

Re: Unable to match function definition to an existing declaration

Source: <http://www.tech-archive.net/Archive/VC/microsoft.public.vc.language/2004-11/0679.html>

From: Victor Bazarov (v.Abazarov_at_comAcast.net)

Date: 11/17/04

Date: Wed, 17 Nov 2004 15:07:40 -0500

Murrgon wrote:

> Consider the following for VC7.1

>

> //

>

=====

> #include <vector>

> using namespace std;

>

> template <class TYPE>

> class TArray : public vector<TYPE>

> {

> public:

Add here:

```
    typedef typename vector<TYPE>::size_type size_type;
```

```
> inline size_type GetAllocSize() const;
```

```
> };
```

```
>
```

```
> template<class TYPE>
```

```
> inline typename vector<TYPE>::size_type TArray<TYPE>::GetAllocSize() const
```

Change to

```
inline
```

```
    typename TArray<TYPE>::size_type
```

```
        TArray<TYPE>::GetAllocSize() const
```

```
> {
```

```
> return(vector<TYPE>::capacity());
```

```
> }
```

```
>
```

```
>
```

```
> //
```

```
>
=====
> // Main
>
=====
> //
>
=====
> int main()
> {
> TArray<int> aIntegers;
> return 0;
> }
>
> //
>
=====
>
> This gives me a C2244 error: "unable to match function definition to an
> existing declaration". The help on error C2244 says "An unusual use of
> the unary + operator was used in front of a function call that did not have
> parenthesis."
>
> What?
>
> I don't see any + operator in use here. Can someone tell me what is
> actually
> wrong here?
```

It can't find 'size_type' when processing the declaration. There must be some weirdness in the way dependent types are resolved in VC++, but I am too lazy now to track them down. This:

```
-----
#include <vector>
using namespace std;

template <class TYPE>
class TArray : public vector<TYPE>
{
public:
    typename vector<TYPE>::size_type GetAllocSize() const;
};

template<class TYPE>
inline typename vector<TYPE>::size_type TArray<TYPE>::GetAllocSize() const
{
    return vector<TYPE>::capacity();
}

int main()
{
```

microsoft.public.vc.language: Re: Unable to match function definition to an existing declaration

```
TArray<int> aIntegers;  
}
```

doesn't compile (although it should). It can't find that "vector<TYPE>"
is the actual base class of 'TArray<TYPE>'.

I checked with VC++ v8, and it can't compile it either. I am not going
to submit a bug report for that, too lazy. Besides, it probably has
already been reported.

V