Subject: TheIde CodeEditor and Python
Posted by pivica on Wed, 29 Mar 2006 17:23:12 GMT
View Forum Message <> Reply to Message

TheIde is great editor for programing c/c++. But I'm also doing some projects in python and frankly all existing Linux editors for python don't satisfy me.
So I am willing to try to develop python module for TheIde. Stuff I wont are syntax highlighting and code completion and some kind of project management.

First question is how difficult is this?
And second, if answer on first question isn't mission imposible;) what classes and files to look in ide package.

Subject: Re: TheIde and Python
Posted by fudadmin on Wed, 29 Mar 2006 17:37:52 GMT
View Forum Message <> Reply to Message

pivica wrote on Wed, 29 March 2006 18:23TheIde is great editor for programing c/c++. But I'm also doing some projects in python and frankly all existing Linux editors for python don't satisfy me.
So I am willing to try to develop python module for TheIde. Stuff I wont are syntax highlighting and code completion and some kind of project management.

First question is how difficult is this?
And second, if answer on first question isn't mission imposible;) what classes and files to look in ide package.

uppsrc/CodeEditor
Syntax.cpp, Highlight.cpp

Subject: Re: TheIde and Python
Posted by fudadmin on Wed, 29 Mar 2006 18:35:30 GMT
View Forum Message <> Reply to Message

I've had a bit of success with Dialect interpreter with Ultimate++
So, I can share my experience...

In CodeEditor.h

look at

```
public:
...
 enum {
  HIGHLIGHT_NONE = -1, HIGHLIGHT_CPP = 0, HIGHLIGHT_USC, HIGHLIGHT_JAVA,
```

```
HIGHLIGHT_T, HIGHLIGHT_CALC,
  HIGHLIGHT_COUNT
 };

#define HL_COLOR(x, a, b)      x,
...
```

I've added HIGHLIGHT_DIALECT before HIGHLIGHT_COUNT.
I  guess you can add HIGHLIGHT_PYTHON...

In Highlight.cpp

I added

```
  static const char *java[] = {
   "abstract", "boolean", "break", "byte", "case",
   "catch", "char", "class", "const", "continue",
   "default", "do", "double", "else", "extends",
   "false", "final", "finally", "float", "for",
   "goto", "if", "implements", "import", "instanceof",
   "int", "interface", "long", "native", "new",
   "null", "package", "private", "protected", "public",
   "return", "short", "static", "super", "switch",
   "synchronized", "this", "throw", "throws", "transient",
   "true", "try", "void", "volatile", "while",
   NULL
  };
//aris added
  static const char *dialect[] = {
   "boolean", "break", "byte",
   "catch", "char", "chunk", "class", "continue",
   "default", "do", "double", "else", "elif", "elseif",
   "false", "final", "finally", "float", "for", "func",
   "goto", "if", "import",
   "int", "long",
   "nil", "string",
   "return", "this", "throw", "throws",
   "true", "try", "void", "while",
   NULL
  };
  static const char *tfile[] = {
   "T_",
   NULL,
  };
  static const char *javan[] = {
   NULL
```

```
  };
  //aris added
  static const char *dialectn[] = {
   NULL
  };
  //keywords- aris added dialect
  static const char **kw[HIGHLIGHT_COUNT] = {
   cpp, usc, java, tfile, usc, dialect  };
  //aris added dialectn
  static const char **nm[HIGHLIGHT_COUNT] = {
   upp, usclib, javan, javan, usclib, dialectn
  };
  for(int i = 0; i < HIGHLIGHT_COUNT; i++) {
   const char **q = kw[i];
   while(*q)
    keyword[i].Add(*q++);
   q = nm[i];
   while(*q)
    name[i].Add(*q++);
  }
```

