Subject: Assist++ future
Posted by Defre on Sat, 04 Aug 2007 13:06:07 GMT
View Forum Message <> Reply to Message

Hi everybody, it's my first post.
First of all, thank you for U++, beautiful piece of software.

I have some interrogation about the future of Assist++:
- when will code completion be improved? (i have encountered some problem with pointer manipulations). More precisely, will preprocessing be included in the analysis?
- is refactoring a part of your plan?

I'm thinking of writing some C++ analysis tools, but nothing serious, just to learn. Concerning preprocessing, since a preprocessor output is generally composed of tens of thousand of lines, performance is an obvious problem. Some kind of "PCH" would be a solution...
Actually i'm just beginning the scanner, there is a long way to go.

Very truly,
Fred

Subject: Re: Assist++ future
Posted by mirek on Sat, 04 Aug 2007 13:16:11 GMT
View Forum Message <> Reply to Message

Code completition: Currently in research phase of #include parsing. I guess you nailed the problem - speed is the trouble. But I believe that it can be solved.

Refactoring - not planned yet. Ask when preprocessing is finished.

BTW, current parser (in CodeBase) is quite a nice thing to play with too... See examples/CodeMetric.

Subject: Re: Assist++ future
Posted by mirek on Sat, 04 Aug 2007 18:17:10 GMT
View Forum Message <> Reply to Message

BTW, thinking about it, I guess it is worth considering to create some semi-automatic testing of Assist++ and to start collecting code that it does not parse properly....

Subject: Re: Assist++ future
Posted by mr_ped on Sat, 04 Aug 2007 20:47:34 GMT
View Forum Message <> Reply to Message

Quote:and to start collecting code that it does not parse properly....

I think I will provide plenty of those over time..

But it's not only about preprocessing, I have some more basic problems, like I don't have local variables parsed.

```
void TestLocalVariablesParsing( int variableone, int anothervariable, int thirdvariable )
{
//here write "vari" and hit Ctrl+Space .. nothing is suggested.
//here write "ano" and hit Ctrl+Space .. nothing is suggested.
String tempstring;
//here write "temps" and hit Ctrl+Space .. nothing is suggested.
}
```

It annoys me a lot personally, I'm used to IDEs which do parse local variables. (MSVS, VSlick).

---

Subject: Re: Assist++ future
Posted by mirek on Sun, 05 Aug 2007 08:05:38 GMT
View Forum Message <> Reply to Message

mr_ped wrote on Sat, 04 August 2007 16:47Quote:and to start collecting code that it does not parse properly....

I think I will provide plenty of those over time..

But it's not only about preprocessing, I have some more basic problems, like I don't have local variables parsed.

```
void TestLocalVariablesParsing( int variableone, int anothervariable, int thirdvariable )
{
//here write "vari" and hit Ctrl+Space .. nothing is suggested.
//here write "ano" and hit Ctrl+Space .. nothing is suggested.
String tempstring;
//here write "temps" and hit Ctrl+Space .. nothing is suggested.
}
```

It annoys me a lot personally, I'm used to IDEs which do parse local variables. (MSVS, VSlick).

I will check soon... but in fact, in this case, it perhaps is not parser bug, just problem in editor. You can easily check: type "tempstring."

Mirek

---

Subject: Re: Assist++ future
Posted by mr_ped on Sun, 05 Aug 2007 10:57:06 GMT
View Forum Message <> Reply to Message

Quote:You can easily check: type "tempstring."

Yes, it does offer the members of String class, so you are right, the parser is ok. I didn't think about the issue into the depth to distinguish it from editor problem.

(it still annoys me anyway )

At least you can see you should think also in terms of what editor should provide out from parser, and if your test procedure should test also editor's ability to work with parser, or parser itself only.

Subject: Standard array not parsed
Posted by andrei-catalin on Sun, 05 Aug 2007 11:29:56 GMT
View Forum Message <> Reply to Message

Standard array is not parsed by Assist

```
//$-
int step[11]={0,1,2,5,10,20,50,100,200,500,1000}; //Whithout parser directives assist don't work
//$+

struct Foo
{
 void ShowStep(int n);
};

//test Alt+C/Ctrl+V
void Foo::ShowStep(int n)
{
 Cout()<<"Step["<<n<<"] = "<<step[n]<<"\n";
}


CONSOLE_APP_MAIN
{
 Cout()<<"Start"<<"\n";
 Foo foo;
 //test autocompletion
 foo.ShowStep(5);
}
```

Andrei

Subject: Re: Assist++ future
Posted by mr_ped on Sun, 14 Oct 2007 23:29:01 GMT
View Forum Message <> Reply to Message

I have problem with Assist++ not showing member of S_A struct in this case:

```
class C_A {
public:
 struct S_A { int x; S_A() : x(1) {} };
};
class C_B : public C_A {
public:
 struct S_B : public S_A { S_B() { x = 3; } };
};
C_B::S_B b;

CONSOLE_APP_MAIN
{
 UPP::Cout().Put( UPP::AsString(b.x) + "\r\n" );
//put "b." here, the "x" of S_A will be not shown
}
```

(P.S. the simple form without C_A and C_B encapsulation works OK, only this more complex case does not work)

Subject: Re: Assist++ future
Posted by zsolt on Mon, 15 Oct 2007 19:49:54 GMT
View Forum Message <> Reply to Message

mr_ped wrote on Sat, 04 August 2007 22:47Quote:and to start collecting code that it does not parse properly....

I think I will provide plenty of those over time..

But it's not only about preprocessing, I have some more basic problems, like I don't have local variables parsed.

```
void TestLocalVariablesParsing( int variableone, int anothervariable, int thirdvariable )
{
//here write "vari" and hit Ctrl+Space .. nothing is suggested.
//here write "ano" and hit Ctrl+Space .. nothing is suggested.
String tempstring;
//here write "temps" and hit Ctrl+Space .. nothing is suggested.
}
```

It annoys me a lot personally, I'm used to IDEs which do parse local variables. (MSVS, VSlick).

Use Ctrl+, for local variables.

---

Subject: Re: Assist++ future
Posted by unodgs on Mon, 15 Oct 2007 20:56:35 GMT
View Forum Message <> Reply to Message

zsolt wrote on Mon, 15 October 2007 15:49
Use Ctrl+, for local variables.
I'm using it too, but without any doubts ctrl+space should show them as well.

---

Subject: Re: Assist++ future
Posted by mr_ped on Thu, 24 Jul 2008 08:24:39 GMT
View Forum Message <> Reply to Message

I got some idea today and I wonder if it makes sense.

What about search function cooperating with Assist++ parsed data, like you would enter "\assignment x" (eventually with whole word flag), and from source

```
void foo( int & x )
{
  int b = x * 10;
  x = b - x;
}
```

it would find only the line "x = " and ignore those x in function parameters and calculation.

Mirek, after parser refactoring, would you store some data similar to this, so you can tell whether some piece of source is assignment / usage / class definition / inheritance definition / etc..?
And if yes, does anyone think such extended search would be practical and good to use?
(*I* can imagine to use it rarely, so I would need easy way to see what options I can use (probably droplist after "\" character, because I would not remember all the options)

---

Subject: Re: Assist++ future
Posted by mrjt on Thu, 24 Jul 2008 10:19:03 GMT
View Forum Message <> Reply to Message

Wouldn't this be more easily implemented as an advanced condition text search? Something like regular expressions, only simpler to use?

I think implementing it in the parser would be very complicated (and it's complicated enough

already ), a would presumably reduce scanning performace.