EBook Gratis

APRENDIZAJE boost

Free unaffiliated eBook created from
Stack Overflow contributors.

#boost

Tabla de contenido

Acerca de

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: boost

It is an unofficial and free boost ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official boost.

The content is released under Creative Commons BY-SA, and the list of contributors to each chapter are provided in the credits section at the end of this book. Images may be copyright of their respective owners unless otherwise specified. All trademarks and registered trademarks are the property of their respective company owners.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Capítulo 1: Empezando con boost

Observaciones

¿Qué es Boost?

Boost es una gran colección de bibliotecas de C ++ gratuitas y de alta calidad que cubren una amplia gama de temas. A menudo se considera una "segunda biblioteca estándar" para C ++, ya que muchos problemas comunes en C ++ se resuelven utilizando Boost.

Desde boost.org :

> Boost proporciona bibliotecas gratuitas de código fuente de C ++ portátiles revisadas por pares.
>
> Hacemos hincapié en las bibliotecas que funcionan bien con la biblioteca estándar de C ++. Las bibliotecas Boost están destinadas a ser ampliamente útiles y se pueden utilizar en un amplio espectro de aplicaciones. La licencia Boost fomenta el uso comercial y no comercial.

Algunas bibliotecas Boost incluso han llegado a la biblioteca estándar de C ++ 11, y algunas otras, como Boost.Optional y Boost.Variant , serán parte de C ++ 17.

¿Qué puede hacer Boost?

Boost cubre la mayoría de los rincones de la programación. Desde la wiki de la etiqueta boost aquí en Stack Overflow:

> Incluye bibliotecas para:
>
> - Procesamiento de cadenas y textos.
> - Contenedores
> - Iteradores
> - Algoritmos
> - Objetos de función y programación de orden superior.
> - Programación genérica
> - Plantilla de metaprogramación
> - Preprocesamiento de metaprogramación
> - Programación concurrente
> - Matematicas y numericas
> - Corrección y pruebas.
> - Estructuras de datos
> - Procesamiento de imágenes
> - De entrada y salida
> - Soporte entre idiomas
> - Memoria

- Análisis
- Interfaces de programación
- Diverso
- Soluciones del compilador roto

Versiones

Versión	Nuevas bibliotecas	Notas de lanzamiento	Fecha de lanzamiento
1.10.0		notas	1999-12-14
1.11.0	Número racional	notas	2000-02-01
1.12.0		notas	2000-02-23
1.13.0	Utilidad, rasgos de tipo, rasgos de llamada, par comprimido	notas	2000-02-29
1.14.0		notas	2000-03-05
1.15.0	Número aleatorio	notas	2000-06-17
1.16.0	Funcional, cabecera iterador.	notas	2000-06-28
1.17.0	Formación	notas	2000-08-03
1.18.0	Gráfica, Expresión Regular	notas	2000-09-28
1.19.0	Verificación de concepto, Python, afirmación estática, Conceptos de mapa de propiedad	notas	2000-12-10
1.20.0	Conversión	notas	2001-01-06
1.21.0	Adaptador de iterador, piscina, prueba	notas	2001-03-09
1.22.0	CRC	notas	2001-05-25
1.23.0	Cualquier, Función, Tokenizer, Funciones Especiales, Octoniones, Cuaterniones.	notas	2001-07-06
1.24.0	Tupla	notas	2001-08-19
1.25.0	Hilo, Bind	notas	2001-10-01
1.26.0	Factor común, preprocesador	notas	2001-11-30

