boost
C++ LIBRARIES

Boost.FileSystem

Dmitri Nesteruk
http://activemesa.com
dn@activemesa.com

pluralsight
hardcore developer training

Overview

- **Work with the local file system in a *portable* manner**
- **Typical tasks**
  - Determine if file or directory exists
  - Measure file size and determine its properties
  - Navigate directory hierarchy
- **Consists of**
  - Objects representing file structure elements (e.g., `path`)
  - Utility functions (e.g., `exists()`)
- **Error handling via C++ exceptions**
- **In the process of migration to the STL**
  - `std::tr2`

Concepts

- **File**
  - Unit of physical storage
  - Source code is made up of files
  - Executable is a file
- **Directory**
  - A.k.a. ‘folder’ on Windows
  - Storage container
  - Can contain files or directories
- **Path**
  - Textural representation of location of a file or directory
  - Different notation on different OSs
    `c:\foo\bar.txt` vs `/foo/bar.txt`

Fundamentals

- **Path represented with `path` class**
  - Can represent either directory or file
  - Can get text with `string()`
  - Can extract the name with `filename()`
  - `parent_path()` moves up the hierarchy
  - `root_path()` gets us to the very top
  - path objects are decomposable
    - `for (auto& child : path)`
- **Determine existence and type of path**
  - `exists(), is_directory(), is_regular_file()`
- **Find out size of file with `file_size()`**
- **Operations on non-existent entries throw a `filesystem_error`**

File Status

- **Different types of file in addition to 'regular'**
  - E.g., symlink
- **`file_status` represents info about a file**
  - Acquired by `status()`
  - A `file_type` value returned by `type()`
  - A set of permissions (`perms`) returned by the `permissions()` function

Directory Navigation

- `directory_iterator(path)`
- Default iterator as stopping condition
- Iterator provides a `path()`

Summary

- `#include <boost/filesystem.hpp>` or
- `#include <filesystem>`
  `using namespace std::tr2;`
- Create a path object to wrap around a path
- Use `is_regular_file()/is_directory()` to determine type
- Use `file_size()` to determine size
- Use `status()` to get type and permissions
- To navigate directory structure
  - Use `parent_path()` or `root_path()`
  - Use a `directory_navigator` to get contents of a directory