Obsolete 2 Feb 2012

Instead refer to TCT/BoostUsageProposal

Obsolete 2 Feb 2012

Boost

From: LSST Toolsets

To Quote the ?boost home page:

> Boost provides free peer-reviewed portable C++ source libraries.
>
> We emphasize libraries that work well with the C++ Standard Library. Boost libraries are intended to be widely useful, and usable across a broad spectrum of applications. The Boost license encourages both commercial and non-commercial use.
>
> We aim to establish "existing practice" and provide reference implementations so that Boost libraries are suitable for eventual standardization. Ten Boost libraries are already included in the C++ Standards Committee's Library Technical Report (TR1) as a step toward becoming part of a future C++ Standard. More Boost libraries are proposed for the upcoming TR2.

The LSST project's DM group endorse the attempt to produce standardized, high quality libraries. Accordingly, we plan to use ?Boost whenever appropriate.

If you find functionality in Boost but would prefer not to use it, please check with DMManagement before taking a decision.

LSST's Choice of Boost Libraries

Approved version of boost libraries: Boost 1.35.0 (as of 7 May 2008)

Libraries blessed for use in LSST code:

?any
: Safe, generic container for single values of different value types

?cstdint
: The typedef's useful for writing portable code that requires certain integer widths.

?current_function

?filesystem TR2
: Portable paths, iteration over directories, and other useful filesystem operations.

?format TR2
: Type-safe 'printf-like' format operations

?GIL
: The Generic Image Library (CCB, 2008-09-03).

?iterator/zip_iterator

?lambda TR2

Lambda functions for use with generic programming (e.g. `std::transform`)

?MPI
: Message Passing Interface library, for use in distributed-memory parallel application programming

?mpl/assert

?mpl/at

?mpl/bool

?mpl/for_each

?mpl/or

?mpl/vector

?multi_index
: Containers with multiple STL-compatible access interfaces.

?noncopyable
: Class noncopyable plus checked_delete(), checked_array_delete(), next(), prior() function templates, plus base-from-member idiom::

?regex
: TR1, Perl-style regular expressions.

?scoped_ptr
: TR2 , Stores a pointer to a dynamically allocated object.::

?serialization
: Serialization for persistence and marshalling. serialization archives:: Text (or XML) archives render data as text (or XML) and are portable across platforms

?shared_array
: TR2, Array ownership shared among multiple pointers

?shared_ptr
: TR1, Object ownership shared among multiple pointers

?static_assert C++0X
: Static assertions (compile time assertions)::

?test
: Support for simple program testing, full unit testing, and for program execution monitoring. Approval limited to 'header-only' version, not compiled boost:unittest library.::

?tokenizer
: Break of a string or other character sequence into a series of tokens. Proposal for inclusion: TokenizerProposal::

?type_traits TR1
: Templates for fundamental properties of types

?weak_ptr
: Non-owning observers of an object owned by shared_ptr

Libraries that may not be used:

?config

?throw_exception

Libraries under consideration:

?numeric/ublas

?operators

Boost Libraries in LSST

2009-01-07

?any (Status: approved)
afw daf/base pex/logging pex/policy utils

?config (Status: reject) Helps boost library developers adapt to compiler idiosyncrasies; not intended for library users.
pex/policy

RHL comments: This is only used in a hacked-up version of any.h. The authors should revisit it

?cstdint (Status: approved) The typedef's useful for writing portable code that requires certain integer widths.
afw isr meas/algorithms meas/astrom

?current_function (Status: approved)
pex/exceptions

?filesystem TR2 (Status: approved) Portable paths, iteration over directories, and other useful filesystem operations.
afw pex/policy

?format TR2 (Status: approved)
afw daf/base daf/data isr meas/algorithms pex/exceptions pex/logging utils

?iterator/zip_iterator (Status: approved)
afw

?lambda TR2 (Status: approved)
afw

?mpl/assert (Status: approved)
afw

RHL notes: this seems to be only used by me

?mpl/at (Status: approved)
afw

RHL notes: this seems to be only used by me

?mpl/bool (Status: approved)
afw

RHL notes: this seems to be only used by me

?mpl/for_each (Status: approved)
afw

RHL notes: this seems to be only used by me

?mpl/or (Status: approved)
afw

RHL notes: this seems to be only used by me

?mpl/vector (Status: approved)
afw

RHL notes: this seems to be only used by me

?multi_index (Status: approved) Containers with multiple STL-compatible access interfaces.
utils

RHL notes: this seems to be only used by me

?noncopyable (Status: approved) Class noncopyable plus checked_delete(), checked_array_delete(), next(), prior() function templates, plus base-from-member idiom.
daf/base

?numeric/ublas (Status: unknown)
meas/astrom

?operators (Status: unknown)
isr

?regex TR1 (Status: approved) Perl-style regular expressions.
afw daf/persistence pex/logging pex/policy utils

?scoped_ptr TR2 (Status: approved) Stores a pointer to a dynamically allocated object.
afw daf/base daf/persistence pex/policy

?serialization (Status: approved) Serialization for persistence and marshalling. serialization archives
Text (or XML) archives render data as text (or XML) and are portable across platforms. :: afw daf/persistence

?shared_array TR2 (Status: approved)
daf/persistence

?shared_ptr TR1 (Status: approved)
afw daf/base daf/data daf/persistence isr meas/algorithms meas/astrom pex/logging pex/policy security

?static_assert C++0X (Status: approved) Static assertions (compile time assertions).
afw pex/policy

?test (Status: approved)

afw daf/base pex/exceptions

?throw_exception (Status: reject) Intended to be used in Boost libraries that need to throw exceptions, but support configurations and platforms where exceptions aren't available
pex/policy

RayPlante comments: I don't recall where this was used, but it seems to me that its use should be replaced by the recommended pex/exceptions usage; I suggest we put this on the "do not use" list.

?tokenizer (Status: approved) Break of a string or other character sequence into a series of tokens. Proposal for inclusion: TokenizerProposal
pex/logging

?type_traits TR1 (Status: approved) Templates for fundamental properties of types.
afw pex/policy

RHL notes that he's recommended all the libraries that appear in TR1, TR2, or the C++0X standard. Other libraries (in particular the mpl) ones are more a matter of taste; his taste has been to permit them.

There are four libraries marked "unknown":

?config - rejected ?numeric/ublas ?operators ?throw_exception - rejected

I'll ask the developers using them to comment.