespace=1

view.indentation.guides=1

view.indentation.examine=3

highlight.indentation.guides=1

caret.additional.blinks=0

indentation.lines.tocheck=1000 # limited to 100 <= n <= 3000

#caret.width=2

selection.fore=#000000

selection.back=#86C7F7

caret.fore=#000000

caret.back=#DEFEE6

#caret.line.back=#FFFED8

#caret.line.back=#00FF00

caret.line.back.alpha=256

caretline.visible=1

edge.column=80

edge.mode=0

edge.colour=#66FAFA

braces.check=1

braces.sloppy=1

#selection.fore=#006000

# DADADA used as background because it yields standard silver C0C0C0

# on low colour displays and a reasonable light grey on higher bit depths

#if PLAT\_WINNT

selection.alpha=256

# alpha value between 0 and 256 (100% colour)

#finding

find.replace.regexp.posix=1

# Scripting

ext.lua.startup.script=$(SciteUserHome)/SciTEStartup.lua

ext.lua.auto.reload=1

#ext.lua.reset=1

# Indentation

tabsize=4

#indent.size=4

use.tabs=0

#indent.auto=1

indent.automatic=1

indent.opening=0

indent.closing=0

#tab.indents=0

backspace.unindents=0

indent.closing=0

# Folding

# enable folding, and show lines below when collapsed.

fold=1

fold.compact=1

fold.flags=16

fold.symbols=1

#fold.on.open=1

fold.comment=1

fold.preprocessor=1

fold.cpp=1

fold.num.to.save=500

# turn of evaluating preprocessor to eventually grey out text

lexer.cpp.track.preprocessor=0

# Behaviour

#eol.mode=LF

eol.auto=1

clear.before.execute=0

#vc.home.key=1

#wrap.aware.home.end.keys=1

caret.policy.xslop=1

caret.policy.width=20

caret.policy.xstrict=0

caret.policy.xeven=0

caret.policy.xjumps=0

caret.policy.yslop=1

caret.policy.lines=1

caret.policy.ystrict=1

caret.policy.yeven=1

caret.policy.yjumps=0

#visible.policy.strict=1

#visible.policy.slop=1

#visible.policy.lines=4

#time.commands=1

#caret.sticky=1

#properties.directory.enable=1

# Internationalisation

# Japanese input code page 932 and ShiftJIS character set 128

#code.page=932

#character.set=128

# Unicode

#code.page=65001

#character.set=204

# ANSI code page

code.page=0

# Required for Unicode to work on GTK+:

#LC\_CTYPE=en\_US.UTF-8

# Define values for use in the imported properties files

chars.alpha=abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ

chars.numeric=0123456789

chars.accented=姎寽�类玲骡勉匿佩奇晴辱砷赎穗天晚晤巷叙疡因芋贼挣重龅

# This is a better set for Russian:

#chars.accented=类玲骡勉匿佩ǜ奇晴辱砷赎穗天晚晤巷叙疡因芋贼挣嘱作伉蠃邡埯茳蔟撖�

# Give symbolic names to the set of fonts used in the standard styles.

#if PLAT\_WIN

font.base=font:Verdana,size:10

font.small=font:Verdana,size:10

font.smaller=font:Verdana,size:10

font.comment=font:Verdana,size:10

font.code.comment.box=$(font.comment)

font.code.comment.line=$(font.comment)

font.code.comment.doc=$(font.comment)

font.code.comment.nested=$(font.comment)

font.text=font:Times New Roman,size:10

font.text.comment=font:Verdana,size:10

font.embedded.base=font:Verdana,size:10

font.embedded.comment=font:Comic Sans MS,size:10

# text and comment font monospace

font.monospace=font:Courier New,size:10

font.vbs=font:Lucida Sans Unicode,size:10

# Give symbolic names to the set of colours used in the standard styles.

colour.code.comment.box=fore:#007F00

colour.code.comment.line=fore:#007F00

colour.code.comment.doc=fore:#3F703F

colour.code.comment.nested=fore:#A0C0A0

colour.text.comment=fore:#0000FF,back:#D0F0D0

colour.other.comment=fore:#007F00

colour.embedded.comment=back:#E0EEFF

colour.embedded.js=back:#F0F0FF

colour.notused=back:#FF0000

colour.number=fore:#007F7F

colour.keyword=fore:#0000FF

colour.string=fore:#7F007F

colour.char=fore:#7F007F

colour.operator=fore:#000000

colour.preproc=fore:#7F7F00

colour.error=fore:#FFFF00,back:#FF0000

# colour for tooltip

colour.tip=fore:#000000,back:#FFFFE6

monospaced.font.cpp=1

monospaced.font.asm=1

monospaced.font.txt=1

# Global default styles for all languages

# Default

style.\*.32=$(font.base)