Versión	Nuevas bibliotecas	Notas de lanzamiento	Fecha de lanzamiento
1.27.0		notas	2002-02-05
1.28.0	Lambda, ahorrador de estado de E / S	notas	2002-05-15
1.29.0	Fecha-Hora, Bitset Dinámico, Formato	notas	2002-10-10
1.30.0	Sistema de archivos, Opcional, Intervalo, MPL, Spirit	notas	2003-03-19
1.31.0	enable_if, Variante	notas	2004-01-26
1.32.0	Asignación, minmax, multi-índice, Conversión Numérica, Opciones de Programa, Rango, Serialización, String, Tribool	notas	2004-11-19
1.33.0	Iostream, Hash, Parámetro, Contenedor de puntero, ola	notas	2005-08-11
1.34.0	Foreach, Statechart, TR1, Typeof, Xpressive	notas	2007-05-12
1.35.0	Asio, Bimap, Buffer Circular, Tipos de Función, Fusión, GIL, Interproceso, Intrusivo, Matemáticas / Funciones Especiales, Matemáticas / distribuciones estadísticas, MPI, sistema	notas	2008-03-29
1.36.0	Acumuladores, Excepción, Unidades, Desordenados	notas	2008-08-14
1.37.0	Proto	notas	2008-11-03
1.38.0	Peso mosca, ScopeExit, Swap	notas	2009-02-08
1.39.0	Señales2	notas	2009-05-02
1.40.0		notas	2009-08-27
1.41.0	Árbol de propiedad	notas	2009-11-17
1.42.0	Uuid	notas	2010-02-02
1.43.0	Funcional / Factor, Funcional / Adelante	notas	2010-05-06
1.44.0	Máquina de estados meta, polígono	notas	2010-08-13

Versión	Nuevas bibliotecas	Notas de lanzamiento	Fecha de lanzamiento
1.45.0		notas	2010-11-19
1.46.0	Icl	notas	2011-02-21
1.46.1		notas	2011-03-12
1.47.0	Crono, Geometría, Fénix, Ratio	notas	2011-07-11
1.48.0	Contenedor, configuración regional, mover	notas	15/11/2011
1.49.0	Montón	notas	2012-02-24
1.50.0	Algoritmo, Funcional / OverloadedFunction, LocalFunction, Utility / IdentityType	notas	2012-06-28
1.51.0	Contexto	notas	2012-08-20
1.52.0		notas	2012-11-05
1.53.0	Atomic, Coroutine, Lockfree, Multiprecision, Odeint	notas	2013-02-04
1.54.0	Registro, TTI, Borrado de Tipo	notas	2013-07-01
1.55.0	Predef	notas	2013-11-11
1.56.0	Alinear, TypeIndex	notas	2014-08-07
1.57.0		notas	2014-11-03
1.58.0	Endian, ordenar	notas	2015-04-17
1.59.0	Convertir, Coroutine2	notas	2015-08-13
1.60.0	VMD	notas	2015-12-17
1.61.0	Calcular, DLL, Hana, Metaparse	notas	2016-05-13
1.62.0	Fibra, QVM	notas	2016-09-28
1.63.0		notas	2016-12-26
1.64.0	Proceso	notas	2017-04-19

