Article
Henrique Dias · Apr 20, 2020 7m read

Open Exchange

IRIS History Monitor using custom built-in REST API /api/monitor/metrics

Hi everyone,

The project IRIS History Monitor received an update, using **ZPM** and the built-in REST API **/api/monitor/metrics**.

I created a new csp page to show the capabilities of the API and the possibilities that came with IRIS 2019.4. So, for those who are still using the project in Caché installations or IRIS before 2019.4, can keep using it with no worries.

These two excellent articles by @Murray Oldfield described the built-in API and how to use it:

https://community.intersystems.com/post/monitoring-intersystems-iris-using-built-rest-api

https://community.intersystems.com/post/example-review-monitor-metrics-intersystems-iris-using-default-rest-api

The endpoint /metrics give us a big list, but to provide the data source to all existent charts, I needed even more information.

And to our rescue, the documentation provides what I was looking for: Create Application Metrics!

https://docs.intersystems.com/irislatest/csp/docbook/Doc.View.cls?KEY=GCMrest#GCMrestmetricsapplication

To add custom application metrics to those returned by the /metrics endpoint:

1. Create a new class that inherits from %SYS.Monitor.SAM.Abstract.
2. Define the PRODUCT parameter as the name of your application.
3. Implement the GetSensors() method to define the desired custom metrics. The method must return $$$OK if successful, and contains one or more calls to the SetSensor() method, which sets the name, value, and an optional label for each metric.

Following the directions provided by documentation, I create the class ***diashenrique.historymonitor.util.customSensors*** that inherits from **%SYS.Monitor.SAM.Abstract**.

```
/// Example of a custom class for the /metric API
Class diashenrique.historymonitor.util.customSensors Extends %SYS.Monitor.SAM.Abstrac
t
{

Parameter PRODUCT = "irismonitor";

/// Collect metrics from
Method GetSensors() As %Status
{
    Do ..getDashboardWS(.dashboard)

    Do ..SetSensor("systemuptime",dashboard.SystemUpTime)
```

```
    Do ..SetSensor("lastbackup",dashboard.LastBackup)
    Do ..SetSensor("locktable",dashboard.LockTable)
    Do ..SetSensor("journalspace",dashboard.JournalSpace)
    Do ..SetSensor("journalstatus",dashboard.JournalStatus)
    Do ..SetSensor("ecpappserver",dashboard.ECPAppServer)
    Do ..SetSensor("ecpdataserver",dashboard.ECPDataServer)
    Do ..SetSensor("writedaemon",dashboard.WriteDaemon)
    Do ..SetSensor("licensecurrent",dashboard.LicenseCurrent)
    Do ..SetSensor("licensecurrentpct",dashboard.LicenseCurrentPct)
    Do ..SetSensor("licensehigh",dashboard.LicenseHigh)
    Do ..SetSensor("licensehighpct",dashboard.LicenseHighPct)
    Do ..SetSensor("licenselimit",dashboard.LicenseLimit)
    Do ..SetSensor("applicationerrors",dashboard.ApplicationErrors)

    Return $$$OK
}

ClassMethod getDashboardWS(Output dashboard)
{
    New $Namespace
    Set $Namespace = "%SYS"

    Do ##class(Config.Startup).Get(.Prop)
    Set webPort = Prop("WebServerPort")

    Set client = ##class(SYS.WSMon.Client).%New()
    Set client.Location = "http://localhost:"_webPort_"/csp/sys/SYS.WSMon.Service.cls
"
    Set dashboard = client.GetDashboard()

    Quit $$$OK
}
}
```

For the source of additional information, I chose the Web Service: **SYS.WSMon.Service**.

