File types

There is a wide variety of checks which we can use to get information about a file and the filesystem. They have been explained in this lesson.

WE'LL COVER THE FOLLOWING

- Getting the type of a file

By using the following predicates, you can easily ask for the type of file.

{title="The file types of the filesystem"}

file types	**Description**
`is_block_file`	Checks if the path refers to a block file.
`is_character_file`	Checks if the path refers to a character file.
`is_directory`	Checks if the path refers to a directory.
`is_empty`	Checks if the path refers to an empty file or directory.
`is_fifo`	Checks if the path refers to a named pipe.
`is_other`	Checks if the path refers to another file.

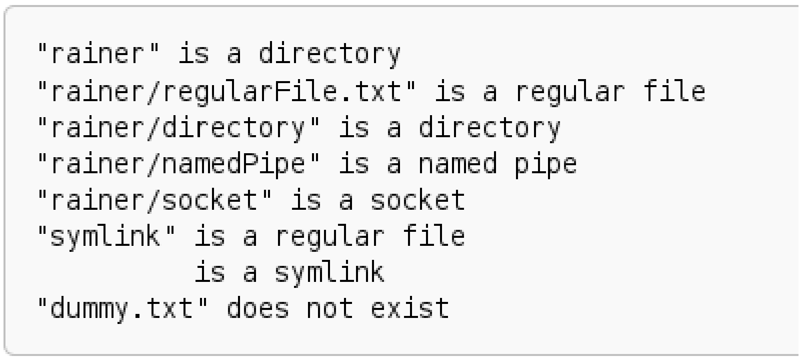
`is_regular_file`	Checks if the path refers to a regular file.
`is_socket`	Checks if the path refers to an IPC socket.
`is_symlink`	Checks if the path refers to a symbolic link.
`status_known`	Checks if the file status is known.

Getting the type of a file

The predicates give you the information on the type of a file. More than one predicate may be right for one file. In the next example, a symbolic link referencing a regular file is both: a regular file and a symbolic link.

```
#include <cstdio>
#include <cstring>
#include <unistd.h>
#include <filesystem>
#include <iostream>
#include <fstream>
#include <sys/socket.h>
#include <sys/stat.h>
#include <sys/un.h>

namespace fs = std::filesystem;

void printStatus(const fs::path& path_){
    std::cout << path_;
    if(!fs::exists(path_)) std::cout << " does not exist";
    else{
        if(fs::is_block_file(path_)) std::cout << " is a block file\n";
        if(fs::is_character_file(path_)) std::cout << " is a character device\n";
        if(fs::is_directory(path_)) std::cout << " is a directory\n";
        if(fs::is_fifo(path_)) std::cout << " is a named pipe\n";
        if(fs::is_regular_file(path_)) std::cout << " is a regular file\n";
        if(fs::is_socket(path_)) std::cout << " is a socket\n";
        if(fs::is_symlink(path_)) std::cout << "         is a symlink\n";
    }
}

int main(){

    fs::create_directory("rainer");
    printStatus("rainer");
    std::ofstream("rainer/regularFile.txt");
    printStatus("rainer/regularFile.txt");


    fs::create_directory("rainer/directory");
    printStatus("rainer/directory");

    mkfifo("rainer/namedPipe", 0644);
    printStatus("rainer/namedPipe");

    struct sockaddr_un addr;
    addr.sun_family = AF_UNIX;
    std::strcpy(addr.sun_path, "rainer/socket");
    int fd = socket(PF_UNIX, SOCK_STREAM, 0);
    bind(fd, (struct sockaddr*)&addr, sizeof addr);
    printStatus("rainer/socket");

    fs::create_symlink("rainer/regularFile.txt", "symlink");
    printStatus("symlink");

    printStatus("dummy.txt");

    fs::remove_all("rainer");

}
```

Type of a file



```
"rainer" is a directory
"rainer/regularFile.txt" is a regular file
"rainer/directory" is a directory
"rainer/namedPipe" is a named pipe
"rainer/socket" is a socket
"symlink" is a regular file
          is a symlink
"dummy.txt" does not exist
```