# setup_logging

## 1. Introduction

This setup function provides a clean logging setup for the `log` crate using `fern`.

- Messages with linebreaks are properly indented.
- Timestamps are only printed if they changed from timestamp of previous line.

Console only:

- "\r" at the beginning of a message overwrites previous line.
- Logging levels are colour-coded.

## 2. Table of Contents

# 3. Installation

### 3.1. Bash

1. `cargo add setup_logging --git https://github.com/9-FS/setup_logging`
2. In the created `Cargo.toml` entry, replace "version" with "tag" and overwrite the field with the desired version number.

### 3.2. Manual

1. Paste the following example `Cargo.toml` entry into your `Cargo.toml` beneath `[dependencies]`:

   ```toml
   setup_logging = { git = "https://github.com/9-FS/setup_logging", tag = "1.0.0" }
   ```

2. Overwrite the desired version number into the `tag` field. This example entry version will not be updated.

> Info
>
> Cargo does not support automatic versioning for GitHub dependencies. Manual updates are required in the `Cargo.toml` file using `tag`.

# 4. Usage

1. Execute `setup_logging` with the desired minimum log level and log filepath format according to the `chrono` crate.
2. `log` can now be used as usual with the provided macros.

### 4.1. Example

```rust
setup_logging::setup_logging(log::Level::Info, "./log/%Y-%m-%d.log");

log::debug!("debug message"); // not printed
log::info!("info message"); // printed
log::warn!("warn message"); // printed without timestamp
log::error!("error message"); // printed without timestamp
```

### 4.2. Example

```rust
setup_logging::setup_logging(log::Level::Debug, "./log/%Y-%m-%dT%H_%M.log");

log::debug!("debug message"); // printed
log::info!("info message"); // printed without timestamp
log::warn!("warn message"); // printed without timestamp
log::error!("error message"); // printed without timestamp
```