Article
Evgeny Shvarov · Jan 25, 2020 4m read

Open Exchange

Working with Several ObjectScript Projects simultaneously Using VSCode and Docker in InterSystems IRIS

Hi Developers!

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

I want to share with you a nice new feature I came across in a new 0.8 release of VSCode ObjectScript plugin by @Dmitry Maslennikov and CaretDev.

The release comes with a new configuration setting "docker-compose" which solves the issue with ports you need to set up to make your VSCode Editor connect to IRIS. It was not very convenient if you had more than one docker container with IRIS running on the same machine. Now, this is solved!

Read below how it works now.

The concept of using docker locally for development with IRIS supposes that you have dockerfile and docker-compose.yml in the repository which you run to build the environment of the project and load all the ObjectScript code into IRIS container pulled from Docker Hub. And you also have the VSCode .vscode/settings.json file in the repository where you point the IRIS web-server port you connect to (along with other connection settings such as URL, Namespace and login credentials).

The question is - what is the port VSCode should connect to?

You can use the port 52773 which is default IRIS port for web server. But if you try to launch the second docker container it will fail cause you cannot run two docker containers on same machine which wait connections on the same port. But you can expose an external port for Docker container and this could be set up via docker-compose.yml file, here is an example (mapped port is bold):

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

So you look into the docker-compose and type the same port to .vscode/settings.json:

But where is the problem you ask me?

The problem is when you expose your project as a library or demo and invite people to run and edit code with VSCode you don't want them to setup the port manually and you want them to clone the repo, run docker and have the option to collaborate immediately. Here comes the question: what should you map your project to in docker-compose which will not have a conflict in someone's environment?

Even if you don't expose it to anyone but use for yourself - what port do you put in the .vscode/settings.jsonn connection settings?

The answer is that when you launch a new docker with IRIS, you see the error message that port has been already taken and you either stop other containers or invent a new port which is probably not taken and try with it in docker-compose and settings.json.

Boring. Time-consuming, useless operation. Nobody likes it.

And you introduce same if you expose the library.

The relief comes with the new 0.8 VSCode ObjectScript release where you can introduce docker-compose section, which solves the issue forever:

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

it contains service and internalPort parameters which say to VSCode that in order find the port to connect to it should look into the docker-compose.yml file we have in the same repo and find "iris" service section there and then get the port which is mapped for internal 52773 port.

Yey!

And what is even cooler, Docker now has the mode for docker-compose.yml when you are able not to set up any port in docker, but leave the "-" sign form mapped port, which means that docker will use a random available port.

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

Yey, two times! Cause this gives you an option not to bother about IRIS web-server ports VSCode connects to anymore whatsoever.

How it works in this case. We run docker-compose.yml, docker takes the random web-server port and runs IRIS with it, VSCode gets this port form docker and connects to IRIS with this port and you are able to edit and compile code immediately. With no additional settings.

Profit!

And you can reproduce the same fantastic feelings with the following template I updated recently according to the new feature of VSCode ObjectScript 0.8 which has the updated settings.json and docker-compose.yml. To test the template run the following commands in terminal (tested on Mac). You also need git and Docker Desktop to be installed.

