Artículo
Ricardo Paiva · 14 ene, 2022 Lectura de 6 min

Open Exchange

Configuración del entorno con config-api

Hola desarrolladores,

Escribir un *script* para el despliegue de una aplicación puede ser muy interesante para garantizar un despliegue rápido sin olvidarse de nada.
config-api es una biblioteca para ayudar a los desarrolladores a escribir *scripts* de configuración basados en un documento JSON.

Características implementadas:

- Establecer la configuración del sistema
- Establecer la configuración de seguridad
- Habilitar servicios
- Configurar *namespaces*, bases de datos y mapeos
- Exportar configuración existente
- Todas las funciones están expuestas con una API RESTful

Esta biblioteca se centra en la configuración de IRIS para ayudar a la implementación de aplicaciones. Por lo tanto, config-api no importa/compila la función de código, considerando queesa debería ser la función del módulo de instalación de tu aplicación o el registro del cliente. config-api podría usarse con el cliente ZPM para configurar los ajustes de IRIS en la implementación del módulo; aprenderemos cómo combinar esta biblioteca con ZPM en otro artículo.

Instalación

```
zpm “install config-api”
```

Si no eres usuario de ZPM, descarga la última versión en formato XML con dependencias página de publicación importación y compilación.

Primer paso

Vamos a escribir un documento JSON de configuración simple, para establecer algunas configuraciones del sistema.
En este primer documento:

- Habilitamos *journal freeze* en caso de error.
- Establecemos el tamaño límite de *journal* en 256 MB.
- Establecemos SystemMode para desarrollo.
- Aumentamos el locksiz.
- Aumentamos el 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)
```

Estructura del documento JSON de configuración

Las claves de primer nivel (Journal,SQL,config,Startup) están relacionadas con las clases en el *namespace* %SYS (usando una clase intermedia en el paquete Api.Config.Services). Significa que Journal admite todas las propiedades disponibles en Config.Journal, SQLtodas las propiedades en Config.SQL, etc...

Salida:

```
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
```

Truco: el método Load es compatible con un argumento de cadena, en este caso, la cadena debe ser un nombre de archivo para un documento de configuración JSON (también se permite objetos *stream*).

Crear un entorno de aplicación

En esta sección, escribimos un documento de configuración para crear:

- Un *namespace* "MYAPP"
- 4 bases de datos (MYAPPDATA, MYAPPCODE, MYAPPARCHIVE,MYAPPLOG)
- 1 aplicación web CSP (/csp/zwebapp)
- 1 aplicación web REST (/csp/zrestapp)
- Configuración de mapeo de *globals*

```
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)
```

Salida:

```
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
```

¡Funciona! La configuración se cargó con éxito.

En el próximo artículo, aprenderemos a usar config-api con ZPM para implementar tu aplicación.

#Despliegue #DevOps #Mejores prácticas #InterSystems IRIS
Ir a la aplicación en InterSystems Open Exchange