Examples

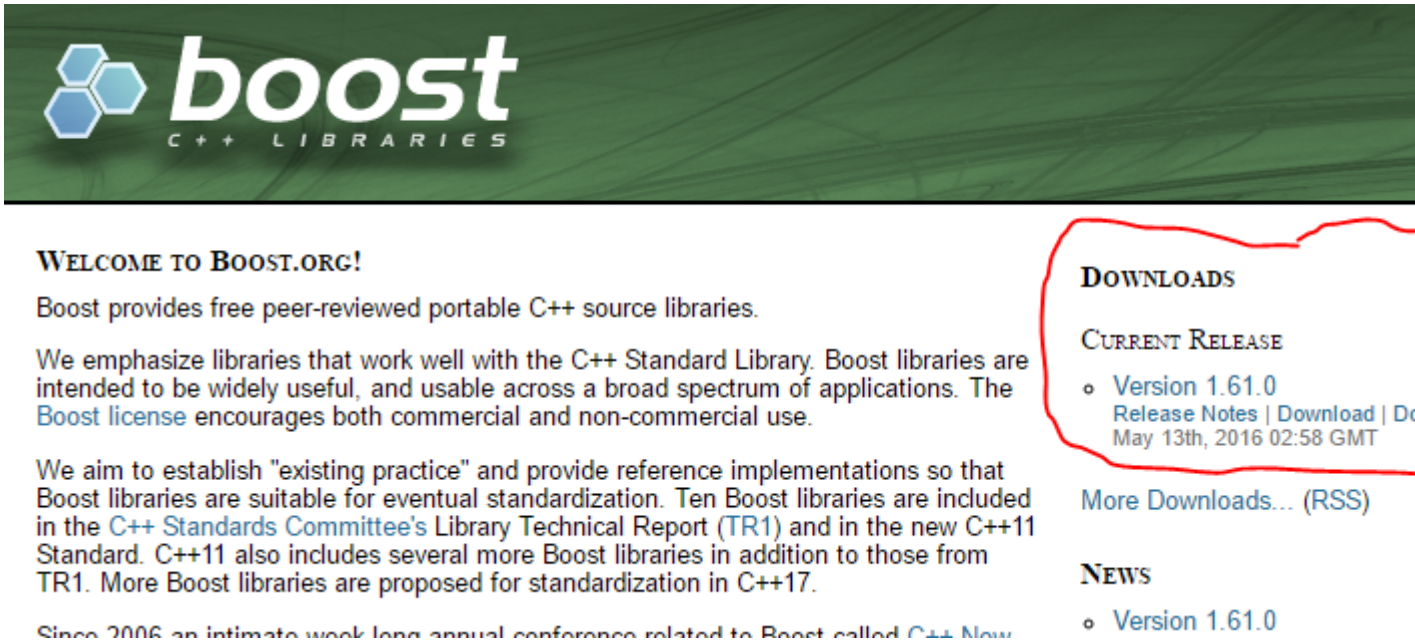
Instalación o configuración

Ver Boost Getting Started .

La mayoría de las bibliotecas de Boost son solo para encabezados, lo que significa que no hay nada que compilar o enlazar.

Asegúrate de obtener la versión más reciente de Boost:

1. Visite www.boost.org
2. Busque la descarga de la versión actual. Actualmente, este enlace aquí .



3. Seleccione el archivo de almacenamiento apropiado para su sistema operativo y descárguelo.

Las bibliotecas de solo encabezado pueden utilizarse simplemente incluyendo los archivos de encabezado respectivos.

Algunas bibliotecas de Boost requieren compilación:

- Boost.Chrono
- Boost.Context
- Boost.Filesystem
- Boost.GraphParallel
- Boost.IOStreams
- Boost.Locale
- Boost.MPI
- Boost.ProgramOptions
- Boost.Python
- Boost.Regex
- Boost.Serialization

- Boost.Signals
- Boost.Sistema
- Boost.Hilo
- Boost.Timer
- Boost.Wave

Además, las siguientes bibliotecas tienen componentes que deben ser compilados:

- Boost.DateTimeTime
- Boost.Graph
- Boost.Math
- Boost.Random
- Prueba de alza
- Boost.Exception

La fuente de Boost se puede obtener a través del enlace de descarga en el sitio , que volverá a dirigir a su página de SourceForge para obtener la última versión (1.61.0 a partir de julio de 2016). Esto se puede descomprimir (o desempaquetar, etc.) a un directorio (como C: \ local \ boost_1_61_0). Este directorio se puede agregar a la ruta de inclusión para el software que está creando. Después de esto, puede incluir encabezados Boost en archivos C ++ con `#include <boost/header/path.hpp>` .

La mayoría de las bibliotecas en Boost son solo de encabezado. Si solo necesita estos, la distribución de la fuente anterior es todo lo que se necesita. Sin embargo, si necesita usar una de las bibliotecas que requiere la creación de un binario compilado, también lo necesitará.

En cualquier sistema, la forma más confiable de obtener los binarios correctos es construirlos usted mismo. Estas direcciones son algo diferentes para Windows o Linux / Unix / POSIX .

En Windows con Visual Studio, una alternativa a la creación de las bibliotecas usted mismo es descargar bibliotecas previamente creadas desde la página de SourceForge de Boost (1.61.0 a partir de julio de 2016). En esa página, puede seleccionar un instalador que instalará una versión para una compilación específica de Visual Studio o el archivo 7-zip (boost_X_XX_X-bin-all-32-64.7z) que contiene los archivos binarios para todas las versiones de Visual Studio compatibles. Cualquiera de estas opciones incluye la fuente / encabezados, así como los binarios, por lo que no es necesario haber descargado la distribución de fuente anterior. Una vez que lo tenga, extraiga / instale en un directorio (como C: \ local \ boost_1_61_0) y agregue ese directorio a su ruta de inclusión, luego agregue el directorio que contiene los archivos binarios que corresponden a su versión de Visual Studio (por ejemplo, C: \ local \ boost_1_61_0 \ lib32-msvc-12.0 para proyectos de Visual Studio 2013 de 32 bits) a la ruta de la biblioteca.

