Chapter 34. Boost.IOStreams

Table of Contents

This chapter introduces the library Boost.IOStreams. Boost.IOStreams breaks up the well-known streams from the standard library into smaller components. The library defines two concepts: *device*, which describes data sources and sinks, and *stream*, which describes an interface for formatted input/output based on the interface from the standard library. A stream defined by Boost.IOStreams isn't automatically connected to a data source or sink.

Boost.IOStreams provides numerous implementations of the two concepts. For example, there is the device `boost::iostreams::mapped_file`, which loads a file partially or completely into memory. The stream `boost::iostreams::stream` can be connected to a device like `boost::iostreams::mapped_file` to use the familiar stream operators `operator<<` and `operator>>` to read and write data.

In addition to `boost::iostreams::stream`, Boost.IOStreams provides the stream `boost::iostreams::filtering_stream`, which lets you add data filters. For example, you can use `boost::iostreams::gzip_compressor` to write data compressed in the GZIP format.

Boost.IOStreams can also be used to connect to platform-specific objects. The library provides devices to connect to a Windows handle or a file descriptor. That way objects from low-level APIs can be made available in platform-independent C++ code.

The classes and functions provided by Boost.IOStreams are defined in the namespace `boost::iostreams`. There is no master header file. Because Boost.IOStreams contains more than header files, it must be prebuilt. This can be important because, depending on how Boost.IOStreams has been prebuilt, support for some features could be missing.