<http://python.jobbole.com/87300/>

<http://python.jobbole.com/86887/>

logging框架中主要由四个部分组成：

* Loggers: 可供程序直接调用的接口
* Handlers: 决定将日志记录分配至正确的目的地
* Filters: 提供更细粒度的日志是否输出的判断
* Formatters: 制定最终记录打印的格式布局

每个logger都有一个日志的级别。logging中定义了如下级别

| **Level** | **Numeric value** |
| --- | --- |
| NOTSET | 0 |
| DEBUG | 10 |
| INFO | 20 |
| WARNING | 30 |
| ERROR | 40 |
| CRITICAL | 50 |

当一个logger收到日志信息后先判断是否符合level，如果决定要处理就将信息传递给Handlers进行处理。

#### **Formatter日志格式**

Formatter对象定义了log信息的结构和内容，构造时需要带两个参数：

* 一个是格式化的模板fmt，默认会包含最基本的level和 message信息
* 一个是格式化的时间样式datefmt，默认为 2003-07-08 16:49:45,896 (%Y-%m-%d %H:%M:%S)

fmt中允许使用的变量可以参考下表。

* ****%(name)s**** Logger的名字
* ****%(levelno)s**** 数字形式的日志级别
* ****%(levelname)s**** 文本形式的日志级别
* ****%(pathname)s**** 调用日志输出函数的模块的完整路径名，可能没有
* ****%(filename)s**** 调用日志输出函数的模块的文件名
* ****%(module)s**** 调用日志输出函数的模块名|
* ****%(funcName)s**** 调用日志输出函数的函数名|
* ****%(lineno)d**** 调用日志输出函数的语句所在的代码行
* ****%(created)f**** 当前时间，用UNIX标准的表示时间的浮点数表示|
* ****%(relativeCreated)d**** 输出日志信息时的，自Logger创建以来的毫秒数|
* ****%(asctime)s**** 字符串形式的当前时间。默认格式是“2003-07-08 16:49:45,896”。逗号后面的是毫秒
* ****%(thread)d**** 线程ID。可能没有
* ****%(threadName)s**** 线程名。可能没有
* ****%(process)d**** 进程ID。可能没有
* ****%(message)s**** 用户输出的消息

#### **Handler 日志处理器**

最常用的是StreamHandler和FileHandler, Handler用于向不同的输出端打log。  
Logging包含很多handler, 可能用到的有下面几种

* ****StreamHandler**** instances send error messages to streams (file-like objects).
* ****FileHandler**** instances send error messages to disk files.
* ****RotatingFileHandler**** instances send error messages to disk files, with support for maximum log file sizes and log file rotation.
* ****TimedRotatingFileHandler**** instances send error messages to disk files, rotating the log file at certain timed intervals.
* ****SocketHandler**** instances send error messages to TCP/IP sockets.
* ****DatagramHandler**** instances send error messages to UDP sockets.
* ****SMTPHandler**** instances send error messages to a designated email address.

logging的配置大致有下面几种方式。

1. 通过代码进行完整配置，参考开头的例子，主要是通过getLogger方法实现。
2. 通过代码进行简单配置，下面有例子，主要是通过basicConfig方法实现。
3. 通过配置文件，下面有例子，主要是通过 logging.config.fileConfig(filepath)

##### **logging.basicConfig**

basicConfig()提供了非常便捷的方式让你配置logging模块并马上开始使用，可以参考下面的例子。

|  |  |
| --- | --- |
| 2  3  4  5  6  7 | logging.basicConfig(      level=logging.NOTSET,      format=LOGGING\_FORMAT,      datefmt=DATE\_FORMAT,      filename='log/test.log',      filemode='a'  ) |

