Added compilation workarounds for MSVC 2015 CTP6.

Browse files

master boost-1.59.0 boost-1.58.0

Lastique authored on 1 Mar 1 parent 7ebfd3b commit 4751a165366dd9d1cf55252c290eaacf39a59552

Showing 3 changed files with **13 additions** and **4 deletions**. Unified Split

4 include/boost/log/detail/code_conversion.hpp View

```
@@ -35,6 +35,9 @@ namespace aux {
 BOOST_LOG_API void code_convert(const wchar_t* str1, std::size_t len, std::string& str2, std::locale const& loc = std::loc
 //! The function converts one string to the character type of another
 BOOST_LOG_API void code_convert(const char* str1, std::size_t len, std::wstring& str2, std::locale const& loc = std::local
+
+// Note: MSVC 2015 (aka VC14) implement char16_t and char32_t types but not codecvt locale facets
+#if !defined(BOOST_MSVC)
 #if !defined(BOOST_NO_CXX11_CHAR16_T)
 //! The function converts one string to the character type of another
 BOOST_LOG_API void code_convert(const char16_t* str1, std::size_t len, std::string& str2, std::locale const& loc = std::lc
@@ -47,6 +50,7 @@ BOOST_LOG_API void code_convert(const char32_t* str1, std::size_t len, std::stri
 //! The function converts one string to the character type of another
 BOOST_LOG_API void code_convert(const char* str1, std::size_t len, std::u32string& str2, std::locale const& loc = std::loc
 #endif
+#endif // !defined(BOOST_MSVC)

 //! The function converts one string to the character type of another
 template< typename CharT, typename SourceTraitsT, typename SourceAllocatorT, typename TargetTraitsT, typename TargetAlloca
```

5 src/code_conversion.cpp View

```
@@ -125,6 +125,9 @@ BOOST_LOG_API void code_convert(const char* str1, std::size_t len, std::wstring&
     code_convert(str1, str1 + len, str2, std::use_facet< std::codecvt< wchar_t, char, std::mbstate_t > >(loc));
 }

+// Note: MSVC 2015 (aka VC14) implement char16_t and char32_t types but not codecvt locale facets
+#if !defined(BOOST_MSVC)
+
 #if !defined(BOOST_NO_CXX11_CHAR16_T)

 //! The function converts one string to the character type of another
@@ -157,6 +160,8 @@ BOOST_LOG_API void code_convert(const char* str1, std::size_t len, std::u32strin

 #endif

+#endif // !defined(BOOST_MSVC)
+
 } // namespace aux

 BOOST_LOG_CLOSE_NAMESPACE // namespace log
```

8 src/unhandled_exception_count.cpp View

```
@@ -49,8 +49,8 @@ extern "C" void* __cxa_get_globals();
 #elif defined(_MSC_VER)
 #if _MSC_VER >= 1900
 // Visual Studio 14 has redesigned CRT
-#define BOOST_LOG_HAS_VCRT_GETPTD
-extern "C" void* __vcrt_getptd();
+#define BOOST_LOG_HAS_PROCESSING_THROW
+extern "C" int* __processing_throw();
 #elif _MSC_VER >= 1400
 #define BOOST_LOG_HAS_GETPTD
 extern "C" void* _getptd();
@@ -68,9 +68,9 @@ BOOST_LOG_API unsigned int unhandled_exception_count() BOOST_NOEXCEPT
```

```
68  68  #elif defined(BOOST_LOG_HAS_GETPTD)
69  69      // MSVC specific. Tested on {MSVC2005SP1,MSVC2008SP1,MSVC2010SP1,MSVC2012}x{x32,x64}.
70  70      return *(reinterpret_cast< const unsigned int* >(static_cast< const char* >(_getptd()) + (sizeof(void*) == 8 ? 0x100 :
71      -#elif defined(BOOST_LOG_HAS_VCRT_GETPTD)
    71  +#elif defined(BOOST_LOG_HAS_PROCESSING_THROW)
72  72      // MSVC specific. Tested on {MSVC 14 CTP}x{x32,x64}.
73      -    return *(reinterpret_cast< const unsigned int* >(static_cast< const char* >(__vcrt_getptd()) + (sizeof(void*) == 8 ? 0
    73  +    return static_cast< unsigned int >(*__processing_throw());
74  74  #else
75  75      // Portable implementation. Does not allow to detect multiple nested exceptions.
76  76      return static_cast< unsigned int >(std::uncaught_exception());
```

0 comments on commit `4751a16`

Please sign in to comment.