23.0 Service Group APIs

WAS THIS PAGE HELPFUL? Leave Feedback

23.0 Service Group APIs

23.1 Query Service Groups

Retrieve service groups by query. There are two kinds of queries supported:

1. Retrieving a single service group. Retrieves exactly one service group by name.
2. Retrieving one or more service groups. Retrieves 1..n service group objects wrapped by an XML `<serviceGroups>` collection or a JSON object with a `serviceGroups` array member.

23.1.1 Method: GET a Single Service Group by Unique Name

GET /api/servicegroups/{serviceGroupName}

23.1.2 HTTP Query and Path Parameters

Field	Type	Description	Required
serviceGroupName	Path	the unique name of the service group	yes

23.1.3 Method: GET Service Groups

GET /api/servicegroups?query=(query criteria see below)

23.1.4 HTTP Query and Path Parameters

Field	Type	Description	Required
query	Query	An encoded query string (where clause)	no **
first	Query	Paging for query. First record to start from	no
count	Query	Paging for query. Number of records to include when paging	no
appType	Query	Application type name filter for service groups	no
agentId	Query	Agent id filter for service groups	no

***Note: If a query parameter is not provided, all service groups will be retrieved.*

23.1.5 HTTP Headers

Header	Valid Values	Required
Content-Type	application/xml or application/json	True
GWOS-API-TOKEN	a valid token returned from login	True
GWOS-APP-NAME	your application name	True

23.1.6 Query Fields

Because service groups are implemented as categories, supported query fields are identical. However, entityType shortcuts and fully qualified fields are hardcoded for service groups and should not be used in a query.

Field	Description	Alias
id	Service group integer id	categoryId

name	The service group primary unique name	
description	The description of this service group	
root	Boolean flag indicating a root service group of a hierarchy	
categoryEntities	A prefix for the services associated with this service group. Valid sub-fields are: categoryEntities.categoryEntityID categoryEntities.objectID categoryEntities.entityType.name categoryEntities.entityType.entityTypeId categoryEntities.entityType.description categoryEntities.entityType.LogicalEntity categoryEntities.entityType.applicationTypeSupported	
ancestors	A prefix for the ancestor parent service groups. Valid sub-fields include all query fields recursively	
parents	A prefix for the parent service groups. Valid sub-fields include all query fields recursively	
children	A prefix for the child service groups. Valid sub-fields include all query fields recursively	

Note: Query fields are case-insensitive, thus camelCase, or all lower case will both work fine.

23.1.7 Example Queries

These examples are not HTTP encoded for readability. In practice queries must be encoded.

1. *query for all service groups*
   GET /api/servicegroups
2. *query for all service groups order by name descending*
   GET /api/servicegroups?query=order by name desc
3. *query for a single service group named 'TEST-SERVICE-GROUP-0'*
   GET /api/servicegroups/TEST-SERVICE-GROUP-0
4. *a like query to find all service groups starting with 'TEST-'*
   GET /api/servicegroups?query=name like 'TEST-%'
5. *query for one or more service group names using IN query syntax*
   GET /api/servicegroups?query=name in ('WEB','DATABASE')
6. *query for all service groups with application type 'VEMA'*
   GET /api/servicegroups/?appType=VEMA
7. *query for service groups with a specific service entity and agent id '007'*
   GET /api/servicegroups/?query=categoryEntities.objectID = 1024 and categoryEntities.entityType.name = 'SERVICE_STATUS'&agentId=007

23.1.8 HTTP Status Codes

200 - One or more query results returned
404 - No query results found
500 - An internal server error occurred

