baidu

BIDL VIM Plugin

百度接口定义语言 VIM 插件

百度在线网络技术（北京）有限公司
（版权所有, 翻版必究）

liuxupeng
2012/10/17

目录

BIDL VIM Plugin

1 介绍

1.1 背景

百度接口定义语言（Baidu Interface Definition Language，以下简称为 BIDL），是一种规范性语言，采用类 C/C++语法来定义 BGCC 客户程序与服务器程序之间的通信接口。BIDL 提供了一套通用的基本数据类型，并以这些基本数据类型来定义更为复杂的数据类型，剥离了语言与平台的差异性，使得客户程序与服务程序均不感知接口背后的实现细节。特别地，分离了业务逻辑与通信逻辑，大大提升了客户程序与服务程序的运行可靠性，并降低了编写客户程序与服务程序的复杂性。

为了能方便快捷地编写 BIDL 文件，故开发了此 VIM 插件。

1.2 内容

本手册涉及的内容包括 BIDL VIM Plugin 的安装与使用指导。

1.3 功能

本 VIM 插件包含的功能有：

1) BIDL 语法高亮
2) BIDL 关键字自动补全

2 安装与使用

bidl.vim 包含 BIDL 语法高亮 VIM 插件，bidl_words.txt 包含 BIDL 关键字单词表，用于自动补全。测试环境：vim7.3。bidl.vim 与 bidl_words.txt 见附录。

2.1 安装

1) 将 bidl.vim 放置在$HOME/.vim/ftplugin 目录中。
2) 将 bidl_words.txt 放置在$HOME/.vim 目录中。

3) 在$HOME/.vimrc 中添加如下行

syntax on

au BufRead,BufNewFile *.bidl setfiletype bidl

set dict+=~/.vim/bidl_words.txt

imap <c-l> <c-x><c-k>

2.2 使用

1) 新建 BIDL 文件 xx.bidl。

2) 在 VIM Insert 模式下，输入 se，然后按<Ctrl-l>，有如下显示：

3) 分别键入<Ctrl-n>和<Ctrl-p>，可在多个选项间跳转。

4) BIDL 语法高亮如下所示：

3　附录

bidl.vim

```
" Vim syntax file
" Language: bidl
" Maintainer:    liuxupeng<liuxupeng@baidu.com>
```

```
" Last Change:    2012 Oct 23

if exists("b:current_syntax")
    finish
endif


syn keyword      cStatement      namespace const in out all typedef
syn keyword      cStructure  struct enum class
syn keyword      cType           void boolean int8 int16 int32 int64 float string binary
syn keyword      cType           map sequence set
syn keyword      cConstant  true TRUE false FALSE


syn keyword cReservedError   _CLASS__       __DIR__    __FILE__    __LINE__    __METHOD__
__NAMESPACE__       abstract
syn keyword cReservedError   alias       alignas     alignof     and     and_eq      args    as
syn keyword cReservedError   asm         assert      auto BEGIN      bitand      bitor bool
syn keyword cReservedError   break       byte  case  catch char  char16_t    char32_t
syn keyword cReservedError   clone       compl       const_cast constexpr continue      cpp   cpp_type
syn keyword cReservedError   declare     decltype    def  default        del     delete      do
syn keyword cReservedError   double      dynamic     dynamic_cast    elif    else  elseif elsif
syn keyword cReservedError   END         enddeclare endfor       endforeachendif endswitch endwhile
syn keyword cReservedError   ensure      except      exec explicit      export      extends     extern
syn keyword cReservedError   final       finally     for  foreach        friend      function    global
syn keyword cReservedError   goto        if      implements      import      inline instanceof int
syn keyword cReservedError   interface is     java lambda      long module      mutable
syn keyword cReservedError   native      new next nil      noexcept    not  not_eq
syn keyword cReservedError   null nullptr    operator    optional    or    or_eq       package
syn keyword cReservedError   pass        print private        protected protocol    public      raise
syn keyword cReservedError   redo        register        reinterpret_cast request     request_len
required    rescue
syn keyword cReservedError   retry       return      self short signed      sizeof      static
syn keyword cReservedError   static_assert    static_cast strictfp     superswitch     synchronized
template
syn keyword cReservedError   then        this thread_local       throw       throws      transient   try
syn keyword cReservedError   typeid      typename undef       unionunless     unsigned    until
syn keyword cReservedError   use using var   virtual      volatile    wchar_t     when
syn keyword cReservedError   while       with xor  xor_eq      yield


syn region  cCppString start=+L\="+ skip=+\\\\\|\\"\|\\$+ excludenl end=+"+ end='$'


syn match   cNumbers  display transparent "\<\d\|\.\d" contains=cNumber,cFloat
syn match   cNumber         display contained "\d\+\(u\=l\{0,2}\|ll\=u\)\>"
syn match   cFloat          display contained "\d\+f"
syn match   cFloat          display contained "\d\+\.\d*\(e[-+]\=\d\+\)\=[fl]\="
```

```
syn match    cFloat          display contained "\.\d\+\(e[-+]\=\d\+\)\=[fl]\=\>"
syn match    cFloat          display contained "\d\+e[-+]\=\d\+[fl]\=\>"

syn region   cCommentL      start="//" skip="\\$" end="$" keepend
syn region   cComment matchgroup=cCommentStart start="/\*" end="\*/" fold extend
syntax match      cCommentError  display "\*/"

syn region   cIncluded   display contained start=+"+ skip=+\\\\\|\\"+ end=+"+
syn match    cInclude     display "^\s*include\>\s*["<]" contains=cIncluded

hi def link cCppString          String
hi def link cCommentL           cComment
hi def link cCommentStart       cComment
hi def link cNumber             Number
hi def link cFloat          Float
hi def link cCommentError       Error
hi def link cReservedError      Error
hi def link cStructure          Structure
hi def link cInclude            Include
hi def link cIncluded           String
hi def link cStatement          Statement
hi def link cType           Type
hi def link cConstant           Constant
hi def link cString         String
hi def link cComment            Comment

let b:current_syntax = "bidl"

" vim: ts=4
```

bidl_words.txt

```
include

int64

typedef

true

namespace

float

struct

TRUE

void

string

enum

false

boolean
```

```
binary

const

FALSE

int8

map

in

int16

sequence

out

int32

set

all

class
```