Instalación y ejecución de Boost (Cygwin)

(Nivel principiante; IDE: CLion)

Primero, instale boost desde el espejo de Cygwin: abra el exe de instalación, busque boost, instale los paquetes.

Después de que se instale boost: se ubicará en `/usr/include/boost` . Aquí es donde está todo. Todas las declaraciones `#include` serán una ruta desde la carpeta boost, como en: `#include <boost/archive/text_oarchive.hpp>` .

Una vez que incluya los archivos boost de su elección en sus archivos `.cpp` , su código aún no se compilará en el IDE de su elección hasta que **vincule** las bibliotecas y le diga a cmake que busque el código boost descargado .

Con el fin de obtener cmake para buscar su código de impulso,

```
find_package(Boost 1.60.0 COMPONENTS components_you_want)

# for example:
find_package(Boost 1.60.0 COMPONENTS serialization)
```

Luego, incluya los directorios: `include_directories(${Boost_INCLUDE_DIRS})`

Finalmente, agrega tu ejecutable y vincula las bibliotecas:

```
add_executable(your_target ${SOURCE_FILES})
target_link_libraries(your_target ${Boost_LIBRARIES} -any_missing_boostlibs)
```

Antes de iniciar su programa, evite un volcado de error haciendo pruebas para ver si se ha encontrado un impulso antes de incluir algo o ejecutar su código:

```
if (Boost_FOUND)
    include_directories(${Boost_INCLUDE_DIRS})
    add_executable(YourTarget ${SOURCE_FILES})
    target_link_libraries(your_target ${Boost_LIBRARIES} -missing_libs)
endif()
```

`-missing_libs` porque un error que puede encontrar es que alguna biblioteca boost u otra podría no estar vinculada, y debe agregarla manualmente, por ejemplo, el enlace al que hice referencia anteriormente.

Todos juntos, un archivo final CMakeLists.txt puede verse algo así como:

```
cmake_minimum_required(VERSION 3.7)
project(your_project)

set(CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} -std=c++11")
set(SOURCE_FILES main.cpp tmap.cpp tmap.h)
find_package(Boost 1.60.0 COMPONENTS serialization)
```

```
if(Boost_FOUND)
    include_directories(${Boost_INCLUDE_DIRS})
    add_executable(your_project ${SOURCE_FILES})
    target_link_libraries(your_project ${Boost_LIBRARIES})
endif()
```

Lea Empezando con boost en línea: https://riptutorial.com/es/boost/topic/1167/empezando-con-boost

