本服务为其他服务提供日志管理的功能，主要包含以下三个方面：

- 接收日志：其他服务通过在配置文件.yml中增加日志映射属性，即可将产生的日志自动传送至本服务，详见服务配置日志方法.
- 查询日志：访问本服务提供的api，可查询日志列表.
- 删除日志：访问本服务提供的api，可删除日志列表

此外，服务还拥有如下功能：

- 日志分析：针对某服务的近期日志进行一定的分析.
- 应用层日志的接收与查询.

服务可通过logmgmt.yml文件更改下列配置信息：

```
name: logmgmt
url: http://123.207.73.150
ip: 119.29.228.21
logmgmtPort: 9999
syslogPort: 9898
mongodbHost: 119.29.228.21
mongodbPort: 8610
mongodbUserName: logmgt
mongodbPassword: testDb

server:
  type: simple
  connector:
    type: http
    port: 9999
```

各属性的说明如下表所示：

属性名	说明	默认值
name	本服务名称	logmgmt
url	注册api的url	http://123.207.73.150
ip	注册api的ip地址	119.29.228.21
logmgmtPort	注册api的端口（进行日志**查询**与**删除**）	9999
syslogPort	接收（**增加**）日志的端口	9898
mongodbHost	数据库主机	119.29.228.21
mongodbPort	数据库端口	8610
mongodbUserName	数据库管理员用户名	logmgt
mongodbPassword	数据库管理员用户名	testDb

在服务器上进行首次配置时，应依次执行下列命令：

1. MongoDB安装
2. 创建数据库
   use logs
3. 创建用户
   db.createUser({user:"logmgt",pwd:"logmgt",roles:[{"role":"readWrite","db":"logs"}]})
4. 检查用户权限
   db.auth("logmgt","logmgt")，若返回值为1，则用户权限设置成功
5. 数据库操作

（1）显示所有数据库：show dbs

（2）创建/切换数据库：use logs

2. 集合（表）操作

（1）显示所有集合：show collections

（2）删除集合：db.[COLLECTION NAME].drop()

（3）查询文档：db.[COLLECTION NAME].find().pretty()

（4）查询集合中文档总数：db.[COLLECTION NAME].find().pretty().count()

3. 更多命令详见：MongoDB

日志服务可接收其他服务自动传送的日志，只需在其他服务的.yml配置文件中添加如下信息：

以日志服务为例，原配置文件为：

```
name: courseservice
ip: 119.29.132.15
port: 7000
url: http://123.207.73.150

server:
  type: simple
  connector:
    type: http
    port: 7000
```

添加日志的映射功能后为：

```
name: courseservice
ip: 119.29.132.15
port: 7000
url: http://123.207.73.150

server:
  requestLog:
    appenders:
      - type: syslog
        host: localhost
        port: 9898
        facility: local0
        threshold: ALL

  type: simple
  connector:
    type: http
    port: 7000

logging:

  appenders:
    - type: syslog
      host: localhost
      port: 9898
      threshold: ALL
```

注：在使用时，requestLog与logging的host属性值应与本服务的host相同，port属性值应与本服务的syslogPort属性值相同。
如上方的配置信息所显示，本服务存储的日志共有两类：logging与requestLog。两类日志的介绍，以及资源所对应的URI如下表所示：

类别	名称	说明	URI
logging	运行日志	服务运行时所产生的日志，主要包括INFO信息与**报错信息**等	loggings
request	访问日志	服务的api受外界访问时所产生的日志	requests

类别	HTTP动词	Path
运行日志	GET	/api/v1/loggings
访问日志	GET	/api/v1/requests

访问api的总路径为：[API前缀] Path [参数列表]。

其中[API前缀]见附录，Path如上表所示，[参数列表]将在下文详细叙述。
查询运行日志logging时，需使用如下的参数列表：

参数	说明	示例	备注
serviceName	服务名	courseservice	必选
fromId	起始ID（不包括本身）	5a23732ee6022d0372e6e570	可选
level	日志级别	见下表	可选
host	主机名	localhost	可选
fromTimeStamp	起始日期（包括本身）	见下表	可选
toTimeStamp	截止日期（包括本身）	见下表	可选
errDetails	详细错误信息查看	1	可选
limit	查询总数量	50	可选