The web service provides a lot of Web Methods, and it's all described here:

https://docs.intersystems.com/irislatest/csp/docbook/DocBook.UI.Page.cls?KEY=GCMwsmon

With all set, at the end of our Installer Manifest, our Custom Application Metrics is called:

```
ClassMethod CustomApplicationMetrics() As %Status
{
  New $Namespace
  Set $Namespace = "%SYS"
  Set status = ##class(SYS.Monitor.SAM.Config).AddApplicationClass("diashenrique.hist
orymonitor.util.customSensors","IRISMONITOR")

  Quit status
}
```

After that, our final result for the REST API http://localhost:52773/api/monitor/metrics is:

```
iris_cpu_pct{id="CSPDMN"} 0
iris_cpu_pct{id="CSPSRV"} 0
iris_cpu_pct{id="ECPWorker"} 0
```

```
iris_cpu_pct{id="GARCOL"} 0
iris_cpu_pct{id="JRNDMN"} 0
iris_cpu_pct{id="LICENSESRV"} 0
iris_cpu_pct{id="WDSLAVE"} 0
iris_cpu_pct{id="WRTDMN"} 0
iris_cpu_usage 7
iris_csp_activity{id="127.0.0.1:52773"} 746
iris_csp_actual_connections{id="127.0.0.1:52773"} 10
iris_csp_gateway_latency{id="127.0.0.1:52773"} .585
iris_csp_in_use_connections{id="127.0.0.1:52773"} 1
iris_csp_private_connections{id="127.0.0.1:52773"} 0
iris_csp_sessions 3
iris_cache_efficiency 177.284
iris_db_expansion_size_mb{id="IRISAUDIT"} 0
iris_db_expansion_size_mb{id="IRISLOCALDATA"} 0
iris_db_expansion_size_mb{id="IRISMONITOR"} 0
iris_db_expansion_size_mb{id="IRISSYS"} 0
iris_db_expansion_size_mb{id="IRISTEMP"} 0
iris_db_expansion_size_mb{id="USER"} 0
iris_db_free_space{id="IRISAUDIT"} .27
iris_db_free_space{id="IRISLOCALDATA"} .2
iris_db_free_space{id="IRISMONITOR"} .16
iris_db_free_space{id="IRISSYS"} 9.4
iris_db_free_space{id="IRISTEMP"} 9.6
iris_db_free_space{id="USER"} .38
iris_db_latency{id="IRISAUDIT"} 0.003
iris_db_latency{id="IRISMONITOR"} 0.003
iris_db_latency{id="IRISSYS"} 0.017
iris_db_latency{id="IRISTEMP"} 0.003
iris_db_latency{id="USER"} 0.002
iris_db_max_size_mb{id="IRISAUDIT"} 0
iris_db_max_size_mb{id="IRISLOCALDATA"} 0
iris_db_max_size_mb{id="IRISMONITOR"} 0
iris_db_max_size_mb{id="IRISSYS"} 0
iris_db_max_size_mb{id="IRISTEMP"} 0
iris_db_max_size_mb{id="USER"} 0
iris_db_size_mb{id="USER",dir="/durable/irissys/mgr/user/"} 1
iris_db_size_mb{id="IRISSYS",dir="/durable/irissys/mgr/"} 90
iris_db_size_mb{id="IRISTEMP",dir="/durable/irissys/mgr/iristemp/"} 11
iris_db_size_mb{id="IRISAUDIT",dir="/durable/irissys/mgr/irisaudit/"} 1
iris_db_size_mb{id="IRISMONITOR",dir="/opt/irisapp/IRISMONITOR/"} 11
iris_db_size_mb{id="IRISLOCALDATA",dir="/durable/irissys/mgr/irislocaldata/"} 1
iris_directory_space{id="USER",dir="/durable/irissys/mgr/user/"} 39209
iris_directory_space{id="IRISSYS",dir="/durable/irissys/mgr/"} 39209
iris_directory_space{id="IRISTEMP",dir="/durable/irissys/mgr/iristemp/"} 39209
iris_directory_space{id="IRISAUDIT",dir="/durable/irissys/mgr/irisaudit/"} 39209
iris_directory_space{id="IRISMONITOR",dir="/opt/irisapp/IRISMONITOR/"} 39209
iris_disk_percent_full{id="USER",dir="/durable/irissys/mgr/user/"} 34.45
