Table of Contents

传智播客 www.itcast.cn

更多资源，百度“黑马软件测试学习路线图”bbs.itheima.com/thread-405757-1-1.html
黑马程序员-软件测试

Web自动化测试课程

序号	章节	知识点
1	第一章 Web自动化入门	1. 认识自动化及自动化测试 2. 自动化测试工具选择 3. 环境搭建
2	第二章 Selenium-API操作	1. 元素定位方式 2. 元素和浏览器的操作方法 3. 鼠标和键盘操作 4. 元素等待 5. HTML特殊元素处理 6. 窗口截图 7. 验证码处理
3	第三章 UnitTest框架	1. UnitTest基本使用 2. UnitTest断言 3. 参数化 4. 生成HTML测试报告
4	第四章 PO模式	1. 方法封装 2. PO模式介绍 3. PO模式实战
5	第五章 数据驱动	1. JSON读写 2. 数据驱动介绍 3. 数据驱动实战
6	第六章 日志收集	1. 日志相关概念 2. 日志的基本方法 3. 日志的高级方法
7	第七章 项目实战	1. 自动化测试流程 2. 项目实战演练

课程目标

1. 掌握使用Selenium进行Web自动化测试的流程和方法，并且能够完成自动化测试脚本的编写。
2. 掌握如何通过UnitTest管理用例脚本，并生成HTML测试报告。
3. 掌握使用PO模式来设计自动化测试代码的架构。
4. 掌握使用数据驱动来实现自动化测试代码和测试数据的分离。
5. 掌握使用logging来实现日志的收集。

更多资源，百度“黑马软件测试学习路线图”bbs.itheima.com/thread-405757-1-1.html
黑马程序员-软件测试

第6章-日志收集

目标

1. 理解日志的相关概念
2. 掌握日志的基本用法
3. 掌握日志的高级用法

传智播客 www.itcast.cn

更多资源，百度“黑马软件测试学习路线图”bbs.itheima.com/thread-405757-1-1.html
黑马程序员-软件测试

日志相关概念

目标

1．了解日志的概念
2．理解日志的作用
3．掌握常见的日志级别

---

1. 日志

概念：日志就是用于记录系统运行时的信息，对一个事件的记录；也称为Log。

1.1 日志的作用

- 调试程序
- 了解系统程序运行的情况，是否正常
- 系统程序运行故障分析与问题定位
- 用来做用户行为分析和数据统计

1.2 日志级别

> 思考：是否系统记录的所有日志信息的重要性都一样？

日志级别：是指日志信息的优先级、重要性或者严重程度

常见的日志级别

日志级别	描述
DEBUG	调试级别，打印非常详细的日志信息，通常用于对代码的调试
INFO	信息级别，打印一般的日志信息，突出强调程序的运行过程
WARNING	警告级别，打印警告日志信息，表明会出现潜在错误的情形，一般不影响软件的正常使用
ERROR	错误级别，打印错误异常信息，该级别的错误可能会导致系统的一些功能无法正常使用
CRITICAL	严重错误级别，一个严重的错误，这表明系统可能无法继续运行

说明

- 上面列表中的日志级别是从上到下依次升高的，即：DEBUG < INFO < WARNING < ERROR < CRITICAL；
- 当为程序指定一个日志级别后，程序会记录所有日志级别大于或等于指定日志级别的日志信息，而不是仅仅记录指定级别的日志信息；
- 一般建议只使用DEBUG、INFO、WARNING、ERROR这四个级别

更多资源，百度“黑马软件测试学习路线图”bbs.itheima.com/thread-405757-1-1.html
黑马程序员-软件测试

传智播客 www.itcast.cn

更多资源，百度“黑马软件测试学习路线图”bbs.itheima.com/thread-405757-1-1.html
黑马程序员-软件测试

日志的基本用法

目标

1. 掌握如何设置日志级别
2. 掌握如何设置日志格式
3. 掌握如何将日志信息输出到文件中

---

1. logging模块

Python中有一个标准库模块logging可以直接记录日志

1.1 基本用法

```
import logging

logging.debug("这是一条调试信息")
logging.info("这是一条普通信息")
logging.warning("这是一条警告信息")
logging.error("这是一条错误信息")
logging.critical("这是一条严重错误信息")
```

