
Subject: Assist++ typedef struct analysis problem
Posted by [Xemuth](#) on Sat, 26 Mar 2022 23:49:19 GMT
[View Forum Message](#) <> [Reply to Message](#)

Hello U++,

I noticed that Assist++ fail to parse some struct. Lets take a look at thoses 4 structs def:

test.h:

```
#ifndef _test_assist_test_h_
#define _test_assist_test_h_

typedef struct StructToResolve {
    uint32_t    engineVersion;
    uint32_t    apiVersion;
} StructToResolve;

typedef struct StructToResolve2 {
    uint32_t    engineVersion;
    uint32_t    apiVersion;
};

//Illegal but we test Assist++ here !
struct StructToResolve3 {
    uint32_t    engineVersion;
    uint32_t    apiVersion;
} StructToResolve3;

struct StructToResolve4 {
    uint32_t    engineVersion;
    uint32_t    apiVersion;
} test;

#endif
```

All four struct are found by assist++. However, auto completion don't work on the first. Moreover having a struct having a name at begining and variable declaration with the same name are shown has two different struct by Assist++.

test.cpp

```
#include "test.h"
int main(int argc, const char *argv[])
{
    StructToResolve str;
```

```
str. // Assist++ don't find anything. It don't even open
```

```
StructToResolve2 str2;  
str2.engineVersion; // work fine
```

```
StructToResolve3 str3;  
str3.apiVersion; // Work fine BUT doing a Ctrl+ Space during writting of this type result in 2  
distinct StructToResolve3 type. See the screenshot
```

```
StructToResolve4 str4;  
str4.engineVersion; // Work fine BUT ...
```

```
return 0;  
}
```

this bug is problematic when working with lib definition a huge amount of the structure which have the same declaration as the first one.

Subject: Re: Assist++ typedef struct analysis problem
Posted by [Lance](#) on Sun, 27 Mar 2022 01:38:20 GMT
[View Forum Message](#) <> [Reply to Message](#)

I can see it happens when you have to use some imported C code (first typedef). So it is a bug worth to be fixed.

second typedef is obviously illegal code.

I also noticed that following are valid C++ code

```
struct C{  
};  
C C;
```

I didn't know that before: I thought by declaring C as a class/struct name in C++, it kind of makes it a para-keyword which forbids it be further used as identifier name.

And the following are also valid

```
struct C{};  
struct D{};  
C D;
```

Subject: Re: Assist++ typedef struct analysis problem
Posted by [Lance](#) on Sun, 27 Mar 2022 01:54:09 GMT
[View Forum Message](#) <> [Reply to Message](#)

BTW, xemuth's example code

```
StructToResolve3 str3;
```

is not legal C/C++ code: it won't compile.

It should be

```
struct StructToResolve3 str3;
```

And the following code compiles

```
struct C{  
    void hi(){}  
};  
  
struct D{  
    void hi(){};  
};  
  
int main()  
{  
    C D;  
    struct D d; // the keyword struct cannot be done without  
    D.hi();  
    d.hi();  
}
```

or this also compiles

```
struct C{  
    void hi(){}  
};  
  
struct D{  
    void hi(){};  
};  
  
int main()  
{  
    D d;  
    C D;
```

```
D.hi();
d.hi();
}
```

Conclusion: in a context where a class/struct name is used as identifier (and hence hidden by it), to refer to the class/struct, a leading class or struct keyword needs to be prepended.

Subject: Re: Assist++ typedef struct analysis problem
Posted by [Lance](#) on Sun, 27 Mar 2022 02:09:37 GMT
[View Forum Message](#) <> [Reply to Message](#)

BTW:

```
struct Empty{};

struct Z {
    char c;
    [[no_unique_address]] Empty e1, e2;
};
```

Above c++20 code will confuse assist++ too. I wasn't able to figure a way to fix it.

Subject: Re: Assist++ typedef struct analysis problem
Posted by [Xemuth](#) on Sun, 27 Mar 2022 02:20:01 GMT
[View Forum Message](#) <> [Reply to Message](#)

Hello Lance, yes some of my struct here won't compile.

About this bug. I digged Parser code in order to find a simple way to ignore C struct alias but didn't find a correct way to fix it.
However I have a temporary fix :

.\upp\uppsrc\CppBase\Parser.cpp : 1675

```
if(d.name.GetCount() && d.name != d.type) { // We want to prevent struct alias to be
missinterpreted
```

It's wacky

Maybe I will dig it more tomorrow to understand the root cause of it and find a better way to fix it.

Subject: Re: Assist++ typedef struct analysis problem
Posted by [Lance](#) on Wed, 30 Mar 2022 16:10:51 GMT
[View Forum Message](#) <> [Reply to Message](#)

That part of code is not very pleasant to touch --from my experience to tell parser to ignore `[[identifier]]`.
