

# Verbose Logging in WebSphere Liberty Server

To address your query about the verbose log option for the server command in WebSphere Liberty Profile (WLP), let's clarify what the `server` command does and how verbosity can be controlled.

The `server` command in WLP is used to manage server instances, with actions like `start`, `stop`, `run`, or `status`. For example:

- `server start <serverName>` starts the server in the background, with output redirected to log files (e.g., `logs/console.log`).
- `server run <serverName>` starts the server in the foreground, displaying output directly in the terminal.

Your question asks about a “verbose log option” for the `server` command, which could imply either more detailed output from the command itself or more detailed logging from the server it manages. After exploring the options, it's clear that the `server` command does not have a direct flag like `--verbose` or `-v` to increase its own output verbosity. Instead, verbosity pertains to the server's logging behavior, which can be influenced when invoking the command.

## Enabling Verbose Logging

In WLP, the verbosity of logs is controlled through the server's logging configuration, not directly via a `server` command flag. Here's how you can enable verbose logging:

**1. Configure Logging in `server.xml`** The primary way to enable verbose logging is by adjusting the `<logging>` element in the server's `server.xml` file, typically located in `<WLP_HOME>/usr/servers/<serverName>/`. You can set a detailed trace specification to increase log verbosity. For example:

```
<logging traceSpecification="*=all" />
```

- `*=all` enables all trace points, making the logs extremely verbose (useful for debugging).
- For more targeted verbosity, you can specify components, e.g., `*=info:com.example.*=debug`, setting the default level to `info` but `debug` for the `com.example` package.

Other useful attributes include: - `logLevel`: Sets the general log level (e.g., `INFO`, `DEBUG`, `TRACE`). - `consoleLogLevel`: Controls the level of messages written to `console.log` or the terminal (e.g., `DEBUG` or `TRACE`).

Example with a finer console level:

```
<logging consoleLogLevel="DEBUG" traceSpecification="*=audit" />
```

When you run `server start`, the logs (including verbose output) go to `logs/console.log`. With `server run`, this verbose output appears directly in your terminal.

**2. Use Environment Variables** You can also control log verbosity via environment variables, which override or complement `server.xml` settings. For instance, set the `WLP_LOGGING_CONSOLE_LOGLEVEL` variable before running the `server` command:

```
export WLP_LOGGING_CONSOLE_LOGLEVEL=DEBUG
server start <serverName>
```

- This sets the console log level to `DEBUG`, ensuring more detailed output.
- Valid levels include `INFO`, `DEBUG`, `FINE`, `FINER`, `FINEST`, etc.

For `server run`, this will make the terminal output more verbose; for `server start`, it affects `console.log`.

**3. JVM-Specific Verbose Options** If you're interested in specific verbose outputs like garbage collection logs, you can add JVM options (though this is server runtime-related, not the `server` command itself). Add `-verbose:gc` to the `jvm.options` file in `<WLP_HOME>/usr/servers/<serverName>/`, or include it via `server.xml`:

```
<jvmEntries verbose="gc" />
```

This generates GC-related logs, typically in `console.log` or the terminal with `server run`.

## How This Relates to the `server` Command

- **No Direct `--verbose` Option:** The `server` command itself (e.g., `server start`) doesn't have a `--verbose` or similar flag to detail its execution steps beyond basic status messages (e.g., "Server started").
- **Server Logging Controls Verbosity:** The verbosity you see depends on the server's logging configuration. Use `server run` to observe logs live in the terminal, or check log files after `server start`.

For example: - With `<logging traceSpecification="*=all"/>` in `server.xml`, running `server run <serverName>` floods the terminal with detailed logs. - With `server start <serverName>`, the same verbosity appears in `logs/console.log`.

## Recommendation

To achieve verbose logging when using the `server` command: 1. Edit `server.xml` to include a detailed `traceSpecification` (e.g., `*=all` for maximum verbosity or a specific level like `*=debug`). 2. Alternatively, set `WLP_LOGGING_CONSOLE_LOGLEVEL=DEBUG` in your environment. 3. Use `server run` to see the verbose output directly, or `server start` and check `logs/console.log`.

This approach ensures you get the detailed logs you need, tailored to your debugging or monitoring requirements.

If you meant something more specific (e.g., command execution verbosity rather than server logs), feel free to clarify! Otherwise, configuring the server's logging as described is the standard way to enable verbose logging with the `server` command in WLP.

---

Note: Be careful with the `logs` directory. It contains logs, such as trace files.