23.1.9 Example Responses

Results of requesting a single entity is always wrapped with a single `<serviceGroup>` XML element. Here is an XML example of the result of a query finding one service group. All simple fields are displayed as attributes. Collections of services appear as subelements. Service properties have been elided here for the sake of brevity.

```
<serviceGroup id="24" name="SG-200" description="Updated" appType="CACTI" appTypeDisplayName="CACTI"
agentId="SECRET">
  <services>
    <service id="8" appType="NAGIOS" description="local_load" monitorStatus="OK" lastCheckTime=
"2013-05-22T10:40:34.000-0600" nextCheckTime="2013-05-22T10:50:34.000-0600" lastStateChange=
"2013-05-17T08:47:38.000-0600" hostName="localhost" stateType="HARD" checkType="ACTIVE" lastHardState=
"PENDING" monitorServer="localhost" deviceIdentification="127.0.0.1" lastPlugInOutput=
"1^^^3^^^0^^^05/17/2013 08:47:38 AM^^^OK - load average: 0.13, 0.18,
0.23^^^load1=0.130;5.000;10.000;0; load5=0.180;4.000;8.000;0; load15=0.230;3.000;6.000;0;">
      <properties/>
    </service>
    <service id="20" appType="NAGIOS" description="tcp_http" monitorStatus="OK" lastCheckTime=
"2013-05-22T10:43:49.000-0600" nextCheckTime="2013-05-22T10:53:49.000-0600" lastStateChange=
"2013-05-17T08:53:49.000-0600" hostName="localhost" stateType="HARD" checkType="ACTIVE" lastHardState=
"PENDING" monitorServer="localhost" deviceIdentification="127.0.0.1" lastPlugInOutput=
"1^^^3^^^0^^^05/17/2013 08:53:49 AM^^^HTTP OK: HTTP/1.1 200 OK - 1268 bytes in 0.001 second response
time^^^time=0.000557s;3.000000;5.000000;0.000000 size=1268B;;;0">
      <properties/>
    </service>
  </services>
</serviceGroup>
```

Result of queries are always wrapped in a `<serviceGroups>` XML collection element, with one or more `<serviceGroup>` subelements.

```
<serviceGroups>
  <serviceGroup id="21" name="SG-202" description="Service Group 202" appType="NAGIOS" agentId=
"Agent-008">
    <services>
      <service id="2" appType="NAGIOS" description="local_cpu_java" monitorStatus="OK" lastCheckTime=
"2013-05-22T10:37:15.000-0600" nextCheckTime="2013-05-22T10:47:15.000-0600" lastStateChange=
"2013-05-17T08:57:15.000-0600" hostName="localhost" stateType="HARD" checkType="ACTIVE" lastHardState=
"PENDING" monitorServer="localhost" deviceIdentification="127.0.0.1" lastPlugInOutput=
"1^^^3^^^0^^^05/17/2013 08:57:15 AM^^^OK - total %CPU for process java : 2.8^^^%CPU=2.8;40;50">
        <properties/>
      </service>
    </services>
  </serviceGroup>
</serviceGroups>
```

Here is a JSON example of the result of a query finding one service group. Service properties have been elided here for the sake of brevity.

```
{
  "id" : 15,
  "name" : "SG-200",
  "description" : "Service Group 200",
  "appType" : "VEMA",
  "appTypeDisplayName" : "VEMA",
  "agentId" : "Agent-007",
  "services" : [ {
    "properties" : {
    },
    "id" : 2,
    "appType" : "NAGIOS",
    "description" : "local_cpu_java",
    "monitorStatus" : "OK",
    "lastCheckTime" : "2013-05-22T10:37:15.000-0600",
    "nextCheckTime" : "2013-05-22T10:47:15.000-0600",
    "lastStateChange" : "2013-05-17T08:57:15.000-0600",
    "hostName" : "localhost",
    "stateType" : "HARD",
    "checkType" : "ACTIVE",
    "lastHardState" : "PENDING",
    "monitorServer" : "localhost",
    "deviceIdentification" : "127.0.0.1",
    "lastPlugInOutput" : "1^^^3^^^0^^^05/17/2013 08:57:15 AM^^^OK - total %CPU for process java :
2.8^^^%CPU=2.8;40;50"
  }, {
    "properties" : {
    },
    "id" : 4,
    "appType" : "NAGIOS",
    "description" : "local_cpu_perl",
    "monitorStatus" : "OK",
    "lastCheckTime" : "2013-05-22T10:36:13.000-0600",
    "nextCheckTime" : "2013-05-22T10:46:13.000-0600",
    "lastStateChange" : "2013-05-17T08:46:12.000-0600",
    "hostName" : "localhost",
    "stateType" : "HARD",
    "checkType" : "ACTIVE",
    "lastHardState" : "PENDING",
    "monitorServer" : "localhost",
    "deviceIdentification" : "127.0.0.1",
    "lastPlugInOutput" : "1^^^3^^^0^^^05/17/2013 08:46:12 AM^^^OK - total %CPU for process perl :
0.7^^^%CPU=0.7;40;50"
  } ]
}
```

