
Subject: Nested template question

Posted by [koldo](#) on Sun, 09 Jun 2019 14:16:28 GMT

[View Forum Message](#) <> [Reply to Message](#)

Hi all

I wanted to overload a function to get a zero (it is a sample):

```
void test() {  
    double val    = GetAZero<double>();  
    std::complex<float> valc1 = GetAZero<std::complex<float>>();  
    std::complex<double> valc1 = GetAZero<std::complex<double>>();  
}
```

This code works, but it is not nice:

```
template <class T> T GetAZero() {return 0;}  
  
template <> std::complex<float> GetAZero() {return std::complex<float> (0, 0);}  
template <> std::complex<double> GetAZero() {return std::complex<double>(0, 0);}
```

This code is more compact, but it does not work:

```
template <class T> T GetAZero() {return 0;}
```

```
template <class std::complex<T>> std::complex<T> GetAZero() {return std::complex<T> (0, 0);}  
Is there an adequate way to do this?
```
