Files

**Intro**

I really had no idea as to where I should write about files within my plethora of notes, though seeing as how they are the basis for a significant part of computing, I felt it necessary to include them somewhere. I have decided to simply write about files in a separate document, which will probably end up being a shorter one (we shall see). This document will reference the Linux operating system quite a bit so if you are unfamiliar with those notes, I’d recommend reading at least a bit of that first.

**What is a File?**

I really felt the need to write this document after one day randomly having an epiphany at work. I consulted the philosophy department of my brain as I pondered the metaphysical properties of what a file is in essence. I subconsciously knew that the icon we use to represent files was not representative of what a file actually is, but then I realized I hadn’t thought about what a file actually is, rather than what it isn’t. And the answer is simple really, and you most likely guessed it: a file is simply a series of bytes representing characters stored on some media. Even further though, the text you see when you open a file is not indicative of what a file is either. We have tools called text editors which interpret the bytes a certain way. In this document, I’d like to take a look at file extensions (and what they mean), metadata, charsets, inodes, etc. Hopefully this helps you in your day-to-day life as you code or work on some project that requires some lower-level knowledge of files.

**Extensions**

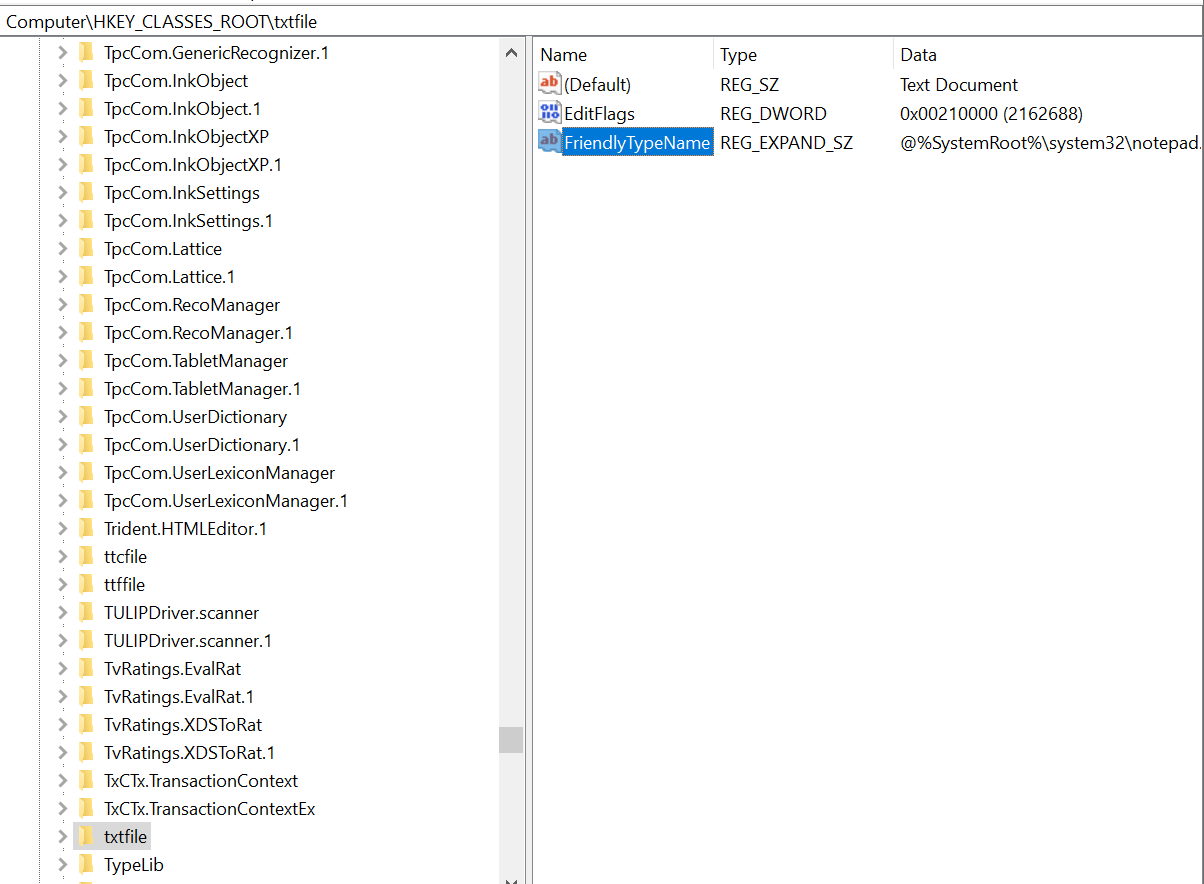
Frankly, I don’t want to get into Windows a whole lot because it sort of makes me want to vomit, however, a lot of people use it, and therefor, it can’t be ignored. Windows (and Mac, though not to the same extent), relies on file extensions to interpret the bytes that make up that file. This is denoted by the period (.) placed after the file name, separating the extension from the name. Windows associates specific file extensions with specific programs. When the user executes the application, it runs executes the program associated with the extension with the file itself as a parameter to operate upon. This mapping of extension to program can be seen in the windows registry as demonstrated below.

Figure . txt extension mapped to notepad

Here are a list of common file extensions and what exactly they are and how they operate:

windows specific files:

**.aac, .adt, .adts –** All of these files are audio files, aac being probably the most common out of the bunch.

**.dll** – A DLL file is a dynamically linked library. This term probably won’t make sense to you if you don’t have a bit of experience in C or some other low level language, but basically, a code library contains many functions and variables that a programmer is able to use for his/her software. In a language like C, we need to compile our code using a compiler (higher level languages have IDEs that do this for us). If we want to include external libraries, we statically link them using what’s called a linker (usually built into the compiler by default). Dynamically linked libraries are very common in windows and is performed during program runtime. This requires less memory space since each program can use the same dynamically linked library stored in virtual memory, but has a much larger margin of error.