The array results of queries are always wrapped in an object with a field named `serviceGroups`.

```
{
  "serviceGroups" : [ {
    "id" : 17,
    "name" : "SG-202",
    "description" : "Service Group 202",
    "appType" : "NAGIOS",
    "agentId" : "Agent-008",
    "services" : [ {
      "properties" : {
      },
      "id" : 2,
      "appType" : "NAGIOS",
      "description" : "local_cpu_java",
      "monitorStatus" : "OK",
      "lastCheckTime" : "2013-05-22T10:37:15.000-0600",
      "nextCheckTime" : "2013-05-22T10:47:15.000-0600",
      "lastStateChange" : "2013-05-17T08:57:15.000-0600",
      "hostName" : "localhost",
      "stateType" : "HARD",
      "checkType" : "ACTIVE",
      "lastHardState" : "PENDING",
      "monitorServer" : "localhost",
      "deviceIdentification" : "127.0.0.1",
      "lastPlugInOutput" : "1^^^3^^^0^^^05/17/2013 08:57:15 AM^^^OK - total %CPU for process java :
2.8^^^%CPU=2.8;40;50"
    }, {
      "properties" : {
      },
      "id" : 4,
      "appType" : "NAGIOS",
      "description" : "local_cpu_perl",
      "monitorStatus" : "OK",
      "lastCheckTime" : "2013-05-22T10:36:13.000-0600",
      "nextCheckTime" : "2013-05-22T10:46:13.000-0600",
      "lastStateChange" : "2013-05-17T08:46:12.000-0600",
      "hostName" : "localhost",
      "stateType" : "HARD",
      "checkType" : "ACTIVE",
      "lastHardState" : "PENDING",
      "monitorServer" : "localhost",
      "deviceIdentification" : "127.0.0.1",
      "lastPlugInOutput" : "1^^^3^^^0^^^05/17/2013 08:46:12 AM^^^OK - total %CPU for process perl :
0.7^^^%CPU=0.7;40;50"
    } ]
  } ]
}
```

23.2 Create or Update Service Groups

Persist a batch (1..n) of service groups in foundation database. Service groups will be created if they do not exist. If a service group exists, it will be updated. You can also specify one or more services to be associated with the service groups. Specified services will replace existing services.

If one or more service group creation operations fails, others may still succeed. This is not an all-or-none transactional operation. The results of each individual creation/update operation is returned back in the resultset described below with a status of success or failure.

23.2.1 Method: POST

POST /api/servicegroups

23.2.2 HTTP Headers

Header	Valid Values	Required
Content-Type	application/xml or application/json	True
GWOS-API-TOKEN	a valid token returned from login	True

GWOS-APP-NAME	your application name	True

23.2.3 Post Data Attributes and Elements

Field	Description	Required	Type
name	The unique service group name	Yes	Attribute/Field
description	The description of this service group	No	Attribute/Field
appType	Application type name of this service group	No	Attribute/Field
agentId	Agent id of this service group	No	Attribute/Field
hostGroupNames	Host group names to be associated with this service group	No	Elements/Array Field
serviceGroupNames	Service keys to be associated with this service group	No	Elements/Array Field