And for the experiments I've started a separate app...
you can try...

```
#include <ACodeEditor/CodeEditor.h>

bool AboveAscii127(const char *fn){
 FileStream fs;
 fs.Open(fn, FileStream::READ);
 String str = fs.GetLine();
 fs.Close();
 for (int i = str.GetLength(); i>=0; i--){
  if(str[i]>127)
     return true;
 }
 return false;
}

bool AllowedExt(const String& ext){
 if ((ext==".txt") ||
     (ext==".html")||
     (ext==".htm") ||
     (ext==".d") ||
     (ext==".uvs")||
     (ext==".h")||
     (ext==".cpp")
   ) return true;
```

```
 else return false;
}
class DirView : public ParentCtrl {
 Splitter   horz;
 TreeCtrl   tree1;
 ArrayCtrl table;
 String contents;

 Label      label;
 StatusBar  info;
 Array<EditString> edit;
 EditField arr_edit;
 CodeEditor wnd_edit;

 EditField path_fld;

public:
 typedef DirView CLASSNAME;
 void OpenFile();
 void OpenDir(int id);
 void CloseDir(int id);
 void ShowPath();
 void ArrayEdit();
 DirView();
 ~DirView() {;}
};


void DirView::ArrayEdit() {
    table.AcceptEnter();
  table.ColumnAt(0).Edit(arr_edit);
 }

void DirView::CloseDir(int id) {
  tree1.RemoveChildren(id);
 }


void DirView::OpenDir(int id) {
  String path = tree1.Get(id);
  Array<FileSystemInfo::FileInfo> root=StdFileSystemInfo().Find(Null);
 if (id==0)
  for(int id = 0; id < root.GetCount(); id++)
   tree1.Add(0,  GetDriveImage(root[id].root_style),
     root[id].filename, root[id].filename, true);
 else
  for(FindFile ff(AppendFileName(path, "*.*")); ff; ff.Next()) {
   String n = ff.GetName();
```

```
   if(n != "." && n != "..")
    tree1.Add(id, ff.IsFolder() ? CtrlImg::Dir() : CtrlImg::File(),
              AppendFileName(path, n), n, ff.IsFolder());
  }
 }

void DirView::ShowPath() {
  info = ~tree1;
 }

void DirView::OpenFile(){
  String path = tree1.Get();
  FindFile ff(path);
  String ext=GetFileExt(path);
  if ( ff.IsFile() && ((AllowedExt(ext)) ||((ext.GetLength()==0) && !AboveAscii127(path))) ){
   table.Add( "1", ff.GetName(), ext );

  if (ext==".cpp" || ext==".h"){

   wnd_edit.Highlight(CodeEditor::HIGHLIGHT_CPP);
  }
  else if (ext==".d") wnd_edit.Highlight(CodeEditor::HIGHLIGHT_DIALECT);
  else wnd_edit.Highlight(CodeEditor::HIGHLIGHT_NONE);
  wnd_edit.SetData(AsString(LoadFile(path)));

  }
}

DirView::DirView() {
  path_fld.SetRect(0,0,350,50);
  Add( path_fld.TopPos(10) );

  horz.Add(tree1);
  horz.Add(table);
  horz.Add(wnd_edit);

  Add(horz.Horz().VSizePos(35,35));

  horz.SetPos(2000,0);
  horz.SetPos(3000,1);

  Font f=Courier(12);

  wnd_edit.LoadHlStyles(LoadFile(ConfigFile("ide.colors")));
  //don't forget to put in exe dir or...
  wnd_edit.SetFont(f);
  wnd_edit.HiliteScope(2);
  wnd_edit.ShowTabs(true);
```

```
  wnd_edit.HiliteBracket(1);
  wnd_edit.LineNumbers(true);
  table.NoHorzGrid();
 #ifdef PLATFORM_WIN32
  String dir =String(GetExeFilePath()[0], 1) + ":\\";
 #else
  String dir = "/usr";
 #endif
  tree1.SetRoot(CtrlImg::Dir(), "My Computer");

  tree1.MultiSelect();

  tree1.WhenOpen = THISBACK(OpenDir);
  tree1.WhenClose = THISBACK(CloseDir);

  tree1.WhenAction = THISBACK(OpenFile);

  tree1.WhenCursor = THISBACK(ShowPath);

  tree1.AddFrame(info);
  tree1.Open(0,true);

  table.AddColumnAt(0,"id",10);
  table.AddColumnAt(1,"path",50);
  table.AddColumnAt(2,"ext",20);

  table.SetEditable(true);
  table.WhenLeftDouble = THISBACK(ArrayEdit);
}


GUI_APP_MAIN
{ TopWindow w;
    DirView dirview;
    dirview.SizePos();
    w.Add(dirview);
    w.SetRect(30,30,950,700);

 w.Zoomable().Sizeable().Run();
}
```

---

Subject: Re: TheIde and Python
Posted by mirek on Wed, 29 Mar 2006 19:20:48 GMT
View Forum Message <> Reply to Message

This is correct, as long as you are interested in highlighting keywords only. Highlighting other lexical elements is a but more complicated....

BTW, maybe, to keep things right, you should overload codeeditor and make another HightlightLine override?

As for more wide Python support, it depends, but one can imagine that - it would at max involved adding syntax highlighting a maybe some sort Python "fake" builder (because in fact you do not need to build anything for python) and adjusting "Run".

However, meanwhile it is true that TheIDE is developed specifically to deal with C++/U++... I am not sure how system of package would play with python...

Mirek

---

Subject: Re: TheIde and Python
Posted by fudadmin on Wed, 29 Mar 2006 22:11:15 GMT
View Forum Message <> Reply to Message

luzr wrote on Wed, 29 March 2006 20:20This is correct, as long as you are interested in highlighting keywords only. Highlighting other lexical elements is a but more complicated....

BTW, maybe, to keep things right, you should overload codeeditor and make another HightlightLine override?

As for more wide Python support, it depends, but one can imagine that - it would at max involved adding syntax highlighting a maybe some sort Python "fake" builder (because in fact you do not need to build anything for python) and adjusting "Run".

However, meanwhile it is true that TheIDE is developed specifically to deal with C++/U++... I am not sure how system of package would play with python...

Mirek

Yes, that is one of the possibilities. But I've been thinking about a customizible syntaxes file (text or xml) to make life easier for more languages... Any hints?

---

Subject: Re: TheIde and Python
Posted by pivica on Thu, 30 Mar 2006 07:59:18 GMT
View Forum Message <> Reply to Message

luzr wrote on Wed, 29 March 2006 14:20This is correct, as long as you are interested in highlighting keywords only. Highlighting other lexical elements is a but more complicated....

There are plenty of other editors that are doing good python highlighting, but code completion is a problem. Is it posible to add some kind of a python parser and code completion to TheIDE?

---

Subject: Re: TheIde and Python
Posted by mirek on Thu, 30 Mar 2006 20:00:14 GMT
View Forum Message <> Reply to Message

pivica wrote on Thu, 30 March 2006 02:59luzr wrote on Wed, 29 March 2006 14:20This is correct, as long as you are interested in highlighting keywords only. Highlighting other lexical elements is a but more complicated....

There are plenty of other editors that are doing good python highlighting, but code completion is a problem. Is it posible to add some kind of a python parser and code completion to TheIDE?

Wow, that is pretty complicated task I guess....

Maybe it would be possible to somehow bind Python syntax to C++ items (to employ curect "code-database format", but I know too little about Python to be sure....

Mirek

---