symlink()

Creates a stream for linking Vinyl objects to the file system.

Usage

```
const { src, symlink } = require('gulp');

function link() {
  return src('input/*.js')
    .pipe(symlink('output/'));
}

exports.link = link;
```

Signature

```
symlink(directory, [options])
```

Parameters

parameter	type	note
directory (required)	string function	The path of the output directory where symbolic links will be created. If a function is used, the function will be called with each Vinyl object and must return a string directory path.
options	object	Detailed in Options below.

Returns

A stream that can be used in the middle or at the end of a pipeline to create symbolic links on the file system. Whenever a Vinyl object is passed through the stream, it creates a symbolic link to the original file on the file system at the given directory.

Whenever a symbolic link is created on the file system, the Vinyl object will be modified.

The cwd, base, and path properties will be updated to match the created symbolic link.

The stat property will be updated to match the symbolic link on the file system.

The contents property will be set to null.

The symlink property will be added or replaced with original path.

Note: On Windows, directory links are created using junctions by default.

The useJunctions option disables this behavior.

Errors

When directory is an empty string, throws an error with the message, "Invalid symlink() folder argument. Please specify a non-empty string or a function."

When directory is not a string or function, throws an error with the message, "Invalid symlink() folder argument. Please specify a non-empty string or a function."

When directory is a function that returns an empty string or undefined, emits an error with the message, "Invalid output folder".

Options

For options that accept a function, the passed function will be called with each Vinyl object and must return a value of another listed type.

name	type	default	note
cwd	string function	process.cwd()	The directory that will be combined with any relative path to form an absolute path. Is ignored for absolute paths. Use to avoid combining directory with path.join().
dirMode	number function		The mode used when creating directories. If not set, the process' mode will be used.
overwrite	boolean function	true	When true, overwrites existing files with the same path.

name	type	default	note
relativeSymlinks	boolean function	false	When false, any symbolic links created will be absolute. Note: Ignored if a junction is being created, as they must be absolute.
useJunctions	boolean function	true	This option is only relevant on Windows and ignored elsewhere. When true, creates directory symbolic link as a junction. Detailed in Symbolic links on Windows below.

Symbolic links on Windows

When creating symbolic links on Windows, a type argument is passed to Node's fs.symlink() method which specifies the type of target being linked. The link type is set to:
'file' when the target is a regular file
'junction' when the target is a directory
'dir' when the target is a directory and the user disables the useJunctions option
If you try to create a dangling (pointing to a non-existent target) link, the link type can't be determined automatically. In these cases, behavior will vary depending on whether the dangling link is being created via symlink() or via dest().
For dangling links created via symlink(), the incoming Vinyl object represents the target, so its stats will determine the desired link type. If isDirectory() returns false then a 'file' link is created, otherwise a 'junction' or 'dir' link is created depending on the value of the useJunctions option.
For dangling links created via dest(), the incoming Vinyl object represents the link - typically loaded from disk via src(..., { resolveSymlinks: false }). In this case, the link type can't be reasonably determined and defaults to using 'file'. This may cause unexpected behavior when creating a dangling link to a directory. Avoid this scenario.