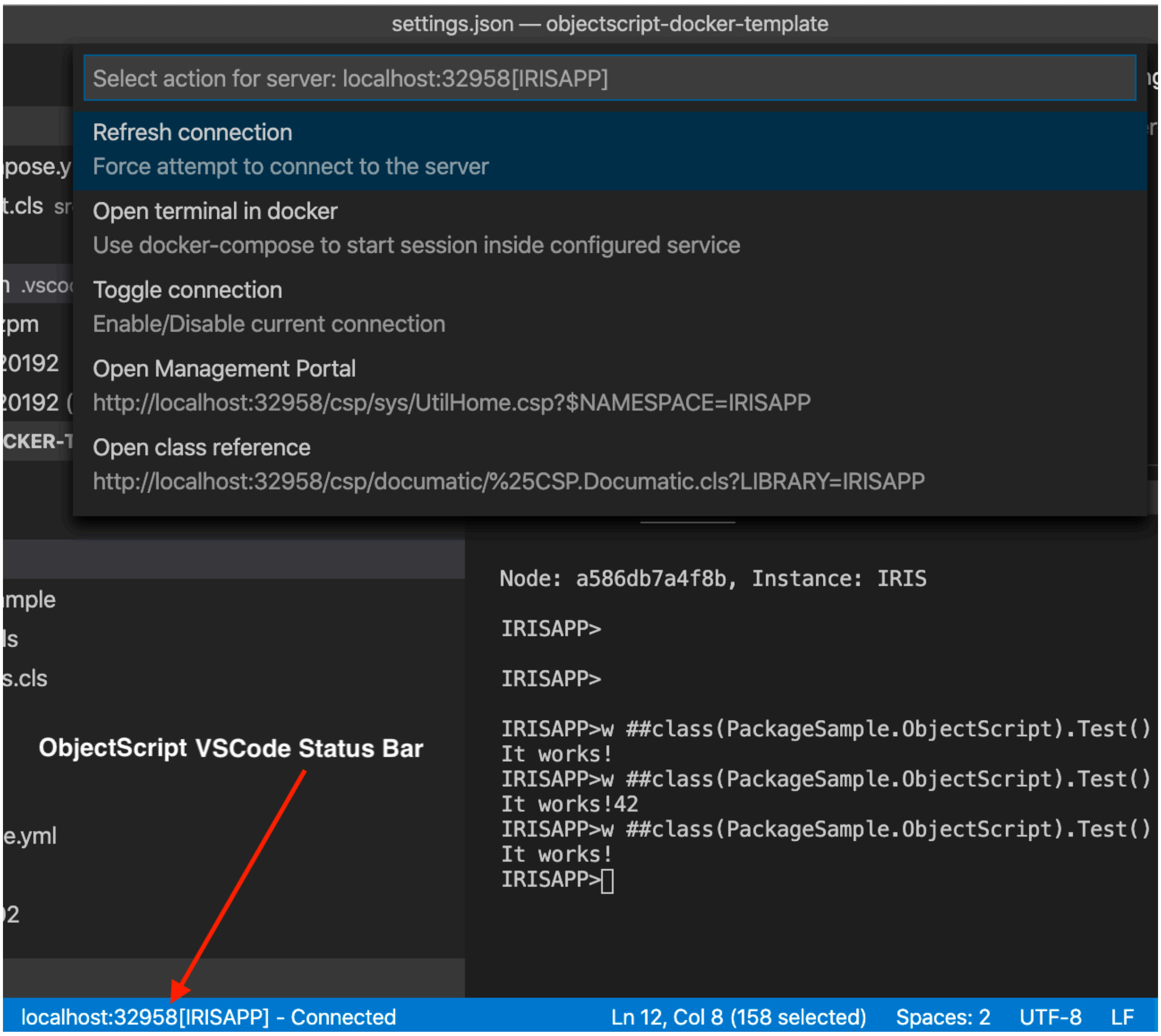
```
$ git clone https://github.com/intersystems-community/objectscript-docker-
template.git

$ cd objectscript-docker-template

$ docker-compose up -d
```

Open this folder in VSCode (make sure you have VScode ObjectScript plugin installed):

Check if VSCode is connected - click on VSCode status line:


settings.json — objectscript-docker-template
Select action for server: localhost:32958[IRISAPP]
Refresh connection
Force attempt to connect to the server
Open terminal in docker
Use docker-compose to start session inside configured service
Toggle connection
Enable/Disable current connection
Open Management Portal
http://localhost:32958/csp/sys/UtilHome.csp?$NAMESPACE=IRISAPP
Open class reference
http://localhost:32958/csp/documatic/%25CSP.Documatic.cls?LIBRARY=IRISAPP
ObjectScript VSCode Status Bar
localhost:32958[IRISAPP] - Connected
Ln 12, Col 8 (158 selected)
Spaces: 2
UTF-8
LF

```
Node: a586db7a4f8b, Instance: IRIS

IRISAPP>

IRISAPP>

IRISAPP>w ##class(PackageSample.ObjectScript).Test()
It works!
IRISAPP>w ##class(PackageSample.ObjectScript).Test()
It works!42
IRISAPP>w ##class(PackageSample.ObjectScript).Test()
It works!
IRISAPP>
```

Reconnect VSCode if needed.

You can open IRIS Terminal in VSCode if needed with the ObjectScript menu.

Done! Now you are able to run, compile and debug the code!

Happy coding!

#Best Practices #Docker #ObjectScript #InterSystems IRIS #Open Exchange #VSCode
Check the related application on InterSystems Open Exchange

Source URL:https://community.intersystems.com/post/working-several-objectscript-projects-simultaneously-using-vscode-and-docker-intersystems-iris