23.2.4 Example Post Data

As with the results of queries, service groups are always wrapped in a `<serviceGroups>` XML collection element, with one or more `<serviceGroup>` subelements. For JSON, service groups are always wrapped in an object with a field named `serviceGroups`.

```
<serviceGroups>
  <serviceGroup name="SG-200" description="Service Group 200" appType="VEMA" agentId="Agent-007">
    <services>
      <service service="local_cpu_java" host="localhost"/>
      <service service="local_cpu_perl" host="localhost"/>
    </services>
  </serviceGroup>
</serviceGroups>
```

```
{
  "serviceGroups" : [ {
    "name" : "SG-200",
    "description" : "Service Group 200",
    "appType" : "VEMA",
    "agentId" : "Agent-007",
    "services" : [ {
      "service" : "local_cpu_java",
      "host" : "localhost"
    }, {
      "service" : "local_cpu_perl",
      "host" : "localhost"
    } ]
  } ]
}
```

23.2.5 HTTP Status Codes

200 - Zero or more service groups were created without any internal server errors
500 - An internal server error occurred

23.2.6 Example Responses

In this example, we use the POST data above: four service groups were successfully created.

Note that the results collection also returns the location of the created or updated service group which can be directly used by the GET operation.

```
<results successful="1" failed="0" entityType="ServiceGroup" operation="Update" warning="0" count="1">
  <result>
    <entity>SG-200</entity>
    <location>http://localhost:8080/foundation-webapp/api/servicegroups/SG-200</location>
<status>success</status>
  </result>
</results>
```

```
{
  "successful" : 1,
  "failed" : 0,
  "entityType" : "ServiceGroup",
  "operation" : "Update",
  "warning" : 0,
  "results" : [ {
    "entity" : "SG-200",
    "status" : "success",
    "location" : "http://localhost:8080/foundation-webapp/api/servicegroups/SG-200"
} ],
  "count" : 1
}
```

23.3 Add or Delete Service Group Members

Services are associated with service groups and can be manipulated via these PUT operations.

23.3.1 Method: PUT adding members with POST Data

PUT /api/servicegroups/addmembers

23.3.2 Method: PUT deleting members with POST Data

PUT /api/servicegroups/deletemembers

23.3.3 HTTP Headers

Header	Valid Values	Required
Content-Type	application/xml or application/json	True
GWOS-API-TOKEN	a valid token returned from login	True
GWOS-APP-NAME	your application name	True

23.3.4 Post Data Attributes and Elements

Field	Description	Required	Type
name	The unique service group name	Yes	Attribute/Field
hostGroupNames	Host group names to be added to or removed from the service group	No	Elements/Array Field
serviceGroupNames	Service keys to be added to or removed from the service group	No	Elements/Array Field

23.3.5 Example POST Data

```
<serviceGroup name="SG-200">
  <services>
    <service service="local_load" host="localhost"/>
    <service service="local_users" host="localhost"/>
    <service service="tcp_http" host="localhost"/>
  </services>
</serviceGroup>
```

```
{
  "name" : "SG-200",
  "services" : [ {
    "service" : "local_load",
    "host" : "localhost"
  }, {
    "service" : "local_users",
    "host" : "localhost"
  }, {
    "service" : "tcp_http",
    "host" : "localhost"
  } ]
}
```

23.3.6 HTTP Status Codes

200 - Service groups were updated without any internal server errors
500 - An internal server error occurred

23.3.7 Example Responses

Note that the results collection also returns the location of the updated service group which can be directly used by the GET operation.

```
<results successful="1" failed="0" entityType="ServiceGroup" operation="Update" warning="0" count="1">
  <result>
    <entity>SG-200</entity>
    <location>http://localhost:8080/foundation-webapp/api/servicegroups/SG-200</location>
<status>success</status>
  </result>
</results>
```

```
{
  "successful" : 1,
  "failed" : 0,
  "entityType" : "ServiceGroup",
  "operation" : "Update",
  "warning" : 0,
  "results" : [ {
    "entity" : "SG-200",
    "status" : "success",
    "location" : "http://localhost:8080/foundation-webapp/api/servicegroups/SG-200"
} ],
  "count" : 1
}
```

23.4 Delete Service Groups

Deletes one or more service groups from the Collage database. Deletes are specified with POST data, passing in service group names.