查询到的结果将按照时间降序排列，即优先显示最近生成的日志。

参数level

其中，level代表运行日志的级别，各level值所对应的日志级别如下所示：

0 Emergency: system is unusable
1 Alert: action must be taken immediately
2 Critical: critical conditions
3 Error: error conditions
4 Warning: warning conditions
5 Notice: normal but significant condition
6 Informational: informational messages
7 Debug: debug-level messages

level值越低的日志，优先级越高。

对level的传参使用**多值查询**，具体使用方法：用8位的二进制数表示是否查询各级别日志，且每个数位对应的级别如下：

Emergency | Alert | Critical | Error | Warning | Notice | Informational | Debug

128 | 64 | 32 | 16 | 8 | 4 | 2 | 1

如：二进制数[00011010]，即参数level = 26表示的是查看Error、 Warning、 Informational这三种类型的日志。

参数timestamp

fromTimeStamp与toTimeStamp可采用如下两种方式：

种类	格式	示例
1	yyyy-MM-dd hh:mm:ss	2017-11-24 23:11:40
2	yyyy-MM-dd	2017-11-24

参数errDetails

errDetails参数的默认值为0，表示**不查看错误细节**，将其值改为1，即可查看错误日志的具体报错堆栈信息。

示例

注：如下所述的示例均为本地数据库查询，在实际使用时需修改API前缀，详见附录说明。

在本地数据库中，查找服务名为test的日志：

localhost:9999/application/api/v1/loggings?serviceName=test&level=16&fromTimeStamp=2017-12-03 11:44:46

查询成功时，返回结果为：


```
{
    "msg": "1 results.",
    "status": 200,
    "data": [
        {
            "id": "5a23732ee6022d0372e6e56f",
            "facility": "16",
            "timestamp": "2017-12-03 11:44:46",
            "level": 3,
            "host": "snow.local",
            "serviceName": "test",
            "className": "io.dropwizard.jersey.errors.LoggingExceptionMapper",
            "message": "[dw-20 - GET /application/api/v2/course] Error handling a request: ef745256c764503d [BASIC ERROR-DETAILS] java.lang.NullPointerException: null",
            "errDetails": null
        }
    ]
}
```


对返回结果的说明，详见附录。
查询访问日志requestLog时，需使用如下的参数列表：

参数	说明	示例	备注
serviceName	服务名	courseservice	必选

参数	说明	示例	备注
fromId	起始ID（包括本身）	5a23732ee6022d0372e6e570	可选
host	主机名	localhost	可选
fromDateTime	起始日期（包括本身）	见下表	可选
toDateTime	截止日期（包括本身）	见下表	可选
method	访问方法	GET	可选
status	返回状态码	500	可选
limit	查询总数量	50	可选

查询到的结果将按照时间降序排列，即优先显示最近生成的日志。

参数datatime

其中，fromDateTime与toDateTime可采用如下两种方式：

种类	格式	示例
1	yyyy-MM-dd hh:mm:ss	2017-11-24 23:11:40
2	yyyy-MM-dd	2017-11-24

参数method

对参数method的查询支持**单值查询**与**多值查询**：

DELETE | PATCH | PUT | POST | GET
16 | 8 | 4 | 2 | 1

其中，单值查询即传入参数DELETE、GET等。对于多值查询，用5位二进制数表示是否查询某个方法。如，二进制数[01011]，即传入参数method = 11表示查询GET、POST、PATCH这三种方法的日志。

示例

在本地数据库中，查找服务名为test的日志：

localhost:9999/application/api/v1/requests?serviceName=test

成功时返回结果：


```
{
    "msg": "3 results.",
    "status": 200,
    "data": [
        {
            "id": "5a237324e6022d0372e6e56d",
            "host": "snow.local",
            "serviceName": "test",
            "className": "dw-21",
            "facility": 16,
            "clientIP": "0:0:0:0:0:0:0:1",
            "datetime": "2017-12-03 11:44:35",
            "method": "GET",
            "url": "/application/api/v2/class",
            "status": 200,
            "client": "PostmanRuntime/6.4.1"
        },
        {
            "id": "5a23732ae6022d0372e6e56e",
            "host": "snow.local",
            "serviceName": "test",
            "className": "dw-27",
            "facility": 16,
            "clientIP": "0:0:0:0:0:0:0:1",
            "datetime": "2017-12-03 11:44:41",
            "method": "GET",
            "url": "/application/api/v2/courseware",
            "status": 200,
            "client": "PostmanRuntime/6.4.1"
        },
        {
            "id": "5a23732ee6022d0372e6e570",
            "host": "snow.local",
```

```
            "serviceName": "test",
            "className": "dw-20",
            "facility": 16,
            "clientIP": "0:0:0:0:0:0:0:1",
            "datetime": "2017-12-03 11:44:46",
            "method": "GET",
            "url": "/application/api/v2/course",
            "status": 500,
            "client": "PostmanRuntime/6.4.1"
        }
    ]
}
```


