» logicmonitor_collectors

Use this datasource to get the ID of an available collector.

» Example Usage

```
# Look up a LogicMonitor collector id
data "logicmonitor_collectors" "collectors" {
  filters {
    property = "hostname"
    operator = "~"
    value = "test"
  },
"most_recent" = true
}
```

» Argument Reference

The following arguments are supported:

- `size` - (Optional) The number of results to display. Max is 1000. Default is 50
- `offset` - (Optional) The number of results to offset the displayed results by. Default is 0
- `most_recent` - (Optional) The most recent collector installed that is online
- `filters` - (Optional) Filters the response according to the operator and value specified. Note that you can use * to match on more than one character. More Info: https://www.logicmonitor.com/support/rest-api-developers-guide/v1/collector-groups/

» Nested filters blocks

Nested `filters` blocks have the following structure: `property{operator}value` * `property` - (Required if using filters) The name of filtered property. Currently the properties supported are `hostname` and `description` * `operator` - (Required if using filters) The type of operator. Currently the operators supported are `:` `~` `!:` `!~` * `value` - (Required if using filters) The value of the filtered property.

» logicmonitor_dashboard

Use this datasource to get the ID of an available dashboard group.

» Example Usage

```
# Look up a LogicMonitor dashboard id
data "logicmonitor_dashboard" "LowerMerion" {
  filters {
    property = "geoLocation"
    operator = ":"
    value = "Philadelphia"
  }
}
```

» Argument Reference

The following arguments are supported:

- `size` - (Optional) The number of results to display. Max is 1000. Default is 50
- `offset` - (Optional) The number of results to offset the displayed results by. Default is 0
- `filters` - (Optional) Filters the response according to the operator and value specified. Note that you can use * to match on more than one character. More Info: https://www.logicmonitor.com/support/rest-api-developers-guide/v1/dashboard-groups/

» Nested filters blocks

Nested `filters` blocks have the following structure: `property{operator}value` * `property` - (Required if using filters) The name of filtered property. Currently the properties supported are `hostname` and `description` * `operator` - (Required if using filters) The type of operator. Currently the operators supported are `:` `~` `!:` `!~` * `value` - (Required if using filters) The value of the filtered property.

» logicmonitor_dashboard_group

Use this datasource to get the ID of an available dashboard group.

» Example Usage

```
# Look up a LogicMonitor dashboard group id
data "logicmonitor_dashboard_group" "TwentyFourLife" {
  filters {
    property = "fullPath"
    operator = ":"
    value = "Teams/Lakers/Guard"
  }
}
```

» Argument Reference

The following arguments are supported:

- `size` - (Optional) The number of results to display. Max is 1000. Default is 50
- `offset` - (Optional) The number of results to offset the displayed results by. Default is 0
- `filters` - (Optional) Filters the response according to the operator and value specified. Note that you can use * to match on more than one character. More Info: https://www.logicmonitor.com/support/rest-api-developers-guide/v1/dashboard-groups/

» Nested filters blocks

Nested `filters` blocks have the following structure: `property{operator}value` * `property` - (Required if using filters) The name of filtered property. Currently the properties supported are `hostname` and `description` * `operator` - (Required if using filters) The type of operator. Currently the operators supported are `: ~ !: !~` * `value` - (Required if using filters) The value of the filtered property.

» logicmonitor_device_group

Use this datasource to get the ID of an available device group.

» Example Usage

```
# Look up a LogicMonitor device group id
data "logicmonitor_device_group" "devicegroups" {
  filters {
    property = "name"
    operator = ":"
    value = "Production"
  },
  filters {
    custom_property_name = "app.user"
    operator = ":"
    custom_property_value = "api"
  }
}
```

» Argument Reference

The following arguments are supported:

- `size` - (Optional) The number of results to display. Max is 1000. Default is 50
- `offset` - (Optional) The number of results to offset the displayed results by. Default is 0
- `filters` - (Optional) Filters the response according to the operator and value specified. Note that you can use * to match on more than one character. More Info: https://www.logicmonitor.com/support/rest-api-developers-guide/v1/device-groups/

» Nested filters blocks

Nested `filters` blocks have the following structure: `property{operator}value` * `property` - (Required if using filters) The name of filtered property. * `operator` - (Required if using filters) The type of operator. * `value` - (Required if using filters) The value of the filtered property.

