Classes

Let's take a look at the different components of the filesystem library.

WE'LL COVER THE FOLLOWING

- Manipulating the permissions of a file

There are many classes encapsulating a specific aspect of the filesystem.

Class	Description
`path`	Represents a path.
`filesystem_error`	Defines an exception object.
`directory_entry`	Represents a directory entry.
`directory_iterator`	Defines a directory iterator.
`recursive_directory_iterator`	Defines a recursive directory iterator.
`file_status`	Stores information about the file.
`space_info`	Represents filesystem information.
`file_type`	Indicates the type of a file.
`perms`	Represents file access permissions.
	Represents options for the function

`perm_options`	`permissions`
`copy_options`	Represents options for the functions `copy`
`directory_options`	Represents options for the functions `directory_iterator`
`file_time_type`	Represents file time.

The various classes the filesystem

Manipulating the permissions of a file

The permissions for a file are represented by the class `std::filesystem::perms`. It is a BitmaskType and can, therefore, be manipulated by bitwise operations. The access permissions are based on POSIX.

The program from en.cppreference.com shows how you can read and manipulate the owner, group, and other (world) bits of a file.

```
#include <fstream>
#include <bitset>
#include <iostream>
#include <filesystem>

namespace fs = std::filesystem;

void printPerms(fs::perms perm){
  std::cout << ((perm & fs::perms::owner_read) != fs::perms::none ? "r" : "-")
            << ((perm & fs::perms::owner_write) != fs::perms::none ? "w" : "-")
            << ((perm & fs::perms::owner_exec) != fs::perms::none ? "x" : "-")
            << ((perm & fs::perms::group_read) != fs::perms::none ? "r" : "-")
            << ((perm & fs::perms::group_write) != fs::perms::none ? "w" : "-")
            << ((perm & fs::perms::group_exec) != fs::perms::none ? "x" : "-")
            << ((perm & fs::perms::others_read) != fs::perms::none ? "r" : "-")
            << ((perm & fs::perms::others_write) != fs::perms::none ? "w" : "-")
            << ((perm & fs::perms::others_exec) != fs::perms::none ? "x" : "-")
            << std::endl;
}


int main(){

    std::ofstream("rainer.txt");

    std::cout << "Initial file permissions for a file: ";
```

```
    printPerms(fs::status("rainer.txt").permissions());

    fs::permissions("rainer.txt", fs::perms::add_perms |
                            fs::perms::owner_all | fs::perms::group_all);
    std::cout << "Adding all bits to owner and group:  ";
    printPerms(fs::status("rainer.txt").permissions());

    fs::permissions("rainer.txt", fs::perms::remove_perms |
                           fs::perms::owner_write | fs::perms::group_write | fs::perms::other
    std::cout << "Removing the write bits for all:     ";
    printPerms(fs::status("rainer.txt").permissions());

    fs::remove("rainer.txt");

}
```

Permissions of a file

Thanks to the call `fs::status("rainer.txt").permissions()`,

1. I get the permissions of the file `rainer.txt` and can display them in the function `printPerms`
2. By setting the type `std::filesystem::add_perms`, I can add permissions to the owner and the group of the file
3. Alternatively, I can set the constant `std::filesystem::remove_perms` for removing permissions (3).

```
Initial file permissions for a file: rw-r--r--
Adding all bits to owner and group:  rwxrwxr--
Removing the write bits for all:     r-xr-xr--
```