类别	HTTP动词	Path
运行日志	DELETE	/api/v1/loggings
访问日志	DELETE	/api/v1/requests

访问api的总路径为：[API前缀] Path [参数列表]。

其中[API前缀]见附录，Path如上表所示，[参数列表]将在下文详细叙述。
删除运行日志logging时，需使用如下的参数列表：

参数	说明	示例	备注
serviceName	服务名	courseservice	必选
level	日志级别	6	可选
host	主机名	localhost	可选
fromTimeStamp	起始日期（包括本身）	如上文所示	可选
toTimeStamp	截止日期（包括本身）	如上文所示	可选

示例

在本地数据库中，删除服务名为courseservice的日志：

```
localhost:9999/application/api/v1/requests?serviceName=courseservice
```

删除失败时，返回结果为：


```
{
    "msg": "NOT FOUND",
    "status": 404,
    "data": ""
}
```


删除访问日志requestLog时，需使用如下的参数列表：

参数	说明	示例	备注
serviceName	服务名	courseservice	必选
host	主机名	localhost	可选
fromDateTime	起始日期（包括本身）	如上文所示	可选
toDateTime	截止日期（包括本身）	如上文所示	可选
method	访问方法	GET	可选
status	返回状态码	500	可选

示例

在本地数据库中，删除服务名为test的日志：

```
localhost:9999/application/api/v1/loggings?serviceName=test&level=6
```

删除成功时，返回结果为：


```
{
    "msg": "12 results has been deleted.",
    "status": 200,
    "data": ""
}
```


本服务提供日志分析功能：通过对某服务一天内产生日志的分析，获得下列反馈信息：

- 该服务最常被调用的api以及数量
- 该服务运行时产生的错误数
- 该服务的异常请求数
- 该服务最近一小时内，每隔5分钟的每秒请求数
- 该服务最近30天每天服务访问量
  日志服务拥有下列两个计划任务：
- 每隔5分钟，需计算一次各服务的每秒访问次数.
- 每隔1小时，需将记录存入MongoDB数据库.

下面将分别阐述两个计划任务对应的api.

每秒访问次数计算

HTTP动词	Path
POST	/api/v1/records/rates

示例

以本地的日志服务为例，每隔5分钟，需要访问下列api：

```
localhost:9999/application/api/v1/records/rates
```

其返回值为：

```
{
    "msg": "Successfully calculate rates for [NUM] services.",
    "status": 200,
    "data": ""
}
```

其中，[NUM]为当前日志服务所管理的服务总数。

每小时记录存储

HTTP动词	Path
POST	/api/v1/records

示例

以本地的日志服务为例，每隔1小时，需要访问下列api：

```
localhost:9999/application/api/v1/records
```

其返回值为：

```
{
    "msg": "1 records have been added.",
    "status": 200,
    "data": [
        {
            "id": null,
            "serviceName": "test",
            "timestamp": "2017-12-22 21:43:17",
            "apiRequestTable": {
                "subject": 3
            },
            "loggingErrors": 0,
            "requestExceptions": 0,
            "hourRequests": 3,
            "secondRequestsRate": [
                {
                    "timescale": "2017-12-22 21:43:14",
                    "requests": 3,
                    "rate": 0
                },
                ...
            ]
        }
    ]
}
```

若没有生成任何记录，则返回值为：

```
{
    "msg": "None Records",
    "status": 404,
    "data": ""
}
```

日志分析接口提供了如下的查询功能：

HTTP动词	Path
GET	/api/v1/records

请求参数列表

参数	说明	示例	备注
serviceName	服务名	courseservice	必选

返回参数说明

参数	说明
timestamp	服务器运行时间
serviceName	服务名
apiRequestTable	API访问表，记录该服务一天内的所有URI访问次数
requestException	该服务一天内的异常请求数
loggingErrors	该服务一天内的运行错误数
latestRequestRates	该服务最近一小时内，每5分钟的请求数
dailyRequests	该服务一天内的访问请求总数
recentDaysRequestsTable	该服务最近一个月内，每天的访问请求数

示例

以本地的日志服务为例，访问如下的api：

```
localhost:9999/application/api/v1/records/analysis?serviceName=test
```

其返回值为：

```
{
    "msg": "The daily analysis of test",
    "status": 200,
    "data": {
        "recentDaysRequestsTable": [
            {
                "timescale": "2017-11-23 23:00:43",
                "requests": 0,
                "rate": 0
            },
            {
                "timescale": "2017-11-24 23:00:43",
                "requests": 0,
                "rate": 0
            },

            ...

            {
                "timescale": "2017-12-22 23:00:43",
                "requests": 16,
                "rate": 0
            }
        ],
        "apiRequestTable": {
            "course": 1,
            "subject": 10,
            "class": 5
        },
        "requestExceptions": 1,
        "loggingErrors": 1,
```

```
        "serviceName": "test",
        "timestamp": "2017-12-22 23:00:43",
        "latestRequestRates": [
            {
                "timescale": "2017-12-22 21:43:14",
                "requests": 3,
                "rate": 0
            },

            ...

        ],
        "dailyRequests": 16
    }
}
```