You can also do custom properties * `custom_property_name` - (Required if using filters and custom properties) The name of filtered custom property. * `operator` - (Required if using filters) The type of operator. * `custom_property_value` - (Required if using filters and custom properties) The value of the filtered custom property.

» logicmonitor_collector

Provides a LogicMonitor collector resource. This can be used to create and manage LogicMonitor collectors.

Note: This resource will only create the collector device in your account. See Downloading a Collector Installer for information on how to download and install an existing collector.

» Example Usage

```
# Create a new LogicMonitor collector
resource "logicmonitor_collector" "collector1" {
  description     = "my terraformed collector"
  enable_failback = true
}
```

» Argument Reference

The following arguments are supported:

- `backup_collector_id` - (Optional) The Id of the failover Collector configured for this Collector
- `collector_group_id` - (Optional) The Id of the group the Collector is in
- `description` - (Optional) The Collector's description
- `enable_failback` - (Optional) Whether or not automatic failback is enabled for the Collector
- `enable_collector_device_failover` - (Optional) Whether or not the device the Collector is installed on is enabled for fail over
- `escalation_chain_id` - (Optional) The Id of the escalation chain associated with this Collector
- `resend_interval` - (Optional) The interval, in minutes, after which alert notifications for the Collector will be resent
- `suppress_alert_clear` - (Optional) Whether alert clear notifications are suppressed for the Collector

» logicmonitor_collector_group

Provides a LogicMonitor collector group resource. This can be used to create and manage LogicMonitor collector groups

» Example Usage

```
# Create a new LogicMonitor collector group
resource "logicmonitor_collector_group" "group1" {
  name  = "collector_group_1"
  description = "a new test group"
}
```

» Argument Reference

The following arguments are supported:

- `name` - (Required) Name of collector group
- `description` - (Optional) Set description of collector group

» logicmonitor_dashboard

Provides a LogicMonitor dashboard resource. This can be used to create and manage LogicMonitor dashboards Currently only creating a dashboard based on an existing JSON template is supported

» Example Usage

```
# Create a new LogicMonitor dashboard
resource "logicmonitor_dashboard" "dash" {
  name = "FiveRings"
  description = "a new dashboard"
  widget_tokens {
    "defaultResourceGroup" = "Lakers/Championships"
  }
}

# Create a new LogicMonitor dashboard based on a template stored in s3
resource "logicmonitor_dashboard" "dash" {
  name = "FiveRings"
  description = "a new dashboard"
  widget_tokens {
    "defaultResourceGroup" = "Lakers/Championships"
  }
  template = "${data.aws_s3_bucket_object.template.body}"
}

data "aws_s3_bucket_object" "template" {
  bucket = "basketball-dashboards"
  key = "Ballers.json"
}
```

» Argument Reference

The following arguments are supported:

- `name` - (Required) Name of dashboard

- `description` - (Optional) Description of dashboard
- `group_id` - (Optional) The id of the parent group for this dashboard
- `public` - (Optional) Defines if it is a public or private dashboard.
- `template` - (Optional) Defines if an existing exported JSON template is used to create dashboard
- `widget_tokens` - (Optional) Dashboard tokens allow users to apply a single dashboard template to different device or website groups simply by changing the tokens' values.

» Import

Device Groups can be imported using their group id or full path

```
$ terraform import logicmonitor_dashboard.dash 12
$ terraform import logicmonitor_dashboard.dash LakersDash
```

» logicmonitor_dashboard_group

Provides a LogicMonitor dashboard group resource. This can be used to create and manage LogicMonitor dashboard groups. Currently only creating a dashboard group based on an existing JSON template is supported

» Example Usage

```
# Create a new LogicMonitor dashboard group
resource "logicmonitor_dashboard_group" "dashgrp" {
  name = "Fadeaway"
  description = "a new dashboard group"
  widget_tokens {
    "defaultResourceGroup" = "Lakers/SignatureMoves"
  }
}

# Create a new LogicMonitor dashboard group based on a template stored in s3
resource "logicmonitor_dashboard_group" "dashgrp" {
  name = "Fadeaway"
  description = "a new dashboard group"
  widget_tokens {
    "defaultResourceGroup" = "Lakers/SignatureMoves"
  }
  template = "${data.aws_s3_bucket_object.template.body}"
}
```

```
data "aws_s3_bucket_object" "template" {
  bucket = "production-dashboards"
  key = "Ballers.json"
}
```

