Article
Lorenzo Scalese · Apr 15, 2021 6m read

Open Exchange

Environment setup with config-api

Hi Developers,

Writing a script for the application deployment can be very interesting to ensure rapid deployment without forgetting anything.
config-api is a library to help developers to write configuration scripts based on a JSON document.

Implemented features :

- Set system settings.
- Set security settings.
- Enable services.
- Configure namespaces, databases, mapping.
- Export existing configuration.
- All features are exposed with a RESTful API.

This library is focused on IRIS configuration to help applications deployment. So, config-api doesn't import\compile code feature, considering it should be the role of your application installer module or the client registry. config-api could be used with the ZPM client to configure IRIS settings on module deployment, we learn how to combine this library with ZPM in another article.

Install

```
zpm “install config-api”
```

If you aren't a ZPM user, download the latest version in XML format with dependencies release page import and compile.

First step

Let ’ s write a simple configuration JSON document to set a few system settings.
In this first document we :

- Enable journal Freeze on error.
- Set limit journal size to 256 MB.
- Set SystemMode to development.
- Increase the locksiz.
- Increase the LockThreshold.

```
Set config = {
  "Journal": {                              /* Service class Api.Config.Journal */
       "FreezeOnError":1,
       "FileSizeLimit":256
   },
```

```
   "SQL": {                                    /* Service class Api.Config.SQL */
       "LockThreshold" : 2500
   },
   "config": {                                 /* Service class Api.Config.config */
       "locksiz" : 33554432
   },
   "Startup":{                                 /* Service class Api.Config.Startup */
       "SystemMode" : "DEVELOPMENT"
   }
}
Set sc = ##class(Api.Config.Services.Loader).Load(config)
```

configuration JSON document structure

The first level keys (Journal,SQL,config,Startup) are related to classes in %SYS namespace (using an intermediate class in Api.Config.Services package). It means Journal support all properties available in Config.Journal, SQLall properties in Config.SQL, etc...

Output :

```
2021-03-31 18:31:54 Start load configuration
2021-03-31 18:31:54 {
  "Journal":{
    "FreezeOnError":1,
            "FileSizeLimit":256
  },
  "SQL":{
    "LockThreshold":2500
  },
  "config":{
    "locksiz":33554432
  },
  "Startup":{
    "SystemMode":"DEVELOPMENT"
  }
}
2021-03-31 18:31:54  * Journal
2021-03-31 18:31:54    + Update Journal ... OK
2021-03-31 18:31:54  * SQL
2021-03-31 18:31:54    + Update SQL ... OK
2021-03-31 18:31:54  * config
2021-03-31 18:31:54    + Update config ... OK
2021-03-31 18:31:54  * Startup
2021-03-31 18:31:54    + Update Startup ... OK
```

Trick : The Load method is compatible with a string argument, in this case, the string must be a filename to a JSON configuration document (stream object is also permitted).

Create an application environment

In this section, we write a configuration document to create :

- A namespace "MYAPP".
- 4 databases (MYAPPDATA, MYAPPCODE, MYAPPARCHIVE,MYAPPLOG)

- 1 CSP Web Application (/csp/zwebapp).
- 1 REST Web Application (/csp/zrestapp).
- Setup globals mapping.

```
Set config = {
    "Defaults":{
        "DBDIR" : "${MGRDIR}",
        "WEBAPPDIR" : "${CSPDIR}",
        "DBDATA" : "${DBDIR}myappdata/",
        "DBARCHIVE" : "${DBDIR}myapparchive/",
        "DBCODE" : "${DBDIR}myappcode/",
        "DBLOG" : "${DBDIR}myapplog/"
    },
    "SYS.Databases":{
        "${DBDATA}" : {"ExpansionSize":128},
        "${DBARCHIVE}" : {},
        "${DBCODE}" : {},
        "${DBLOG}" : {}
    },
    "Databases":{
        "MYAPPDATA" : {
            "Directory" : "${DBDATA}"
        },
        "MYAPPCODE" : {
            "Directory" : "${DBCODE}"
        },
        "MYAPPARCHIVE" : {
            "Directory" : "${DBARCHIVE}"
        },
        "MYAPPLOG" : {
            "Directory" : "${DBLOG}"
        }
    },
    "Namespaces":{
        "MYAPP": {
            "Globals":"MYAPPDATA",
            "Routines":"MYAPPCODE"
        }
    },
    "Security.Applications": {
        "/csp/zrestapp": {
            "DispatchClas" : "my.dispatch.class",
            "Namespace" : "MYAPP",
            "Enabled" : "1",
            "AuthEnabled": "64",
            "CookiePath" : "/csp/zrestapp/"
        },
        "/csp/zwebapp": {
            "Path": "${WEBAPPDIR}zwebapp/",
            "Namespace" : "MYAPP",
            "Enabled" : "1",
            "AuthEnabled": "64",
            "CookiePath" : "/csp/zwebapp/"
        }
    },
    "MapGlobals":{
        "MYAPP": [{
            "Name" : "Archive.Data",
            "Database" : "MYAPPARCHIVE"
```

```
        },{
            "Name" : "App.Log",
            "Database" : "MYAPPLOG"
        }]
    }
}
Set sc = ##class(Api.Config.Services.Loader).Load(config)
```