# Line number

linenumber=1

#style.\*.33=back:#C0C0C0,$(font.base)

style.\*.33=back:#E8E8E8,fore:#000000,$(font.small)

# Brace highlight

style.\*.34=fore:#0000FF,bold

# Brace incomplete highlight

style.\*.35=fore:#FF0000,bold

# Control characters

style.\*.36=

# Indentation guides

style.\*.37=fore:#C0C0C0,back:#FFFFFF

# profiling text margin

style.\*.39=back:#FFFFCC,fore:#000000,$(font.small)

#style.\*.39=back:#FFFFCC,fore:#000000,$(font.monospace)

style.\*.38=$(style.\*.32)

### default settings for C/C++

# C++ styles

# Default

#style.cpp.32=$(font.base),fore:#FF0000,back:#000000

#style.cpp.32=$(font.base)

# White space

style.cpp.0=fore:#808080

# Comment: /\* \*/.

style.cpp.1=fore:#007F00,font:Verdana,size:10

# Line Comment: //.

style.cpp.2=fore:#007F00,font:Verdana,size:10

# Doc comment: block comments beginning with /\*\* or /\*!

#style.cpp.3=$(colour.code.comment.doc),$(font.code.comment.doc)

style.cpp.3=$(style.cpp.1)

# Number

style.cpp.4=$(colour.number)

# Keyword

style.cpp.5=$(colour.keyword)

# Double quoted string

style.cpp.6=$(colour.string)

# Single quoted string

style.cpp.7=$(colour.char)

# UUIDs (only in IDL)

style.cpp.8=fore:#804080

# Preprocessor

style.cpp.9=$(colour.preproc)

# Operators

style.cpp.10=$(colour.operator)

# Identifiers

style.cpp.11=

# End of line where string is not closed

style.cpp.12=fore:#000000,$(font.monospace),back:#E0C0E0,eolfilled

# Verbatim strings for C#

style.cpp.13=fore:#007F00,$(font.monospace),back:#E0FFE0,eolfilled

# Regular expressions for JavaScript

style.cpp.14=fore:#3F7F3F,$(font.monospace),back:#E0F0FF,eolfilled

# Doc Comment Line: line comments beginning with /// or //!.

#style.cpp.15=$(colour.code.comment.doc),$(font.code.comment.doc)

style.cpp.15=$(style.cpp.2)

# Keywords2

style.cpp.16=fore:#B00040

# Comment keyword

style.cpp.17=fore:#3060A0,$(font.code.comment.doc)

# Comment keyword error

style.cpp.18=fore:#804020,$(font.code.comment.doc)

