Artículo
Jose Tomas Salvador · Ene 28, 2020 Lectura de 5 min

Open Exchange

Editar y Compilar múltiples proyectos en InterSystems IRIS con ObjectScript utilizando VSCode y Docker

Hola Desarrolladores!

```
"objectscript.conn" :{
      "ns": "IRISAPP",
      "active": true,
      "docker-compose": {
        "service": "iris",
        "internalPort": 52773
      }
```

Quiero compartir con vosotros una estupenda nueva funcionalidad que he descubierto en la nueva versión 0.8 del plugin de VSCode ObjectScript desarrollado por @Dmitry Maslennikov y CaretDev.

La versión viene con un nuevo parámetro de configuración: "docker-compose"; que resuelve el problema con los puertos que necesitas para conectar tu editor VSCode a IRIS. No era muy cómodo si tenías más de un contenedor docker con IRIS corriendo en la misma máquina. Ahora ¡ esto está resuelto!

Veamos como funciona ahora...

El concepto de utilizar docker localmente para desarrollar con IRIS supone que tienes un *dockerfile* y un *docker-compose.yml* en el repositorio, con los que tu construyes el entorno del proyecto y cargas todo el código ObjectScript dentro del contenedor IRIS descargado de Docker Hub. Y también tienes el archivo *.vscode/settings.json* de VSCode en el repositorio, en el cual indicas el puerto del servidor web de IRIS al que te conectas (junto con otros parámetros de conexión tales como URL, Namespace y credenciales de acceso).

La cuestión es: ¿ cuál es el puerto al que se debería conectar VSCode?

Puedes utilizar el puerto 52773, que es el puerto por defecto del servidor web de IRIS. Pero si intentas lanzar un segundo contenedor docker, fallará porque no puedes ejecutar dos contenedores docker en la misma máquina que comparten conexiones en el mismo puerto. Puedes exponer un puerto externo para un contenedor docker y esto se puede configurar en el archivo *docker-compose.yml*. Aquí vemos un ejemplo (en negrita el puerto mapeado):

```
version: '3.6'
services:
  iris:
    build: .
    restart: always
    ports:
      - 52791:52773
    volumes:
      - ~/iris.key:/usr/irissys/mgr/iris.key
```

```
      - ./:/irisdev/app
```

De este modo, puedes mirar en el *docker-compose* y escribir el mismo puerto en *.vscode/settings.json*:

Entonces, te preguntarás, ¿ cuál es el problema?

El problema está en cuando expones tu proyecto como una librería o demo e invitas a la gente a ejecutar y editar código con *VSCode*. No quieres que tengan que configurar el puerto manualmente y quieres que puedan clonar el repositorio, ejecutar docker y tener la opción de colaborar inmediatamente. Y esta es la cuestión: ¿ a qué puertos deberías mapear tu proyecto en *docker-compose* de modo que no provoque ningún conflicto en el entorno de cualquier otro desarrollador?

Incluso si no lo expones a nadie más que a tí, ¿ qué puerto pondrías en los parámetros de la conexión en *.vscode/settings.json*?

La cuestión es que, cuando lanzas un nuevo docker con IRIS, ves un mensaje de error diciendo que el puerto ya está en uso y, o bien paras el otro contenedor o bien te inventas otro puerto que quizá no esté siendo utilizado y pruebas con él en *docker-compose* y *settings.json*.

Es aburrido, es una pérdida de tiempo, es un esfuerzo inútil. A nadie le gusta.

Y ocurre lo mismo si expones la librería.

La solución la encontramos con la nueva release 0.8 de VSCode ObjectScript, donde puedes introducir una sección *docker-compose*, que resuelve para siempre este tema:

```
"objectscript.conn" :{
      "ns": "IRISAPP",
      "active": true,
      "docker-compose": {
        "service": "iris",
        "internalPort": 52773
      }
```

Contiene los parámetros *service* e *internalPort* que le indican a VSCode que para encontrar el puerto al que debe conectarse debe mirar en el fichero *docker-compose.yml* que tenemos en el mismo repositorio e ir a la sección del servicio *"iris"* donde encontrará el puerto que está mapeado al puerto interno 52773.

¡ Bien!

Y lo que aún es mejor, Docker ahora tiene el modo para *docker-compose.yml* en el que puedes no configurar ningún puerto en docker, y en su lugar dejar el símbolo **"-"** como puerto mapeado, lo que significa que docker utilizará, de forma aleatoria, cualquier puerto que esté disponible.

```
iris:
    build:
      context: .
      dockerfile: Dockerfile-zpm
    restart: always
    ports:
      - 51773
      - 52773
      - 53773
    volumes:
      - ~/iris.key:/usr/irissys/mgr/iris.key
      - ./:/irisdev/app
```

¡ Bien, por partida doble! Porque esto te da la opción de no preocuparte nunca más sobre los puertos del servidor web de IRIS a los que se debe conectar VSCode.

¿ Cómo funciona en este caso? Ejecutamos*docker-compose.yml* , docker coge un puerto aleatorio para el servidor web y ejecuta IRIS con él, VSCode obtiene este puerto desde docker y conecta con IRIS a través de él de modo que tú puedes editar y compilar código inmediatamente. Sin configuraciónes adicionales.

¡ Aprovecha!

Puedes sentir la magia con el siguiente template que modificamos recientemente de acuerdo a la nueva funciontalidad de VSCode ObjectScript 0.8 y que ya tiene modificados los ficheros settings.json y docker-compose.yml. Para probar el template ejecuta los siguientes comandos en el terminal (probado en Windows 10 y en Mac). También necesitas tener instalado git y Docker Desktop.

