Discuss C++ Template Downcast

Table of Contents

This is a discuss in C board in bbs.sjtu.edu.cn, about type down-cast in C++ template.

Original Discuss

http://bbs.sjtu.edu.cn/bbstcon,board,C,reid,13300789

http://bbs.sjtu.edu.cn/bbstcon,board,cpltd,1330078933,file,M.1330078933.A.html

The problem

Today I read a book about we can do cast-down in template, so I write this to test:

```
1  τεμπλατε <βοολ _Τεστ, χλασσ _Τψπε = ϖοιδ>
2  στρυχτ εναβλε_ιφ { };
3
4  τεμπλατε<χλασσ _Τψπε>
5  στρυχτ εναβλε_ιφ<τρυε, _Τψπε> {
6      τψπεδεφ _Τψπε τψπε;
7  };
8
9  χλασσ Α { };
10 χλασσ Β : Α { };
11
12 τεμπλατε <τψπεναμε Τ>
13 στρυχτ τραιτσ { στατιχ ιντ χονστ ϖαλυε = φαλσε; };
14
15 τεμπλατε <>
16 στρυχτ τραιτσ<Α> { στατιχ ιντ χονστ ϖαλυε = τρυε; };
17
18 τεμπλατε <τψπεναμε Τ>
19 ϖοιδ φ(Τ, τψπεναμε εναβλε_ιφ<τραιτσ<Τ>::ϖαλυε>::τψπε* = 0
20
21 τεμπλατε <>
22 ϖοιδ φ<Α>(Α, εναβλε_ιφ<τραιτσ<Α>::ϖαλυε>::τψπε*) { }
23
24
25
26 τεμπλατε <τψπεναμε Τ>
```

```
27  χλασσ BB {};
28
29  τεμπλατε <τψπεναμε T>
30  χλασσ ΔΔ : πυβλιχ BB<T> {};
31
32  τεμπλατε <τψπεναμε T> ϖοιδ φφ(BB<T>) {};
33
34  ιντ μαιν(ιντ αργχ, χηαρ * αργϖ[])
35  {
36      A α; B β;
37      ΔΔ<λονγ> δδ;
38      //φ(β);
39      φφ(δδ);
40  }
```

It is strange when φ it don't allow my specified φ<A> ¯ .

But in φφ it allowed φφ<BB<λονγ>> ¯ .

Tested under VC10 and GCC3.4

My answer to the problem

Let's think ourself as compiler to see what happened there.

Define mark # : A#B is the instantiated result when we put B into the parameter T of A<T> .

First we discuss ff

```
1  ΔΔ<λονγ> δδ;
```

After this sentense, the compiler saw the instantiation of `ΔΔ<λονγ>` , so it instantiate `ΔΔ#λονγ` , and also `BB#λονγ` .

```
1  φφ(δδ);
```

This sentense required the compiler to calculate set of overloading functions.

Step 1 we need to infer `T` of `φφ<T>` from argument `ΔΔ#λονγ –> BB<T>` . Based on the inference rule:

```
Αργυμεντ ωιτη τψπε :χοδε: χλασσ_τεμπλατε_ναμε<T> χαν
```

So compiler inferred `T` as `λονγ` . Here if it is not `BB` but `XX` which is complete un-related, we can also infer, as long as `XX` is a template like `XX<T>` .

Step 2 Template Specialization Resolution. There is only one template here so we matched `φφ<T>` .

Step 3 Template Instantiation

After inferred `λονγ –> T` , compiler instantiated `φφ#λονγ` .

Set of available overloading functions : `{φφ#λονγ}`

Then overloading resolution found the only match

φφ#λονγ¯ , checked its real parameter ΔΔ#λονγ can be down-cast to formal parameter BB#λονγ .

Then we discuss f

```
1  φ(β);
```

Calculate set of overloading functions.

Step 1 infer all template parameters for template φ . According to inference rule:

> Παραμετερ ωιτη τψπε T χαν βε υσεδ το ινφερ T

So B –> T is inferred.

Step 2 Template Specialization Resolution.

Here B is not A so we can not apply specialization of φ<A> , remaining φ<T> as the only alternative.

Step 3 Template Instantiation.

When we put B into φ<T> to instantiate as φ#B , we need to instantiate τραιτσ#B¯ .

There is no specialization for B so we use template τραιτσ<T> , τραιτσ#B::ϖαλυε=φαλσε , so εναβλε_ιφ#φαλσε didn't contains a τψπε , an error occurred.

The only template is mismatch, available overloading functions is empty set. So we got an error.