.TH Project 3
.SH Name
Project 3 \- Wad File Systems

.SH Synopsis
The goal of this project is to implement a userspace filesystem daemon with FUSE,
providing
read-only access with files and directories using the provided WAD data.

.SH Description-
.PP

.PP
.B Wad.h
.PP

.B class Wad
This section contains the declaration of Wad class functions (discussed below).
Additionally, it declares variables such as a buffer that is filled with various
file system attributes, the magic string (defined in `loadWad()`), and a vector for
keeping track of descriptor elements.

.B class Descriptors
The class keeps track of various descriptor attributes, such as an element's offset,
length, and name.

.B Wad.cpp
.PP

.B loadWad\-
The file is loaded and read into the Wad class buffer with error checking. The Wad
data is then recorded, along with the number of descriptors and each offset, for the
purpose of iterating through each descriptor and copying the information obtained
into the class descriptor offset, length, and name. These elements are then added to
the descriptor vector. The file structure is created by iterating through each
descriptor and checking whether a map marker is reached (either with "_END",
"_START", or some variation of "E#M#"), and adding content accordingly. Otherwise,
content is assumed and added to the descriptor vector.

.B string getMagic\-
Magic is returned

.B bool isContent\-
Calls isDirectory() and returns true if false.

.B bool isDirectory\-
This code returns true if the path is "/", or if the size of the path is zero. A
size of zero is interpreted as false, which would indicate a directory and return
true.

.B int getSize\-
Iterates through the elements to match path into a descriptor object. This is done
simply to have access to the size of the descriptor object to be returned.

.B int getContents\-
Checked if path is content. If yes, the Wad object is copied into buffer. Length is returned.

.B int getDirectory-
If path is a directory, the range of directories are parsed through and pushed onto
the vector. Count is returned.

.B wadfs.cpp
.PP

.B Fuse
Fuse was implemented using #ifndef FUSE_USE_VERSION and #define FUSE_USE_VERSION 26
along with compiling my wadfs which also stated the version in usage. Fuse was also
used in myFuse.(getattr, open, read, readdir, opendir, release, releasedir.

.B static int getattr_callback\-
If path is a directory, then dr-xr-xr-x permissions set. Otherwise,
- r--r--r-- permissions set.

.B static int open_callback\-
If path is content, then success is returned.

.B static int read_callback\-
If path is content, getContents() is returned.

.B static int opendir_callback\-
Success is returned.

.B static int releasedir_callback\-
Iterate through the range of directories and fill buffer.

.B int main-
This code checks for errors in argc and loads the wad file into the wadfs object. It

then assigns the argument parameters to a buffer. These, along with the argument
count and fuse operations, are used as parameters when calling fuse_main().

.SH TESTING
.PP
Built and ran wad_dump.cpp against my library
.PP
Tested FUSE and built wadfs.cpp against my library.

.SH BUGS
Issue with DOOM1.WAD with Dir S as it would not entry.
Though not necessary, but there are 70 (bytes) of memory leak.

.SH LINK
https://youtu.be/AqSoE0dYsWA

.SH REFERENCES/CITATIONS
[1]https://engineering.facile.it/blog/eng/write-filesystem-fuse/

[2]https://www.cs.hmc.edu/~geoff/classes/hmc.cs135.201109/homework/fuse/fuse_doc.html

[3]https://man.openbsd.org/fuse_main.3

.SH AUTHOR
Jesse Maki