1.1 设置日志级别

logging中默认的日志级别为WARNING，程序中大于等于该级别的日志才能输出，小于该级别的日志不会被打印出来。

设置日志级别

```
logging.basicConfig(level=logging.DEBUG)
```

如何选择日志级别

- 在开发环境和测试环境中，为了尽可能详细的查看程序的运行状态来保证上线后的稳定性，可以使用DEBUG或INFO级别的日志获取详细的日志信息，这是非常耗费机器性能的。
- 在生产环境中，通常只记录程序的异常信息、错误信息等（设置成WARNING或ERROR级别），这样既可以减小服务器的I/O压力，也可以提高获取错误日志信息的效率和方便问题的排查。

1.2 设置日志格式

默认的日志的格式为：
日志级别:Logger名称:日志内容

自定义日志格式：

更多资源，百度“黑马软件测试学习路线图”bbs.itheima.com/thread-405757-1-1.html
黑马程序员-软件测试

```
logging.basicConfig(format="%(levelname)s:%(name)s:%(message)s")
```

format参数中可能用到的格式化信息：

占位符	描述
%(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”
%(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

示例代码：

```
import logging

fmt = '%(asctime)s %(levelname)s [%(name)s] [%(filename)s(%(funcName)s:%(lineno)d)] - %(message)s'
logging.basicConfig(level=logging.INFO, format=fmt)

logging.debug("调试")
logging.info("信息")
logging.warning("警告")
logging.error("错误")
```

1.3 将日志信息输出到文件中

默认情况下Python的logging模块将日志打印到了标准输出中（控制台）

将日志信息输出到文件中：

```
logging.basicConfig(filename="a.log")
```

示例代码：

```
import logging
```


更多资源，百度“黑马软件测试学习路线图”bbs.itheima.com/thread-405757-1-1.html
黑马程序员-软件测试

```
fmt = '%(asctime)s %(levelname)s [%(name)s] [%(filename)s(%(funcName)s:%(lineno)d)] - %(message)s'
logging.basicConfig(filename="a.log", level=logging.INFO, format=fmt)

logging.debug("调试")
logging.info("信息")
logging.warning("警告")
logging.error("错误")
```

传智播客 www.itcast.cn

更多资源，百度“黑马软件测试学习路线图”bbs.itheima.com/thread-405757-1-1.html
黑马程序员-软件测试

日志的高级用法

目标

```
1. 了解logging日志模块四大组件
2. 掌握如何讲日志输出到多个Handler中
```

> 思考：
>
> 1. 如何将日志信息同时输出到控制台和日志文件中？
> 2. 如何将不同级别的日志输出到不同的日志文件中？
> 3. 如何解决日志文件过大的问题？

1. logging日志模块四大组件

组件名称	类名	功能描述
日志器	Logger	提供了程序使用日志的入口
处理器	Handler	将logger创建的日志记录发送到合适的目的输出
格式器	Formatter	决定日志记录的最终输出格式
过滤器	Filter	提供了更细粒度的控制工具来决定输出哪条日志记录，丢弃哪条日志记录

logging模块就是通过这些组件来完成日志处理的

1.1 组件之间的关系

- 日志器（logger）需要通过处理器（handler）将日志信息输出到目标位置，如：文件、sys.stdout、网络等；
- 不同的处理器（handler）可以将日志输出到不同的位置；
- 日志器（logger）可以设置多个处理器（handler）将同一条日志记录输出到不同的位置；
- 每个处理器（handler）都可以设置自己的格式器（formatter）实现同一条日志以不同的格式输出到不同的地方。
- 每个处理器（handler）都可以设置自己的过滤器（filter）实现日志过滤，从而只保留感兴趣的日志；

简单点说就是：日志器（logger）是入口，真正干活儿的是处理器（handler），处理器（handler）还可以通过过滤器（filter）和格式器（formatter）对要输出的日志内容做过滤和格式化等处理操作。

1.2 Logger类

Logger对象的任务：

- 向程序暴露记录日志的方法
- 基于日志级别或Filter对象来决定要对哪些日志进行后续处理
- 将日志消息传送给所有感兴趣的日志handlers

更多资源，百度“黑马软件测试学习路线图”bbs.itheima.com/thread-405757-1-1.html
黑马程序员-软件测试

如何创建**Logger**对象

```
logger = logging.getLogger()
logger = logging.getLogger("myLogger")
```

logging.getLogger()方法有一个可选参数name，该参数表示将要返回的日志器的名称标识，如果不提供该参数，则返回root日志器对象。 若以相同的name参数值多次调用getLogger()方法，将会返回指向同一个logger对象的引用。