Capítulo 2: Async boost :: process

Examples

Usando los 3 tubos de un proceso hijo de forma asíncrona.

```
#include <vector>
#include <string>
#include <boost/process.hpp>
#include <boost/asio.hpp>
#include <boost/process/windows.hpp>

int Run(
    const std::string& exeName,         ///< could also be UTF-16 for Windows
    const std::string& args,            ///< could also be UTF-16 for Windows
    const std::string& input,           ///< [in] data for stdin
    std::string& output,                ///< [out] data from stdout
    std::string& error                  ///< [out] data from stderr
)
{
    using namespace boost;

    asio::io_service ios;

    // stdout setup
    //
    std::vector<char> vOut(128 << 10);          // that worked well for my decoding app.
    auto outBuffer{ asio::buffer(vOut) };
    process::async_pipe pipeOut(ios);

    std::function<void(const system::error_code & ec, std::size_t n)> onStdOut;
    onStdOut = [&](const system::error_code & ec, size_t n)
    {
        output.reserve(output.size() + n);
        output.insert(output.end(), vOut.begin(), vOut.begin() + n);
        if (!ec)
        {
            asio::async_read(pipeOut, outBuffer, onStdOut);
        }
    };

    // stderr setup
    //
    std::vector<char> vErr(128 << 10);
    auto errBuffer{ asio::buffer(vErr) };
    process::async_pipe pipeErr(ios);

    std::function<void(const system::error_code & ec, std::size_t n)> onStdErr;
    onStdErr = [&](const system::error_code & ec, size_t n)
    {
        error.reserve(error.size() + n);
        error.insert(error.end(), vErr.begin(), vErr.begin() + n);
        if (!ec)
        {
            asio::async_read(pipeErr, errBuffer, onStdErr);
        }
    };
```

```
    // stdin setup
    //
    auto inBuffer{ asio::buffer(input) };
    process::async_pipe pipeIn(ios);

    process::child c(
        exeName + " " + args,                   // exeName must be full path
        process::std_out > pipeOut,
        process::std_err > pipeErr,
        process::std_in < pipeIn
    );

    asio::async_write(pipeIn, inBuffer,
        [&](const system::error_code & ec, std::size_t n)
        {
            pipeIn.async_close();                     //  tells the child we have no more data
        });

    asio::async_read(pipeOut, outBuffer, onStdOut);
    asio::async_read(pipeErr, errBuffer, onStdErr);

    ios.run();
    c.wait();
    return c.exit_code();                           // return the process' exit code.
}
```

IMPORTANTE para boost 1.64

Hay un error / solución para boost 1.64, el error solo afecta a Windows, al parecer.

referencia: https://github.com/klemens-morgenstern/boost-process/issues/90 y https://github.com/klemens-morgenstern/boost-process/commit/74814e46c1614850a8e447fd689c21cf82f36ceb

en el archivo `boost\process\detail\windows\async_pipe.hpp, line 79` :

```
    ~async_pipe()
    {
//fix
        //if (_sink .native()  != ::boost::detail::winapi::INVALID_HANDLE_VALUE_)
        //    ::boost::detail::winapi::CloseHandle(_sink.native());
        //if (_source.native() != ::boost::detail::winapi::INVALID_HANDLE_VALUE_)
        //    ::boost::detail::winapi::CloseHandle(_source.native());
        boost::system::error_code ec;
        close(ec);
//fix
    }
```

Lea Async boost :: process en línea: https://riptutorial.com/es/boost/topic/10822/async-boost----process

Capítulo 3: biblioteca de algoritmos de cadena de impulso

Observaciones

Impulsar la documención en los algroritmos de cuerdas

Examples

boost :: split ()

```
#include <iostream>
#include <vector>
#include <string>
#include <boost/algorithm/string.hpp>

using namespace std;

int main()
{

    // String to split

    string str = "You're supposed to see this!|NOT THIS!!!!!!";

    // Line container

    vector<string> lines;

    // Splits string

    boost::split(lines, str, boost::is_any_of("|"), boost::token_compress_on);

    // Outputs 1 half of the split string

    cout << lines.at(0).c_str() << endl;

    // Waits for input before program exits

    cin.get();

    return 0;
}
```

El siguiente es el programa en *psuedocode* :

Declarar cadena *str* y la **puso** a *"Se supone que ver esto |! NO ESTA !!!!!!"* .

Declarar *líneas* vectoriales de **tipo** cadena.

Dividir cadena *str* en *líneas* vectoriales si regex *"|"* es encontrado.

Imprimir objeto en el índice 0 en líneas.

Obtener entrada.

Reemplazar Algrorithms

boost :: replace_all ():

```
#include <iostream>
#include <string>
#include <boost/algorithm/string.hpp>

using namespace std;

int main()
{

    // String to replace characters in

    string str = "Darn you, Darn you to the 5th power!!!";

    // Replace "Darn" with "*CENSORED*"

    boost::replace_all(str, "Darn", "*CENSORED*");

    // Print str

    cout << str.c_str() << endl;

    // Waits for program to exit

    cin.get();

    return 0;
}
```

boost :: replace_first ():

```
#include <iostream>
#include <string>
#include <boost/algorithm/string.hpp>

using namespace std;

int main()
{

    // String to replace characters in

    string str = "Darn you, Darn you to the 5th power!!!";

    // Replace the first instance of "Darn" with "*CENSORED*"

    boost::replace_first(str, "Darn", "*CENSORED*");
```

```
    // Print str

    cout << str.c_str() << endl;

    // Waits for program to exit

    cin.get();

    return 0;
}
```

