CHAPTER

EIGHT

LOGGING HANDLER

Rich supplies a *logging handler* which will format and colorize text written by Python's logging module.

Here's an example of how to set up a rich logger:

```python
import logging
from rich.logging import RichHandler

FORMAT = "%(message)s"
logging.basicConfig(
    level="NOTSET", format=FORMAT, datefmt="[%X]", handlers=[RichHandler()]
)

log = logging.getLogger("rich")
log.info("Hello, World!")
```

Rich logs won't render *Console Markup* in logging by default as most libraries won't be aware of the need to escape literal square brackets, but you can enable it by setting `markup=True` on the handler. Alternatively you can enable it per log message by supplying the `extra` argument as follows:

```python
log.error("[bold red blink]Server is shutting down![/]", extra={"markup": True})
```

Similarly, the highlighter may be overridden per log message:

```python
log.error("123 will not be highlighted", extra={"highlighter": None})
```

8.1 Handle exceptions

The *`RichHandler`* class may be configured to use Rich's *`Traceback`* class to format exceptions, which provides more context than a built-in exception. To get beautiful exceptions in your logs set `rich_tracebacks=True` on the handler constructor:

```python
import logging
from rich.logging import RichHandler

logging.basicConfig(
    level="NOTSET",
    format="%(message)s",
    datefmt="[%X]",
    handlers=[RichHandler(rich_tracebacks=True)]
)
```

(continues on next page)