


## Logging

 Transformers has a centralized logging system, so that you can setup the verbosity of the library easily.

Currently the default verbosity of the library is `WARNING`.

To change the level of verbosity, just use one of the direct setters. For instance, here is how to change the verbosity to the `INFO` level.

```
import transformers

transformers.logging.set_verbosity_info()
```

You can also use the environment variable `TRANSFORMERS_VERBOSITY` to override the default verbosity. You can set it to one of the following: `debug`, `info`, `warning`, `error`, `critical`. For example:

```
TRANSFORMERS_VERBOSITY=error ./myprogram.py
```

Additionally, some warnings can be disabled by setting the environment variable `TRANSFORMERS_NO_ADVISORY_WARNINGS` to a true value, like `1`. This will disable any warning that is logged using `logger.warning_advice`. For example:

```
TRANSFORMERS_NO_ADVISORY_WARNINGS=1 ./myprogram.py
```

Here is an example of how to use the same logger as the library in your own module or script:

```
from transformers.utils import logging

logging.set_verbosity_info()
logger = logging.get_logger("transformers")
logger.info("INFO")
logger.warning("WARN")
```

All the methods of this logging module are documented below, the main ones are `logging.get_verbosity()` to get the current level of verbosity in the logger and `logging.set_verbosity()` to set the verbosity to the level of your choice. In order (from the least verbose to the most verbose), those levels (with their corresponding int values in parenthesis) are:

- `transformers.logging.CRITICAL` or `transformers.logging.FATAL` (int value, 50): only report the most critical errors.
- `transformers.logging.ERROR` (int value, 40): only report errors.
- `transformers.logging.WARNING` or `transformers.logging.WARN` (int value, 30): only reports error and warnings. This the default level used by the library.
- `transformers.logging.INFO` (int value, 20): reports error, warnings and basic information.
- `transformers.logging.DEBUG` (int value, 10): report all information.

By default, tqdm progress bars will be displayed during model download. `logging.disable_progress_bar()` and `logging.enable_progress_bar()` can be used to suppress or unsuppress this behavior.

## logging vs warnings

Python has two logging systems that are often used in conjunction: logging, which is explained above, and warnings, which allows further classification of warnings in specific buckets, e.g., FutureWarning for a feature or path that has already been deprecated and DeprecationWarning to indicate an upcoming deprecation.

We use both in the transformers library. We leverage and adapt logging's captureWarning method to allow management of these warning messages by the verbosity setters above.

What does that mean for developers of the library? We should respect the following heuristic:

- warnings should be favored for developers of the library and libraries dependent on transformers
- logging should be used for end-users of the library using it in every-day projects

See reference of the captureWarnings method below.

 [transformers.utils.logging.captureWarnings](#)

< source >

( capture )

Calls the captureWarnings method from the logging library to enable management of the warnings emitted by the warnings library.

Read more about this method here: <https://docs.python.org/3/library/logging.html#integration-with-the-warnings-module>

All warnings will be logged through the `py.warnings` logger.

Careful: this method also adds a handler to this logger if it does not already have one, and updates the logging level of that logger to the library's root logger.

## Base setters

 [transformers.utils.logging.set\\_verbosity\\_error](#)

< source >

( )

Set the verbosity to the ERROR level.

 [transformers.utils.logging.set\\_verbosity\\_warning](#)

< source >

( )

Set the verbosity to the WARNING level.

 [transformers.utils.logging.set\\_verbosity\\_info](#)

< source >

( )

Set the verbosity to the INFO level.

 [transformers.utils.logging.set\\_verbosity\\_debug](#)

< source >

( )

Set the verbosity to the DEBUG level.

## Other functions

 [transformers.utils.logging.get\\_verbosity](#)

< source >

( ) → int

**Returns** int

The logging level.

Return the current level for the 🤖 Transformers's root logger as an int.

🤖 Transformers has following logging levels:

- 50: `transformers.logging.CRITICAL` or `transformers.logging.FATAL`
- 40: `transformers.logging.ERROR`

- 30: transformers.logging.WARNING or transformers.logging.WARN
- 20: transformers.logging.INFO
- 10: transformers.logging.DEBUG

## transformers.utils.logging.set\_verbosity

[< source >](#)

```
( verbosity: int )
```

### Parameters

- **verbosity** (int) — Logging level, e.g., one of:
  - transformers.logging.CRITICAL or transformers.logging.FATAL
  - transformers.logging.ERROR
  - transformers.logging.WARNING or transformers.logging.WARN
  - transformers.logging.INFO
  - transformers.logging.DEBUG

Set the verbosity level for the 🤖 Transformers's root logger.

## transformers.utils.logging.get\_logger

[< source >](#)

```
( name: Optional = None )
```

Return a logger with the specified name.

This function is not supposed to be directly accessed unless you are writing a custom transformers module.

 [transformers.utils.logging.enable\\_default\\_handler](#)

< source >

( )

Enable the default handler of the HuggingFace Transformers's root logger.

 [transformers.utils.logging.disable\\_default\\_handler](#)

< source >

( )

Disable the default handler of the HuggingFace Transformers's root logger.

 [transformers.utils.logging.enable\\_explicit\\_format](#)

< source >

( )

Enable explicit formatting for every HuggingFace Transformers's logger. The explicit formatter is as follows:

```
[LEVELNAME | FILENAME | LINE NUMBER] TIME >> MESSAGE
```

All handlers currently bound to the root logger are affected by this method.

 [transformers.utils.logging.reset\\_format](#)

< source >

( )

Resets the formatting for HuggingFace Transformers's loggers.

All handlers currently bound to the root logger are affected by this method.

 [transformers.utils.logging.enable\\_progress\\_bar](#)

< source >

( )

Enable tqdm progress bar.

 [transformers.utils.logging.disable\\_progress\\_bar](#)

< source >

( )

Disable tqdm progress bar.

<> [Update](#) on GitHub



← Keras callbacks

Models →