Ruby master - Feature #17001

[Feature] Dir.scan to yield dirent for efficient and composable recursive directory scaning

06/30/2020 07:57 PM - byroot (Jean Boussier)

Status:	Open
Priority:	Normal
Assignee:	
Target version:	

Description

Use case

When you need to recusrsively scan a directory, you either have to use Dir[] / Dir.glob, which is fine for small directories or simple patterns,
but can easily take several seconds to complete for large repositories or complex patterns and returns a very large array which tend to trash GC.

Or you can use Dir.each_entry / Dir.foreach recursively, but then you need to stat each entry to know wether it's a directory, or even symlink if you want to follow them.
This means one syscall per directory, and one per file and directories. This is particularly impactful on OSX where stat() is several times slower than on Linux because of various sandboxing features.

There's a typical example of this use case in Bootsnap.

Proposal

Python introduced os.scandir a few years ago for exactly this purpose. It is functionaly similar to Dir.foreach / Dir.each_child, except it yields
DirEntry instances which are a wrapper around the libc dirent struct.

I reduced the Bootsnap code into a simplified benchmark, and using os.scandir() Python scan our main repo in a bit over 1s, which 3 to 4 times faster
than Ruby can with Dir.foreach (3-4s). For comparison sake Dir['**/*.rb'] also complete in about 1s.

So I beleive that exposing a similar Dir.scan method, returning Dir::Entry instances, with methods inspired from File::Stat such as directory? would allow for more performant file system scaning
when the query is not easily expressed with a glob pattern.