Output :

```
2021-03-31 20:20:07 Start load configuration
2021-03-31 20:20:07 {
  "SYS.Databases":{
    "/usr/irissys/mgr/myappdata/":{
      "ExpansionSize":128
    },
    "/usr/irissys/mgr/myapparchive/":{
    },
    "/usr/irissys/mgr/myappcode/":{
    },
    "/usr/irissys/mgr/myapplog/":{
    }
  },
  "Databases":{
    "MYAPPDATA":{
      "Directory":"/usr/irissys/mgr/myappdata/"
    },
    "MYAPPCODE":{
      "Directory":"/usr/irissys/mgr/myappcode/"
    },
    "MYAPPARCHIVE":{
      "Directory":"/usr/irissys/mgr/myapparchive/"
    },
    "MYAPPLOG":{
      "Directory":"/usr/irissys/mgr/myapplog/"
    }
  },
  "Namespaces":{
    "MYAPP":{
      "Globals":"MYAPPDATA",
      "Routines":"MYAPPCODE"
    }
  },
  "Security.Applications":{
    "/csp/zrestapp":{
      "DispatchClas":"my.dispatch.class",
      "Namespace":"MYAPP",
      "Enabled":"1",
      "AuthEnabled":"64",
      "CookiePath":"/csp/zrestapp/"
    },
    "/csp/zwebapp":{
      "Path":"/usr/irissys/csp/zwebapp/",
      "Namespace":"MYAPP",
      "Enabled":"1",
      "AuthEnabled":"64",
      "CookiePath":"/csp/zwebapp/"
```

```
    }
  },
  "MapGlobals":{
    "MYAPP":[
      {
        "Name":"Archive.Data",
        "Database":"MYAPPARCHIVE"
      },
      {
        "Name":"App.Log",
        "Database":"MYAPPLOG"
      }
    ]
  }
}
2021-03-31 20:20:07  * SYS.Databases
2021-03-31 20:20:07    + Create /usr/irissys/mgr/myappdata/ ... OK
2021-03-31 20:20:07    + Create /usr/irissys/mgr/myapparchive/ ... OK
2021-03-31 20:20:07    + Create /usr/irissys/mgr/myappcode/ ... OK
2021-03-31 20:20:07    + Create /usr/irissys/mgr/myapplog/ ... OK
2021-03-31 20:20:07  * Databases
2021-03-31 20:20:07    + Create MYAPPDATA ... OK
2021-03-31 20:20:07    + Create MYAPPCODE ... OK
2021-03-31 20:20:07    + Create MYAPPARCHIVE ... OK
2021-03-31 20:20:07    + Create MYAPPLOG ... OK
2021-03-31 20:20:07  * Namespaces
2021-03-31 20:20:07    + Create MYAPP ... OK
2021-03-31 20:20:07  * Security.Applications
2021-03-31 20:20:07    + Create /csp/zrestapp ... OK
2021-03-31 20:20:07    + Create /csp/zwebapp ... OK
2021-03-31 20:20:07  * MapGlobals
2021-03-31 20:20:07    + Create MYAPP Archive.Data ... OK
2021-03-31 20:20:07    + Create MYAPP App.Log ... OK
```

It works! The configuration is successfully loaded.

In the next article, we learn how to use config-api with ZPM to deploy your application.

The app is submitted for the contest, vote for it if you like it ;-) .

Thanks for reading.

#Best Practices #Deployment #DevOps #InterSystems IRIS
Check the related application on InterSystems Open Exchange