boost_replace_last ():

```
#include <iostream>
#include <string>
#include <boost/algorithm/string.hpp>

using namespace std;

int main()
{

    // String to replace characters in

    string str = "Darn you, Darn you to the 5th power!!!";

    // Replace the last instance of "Darn" with "*CENSORED*"

    boost::replace_last(str, "Darn", "*CENSORED*");

    // Print str

    cout << str.c_str() << endl;

    // Waits for program to exit

   cin.get();

   return 0;
}
```

Métodos de conversión de casos

to_upper ():

```
#include <iostream>
#include <string>
#include <boost/algorithm/string.hpp>

using namespace std;

int main()
{
```

```
    // String to convert characters to uppercase

    string str = "ThIS iS SUpPoSEd tO Be UpPeR CAsE.";

    // Convert characters in str to upper case

    boost::to_upper(str);

    // Print str

    cout << str.c_str() << endl;

    // Waits for program to exit

    cin.get();

    return 0;
}
```

reducir():

```
#include <iostream>
#include <string>
#include <boost/algorithm/string.hpp>

using namespace std;

int main()
{

    // String to convert characters to lowercase

    string str = "ThIS iS SUpPoSEd tO Be LoWer CAsE.";

    // Convert characters in str to lower case

    boost::to_lower(str);

    // Print str

    cout << str.c_str() << endl;

    // Waits for program to exit

    cin.get();

    return 0;
}
```

Lea biblioteca de algoritmos de cadena de impulso en línea:
https://riptutorial.com/es/boost/topic/7239/biblioteca-de-algoritmos-de-cadena-de-impulso

Capítulo 4: Boost Acumuladores Framework

Examples

Informática media y varianza.

```
#include <iostream>
#include <boost/accumulators/accumulators.hpp>
#include <boost/accumulators/statistics/stats.hpp>
#include <boost/accumulators/statistics/mean.hpp>
#include <boost/accumulators/statistics/variance.hpp>

int main()
{
  using namespace boost::accumulators;
  accumulator_set<int, stats<tag::mean, tag::variance>> acc;

  for(int i = 1; i <= 6; i++)
    acc(i);

  std::cout << "mean=" << mean(acc) << ", variance=" << variance(acc) << '\n';
  // prints "mean=3.5, variance=2.91667"

  return 0;
}
```

Lea Boost Acumuladores Framework en línea: https://riptutorial.com/es/boost/topic/5718/boost-acumuladores-framework

Capítulo 5: BOOST- Compara Imágenes usando OpevCV

Introducción

Esta documentación explica cómo se puede probar una imagen externa y compararla con la imagen de salida de OpenCV. Por ejemplo, para comparar dos imágenes borrosas y probar si ambas son iguales, desenfocamos una imagen original en un software externo (utilicé el software de procesamiento de imágenes WiT) o simplemente descargamos cualquier imagen borrosa en línea-output1. Crear un proyecto Win32 OpenCV en Visual Studio. Lea la imagen original como una entrada al OpenCV. Difumina la imagen original en OpenCV y compárala con output1.

Examples

Código OpenCV para leer imágenes y comparar.

#include <opencv2 / opencv.hpp> #include

usando namespace cv; utilizando namespace std;

int main (int argc, char ** argv) {Imagen mat; image = imread ("C: \ Users \ Development \ Documents \ Visual Studio 2013 \ Projects \ ImageIn.bmp", CV_LOAD_IMAGE_GRAYSCALE); // leer el archivo

```
if (!image.data)                              // Check for invalid input
{
    cout << "Could not open or find the image" << std::endl;
    return -1;
}



Mat witout = imread("C:\\Users\\Development\\Documents\\Visual Studio
2013\\Projects\\ImageWitOut.bmp", CV_LOAD_IMAGE_GRAYSCALE);;
Mat cvout = Mat(image.size(), image.type(), Scalar(255));

imshow("witout", witout);
imshow("cvout", cvout);

Mat diff = (witout == cvout);

namedWindow("Difference", WINDOW_AUTOSIZE);// Create a window for display.
imshow("Difference", diff);                   // Show our image inside it.

waitKey(0);                                          // Wait for a keystroke in the window
return 0;
```