iris_disk_percent_full{id="IRISSYS",dir="/durable/irissys/mgr/"} 34.45
iris_disk_percent_full{id="IRISTEMP",dir="/durable/irissys/mgr/iristemp/"} 34.45
iris_disk_percent_full{id="IRISAUDIT",dir="/durable/irissys/mgr/irisaudit/"} 34.45
iris_disk_percent_full{id="IRISMONITOR",dir="/opt/irisapp/IRISMONITOR/"} 34.45
iris_ecp_conn 0
iris_ecp_conn_max 2
iris_ecp_connections 0
iris_ecp_latency 0
iris_ecps_conn 0
iris_ecps_conn_max 1
iris_glo_a_seize_per_sec 0
```

```
iris_glo_n_seize_per_sec 0
iris_glo_ref_per_sec 32
iris_glo_ref_rem_per_sec 0
iris_glo_seize_per_sec 0
iris_glo_update_per_sec 4
iris_glo_update_rem_per_sec 0
iris_journal_size 1472
iris_journal_space 36141.84
iris_jrn_block_per_sec 0
iris_jrn_entry_per_sec 0
iris_jrn_free_space{id="WIJ",dir="default"} 36141.84
iris_jrn_free_space{id="primary",dir="/durable/irissys/mgr/journal/"} 36141.84
iris_jrn_free_space{id="secondary",dir="/durable/irissys/mgr/journal/"} 36141.84
iris_jrn_size{id="WIJ"} 100
iris_jrn_size{id="primary"} 1
iris_jrn_size{id="secondary"} 0
iris_license_available 3
iris_license_consumed 2
iris_license_percent_used 40
iris_log_reads_per_sec 25
iris_obj_a_seize_per_sec 0
iris_obj_del_per_sec 2
iris_obj_hit_per_sec 5
iris_obj_load_per_sec 0
iris_obj_miss_per_sec 1
iris_obj_new_per_sec 2
iris_obj_seize_per_sec 0
iris_page_space_per_cent_used 0
iris_phys_mem_per_cent_used 96
iris_phys_reads_per_sec 0
iris_phys_writes_per_sec 0
iris_process_count 32
iris_rtn_a_seize_per_sec 0
iris_rtn_call_local_per_sec 37
iris_rtn_call_miss_per_sec 0
iris_rtn_call_remote_per_sec 0
iris_rtn_load_per_sec 0
iris_rtn_load_rem_per_sec 0
iris_rtn_seize_per_sec 3
iris_sam_get_db_sensors_seconds .000745
iris_sam_get_jrn_sensors_seconds .000811
iris_system_alerts 2
iris_system_alerts_new 1
iris_system_state 0
iris_trans_open_count 0
iris_trans_open_secs 0
iris_trans_open_secs_max 0
iris_wd_buffer_redirty 0
iris_wd_buffer_write 0
iris_wd_cycle_time 706
iris_wd_proc_in_global 0
iris_wd_size_write 0
iris_wd_sleep 9006
iris_wd_temp_queue 81
iris_wd_temp_write 0
iris_wdwij_time 496
iris_wd_write_time 209
iris_wij_writes_per_sec 0
irismonitor_applicationerrors 1
```

```
irismonitor_ecpappserver OK
irismonitor_ecpdataserver OK
irismonitor_journalspace Normal
irismonitor_journalstatus Normal
irismonitor_lastbackup
irismonitor_licensecurrent 2
irismonitor_licensecurrentpct 40
irismonitor_licensehigh 2
irismonitor_licensehighpct 40
irismonitor_licenselimit 5
irismonitor_locktable Normal
irismonitor_systemuptime 0d  1h 18m
irismonitor_writedaemon Normal
```

Remember the parameter PRODUCT in our class?

```
Parameter PRODUCT = "irismonitor";
```

The parameter became the prefix of our customer information.

I hope you enjoyed the example and that it can serve as a reference for your code.

#CSP #Dashboards #Monitoring #InterSystems IRIS
Check the related application on InterSystems Open Exchange

Source URL:https://community.intersystems.com/post/iris-history-monitor-using-custom-built-rest-api-apimonitormetrics