



define_logger.py

```
import structlog

def get_logger(cls: str):
    return structlog.get_logger().bind(cls=cls)
```

Define logger



call_logger.py

```
logger = logger_utils.get_logger(__name__)

logger.info(...)
logger.warning(...)
logger.error(...)
```

Call logger



Decoding ML

 decodingml

How it
looks like:
out of the
box

```
> python show_off.py
```

```
2023-12-29 08:06:42 [debug    ] debugging is hard
2023-12-29 08:06:42 [info     ] informative!
2023-12-29 08:06:42 [warning  ] uh-uh!
2023-12-29 08:06:42 [error    ] omg
2023-12-29 08:06:42 [critical ] wtf
2023-12-29 08:06:42 [exception] poor me
```

```
[some_logger] a_list=[1, 2, 3]
[some_logger] some_key=some_value
[some_logger]
[some_logger] a_dict={'a': 42, 'b': 'foo'}
[some_logger] what=SomeClass(x=1, y='z')
[another_logger]
```

```
Traceback (most recent call last)
/Users/hynek/FOSS/structlog/show_off.py:36 in make_call_stack_more_impressive
```

```
33 def make_call_stack_more_impressive():
34     try:
35         d = {"x": 42}
> 36         print(SomeClass(d["y"], "foo"))
37     except Exception:
38         log2.exception("poor me")
39         log.info("all better now!", stack_info=True)
```

```
locals
d = {'x': 42}
```

```
KeyError: 'y'
```

```
Traceback (most recent call last):
```

```
File "/Users/hynek/FOSS/structlog/show_off.py", line 36, in make_call_stack_more_impressive
    print(SomeClass(d["y"], "foo"))
                    ~^^^^^
```

```
KeyError: 'y'
```

```
2023-12-29 08:06:42 [info     ] all better now!
```

```
[some_logger]
```

```
Stack (most recent call last):
```

```
File "/Users/hynek/FOSS/structlog/show_off.py", line 42, in <module>
```

```
    make_call_stack_more_impressive()
```

```
File "/Users/hynek/FOSS/structlog/show_off.py", line 39, in make_call_stack_more_impressive
```

```
    log.info("all better now!", stack_info=True)
```