# Using the SQL DM for MySQL API

You can access the API by passing parameters to Monyog through its base URL.

For example, if Monyog is running on a system with IP 192.168.1.1, then the parameters need to be passed to: `http://192.168.1.1:5555/`

You can use either of the HTTP methods GET and POST.

## The Parameters

The parameters that you will need to pass are:

- `_object`: This basically addresses the logical object in Monyog that you want to direct your request to.The only acceptable value is MONyogAPI.
- `_action`: This specifies the part of the object specified above that you want to direct your request to. The acceptable values are:
    - `Alerts`
    - `DataCollection`
    - `Sniffer`
    - `LongRunningQueries`
    - `LockedQueries`
    - `LongRunningQueryAction`
    - `AddServer`
    - `EditServer`
    - `RemoveServer`
- `_value`: The operation that you want to perform for the action specified in the _action field. Acceptable values include for:
    - `Alerts`, `DataCollection`, `Sniffer`, `LongRunningQueries`, `LockedQueries`: `enable` and `disable`
    - `LongRunningQueryAction`: `notify`, `kill` and `notifyandkill`
- `_user`: It may be Monyog user, LDAP user or LDAP Group user. In case, no user is supplied, admin account is used by default.
- `_password`: The password for the specified _user.
- `_server`: Name or data directory number of the servers separated by a comma(‘,’) for which the operation to be performed.
- `_tags`: Name of the tag separated by a comma(‘,’) for which the operation to be performed for all the servers under the specified tag.

## API's for server management

To manage servers in Monyog via API, the following parameters need to be passed:

- `_server`: The name or data directory number of the server to be registered
- `_mysqlhost`: MySQL host/ip address
- `_mysqluser`: MySQL user name
- `_mysqlport`: MySQL port
- `_mysqlpassword`: MySQL user password

For example, to add a server:

```
$ curl "192.168.1.1:5555/?_object=MONyogAPI&_action=addserver&_user=admin
       &_server=Production&_mysqlhost=127.0.0.1&_mysqluser=admin
       &_mysqlport=3306&_mysqlpassword=adminpassword"
```

Additional parameters for registering servers are listed here.

For example, suppose you have a server named Production001 registered with Monyog. To stop data collection for this server using the HTTP GET method, the URL would look like:

```
$ curl "http://192.168.1.1:5555/?_object=MONyogAPI&_action=DataCollection
       &_value=disable&_password=mypassword&_server=Production001"
```

In summary, the various URLs that you can use with curl:

### Starts data collection for <server name>

```
$ curl "http://192.168.1.1:5555/?_object=MONyogAPI&_action=DataCollection
       &_value=enable&_user=admin&_password=Password&_server=Production001"
```

### Starts data collection for <multiples servers>

```
$ curl "http://192.168.1.1:5555/?_object=MONyogAPI&_action=DataCollection
       &_value=enable&_user=admin&_password=Password&_server=Production001,Test"
```

### Stops data collection for <server name>(Slave Of Production)

```
$ curl "http://192.168.1.1:5555/?_object=MONyogAPI&_action=DataCollection
       &_value=disable&_user=admin&_password=Password&_server=Slave+Of+Production"
```

### Starts data collection for <tag>

```
$ curl "http://192.168.1.1:5555/?_object=MONyogAPI&_action=DataCollection
       &_value=enable&_user=admin&_password=Password&_tag=Production"
```

### Stops data collection for <tag>

```
$ curl "http://192.168.1.1:5555/?_object=MONyogAPI&_action=DataCollection
       &_value=disable&_user=admin&_password=Password&_tag=Production"
```

### Stops data collection globally for all the servers (Maintenance)

```
$ curl "http://192.168.1.1:5555/?_object=MONyogAPI&_action=datacollection&_value=enable&_user=admin&_password="
```

### Enables alerts for <server name>

```
$ curl "http://192.168.1.1:5555/?_object=MONyogAPI&_action=Alerts
       &_value=enable&_user=admin&_password=Password&_server=Production001"
```

### Disables alerts for <server name>

```
$ curl "http://192.168.1.1:5555/?_object=MONyogAPI&_action=Alerts
       &_value=disable&_user=admin&_password=Password&_server=Production001"
```

### Enables alerts for <tag>

```
$ curl "http://192.168.1.1:5555/?_object=MONyogAPI&_action=Alerts
       &_value=enable&_user=admin&_password=Password&_tag=Production"
```

