Watch and WatchOptions

webpack can watch files and recompile whenever they change. This page explains how to enable this and a couple of tweaks you can make if watching does not work properly for you.

`watch`

`boolean = false`

Turn on watch mode. This means that after the initial build, webpack will continue to watch for changes in any of the resolved files.

webpack.config.js

```
module.exports = {
  //...
  watch: true
};
```

In webpack-dev-server and webpack-dev-middleware watch mode is enabled by default.

`watchOptions`

`object`

A set of options used to customize watch mode:

webpack.config.js

```
module.exports = {
  //...
  watchOptions: {
    aggregateTimeout: 300,
    poll: 1000
  }
};
```

```
};
```

`watchOptions.aggregateTimeout`

`number = 300`

Add a delay before rebuilding once the first file changed. This allows webpack to aggregate any other changes made during this time period into one rebuild. Pass a value in milliseconds:

```
module.exports = {
  //...
  watchOptions: {
    aggregateTimeout: 600
  }
};
```

`watchOptions.ignored`

`RegExp` `anymatch`

For some systems, watching many file systems can result in a lot of CPU or memory usage. It is possible to exclude a huge folder like `node_modules` :

webpack.config.js

```
module.exports = {
  //...
  watchOptions: {
    ignored: /node_modules/
  }
};
```

It is also possible to have and use multiple anymatch patterns:

webpack.config.js

```
module.exports = {
  //...
  watchOptions: {
    ignored: ['files/**/*.js', 'node_modules']
  }
};
```

If you use `require.context`, webpack will watch your entire directory. You will need to ignore files and/or directories so that unwanted changes will not trigger a rebuild.

`watchOptions.poll`

`boolean = false` `number`

Turn on polling by passing `true`, or specifying a poll interval in milliseconds:

webpack.config.js

```
module.exports = {
  //...
  watchOptions: {
    poll: 1000 // Check for changes every second
  }
};
```

If watching does not work for you, try out this option. Watching does not work with NFS and machines in VirtualBox.

`info-verbosity`

`string: 'none' | 'info' | 'verbose'`

Controls verbosity of the lifecycle messaging, e.g. the `Started watching files...` log. Setting `info-verbosity` to `verbose` will also message to console at the beginning and the end of incremental build. `info-verbosity` is set to `info` by default.

```
webpack --watch --info-verbosity verbose
```

Troubleshooting

If you are experiencing any issues, please see the following notes. There are a variety of reasons why webpack might miss a file change.

Changes Seen But Not Processed

Verify that webpack is not being notified of changes by running webpack with the --progress flag. If progress shows on save but no files are outputted, it is likely a configuration issue, not a file watching issue.

```
webpack --watch --progress
```

Not Enough Watchers

Verify that you have enough available watchers in your system. If this value is too low, the file watcher in Webpack won't recognize the changes:

```
cat /proc/sys/fs/inotify/max_user_watches
```

Arch users, add `fs.inotify.max_user_watches=524288` to `/etc/sysctl.d/99-sysctl.conf` and then execute `sysctl --system`. Ubuntu users (and possibly others), execute: `echo fs.inotify.max_user_watches=524288 | sudo tee -a /etc/sysctl.conf && sudo sysctl -p`.

macOS fsevents Bug

On macOS, folders can get corrupted in certain scenarios. See this article.

Windows Paths

Because webpack expects absolute paths for many config options such as `__dirname + '/app/folder'` the Windows `\` path separator can break some functionality.

Use the correct separators. I.e. `path.resolve(__dirname, 'app/folder')` or `path.join(__dirname, 'app', 'folder')`.

Vim

On some machines Vim is preconfigured with the backupcopy option set to `auto`. This could potentially cause problems with the system's file watching mechanism. Switching this option to `yes` will make sure a copy of the file is made and the original one overwritten on save.

```
:set backupcopy=yes
```

Saving in WebStorm

When using the JetBrains WebStorm IDE, you may find that saving changed files does not trigger the watcher as you might expect. Try disabling the `safe write` option in the settings, which determines whether files are saved to a temporary location first before the originals are overwritten: uncheck `File > {Settings|Preferences} > Appearance & Behavior > System Settings > Use "safe write" (save changes to a temporary file first)`.