Subject: Bug in Core06.cpp or in U++ Core value types tutorial
Posted by Werner on Sat, 01 Dec 2007 11:54:25 GMT
View Forum Message <> Reply to Message

Core06.exe yields:

Quote:...
a.Is<int>() = false
a.Is<double>() = false
b.Is<double>() = false
c.Is<int>() = false
c.Is<Date>() = true
d.Is<String>() = true
...

U++ Core value types tutorial, 6. Value, says:

Quote:
a.Is<int>() = true
a.Is<double>() = false
b.Is<double>() = true
c.Is<int>() = false
c.Is<Date>() = true
d.Is<String>() = true

Who is right?

Werner

Subject: Re: Bug in Core06.cpp or in U++ Core value types tutorial
Posted by mirek on Wed, 05 Dec 2007 11:07:31 GMT
View Forum Message <> Reply to Message

This is a complete output from quick test:

* c:\out\MSC71cdb.Debug_full\Core06.exe 05.12.2007 12:04:53, user: Luzr

x = 1
y = 2.34
z = 12/05/2007
s = hello
i = 1

j = 2
k = 12/05/2007
t = hello
a.Is<int>() = true
a.Is<double>() = false
b.Is<double>() = true
c.Is<int>() = false
c.Is<Date>() = true
d.Is<String>() = true
IsNumber(a) = true
IsNumber(b) = true
IsDateTime(c) = true

Does seem all right to me. However, tested with MSC71 in WinXP.

It is however possible that the problem is compiler / platform related. Can you perhaps provide more information please? Also about U++ version?

Mirek

Subject: Re: Bug in Core06.cpp or in U++ Core value types tutorial
Posted by Werner on Wed, 05 Dec 2007 13:29:37 GMT
View Forum Message <> Reply to Message

luzr wrote on Wed, 05 December 2007 12:07This is a complete output from quick test:

* c:\out\MSC71cdb.Debug_full\Core06.exe 05.12.2007 12:04:53, user: Luzr

x = 1
y = 2.34
z = 12/05/2007
s = hello
i = 1
j = 2
k = 12/05/2007
t = hello
a.Is<int>() = true
a.Is<double>() = false
b.Is<double>() = true
c.Is<int>() = false
c.Is<Date>() = true
d.Is<String>() = true
IsNumber(a) = true
IsNumber(b) = true
IsDateTime(c) = true

Does seem all right to me. However, tested with MSC71 in WinXP.

It is however possible that the problem is compiler / platform related. Can you perhaps provide more information please? Also about U++ version?

Mirek

Sorry for the wrong forum section.

I'm running upp-mingw-711-dev2.

I got my results building the original core06.cpp example using gcc 4.2.1 under Windows XP.

Werner

Subject: Re: Bug in Core06.cpp or in U++ Core value types tutorial
Posted by mirek on Wed, 05 Dec 2007 13:59:35 GMT
View Forum Message <> Reply to Message

[quote title=Werner wrote on Wed, 05 December 2007 08:29]luzr wrote on Wed, 05 December 2007 12:07This is a complete output from quick test:

I'm running upp-mingw-711-dev2.

I got my results building the original core06.cpp example using gcc 4.2.1 under Windows XP.

Werner

Thank you. Bug reproduced and fixed (templates are implemented according to standard in GCC... while MSC interpreted certain code as I wanted.... .

To be sure, please replace these in Value.h:

```
template <class T>
inline dword ValueTypeNo(const T&)       { return StaticTypeNo<T>() + 0x8000000;; }

template<> inline dword ValueTypeNo(const int&)     { return INT_V; }
template<> inline dword ValueTypeNo(const int64&)   { return INT64_V; }
template<> inline dword ValueTypeNo(const double&)  { return DOUBLE_V; }
template<> inline dword ValueTypeNo(const bool&)    { return BOOL_V; }
template<> inline dword ValueTypeNo(const String&)  { return STRING_V; }
template<> inline dword ValueTypeNo(const WString&) { return WSTRING_V; }
```

```
template<> inline dword ValueTypeNo(const Date&)    { return DATE_V; }
template<> inline dword ValueTypeNo(const Time&)    { return TIME_V; }

template <class T, dword type, class B = EmptyClass>
class AssignValueTypeNo : public B {
public:
 friend dword ValueTypeNo(const T&)                { return type; }

 void operator=(const AssignValueTypeNo&) {} // MSC 6.0 empty base class bug fix
};

template <class T>
bool IsType(const Value& x, T* = 0)          { return ValueTypeNo(*(T *)NULL) == x.GetType(); }

template <class T>
inline bool Value::Is() const
{
 return IsType<T>(*this);
}
```

..and test..

Mirek

Subject: Re: Bug in Core06.cpp or in U++ Core value types tutorial
Posted by Werner on Wed, 05 Dec 2007 17:29:21 GMT
View Forum Message <> Reply to Message

luzr wrote on Wed, 05 December 2007 14:59
Thank you. Bug reproduced and fixed (templates are implemented according to standard in GCC... while MSC interpreted certain code as I wanted.... .

To be sure, please replace these in Value.h:

```
template <class T>
inline dword ValueTypeNo(const T&)       { return StaticTypeNo<T>() + 0x8000000;; }

template<> inline dword ValueTypeNo(const int&)     { return INT_V; }
template<> inline dword ValueTypeNo(const int64&)   { return INT64_V; }
template<> inline dword ValueTypeNo(const double&)  { return DOUBLE_V; }
template<> inline dword ValueTypeNo(const bool&)    { return BOOL_V; }
template<> inline dword ValueTypeNo(const String&)  { return STRING_V; }
template<> inline dword ValueTypeNo(const WString&) { return WSTRING_V; }
template<> inline dword ValueTypeNo(const Date&)    { return DATE_V; }
template<> inline dword ValueTypeNo(const Time&)    { return TIME_V; }
```

```
template <class T, dword type, class B = EmptyClass>
class AssignValueTypeNo : public B {
public:
 friend dword ValueTypeNo(const T&)                { return type; }

 void operator=(const AssignValueTypeNo&) {} // MSC 6.0 empty base class bug fix
};

template <class T>
bool IsType(const Value& x, T* = 0)          { return ValueTypeNo(*(T *)NULL) == x.GetType(); }

template <class T>
inline bool Value::Is() const
{
 return IsType<T>(*this);
}
```

..and test..

Mirek

Thanks. Now works as expected.

Just to do a quick test, I built TheIDE using gcc 4.2.1 with "MinGW Debug" and "MinGW Optimal". No problems, too.

Werner