Understanding Resource Acquisition is Initialization (RAII)

It's not just for C++

By [David Bolton](http://cplus.about.com/bio/David-Bolton-20620.htm),

RAII stands for Resource Acquisition Is Initialization. it's a useful way of handling resources, such as memory, fonts, files etc to ensure that they are released when no longer needed or when a process terminates due to an error. Resources are generally tied to objects. A file handling class will probably require a file handle to access a file, for opening, reading or writing the file etc. Once the object is finished with the file, the resource must be released.

The basic problem RAII solves is that if the file handling object terminates say due to an exception then the resource must be freed if it has been acquired. After an exception in C++, the only code that can run is the object's destructor. Constructor's can fail as well so it makes sense to return an exception.

The file class manages the file handle and relinquishes it in the destructor.

#include <cstdio>
#include <stdexcept> // std::runtime\_error

class file {
public:
    file (const char\* filename)
        : file\_(std::fopen(filename, "w+")) {
        if (!file\_) {
            throw std::runtime\_error("file open failure") ;
        }
    }

    ~file() {
        if (std::fclose(file\_)) {
// an error?
        }
    }

    void write (const char\* str) {
        if (EOF == std::fputs(str, file\_)) {
            throw std::runtime\_error("file write failure") ;
        }
    }

private:
    std::FILE\* file\_;

    file (const file &) ;
    file & operator= (const file &) ;
};

int main(int arc,char \* argv[]) {
    file logfile("c:\\fred.txt") ; //
    logfile.write("hello world!") ;
return 0;
}