能够通过api访问接收应用层（客户端）传来的日志，日志的基本格式如下：

属性	类型	说明
ip	String	访问客户端ip
userId	String	操作用户id
responseTime	long	响应时间
api	String	调用api及其参数
status	String	返回状态码
error	String	错误
msg	String	消息

通过传入日志的相关参数来进行日志接收。

HTTP动词	Path
POST	/api/v1/clients

添加一条应用层日志，所需参数如下：

参数	说明	备注
ip	访问客户端ip	必选
userId	操作用户id	必选
responseTime	响应时间	必选
api	调用api及其参数	必选
status	返回状态码	必选
error	错误	必选
msg	消息	必选

示例

以本地的日志服务为例，添加一条应用层日志：

```
localhost:9999/application/api/v1/clients?ip=192.168.1.101&userId=snow&responseTime=2000&api=courseserive&status=200&error=none&msg=normal
```

添加成功时，返回：


```
{
    "msg": "The log have been added.",
    "status": 200,
    "data": [
        {
            "id": "5a4b414de6022d07699513d2",
            "timestamp": "2018-01-02 16:22:37",
            "ip": "192.168.1.101",
            "userId": "snow",
            "responseTime": 2000,
            "api": "courseserive",
            "status": "200",
            "error": "none",
            "msg": "normal"
        }
    ]
}
```

通过传入日志的相关参数来进行应用层日志的条件查询。

HTTP动词	Path
GET	/api/v1/clients

查询应用层日志，可以使用的参数如下：

参数	说明	备注
fromTimestamp	起始时间	可选
toTimestamp	结束时间	可选
ip	访问客户端ip	可选
userId	操作用户id	可选
api	调用api及其参数	可选
status	返回状态码	可选
limit	返回的日志数量	可选

查询到的结果将按照时间降序排列，即优先显示最近生成的日志。

示例

以本地的日志服务为例，进行如下的条件查询：

localhost:9999/application/api/v1/clients?fromTimestamp=2018-01-01&toTimestamp=2018-01-03&ip=192.168.1.100

添加成功时，返回：


```
{
    "msg": "3 results.",
    "status": 200,
    "data": [
        {
            "id": "5a4a5e5fe6022d0eedbc6498",
            "timestamp": "2018-01-02 00:14:23",
            "ip": "192.168.1.100",
            "userId": "snow",
            "responseTime": 2000,
            "api": "courseserive",
            "status": "200",
            "error": "none",
            "msg": "normal"
        },
        {
            "id": "5a4a5e63e6022d0eedbc6499",
            "timestamp": "2018-01-02 00:14:27",
            "ip": "192.168.1.100",
            "userId": "snow",
            "responseTime": 2000,
            "api": "courseserive",
            "status": "200",
            "error": "none",
            "msg": "normal"
        },
        {
            "id": "5a4a5ef0e6022d0eedbc649a",
            "timestamp": "2018-01-02 00:16:48",
            "ip": "192.168.1.100",
            "userId": "snow",
            "responseTime": 2000,
            "api": "courseserive",
            "status": "200",
            "error": "none",
            "msg": "normal"
        }
    ]
}
```


无记录时，返回值为：


```
{
    "msg": "None Logs",
    "status": 404,
    "data": ""
```

```
}
```

服务器异常时，返回值为：

```
{
    "msg": "Errors in querying logs.",
    "status": 500,
    "data": ""
}
```

返回结果格式如下：

```
{
    "msg" : 字段,
    "status" : 状态码,
    "data" : 数组或对象
}
```

状态码status说明：

Successful 200

Client Error 4XX:
404 NOT FOUND - [*]：用户发出的请求针对的是不存在的记录，服务器没有进行操作，该操作是幂等的。

Server Error 5XX:
500 INTERNAL SERVER ERROR - [*]：服务器发生错误，用户将无法判断发出的请求是否成功。

访问本服务时，API前缀为：[主机名]:[logmgmtPort]/application.

如：在本地访问时，前缀为localhost:9999/application.
各level值所对应的日志级别如下所示：

0	Emergency: system is unusable
1	Alert: action must be taken immediately
2	Critical: critical conditions
3	Error: error conditions
4	Warning: warning conditions
5	Notice: normal but significant condition
6	Informational: informational messages
7	Debug: debug-level messages

其中，level值越低的日志，优先级越高。