}

Lea BOOST- Compara Imágenes usando OpevCV en línea:
https://riptutorial.com/es/boost/topic/10703/boost--compara-imagenes-usando-opevcv

Capítulo 6: Opciones de programa de impulso

Examples

Uso básico

Las opciones del programa Boost proporcionan una forma sencilla y segura de analizar y manejar los argumentos de la línea de comandos.

```
#include <boost/program_options.hpp>
#include <string>
#include <iostream>

int main(int argc, char** argv) {
  namespace po = boost::program_options;

  po::variables_map vm;
  po::options_description desc("Allowed Options");

  // declare arguments
  desc.add_options()
    ("name", po::value<std::string>()->required(), "Type your name to be greeted!");

  // parse arguments and save them in the variable map (vm)
  po::store(po::parse_command_line(argc, argv, desc), vm);

  std::cout << "Hello " << vm["name"].as<std::string>() << std::endl;

  return 0;
}
```

Compila y ejecuta con:

```
 $ g++ main.cpp -lboost_program_options && ./a.out --name Batman
Hello Batman
```

Puede generar un objeto `boost::program_options::options_description` para imprimir el formato de argumento esperado:

```
std::cout << desc << std::endl;
```

produciría

```
Allowed Options:
  --name arg                Type your name to be greeted!
```

Manejo de errores

`boost::program_options::notify` se puede usar para reportar cualquier error en los parámetros que pasan

```
#include <boost/program_options.hpp>
#include <string>
#include <iostream>

int main(int argc, char** argv) {
  namespace po = boost::program_options;

  po::variables_map vm;
  po::options_description desc("Allowed Options");

  // declare options
  desc.add_options()
    ("name", po::value<std::string>()->required(), "Type your name to be greeted!");

  // parse arguments
  po::store(po::parse_command_line(argc, argv, desc), vm);

  // check arguments
  try {
    po::notify(vm);
  } catch (std::exception& e) {
    std::cout << "Error: " << e.what() << std::endl;
    std::cout << desc << std::endl;
    return 1;
  }

  // program logic
  std::cout << "Hello " << vm["name"].as<std::string>() << std::endl;

  return 0;
}
```

Pasar argumentos ilegales produce mensajes de error útiles

```
 $ ./a.out
Error: the option '--name' is required but missing
Allowed Options:
  --name arg            Type your name to be greeted!
```

Valores predeterminados

Un argumento de línea de comando con valor predeterminado se puede especificar fácilmente:

```
// declare options
desc.add_options()
  ("name", po::value<std::string>()->required(), "Type your name to be greeted!")
  ("rank", po::value<std::string>()->default_value("Dark Knight"), "Your rank");
```

Su valor también se agrega al mapa variable:

```
  std::cout << "Hello " << vm["name"].as<std::string>() << " " << vm["rank"].as<std::string>()
<< std::endl;
```

El valor predeterminado se muestra en la descripción ...

```
$ ./a.out
Error: the option '--name' is required but missing
Allowed Options:
  --name arg                Type your name to be greeted!
  --rank arg (=Dark Knight) Your rank
```

... y se usa si no se especifica ...

```
$ ./a.out --name Batman
Hello Batman Dark Knight
```

... pero se puede sobrescribir en la línea de comando:

```
 $ ./a.out --name Batman --rank FlyingSquirrel
Hello Batman FlyingSquirrel
```

Interruptores

Un interruptor es un argumento de línea de comando que no toma ningún valor. Se puede especificar con:

```
desc.add_options()
  ("hidden", po::bool_switch()->default_value(false), "Hide your name");
```

Y usado con:

```
if (vm["hidden"].as<bool>())
   std::cout << "Hello *****" << std::endl;
```

desde la línea de comando:

```
 $ ./a.out --name Batman --hidden
Hello *****
```

y en la descripción se muestra como:

```
Allowed Options:
  --name arg                Type your name to be greeted!
  --rank arg (=Dark Knight) Your rank
  --hidden                  Hide your name
```

Lea Opciones de programa de impulso en línea:
https://riptutorial.com/es/boost/topic/5359/opciones-de-programa-de-impulso

Capítulo 7: Usando boost.python

Examples

Ejemplo introductorio en Boost.Python

Las cosas son fáciles cuando tienes que usar una biblioteca de C ++ en un proyecto de Python. Solo puedes usar Boost.

