

## Re: Fast File Searching

**Source:**

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.csharp.general/2005-02/0053.html>

---

**From:** Joel Thornton ([joelpt\\_at\\_gmail.com](mailto:joelpt_at_gmail.com))

**Date:** 02/11/05

Date: 11 Feb 2005 15:22:36 -0800

A database can be extremely fast, because you can load all of those files' fields into a single database table like so:

ID Filename FieldValue

=====

```
1 1001.txt dog
2 1001.txt duck
3 1001.txt fish
4 1002.txt dog
5 1002.txt mule
```

...

Then you build a database index against FieldValue. Basically the index builds a data tree of your words, so when you search for "dog", instead of searching through thousands upon thousands of files, it searches like this:

Find the start of the "d" values in the index

Find the "do" values in the index

Find the "dog" values in the index

Return every row (or in your case filename) with "dog" appearing in it

In other words, instead of looking for "dog" in thousands of places, you look for it in just one place, then get a list of WHERE it occurs instead.

This takes some extra work for you, since you would need to create a process that periodically or continuously scans those directories and files and populates your database for you. But basically, the difference here is that you are scanning all those files ONCE per file (or file change), instead of EVERY TIME somebody wants to do a new search.

It is a little hard to appreciate the speed benefit of going this route without experiencing it, but to give you an example -- once upon a time I built a website composed of thousands of pages of text, and of course wanted to search those pages at will. At first, I just searched every

file every time a search request came in. It took roughly 60 seconds per search.

After building an indexer, I could rebuild the entire index from scratch in about 5 minutes (which you don't have to do if you incrementally update the index by watching for file changes). But as a result, each search request which ran against my index took less than a second to run.

Hope that gives you an idea.

Joel