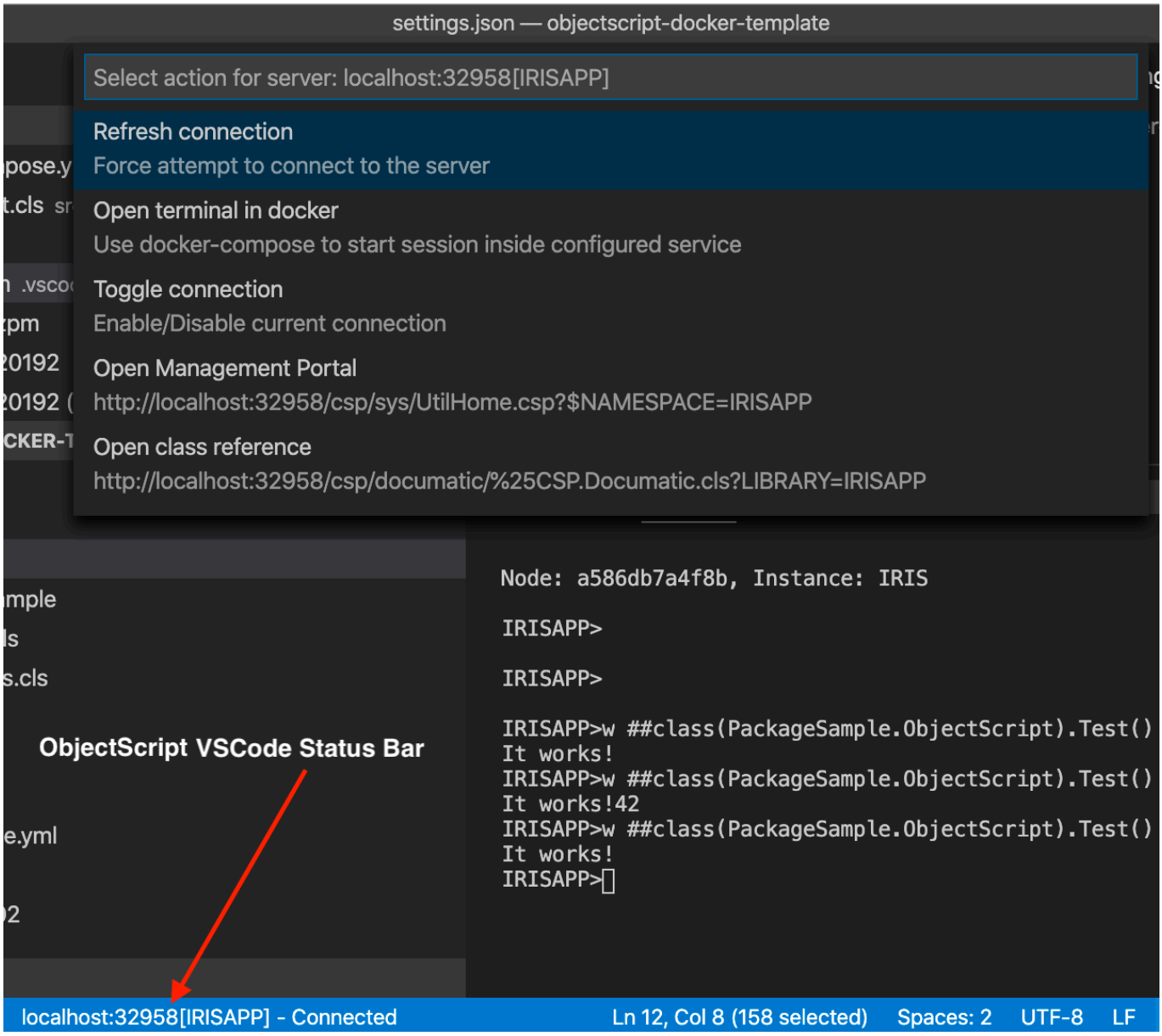
```
$ git clone https://github.com/intersystems-community/objectscript-docker-template.git

$ cd objectscript-docker-template

$ docker-compose up -d
```

Abre este directorio en VSCode (asegúrate de que tienes el plugin VScode ObjectScript instalado):

Comprueba si VSCode está conectado - haz click en la línea de estado de VSCode:



Reconecta VSCode si es necesario.

Si lo necesitas también puedes abrir un terminal IRIS en VSCode con el menú de ObjectScript (*).

¡ Hecho! Ahora puedes ejecutar, compilar y debugar el código!

¡ Disfruta!

Nota.- A fecha de hoy, la opción de abrir un terminal de IRIS directamente desde VSCode puede dar un error. No obstante, se puede entrar en la consola de IRIS desde VSCode abriendo un terminal y ejecutando el comando:

docker-compose exec iris /bin/bash -c 'command -v ccontrol >/dev/null 2>&1 && ccontrol session $ISC_PACKAGE_INSTANCENAME -U IRISAPP ¦¦ iris session $ISC_PACKAGE_INSTANCENAME -U IRISAPP'

Este problema será resuelto en posteriores versiones. Si te lo encuentras, añade un comentario para hacerle seguimiento.

#Docker #ObjectScript #VSCode #InterSystems IRIS #Open Exchange
Compruebe la aplicación relacionada en InterSystems Open Exchange

URL de fuente: https://es.community.intersystems.com/post/editar-y-compilar-m%C3%BAltiples-proyectos-en-intersystems-iris-con-objectscript-utilizando-vscode-y