System obsługi plików
<filesystem>

Daniel Wikarek

Patryk Rossa

Grupa 2, sekcja 2

Co to jest <filesystem> i do czego służy?

<filesystem> to biblioteka dodana w standardzie C++17, pozwalająca na łatwą obsługę plików oraz ich składowych, takich jak: ścieżki, pliki regularne oraz katalogi.

Podstawowe pojęcia obsługi plików:

- **plik (file)** - obiekt systemu plików, który przechowuje dane, używany do odczytu i/lub zapisu. Typy plików:
  - **katalog (directory)** - plik, który jest kontenerem dla innych obiektów plikowych
  - **trwały link (hard link)** - obiekt katalogu, który wiąże nazwę z istniejącym plikiem.
  - **link symboliczny (symbolic link)** - obiekt katalogowy, który wiąże nazwę ze ścieżką, która istnieje lub nie
  - **regularny plik (regular file)** - plik, który nie jest jednym z powyżej zdefiniowanych rodzajów plików

- **nazwa pliku (filename)** - ciąg znaków, który określa nazwę pliku. Nazwy . oraz .. mają specjalne znaczenie w bibliotece

- **ścieżka (path)** - sekwencja elementów identyfikująca plik.

Typy ścieżek:

- **absolutna (absolut path)** - ścieżka, która jednoznacznie identyfikuje lokalizację pliku
- **kanoniczna (canonical path)** - ścieżka absolutna, która nie zawiera linków symbolicznych, . lub ..
- **względna (relative path)** - ścieżka, która identyfikuje lokalizację pliku w odniesieniu do danej lokalizacji w systemie plików

Obiekt std::filesystem::path

Podstawowym elementem biblioteki <filesystem> jest obiekt **path**. Daje nam on dostęp do wielu użytecznych funkcji. W obiekcie tym występuje domniemana konwersja na typ std::string, przez co możemy się łatwo posługiwać operatorami strumieniowymi.

Najczęściej wykorzystywanymi metodami klasy są:

- funkcje składowe dekompozycji ścieżki
- operacje append(), operator /()
- operacje concat(), operator +()

std::filesystem::current_path()

Zwraca ścieżkę folderu, jako obiekt
std::filesystem::path, w którym znajduje się
plik, w którym wywołaliśmy tą funkcję.

Przykład:

```
std::filesystem::path pth = std::filesystem::current_path()

cout << pth << endl  // C:\Users\user1\Desktop\Laboratorium 3\Laboratorium 3
```

Funkcje dekompozycji ścieżki

- root_name() – zwraca nazwę korzenia ścieżki w formacie ogólnym
- root_directory() - zwraca katalog główny ścieżki w formacie ogólnym.
- root_path() – zwraca główną ścieżkę ścieżki
- relative_path() – zwraca ścieżkę względną do głównej
- parent_path() – zwraca ścieżkę do katalogu nadrzędnego
- filename() – zwraca nazwę pliku z rozszerzeniem
- stem() – zwraca nazwę pliku bez rozszerzenia
- extension() – zwraca rozszerzenie pliku

Operacje append() i operator /()

Przykład:

```
path /= "Training”;
path = path / "Modern" / "Cpp17”;
path.append("Programming”);
cout << path;
// C:\Users\Infotraining\Documents\Training\Modern\Cpp17\Programming
```

Operacje concat() i operator +()

Przykład:

```
fs::path p2(„C:\\temp\\”);
p2 += "user”;
p2.concat(„data”);
cout << p2;
// C:\temp\userdata
```

Iteracja po elementach katalogu

Iteracja po elementach katalogu może zostać zrealizowana przy pomocy ***iteratorów katalogów*** oraz pętli.

Przykład:

```
auto path = fs::current_path() / "temp”;

cout << "\nZawartość ścieżki: " << path << endl;
for(const fs::directory_entry& dir_entry : fs::directory_iterator(path))
{
    cout << dir_entry.path() << endl;
}
```

Modyfikacja stuktury plików

Biblioteka <filesystem> zawiera zbiór funkcji umożliwiających modyfikację struktury plików:

- kopiowanie - std::filesystem::copy()
- usuwanie - std::filesystem::remove() i std::filesystem::remove_all()
- tworzenie katalogów - std::filesystem::create_directory() i std::filesystem::create_directories()
- tworzenie linków - std::filesystem::create_symlink() i std::filesystem::create_directory_symlink()
- zmiana nazwy - std::filesystem::rename()

Przykład:

```
#include <cstdlib> // system()
#include <iostream>
#include <fstream> // std::ofstream
#include <filesystem>

namespace fs = std::filesystem;

int main()
{
    fs::create_directories("sandbox/dir/subdir”); // tworzy katalogi podane w parametrze
    std::ofstream("sandbox/file1.txt”).put('a’); // tworzy plik i zapisuje tam znak 'a’
    fs::copy("sandbox/file1.txt”, "sandbox/file2.txt”); // kopiuje plik
    fs::copy("sandbox/dir”, "sandbox/dir2”); // kopiuje katalog bez podfolderów (nierekurencyjnie)
    fs::copy("sandbox”, "sandbox_copy” , fs::copy_options::recursive); // kopiuje katalog z podfolderami (rekurencyjnie)
    static_cast<void>(std::system("tree”)); // funkcja wyświetlająca w konsoli strukturę plików
    fs::rename("sandbox/file1.txt”, "sandbox/plik1.txt”); // zmienia nazwę pliku
    fs::remove_all("sandbox”); // usuwa całą zawartość ścieżki podanej w parametrze
    fs::remove_all("sandbox_copy”); // usuwa całą zawartość ścieżki podanej w parametrze
}
```