# Raw strings for C++0x

style.cpp.20=$(colour.string),back:#FFF3FF,eolfilled

# Triple-quoted strings for Vala

style.cpp.21=$(font.monospace),fore:#007F00,back:#E0FFE0,eolfilled

#brace light

style.cpp.34=fore:#500000,back:#20FFFF

#badbrace

style.cpp.35=fore:#FF0000,back:#DCDCDC

# Inactive states are 64 greater than their active counterparts

# White space

style.cpp.64=fore:#C0C0C0

# Comment: /\* \*/.

style.cpp.65=$(style.cpp.1),fore:#90B090

# Line Comment: //.

style.cpp.66=$(style.cpp.2),fore:#90B090

# Doc comment: block comments beginning with /\*\* or /\*!

style.cpp.67=$(style.cpp.3),fore:#D0D0D0

# Number

style.cpp.68=$(style.cpp.4),fore:#90B0B0

# Keyword

style.cpp.69=$(style.cpp.5),fore:#9090B0

# Double quoted string

style.cpp.70=$(style.cpp.6),fore:#B090B0

# Single quoted string

style.cpp.71=$(style.cpp.7),fore:#B090B0

# UUIDs (only in IDL)

style.cpp.72=$(style.cpp.8),fore:#C0C0C0

# Preprocessor

style.cpp.73=$(style.cpp.9),fore:#B0B090

# Operators

style.cpp.74=$(style.cpp.10),fore:#B0B0B0

# Identifiers

style.cpp.75=$(style.cpp.11),fore:#B0B0B0

# End of line where string is not closed

style.cpp.76=$(style.cpp.12),fore:#000000

# Verbatim strings for C#

style.cpp.77=$(style.cpp.13),fore:#90B090

# Regular expressions for JavaScript

style.cpp.78=$(style.cpp.14),fore:#7FAF7F

# Doc Comment Line: line comments beginning with /// or //!.

style.cpp.79=$(style.cpp.15),fore:#C0C0C0

# Keywords2

style.cpp.80=$(style.cpp.16),fore:#C0C0C0

# Comment keyword

style.cpp.81=$(style.cpp.17),fore:#C0C0C0

# Comment keyword error

style.cpp.82=$(style.cpp.18),fore:#C0C0C0

# Raw strings for C++0x

style.cpp.84=$(style.cpp.20),fore:#B090B0

# Triple-quoted strings for Vala

style.cpp.85=$(style.cpp.21),fore:#90B090

# Braces are only matched in operator style

braces.cpp.style=10

###

# Printing - only works on Windows

#if PLAT\_WIN

#print.colour.mode=0

print.syntax.coloring=1

print.magnification=-1

# Setup: left, right, top, bottom margins, in local units:

# hundredths of millimeters or thousandths of inches

print.margins=1500,1000,1000,1500

# Header/footer:

# && = &; &p = current page

# &f = file name; &F = full path

# &d = file date; &D = current date

# &t = file time; &T = full time

# print.header.format=$(FileNameExt) -- Printed on $(CurrentDate), $(CurrentTime) -- Page $(CurrentPage)

# print.footer.format=$(FilePath) -- File date: $(FileDate) -- File time: $(FileTime)

print.footer.format=$(FilePath)

# Header/footer style

#print.header.style=font:Arial,size:12,bold

print.header.style=font:Arial,size:10

print.footer.style=font:Arial Narrow,size:10,italics

# Warnings - only works on Windows and needs to be pointed at files on machine

#if PLAT\_WIN

# warning.findwrapped=100,E:\Windows\Media\SFX\Boing.wav

# warning.notfound=0,Effect.wav

# warning.wrongfile=0,Glass.wav

# warning.executeok=0,Fanfare.wav

# warning.executeko=100,GlassBreak.wav

# warning.nootherbookmark=100,Boing2.wav

# User defined key commands

user.shortcuts=\

Ctrl+Shift+V|IDM\_PASTEANDDOWN|\

Ctrl+PageUp|IDM\_PREVFILE|\

Ctrl+PageDown|IDM\_NEXTFILE|

#KeypadPlus|IDM\_EXPAND|\

#KeypadMinus|IDM\_BLOCK\_COMMENT|

#user.context.menu=\

#||\

#Next File|IDM\_NEXTFILE|\

#Prev File|IDM\_PREVFILE|

# C/CPP Editor files

style.cpp.32=font:Courier New,size:10,fore:#000000,back:#FFFFFF

style.cpp.4=font:Courier New,size:10,fore:#007F7F

style.cpp.10=font:Courier New,size:10,fore:#000000

style.cpp.1=font:Courier New,size:10,fore:#007F00

style.cpp.2=font:Courier New,size:10,fore:#007F00

style.cpp.5=font:Courier New,size:10,fore:#0000FF

style.cpp.6=font:Courier New,size:10,fore:#7F007F

style.cpp.11=font:Courier New,size:10,fore:#000000

style.cpp.9=font:Courier New,size:10,fore:#7F7F00

style.cpp.7=font:Courier New,size:10,fore:#7F007F

style.cpp.34=font:Courier New,size:10,fore:#500000,back:#20FFFF

style.cpp.35=font:Courier New,size:10,fore:#FF0000

style.cpp.16=font:Courier New,size:10,fore:#B00040

# Asm Editor files

style.asm.32=font:Courier New,size:10,fore:#000000,back:#FFFFFF

style.asm.1=font:Courier New,size:10,fore:#616161

style.asm.2=font:Courier New,size:10,fore:#FF0000

style.asm.3=font:Courier New,size:10,fore:#7F007F

style.asm.4=font:Courier New,size:10,fore:#000000

style.asm.5=font:Courier New,size:10,fore:#000000

style.asm.6=font:Courier New,size:10,fore:#0000FF

style.asm.7=font:Courier New,size:10,fore:#0000FF

style.asm.9=font:Courier New,size:10,fore:#0000FF

style.asm.10=font:Courier New,size:10,fore:#0000FF

style.asm.11=font:Courier New,size:10,fore:#007F00

style.asm.12=font:Courier New,size:10,fore:#7F007F

style.asm.8=font:Courier New,size:10,fore:#46AA03

# other files

style.txt.32=$(font.base),fore:#000000,back:#FFFFFF

usemonofont.txt=0

function.scanner.project=1

function.scanner.files=1

function.scanner.modules=1

# specification for comment

comment.block.cpp=//

comment.block.at.line.start.cpp=1

comment.stream.start.cpp=/\*

comment.stream.end.cpp=\*/

comment.box.start.cpp=/\*

comment.box.middle.cpp= \*

comment.box.end.cpp= \*/

comment.block.ch=//~

comment.stream.start.ch=/\*

comment.stream.end.ch=\*/

comment.box.start.ch=/\*

comment.box.middle.ch= \*

comment.box.end.ch= \*/

# keywords3 is for doc comment keywords, highlighted in 17

#CPP doxygen

#file.patterns.doxygen.langs=$(file.patterns.cpp);$(file.patterns.java)

