This folder contains modules that help communicating with the file-system.

`files` vs `fs` and `files.path*` vs `path`

Since the Meteor tool was originally written to work on Mac OS X and Linux but now is also required to work on Windows, there has been a decision to abstract the file-system calls to `fs` and `path` modules and make them go through the `files.js` lib.

All path and files manipulations in the `tools` code assumes it is running in a unixy environment, where the path separator is `/` and the default line-break symbol is `\n`; calls like `rename` and `unlink` are atomic and the file-system always works as you expect.

The `files.js` file tries its best to simulate this behavior on Windows, converting slashes, file contents and running FS operations in a "try/sleep/repeat" loop when an `EBUSY` error is returned. Operations on Windows happen to be slower, especially moving folders and symlinking (which is done by copying the directory instead).

It is advised to use `files.readFile` and others instead of `fs.readFileSync`. The methods are Fiberized and are converted on Windows.

Also `files.pathJoin` instead of `path.join` and others to properly preserve the unixy feel of paths: `/C/Users/IEUser/AppData/Local` instead of `C:\Users\IEUser\AppData\Local`.

File watching

Since node.js doesn't ship a stable library to watch a folder on all file-systems, a wrapper is used. The wrapper checks if the native functionality works, if not (while on Windows, or a virtualized shared file-system like in VirtualBox), polling is used.

Watchset

A specific data-structure that is a set of files and directories paths observed by the file-watcher.