Note that while deleting hierarchical service groups is supported for backward compatibility, this is not recommended. Hierarchical service group delete operations should be carried out using the hierarchy-specific APIs for categories.

23.4.1 Method: DELETE with POST Data

DELETE /api/servicegroups

23.4.2 HTTP Headers

Header	Valid Values	Required
Content-Type	application/xml or application/json	True
GWOS-API-TOKEN	a valid token returned from login	True
GWOS-APP-NAME	your application name	True

23.4.3 Post Data Attributes and Elements

Field	Description	Required	Type
name	The unique service group name	Yes	Attribute/Field

23.4.4 Example POST Data

The post data should be formatted like a POST payload, but the only required field is the service group name.

```
<serviceGroups>
  <serviceGroup name="SG-200"/>
  <serviceGroup name="SG-201"/>
  <serviceGroup name="SG-202"/>
  <serviceGroup name="SG-203"/>
</serviceGroups>
```

```
{
  "serviceGroups" : [ {
    "name" : "SG-200"
  }, {
    "name" : "SG-201"
  }, {
    "name" : "SG-202"
  }, {
    "name" : "SG-203"
  } ]
}
```

23.4.5 HTTP Status Codes

200 - Service groups were deleted successfully
500 - An internal server error occurred

23.4.6 Example Responses

```
<results successful="4" failed="0" entityType="ServiceGroup" operation="Delete" warning="0" count="4">
  <result>
    <entity>SG-200</entity>
    <message>Service Group SG-200 deleted.</message>
    <status>success</status>
  </result>
  <result>
    <entity>SG-201</entity>
    <message>Service Group SG-201 deleted.</message>
    <status>success</status>
  </result>
  <result>
    <entity>SG-202</entity>
    <message>Service Group SG-202 deleted.</message>
    <status>success</status>
  </result>
  <result>
    <entity>SG-203</entity>
    <message>Service Group SG-203 deleted.</message>
    <status>success</status>
  </result>
</results>
```

```
{
  "successful" : 4,
  "failed" : 0,
  "entityType" : "ServiceGroup",
  "operation" : "Delete",
  "warning" : 0,
  "results" : [ {
    "entity" : "SG-200",
    "status" : "success",
    "message" : "Service Group SG-200 deleted."
  }, {
    "entity" : "SG-201",
    "status" : "success",
    "message" : "Service Group SG-201 deleted."
  }, {
    "entity" : "SG-202",
    "status" : "success",
    "message" : "Service Group SG-202 deleted."
  }, {
    "entity" : "SG-203",
    "status" : "success",
    "message" : "Service Group SG-203 deleted."
  } ],
  "count" : 4
}
```

23.5 Service Group Name Autocomplete

Get limited number of sorted existing Service Group names that match a specified prefix. Matching and sorting is done in a case-insensitive fashion. The special wildcard prefix '*' matches all names. Autocomplete names are updated asynchronously when Service Group names are modified.

23.5.1 Method: GET Service Group Name Autocomplete

GET /api/servicegroups/autocomplete/{prefix}

23.5.2 Path Parameters

Field	Type	Description	Required

prefix	Path	Encoded Service Group name prefix to match	yes

23.5.3 HTTP Headers

Header	Valid Values	Required
Accept	application/xml or application/json	True
GWOS-API-TOKEN	a valid token returned from login	True
GWOS-APP-NAME	your application name	True

23.5.4 HTTP Status Codes

Code	Description
200	Autocomplete names for prefix returned
404	No autocomplete names for prefix
401	Authentication/authorization error occurred
500	An internal server error occurred while processing autocomplete prefix

23.5.5 Example Responses

Here is an XML example of the autocomplete names returned.

XML results are always wrapped in an `<names>` collection element, with one or more `<name>` subelements.

```
<names>
    <name>SG1</name>
    <name>SG2</name>
</names>
```

Here is a JSON example of the autocomplete names returned.

JSON results are always wrapped in an object with a `names` array member, with one or more name strings.

```
{
  "names" : [ "SG1", "SG2" ]
}
```