### Disables alerts for <tag>

```
$ curl "http://192.168.1.1:5555/?_object=MONyogAPI&_action=Alerts
       &_value=disable&_user=admin&_password=Password&_tag=Production"
```

### Disables alerts globally for all the servers(Maintenance)

```
$ curl "http://192.168.1.1:5555/?_object=MONyogAPI&_action=Alerts&_value=disable&_user=admin&_password="
```

### Enables alerts globally for all the servers(Maintenance)

```
$ curl "http://192.168.1.1:5555/?_object=MONyogAPI&_action=Alerts&_value=enable&_user=admin&_password="
```

### Enables Sniffer for <server name>

```
$ curl "http://192.168.1.1:5555/?_object=MONyogAPI&_action=sniffer
       &_value=enable&_server=Production001"
```

### Disables Sniffer for <server name>

```
$ curl "http://192.168.1.1:5555/?_object=MONyogAPI&_action=sniffer
       &_value=disable&_server=Production001"
```

### Add Server

```
$ curl "http://192.168.1.1:5555/?_object=MONyogAPI&_action=addserver
       &_mysqluser=msandbox&_mysqlhost=127.0.0.1&_mysqlport=3306&_tags=Production
       &_server=Test&_mysqlpassword=msandbox&_connectontype=direct
       &_user=admin&_password=Password"
```

### Add Server with SSH Tunnel

```
$ curl "http://192.168.1.1:5555/?_object=MONyogAPI&_action=addserver
       &_mysqluser=msandbox&_mysqlhost=127.0.0.1&_mysqlport=3306
       &_tags=Production&_server=Test&_mysqlpassword=msandbox
       &_connectiontype=ssh&_sshhost=192.168.1.86&_sshuser=username
       &_sshpassword=sshpassword&_sshport=22&_user=admin&_password=Password"
```

### Edit Server

```
$ curl "http://192.168.1.1:5555/?_object=MONyogAPI&_action=editserver
       &_mysqluser=msandbox&_mysqlhost=127.0.0.1&_mysqlport=3306
       &_tags=Production&_server=Test&_mysqlpassword=msandbox
       &_connectontype=direct&_user=admin&_password=Password"
```

### Delete Server

```
$ curl "http://192.168.1.1:5555/?_object=MONyogAPI&_action=removeserver&_server=Test"
```

### Delete all the servers under <tag>

```
curl "http://192.168.1.1:5555/?_object=MONyogAPI&_action=removeserver&_tag=Production"
```

### Delete multiple servers

```
curl "http://192.168.1.1:5555/?_object=MONyogAPI&_action=removeserver&_server=Server1,Server2"
```

## Return Codes

Assuming that the connection to SQL DM for MySQL was successful, it returns a text message. The message is in the JSON format:

```
{"STATUS": "SUCCESS/FAILURE", "RESPONSE" : "<Response text>"}
```

Your application can parse this message and determine whether the operation was successfully carried out or not.

> ⚠ Since version 5.21 we have deprecated the API calls to `_object=ConnectionMgr`. Instead use `_object=MONyogAPI`.

## Applications

The Monyog API is very flexible and can be accessed from other programming languages including scripting languages such as Perl, VBScript, etc. Here is a very generic Perl script that accepts the required parameters from the command line and executes the specified action:

```
#! /usr/bin/perl
use LWP 5.64;
# USAGE: MONyog.pl <hostname>:<port> <user> <password> <connection_name/ID> <action> <value>
# $ARGV[0] = hostname:port of server running Monyog
# $ARGV[1] = Monyog user
# $ARGV[2] = Monyog password
# $ARGV[3] = connection name
# $ARGV[4] = action
# $ARGV[5] = value
my $numArgs = $#ARGV + 1;
if($numArgs < 5) {
   die 'USAGE: MONyog.pl <hostname>:<port> <user> <password> <connection_name/ID> <action>';
}

my $browser = LWP::UserAgent->new;

# The request URL
my $url = URI->new('http://' . $ARGV[0] . '/');

# The form data pairs:
$url->query_form(
   '_object' => 'MONyogAPI',
   '_action' => $ARGV[4],
   '_user' => $ARGV[1]
   '_password' => $ARGV[2],
   '_server' => $ARGV[3],
   '_value' => $ARGV[5]
);

# The response object
$response = $browser->post($url);

if (!$response->is_success) {# Error connecting to MONyog
   die $response->status_line . "\n";
} else {

   # Successfully connected to MONyog; print MONyog's response
   print $response->content . "\n";

}
```