En primer lugar, aquí hay una lista de los componentes que necesita:

- Un archivo CMakeList.txt, porque vas a utilizar CMake.
- Los archivos C ++ del proyecto C ++.
- El archivo python - este es tu proyecto python.

Vamos a empezar con un pequeño archivo de C ++. Nuestro proyecto C ++ tiene solo un método que devuelve una cadena "Este es el primer intento". Llámalo *CppProject.cpp*

```
char const *firstMethod() {
    return "This is the first try.";
}

BOOST_PYTHON_MODULE(CppProject) {
    boost::python::def("getTryString", firstMethod); // boost::python is the namespace
}
```

Tenga un archivo CMakeLists.txt a continuación:

```
cmake_minimum_required(VERSION 2.8.3)
FIND_PACKAGE(PythonInterp)
FIND_PACKAGE(PythonLibs)
FIND_PACKAGE(Boost COMPONENTS python)

INCLUDE_DIRECTORIES(${Boost_INCLUDE_DIRS} ${PYTHON_INCLUDE_DIRS})

PYTHON_ADD_MODULE(NativeLib CppProject)
FILE(COPY MyProject.py DESTINATION .) # See the whole tutorial to understand this line
```

Por esta parte del tutorial todo es muy fácil. puede importar la biblioteca y el método de llamada en su proyecto python. Llame a su proyecto de Python *MyProject.py* .

```
import NativeLib
print (NativeLib.getTryString)
```

Para ejecutar su proyecto siga las instrucciones a continuación:

- Crear un directorio con el nombre de *construcción* .
- Entra en ese directorio.
- Dé el comando `cmake -DCMAKE_BUILD_TYPE=Release ..`

- `make`
- `python MyProject.py` . Ahora, tiene que ver la cadena que devuelve el método en su proyecto C ++.

Envolviendo std :: vector en boost.python

Si una función devuelve un tipo `std::vector` , y se expone a Python directamente como

```
std::vector<float> secondMethod() {
    return std::vector<float>();
}

BOOST_PYTHON_MODULE(CppProject) {
    boost::python::def("getEmptyVec", secondMethod);
}
```

luego, cuando las funciones se llamen, Python le dirá `No to_python (by-value) converter found for C++ type: std::vector<float, std::allocator<float> >` , porque Python necesita saber cómo lidiar con `std::vector` .

Afortunadamente, boost.python ha proporcionado una función de envoltorio para nosotros en `vector_indexing_suite.hpp` . El valor de retorno puede manejarse como un objeto `FloatVec` cuyo elemento puede acceder el operador `[]` , exponiendo la función de envoltura correspondiente de la siguiente manera.

```
std::vector<float> secondMethod() {
    return std::vector<float>();
}

BOOST_PYTHON_MODULE(CppProject) {
    // wrapper function
    class_<std::vector<float> >("FloatVec")
        .def(vector_indexing_suite<std::vector<float> >());
    boost::python::def("getEmptyVec", secondMethod);
}
```

El resultado se puede convertir aún más en una lista de Python o una matriz de Numpy simplemente llamando a la `list()` y `numpy.asarray()` .

Lea Usando boost.python en línea: https://riptutorial.com/es/boost/topic/6433/usando-boost-python

Creditos

S. No	Capítulos	Contributors
1	Empezando con boost	bordeo, Community, Igor R., ildjarn, Justin, Null, teeks99
2	Async boost :: process	Michaël Roy
3	biblioteca de algoritmos de cadena de impulso	Zopesconk
4	Boost Acumuladores Framework	Philipp Claßen
5	BOOST- Compara Imágenes usando OpevCV	DivyaMaheswaran
6	Opciones de programa de impulso	paul-g
7	Usando boost.python	dontloo, Onur Tuna