Logger常用的方法

方法	描述
logger.debug() logger.info() logger.warning() logger.error() logger.critical()	打印日志
logger.setLevel()	设置日志器将会处理的日志消息的最低严重级别
logger.addHandler()	为该logger对象添加一个handler对象
logger.addFilter()	为该logger对象添加一个filter对象

1.3 **Handler**类

Handler对象的作用是将消息分发到handler指定的位置，比如：控制台、文件、网络、邮件等。 Logger对象可以通过addHandler()方法为自己添加多个handler对象。

如何创建**Handler**对象

在程序中不应该直接实例化和使用Handler实例，因为Handler是一个基类，它只定义了Handler应该有的接口。 应该使用Handler实现类来创建对象，logging中内置的常用的Handler包括：

Handler	描述
logging.StreamHandler	将日志消息发送到输出到Stream，如std.out, std.err或任何file-like对象。
logging.FileHandler	将日志消息发送到磁盘文件，默认情况下文件大小会无限增长
logging.handlers.RotatingFileHandler	将日志消息发送到磁盘文件，并支持日志文件按大小切割
logging.hanlders.TimedRotatingFileHandler	将日志消息发送到磁盘文件，并支持日志文件按时间切割
logging.handlers.HTTPHandler	将日志消息以GET或POST的方式发送给一个HTTP服务器
logging.handlers.SMTPHandler	将日志消息发送给一个指定的email地址

Handler常用的方法

方法	描述
handler.setLevel()	设置handler将会处理的日志消息的最低严重级别

更多资源，百度“黑马软件测试学习路线图”bbs.itheima.com/thread-405757-1-1.html
黑马程序员-软件测试

handler.setFormatter()	为handler设置一个格式器对象
handler.addFilter()	为handler添加一个过滤器对象

1.4 Formatter类

Formatter对象用于配置日志信息的格式。

如何创建Formatter对象

```
formatter = logging.Formatter(fmt=None, datefmt=None, style='%')
    fmt：指定消息格式化字符串，如果不指定该参数则默认使用message的原始值
    datefmt：指定日期格式字符串，如果不指定该参数则默认使用"%Y-%m-%d %H:%M:%S"
    style：Python 3.2新增的参数，可取值为 '%', '{'和 '$'，如果不指定该参数则默认使用'%'
```

---

2. 将日志信息同时输出到控制台和文件中

实现步骤分析

```
1.创建日志器对象
2.创建控制台处理器对象
3.创建文件处理器对象
4.创建格式化器对象
5.把格式化器添加到处理器中
6.把处理器添加到日志器中
```

定义日志格式

```
fmt = '%(asctime)s %(levelname)s [%(name)s] [%(filename)s(%(funcName)s:%(lineno)d)] - %(message)s'
formatter = logging.Formatter(fmt)
```

把日志输出到控制台

```
logger = logging.getLogger()
sh = logging.StreamHandler()
sh.setFormatter(formatter)
logger.addHandler(sh)
```

把日志输出到文件中

```
fh = logging.FileHandler("./b.log")
fh.setFormatter(formatter)
logger.addHandler(fh)
```

---

3. 每日生成一个日志文件

定义Handler对象

更多资源，百度“黑马软件测试学习路线图”bbs.itheima.com/thread-405757-1-1.html
黑马程序员-软件测试

```
fh = logging.handlers.TimedRotatingFileHandler(filename, when='h', interval=1, backupCount=0)
    将日志信息记录到文件中，以特定的时间间隔切换日志文件。
    filename: 日志文件名
    when: 时间单位，可选参数
        S - Seconds
        M - Minutes
        H - Hours
        D - Days
        midnight - roll over at midnight
        W{0-6} - roll over on a certain day; 0 - Monday
    interval: 时间间隔
    backupCount: 日志文件备份数量。如果backupCount大于0，那么当生成新的日志文件时，
                 将只保留backupCount个文件，删除最老的文件。
```

示例代码：

```python
import logging.handlers

logger = logging.getLogger()
logger.setLevel(logging.DEBUG)

# 日志格式
fmt = "%(asctime)s %(levelname)s [%(filename)s(%(funcName)s:%(lineno)d)] - %(message)s"
formatter = logging.Formatter(fmt)

# 输出到文件，每日一个文件
fh = logging.handlers.TimedRotatingFileHandler("./a.log", when='MIDNIGHT', interval=1, backupCount=3)
fh.setFormatter(formatter)
fh.setLevel(logging.INFO)
logger.addHandler(fh)
```

传智播客 www.itcast.cn

更多资源，百度“黑马软件测试学习路线图”bbs.itheima.com/thread-405757-1-1.html
黑马程序员-软件测试