Boost и C++11/C++14

Новости с фронта,
или обо всём понемногу.

Полухин Антон
Antony Polukhin

Ведущий разработчик Boost.LexicalCast, Boost.Any.
+ Boost.CircularBuffer, Boost.Variant, Android тестирование.

Содержание

- Вредный в C++11 совет
- Variadic templates – неозвученные плюсы
- Rvalue references в цифрах
- Android и Boost
- GIT == мерзкий тип (впечатления от модуляризации Boost)
- Что ждать в следующей версии Boost

Вредный в C++11 совет

```
const std::string foo();
```

Вредный в C++11 совет

```
std::vector<std::string> vect;
vect.push_back(foo());
```

C++03	C++11
vector<string>::push_back(const string&);	vector<string>::push_back(const string&); vector<string>::push_back(string&&);
const string foo();	const string&& foo();
vect.push_back(foo());	vect.push_back(foo()); // Ой-ой!

Variadic templates:
Объявление типов

C++03	C++11
boost::variant<T0, boost::detail::variant::void_, boost::detail::variant::void_, boost::detail::variant::void_, boost::detail::variant::void_, boost::detail::variant::void_,boost::detail::variant::void_, boost::detail::variant::void_, boost::detail::variant::void_,boost::detail::variant::void_, boost::detail::variant::void_, boost::detail::variant::void_,boost::detail::variant::void_, boost::detail::variant::void_, boost::detail::variant::void_,boost::detail::variant::void_, boost::detail::variant::void_, boost::detail::variant::void_,boost::detail::variant::void_, boost::detail::variant::void_>	boost::variant<T0, TN...>

Variadic templates:
BOOST_CURRENT_FUNCTION/__PRETTY_FUNCTION__

C++03	C++11
void foo(const boost::variant<T0, boost::detail::variant::void_,boost::detail::variant::void_,boost::detail::variant::void_,boost::detail::variant::void_,boost::detail::variant::void_,boost::detail::variant::void_,boost::detail::variant::void_,boost::detail::variant::void_,boost::detail::variant::void_,boost::detail::variant::void_,boost::detail::variant::void_,boost::detail::variant::void_,boost::detail::variant::void_,boost::detail::variant::void_,boost::detail::variant::void_,boost::detail::variant::void_,boost::detail::variant::void_,boost::detail::variant::void_,boost::detail::variant::void_> &)	void foo(const T&) [with T = boost::variant<int>]

Variadic templates:
typeid(T).name()

C++03	C++11
N5boost7variantI2T0NS_6detail7variant5void_ES4_S4_S4_S4_S4_S4_S4_S4_S4_S4_S4_S4_S4_S4_S4_S4_S4_S4_S4_EE	N5boost7variantIiIEEE

Исходники GCC:

```
bool std::type_info::operator== (const std::type_info& arg) const noexcept {
  return (&arg == this) || (strcmp(name(), arg.name()) == 0));
}
```

Variadic templates:
Отладочная информация и сообщения об ошибках

	C++03	C++11
Размер бинарного файла	77.0 кБ (76,999 байт)	72.6 кБ (72,642 байта)
Размер “пустого” бинарного файла	34.9 кБ (34,852 байта)	43.7 кБ (43,704 байта)
Размер сообщения об ошибке	14.3 кБ (14,289 байт)	3.2 кБ (3,219 байт)

Variadic templates:
Переносимость

```
template < BOOST_VARIANT_ENUM_PARAMS(typename T) >
std::size_t hash_value(variant< BOOST_VARIANT_ENUM_PARAMS(T) > const& val);
```

BOOST_VARIANT_ENUM_PARAMS(class Something)	class Something0, class... SomethingN
BOOST_VARIANT_ENUM_PARAMS(typename Something)	typename Something0, typename... SomethingN
BOOST_VARIANT_ENUM_PARAMS(Something)	Something0, SomethingN...
BOOST_VARIANT_ENUM_SHIFTED_PARAMS(class Something)	class... SomethingN
BOOST_VARIANT_ENUM_SHIFTED_PARAMS(typename Something)	typename... SomethingN
BOOST_VARIANT_ENUM_SHIFTED_PARAMS(Something)	SomethingN...

Rvalue references в цифрах

5 000 000 итераций	C++03 (миллисек)	C++11 (миллисек)	Относительный прирост
boost::variant(const variant&) boost::variant(variant&[&]) + boost::move	268 114	220 35	268/220 = **~1** 114/35 = **~3.2** 268/35 = **~7.5**
boost::variant=(const variant&) boost::variant=(variant&[&]) + boost::move	115 113	111 34	115/111 = **~1** 113/34 = **~3.3** 115/34 = **~3.3**
boost::variant=(const T&) boost::variant=(T&[&]) + boost::move	353 292	182 28	353/182 = **~2** 292/28 = **~10** 353/28 = **~12.5**

Android

- https://github.com/MysticTreeGames/Boost-for-Android
- https://github.com/apolukhin/regression_android
- http://www.boost.org/development/tests/develop/developer/summary.html

Git и модульность

Boost переехал с SVN на GIT и разбился на модули.

Впечатления:

- GIT неприятен в использовании
- Больше пользователей шлют исправления
- Merge стал проще

В новом Boost

- Variant с поддержкой variadic templates и noexcept
- MultiIndex с rvalue reference и другими C++11 фишками
- Поддержка Android
- C++14 string_view (string_ref)
- Модульность и GIT
- Поддержка MSVC2013

Вместо заключения

Отличная головоломка:

https://svn.boost.org/trac/boost/ticket/8555

```
#include <boost/variant.hpp>

typedef boost::variant<int> my_variant;
struct convertible {
  operator my_variant() const {
        return my_variant();
  }
};

int main() {
  convertible x;
  my_variant y = x;
}
```

Кое-что почитать:

«Boost C++ Application Development Cookbook»

ISBN: 9781849514880