Container Deployment Reference

Deploy FrameworX using containers.

Container Deployment (Reference): Deploying FrameworX in Docker and Kubernetes environments.

Downloading the Image
Creating a Container
Starting an existing container
Licensing

Downloading the Image

To download the desired image, with the container platform running, use the following commands:

- **Full Build** (frameworx)**:**

```
docker pull tatsoftdockerhub/frameworx:latest
```

- **TWebServer Only** (twebservices)**:**

```
docker pull tatsoftdockerhub/twebservices:latest
```

Creating a Container

After downloading the image, create a container using the command below:

```
docker run -it --name <ContainerName> --network <networktype> --mount type=bind,source="<path to data folder>",target=/Documents <ImageName>
```

Parameters:

-it = Enables an interactive terminal.

<ContainerName> = Container Name.

<networktype> = Defines how the network is set up: host, bridge...

--mount type=bind = Binds the container's /Documents directory (where FrameworX stores project files and data) to a folder on your host machine. This ensures your data persists even if the container is removed — container data is otherwise volatile.

<path to data folder> = Local folder path that is mapped to the container's Documents directory, ensuring your data persists even if the container is removed. Ensure this folder exists before running the container.

Note: the folder in the host system needs to be created by the user.
Example:
If your desired host folder is /home/username/FX_Data, use the command below:

```
docker run -it --name fx-server --network host --mount type=bind,source="/home/username/FX_Data",target=/Documents tatsoftdockerhub/frameworx:latest
```

For Windows (Docker Desktop or similar)

When accessing the container on Windows (**Docker Desktop** or similar), use the default bridge network—not `host`—and map ports 10108 and 3101. Then, in SolutionCenter, connect to localhost (127.0.0.1); it will route to the Docker container via the mapped ports. Example:

```
docker run -p 10108:10108 -p 3101:3101 -it --name fx-server --mount type=bind,source="C:\Users\username\Documents\Docker",target=/Documents tatsoftdockerhub/frameworx:latest
```

Or, if you are using TWebServices Image:

```
docker run -it --name fx-server --network host --mount type=bind,source="/home/username/FX_Data",target=
/Documents tatsoftdockerhub/twebservices:latest
```

This will:

- Create a container named fx-server
- Use the full product image
- Share project data between the container and the local /home/username/FX_Data folder
- Run the container

Starting an existing container

To start a previously created and stopped container, use:

```
docker start -ia <ContainerName>
```

Parameters
-ia => Starts the container in interactive mode and attaches to the terminal.

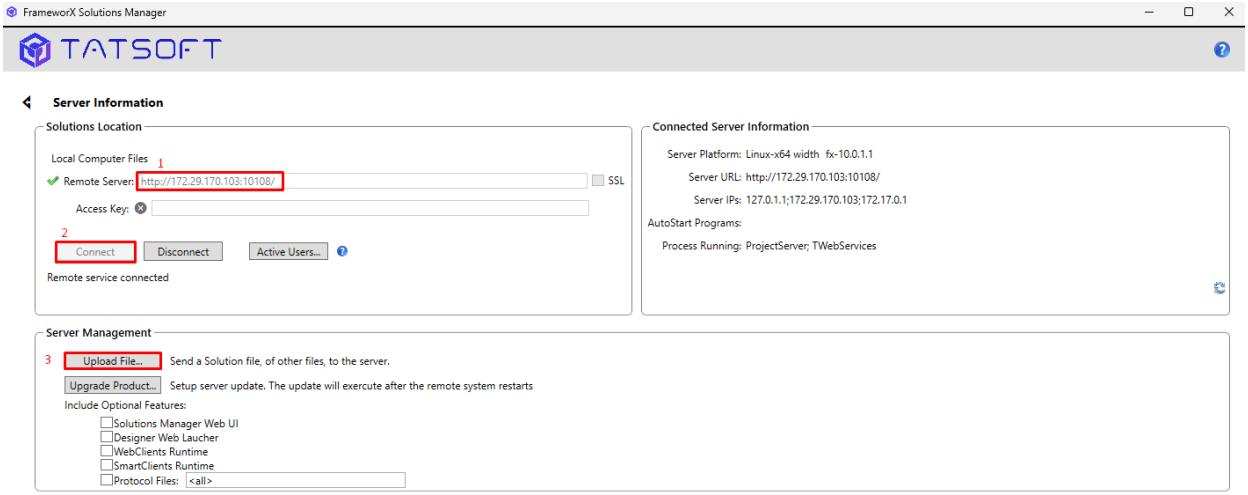
Expected Output

When the container starts, you should see output similar to this:

```
user@userNote:~$ docker start -ia fx-server
04/02/2025 13:12:07.579 - Starting service - Port: 10108...
04/02/2025 13:12:07.686 - TWebServices was started - Port: 10108, Local IPs:
127.0.1.1,172.29.170.103,172.17.0.1...
04/02/2025 13:12:07.691 - Type 'exit' or 'quit' to finish...
```

Connecting from Windows (frameworx Image)

Start the SolutionCenter on a Windows computer, configure the 'Remote Server' with the IP address shown in the Linux terminal, then click Connect, as shown below:



You can upload a solution to the Linux server, by clicking Upload File.