» Argument Reference

The following arguments are supported:

- `name` - (Required) Name of dashboard
- `description` - (Optional) Description of dashboard
- `parent_id` - (Optional) The id of the parent group for this dashboard group
- `force_delete` - (Optional) Force delete the dashboard group
- `template` - (Optional) Defines if an existing exported JSON template is used to create dashboard group
- `widget_tokens` - (Optional) Dashboard tokens allow users to apply a single dashboard group template to different device or website groups simply by changing the tokens' values.

» Import

Device Groups can be imported using their group id or full path

```
$ terraform import logicmonitor_dashboard_group.dashgrp 12
$ terraform import logicmonitor_dashboard_group.dashgrp LakersDash/Players
```

» logicmonitor_device

Provides a LogicMonitor device resource. This can be used to create and manage LogicMonitor devices

» Example Usage

```
# Create a new LogicMonitor device
resource "logicmonitor_device" "host" {
  ip_addr = "10.32.12.18"
  disable_alerting = true
  collector = "2"
  properties = {
   "app" = "haproxy"
   "system.categories" = "a,b,c,d"
  }
}

# Create a new LogicMonitor device and device group with some data source lookups and comput
resource "logicmonitor_device" "host" {
  ip_addr = "10.32.12.18"
  disable_alerting = true
  collector = "${data.logicmonitor_collectors.collectors.id}"
  hostgroup_id = "${logicmonitor_device_group.group1.id}"
  properties = {
   "app" = "haproxy"
   "system.categories" = "a,b,c,d"
  }
}

resource "logicmonitor_device_group" "group1" {
    name = "newgroup"
    properties = {
     "system.categories" = "ec2"
     "jmx.port" = "3008"
     "snmp.version" = "v2c"
    }
}

data "logicmonitor_collectors" "collectors" {
  most_recent = true
}
```

» Argument Reference

The following arguments are supported:

- `ip_addr` - (Required) Ip Address/Hostname of device
- `collector` - (required) The id of the collector that will monitoring the device
- `display_name` - (Optional) Display name of device, (default is ip_addr)
- `disable_alerting` - (Optional) The host is created with alerting disabled (default is true)
- `hostgroup_id` - (Optional) The host group id that specifies which group the device belongs to (multiple host group ids can be added, represented by a comma separated string)
- `properties` - (Optional) The properties associated with this device group. Any string value pair will work (see example).

» Import

Devices can be imported using their device id or ip address/dns name

```
$ terraform import logicmonitor_device.host 751
$ terraform import logicmonitor_device.host server01.us-east-1.logicmonitor.net
```

» logicmonitor_device_group

Provides a LogicMonitor device group resource. This can be used to create and manage LogicMonitor device groups

» Example Usage

```
# Create a new LogicMonitor device group
resource "logicmonitor_device_group" "group" {
    name = "NewTestGroup"
    description = "new test group"
    properties = {
     "group" = "test"
     "system.categories" = "a,b,c,d"
    }
}
```

```
# Create a new LogicMonitor dynamic device group
resource "logicmonitor_device_group" "group1" {
    name = "NewDynamicGroup"
    description = "new dynamic group"
    applies_to = "system.displayname =~ \"Prod\""
}
```

» Argument Reference

The following arguments are supported:

- `name` - (Required) Name of device group
- `description` - (Optional) Description of device group
- `parent_id` - (Optional) The id of the parent group for this device group (the root device group has an Id of 1)
- `applies_to` - (Optional) The Applies to custom query for this group. Setting this field will make this a dynamic group.
- `disable_alerting` - (Optional) Indicates whether alerting is disabled (true) or enabled (false) for this device group

- `properties` - (Optional) The properties associated with this device group. Any string value pair will work (see example).

» Import

Device Groups can be imported using their group id or full path

```
$ terraform import logicmonitor_device_group.group1 451
$ terraform import logicmonitor_device_group.group1 Production/SBA/Linux
```