#keywordclass.doxygen=a addindex addtogroup anchor arg attention \

#author b brief bug c class code date def defgroup deprecated dontinclude \

#e em endcode endhtmlonly endif endlatexonly endlink endverbatim enum example exception \

#f$ f[ f] file fn hideinitializer htmlinclude htmlonly \

#if image include ingroup internal invariant interface latexonly li line link \

#mainpage name namespace nosubgrouping note overload \

#p page par param param[in] param[out] \

#post pre ref relates remarks return retval \

#sa section see showinitializer since skip skipline struct subsection \

#test throw throws todo typedef union until \

#var verbatim verbinclude version warning weakgroup $ @ \ & < > # { }

#keywords3.$(file.patterns.doxygen.langs)=$(keywordclass.doxygen)

word.chars.cxx=$(chars.alpha)$(chars.numeric)$(chars.accented)\_#

word.characters.$(file.patterns.cpp)=$(word.chars.cxx)

calltip.cpp.word.characters=$(chars.alpha)$(chars.numeric)\_

# Folding

#fold.cpp.syntax.based=0

#fold.cpp.comment.multiline=0

fold.cpp.comment.explicit=0

#defaults for fold.cpp.explicit.start=//{ and fold.cpp.explicit.end=//}

# can be replaced by defining custom strings, e.g. //[ and //]

# fold.cpp.explicit.start=

# fold.cpp.explicit.end=

#if fold strings are set to something like /\*{{{ and /\*}}} (Origami/WinF style fold strings), enable

# fold.cpp.explicit.anywhere, allowing explicit fold points being anywhere, not just in line comments

#fold.cpp.explicit.anywhere=1

#fold.at.else=1

word.characters.$(file.patterns.javafx)=$(word.chars.cxx)$-

file.patterns.c.except.cpp=$(file.patterns.cs);$(file.patterns.idl);\*.java;$(file.patterns.flash);$(file.patterns.ch)

statement.indent.$(file.patterns.c.except.cpp)=5 case default do else for if while

statement.indent.$(file.patterns.cpp)=5 case default do else for if private protected public while

statement.end.$(file.patterns.c.like)=10 ;

statement.lookback.$(file.patterns.c.like)=20

block.start.$(file.patterns.c.like)=10 {

block.end.$(file.patterns.c.like)=10 }

// settings for auto completion

autocomplete.cpp.start.characters=.(>,

#autocomplete.cpp.fillups=\t\n\r !\"#$%&'()\*+,-/:;<=?@[\\]^`{|}~

autocompleteword.automatic=0

autocomplete.choose.single=1

autocomplete.MaxFileSize=4000000

# setting for code completion and syntax check

cc.autolist=1

cc.highlightsyntax=1

cc.showparameters=1

cc.triggerlist=1

cc.triggernumchars=3

cc.enter.as.fillup=0

# timer for dynamic syntax check in millisecond

cc.timer.interval=2000

# autosave for editor files

autosave=0

autosave.interval=5

styling.within.preprocessor=1

preprocessor.symbol.$(file.patterns.cpp)=#

preprocessor.start.$(file.patterns.cpp)=if ifdef ifndef

preprocessor.middle.$(file.patterns.cpp)=else elif

preprocessor.end.$(file.patterns.cpp)=endif

lexer.flagship.styling.within.preprocessor=1

# Keywords for C/C++

keywordclass.cpp=alignas alignof and and\_eq asm auto bitand bitor bool break case catch char char16\_t char32\_t class compl \

const constexpr const\_cast continue decltype default delete do double dynamic\_cast else enum explicit export extern false \

float for friend goto if inline int long mutable namespace new noexcept not not\_eq nullptr operator or or\_eq private protected \

public register reinterpret\_cast return short signed sizeof static static\_assert static\_cast struct switch template this thread\_local \

throw true try typedef typeid typename union unsigned using virtual void volatile wchar\_t while xor xor\_eq

keywords.$(file.patterns.cpp)=$(keywordclass.cpp)

# batch files

monospaced.font.batch=1

font.monospace.batch=Courier New