> When using Docker Desktop or a similar tool, connect to localhost (127.0.0.1:10108).

Changed in FrameworX 10.1.6: Windows-target solutions are refused

From FrameworX 10.1.6, a runtime on Linux or in a container refuses to start a solution whose target platform is Windows (.NET Framework 4.8). The refusal applies to every start path: Solution Center, the Designer **Run Startup** button, **Actions Start Runtime**, the command line and the automatic start at boot. The message names the solution and the platform, for example:

```
The solution 'MySolution' targets Windows (.NET Framework 4.8) and cannot run on this computer (Linux container).

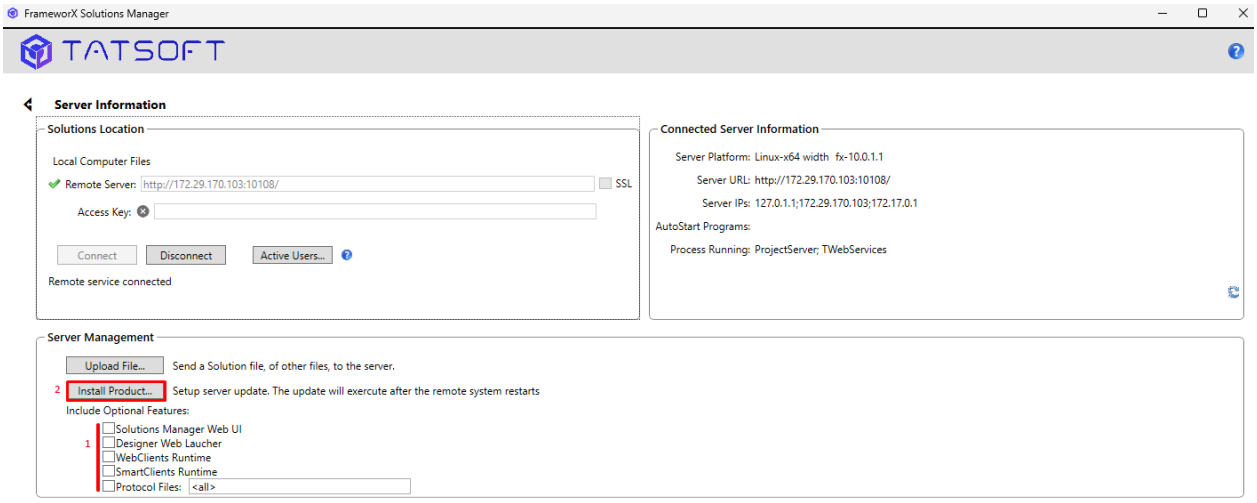
Open the solution in Designer, change the target platform to Multiplatform in Solution Settings, run a Build All, and try again.
```

To fix it, follow the message: in Designer, set **Target Platform** to **Multiplatform** in **Solution Settings**, run **Build All**, and upload the solution again. In 10.1.5, such a solution appears to start but never runs. Windows computers start every solution as before.

Also from 10.1.6, when the Multiplatform launcher stops during start-up, for example with no solution name, a port already in use, or a configuration file it cannot read, it exits instead of waiting indefinitely.

Connecting from Windows (twebservices Image)

When connecting to a container created with the twebservices image, you will see a screen like this:



In this screen:

1. Select the components you want to install.
2. Click **Install Product**.

Additional Feature

This Docker image already includes Python 3.10.12. If you need additional libraries, you can install them or contact Tatsoft Support for assistance.

Licensing

Since Docker works similarly to a virtual machine, the licensing procedure is almost the same.

Steps:

1. Set up a License Server. Instructions can be found at this link.
2. On the client side, the only difference is the location of the folder where the RemoteLicenseService.config file should be placed.
   If you're following the example on the referenced page, you'll have a shared folder with the container located at:
   /home/username/FX_Data
   The RemoteLicenseService.config file must be placed in the following path:
   /home/username/FX_Data/FrameworX/MachineSettings
   Note: The /FrameworX/MachineSettings folder is created automatically after the container is run for the first time.

In this section...