Cordial Docker Setup

Release 1.0

Aug 26, 2020

Contents

Contents:

CHAPTER 1

Setting up QT

1.1 What you'll need

- A monitor
- A mouse and keyboard
- Wireless internet
- An Amazon Web Services account
- A Google account to get an app for the tablet

QT has two ports: one USB-C and one USB-A. You'll need to use the USB-C port for display and figure out how to control a mouse and keyboard. I suggest a USB-C hub that has a display port that you can use (USB-C or HDMI, for example) and has USB-A ports for your mouse and keyboard. Alternatively, you can use a USB-A hub to connect your mouse and keyboard to QT.

Note: If you have trouble using your mouse or keyboard through a USB-C port, try flipping the USB-C input going into QT. In theory, USB-C should go both ways, but in practice, sometimes not.

1.2 Basics

1.2.1 Turning QT on and off

To turn on QT, just plug in power to QT and it will boot.

To turn off QT, there are two options:

1. Press the button on the backside of QT, near its feet.
2. Login to the head computer (see below) and do `sudo shutdown`.

If you do `sudo shutdown` on the body computer, you only turn off the body computer—the head computer is still on.

Note: If you unplug QT to restart QT, it may mess up the boot timing of the two computers. Probably one of them takes longer because it boots in recovery mode. This screws up how the head and body computer network. You will not be able to connect to the head computer from the body computer. If this occurs, simple restart QT by pushing the button on its backside.

1.2.2 Accessing QT's body computer

When you connect a monitor to QT and turn QT on, you will start on QT's body computer.

1.2.3 Accessing QT's head computer

To setup the head, you must Secure-SHell into it (SSH) from QT's body computer. To do this

0. Turn on QT.
1. Open a terminal.
2. Type the following and hit return:

```
ssh qtrobot@192.168.100.1
```

1.3 Head

1.3.1 Turning off the default face

0. If you haven't already, SSH into QT's head computer:

```
ssh qtrobot@192.168.100.1
```

1. Update QT:

```
cd ~/robot/packages/deb
git pull
sudo dpkg -i ros-kinetic-qt-robot-interface_1.1.8-0xenial_armhf.deb
```

Note: If the `git pull` step fails, the head computer might be having trouble with it its network. You can check this with `ping google.com`. If there's nothing, there is a problem with the network. To fix this, the best think we've found is to restart QT: `sudo reboot`.

---

2. Edit a configuration file to turn off QT's default face:

   a. Open the configuration file:

```
sudo nano /opt/ros/kinetic/share/qt_robot_interface/config/qtrobot-interface.
↪yaml
```

   b. Change the line that says `disable_interface: false` to `disable_interface: true`

c. Save and exit `nano` by hitting Ctrl+x, then typing 'y', and then hitting Enter twice to confirm things.

Note: You can reboot to see these changes take effect, or continue on and we'll reboot eventually.

1.3.2 Setting up our code

0. Secure-Shell (SSH) into QT's head computer:

```
ssh qtrobot@192.168.100.1
```

1. Install our project's dependencies:

```
git clone -b master https://github.com/robotpt/cordial-docker.git
bash ~/cordial-docker/scripts/pi_setup.bash
```

2. Increase the swap size, so we're able to build without running out of virtual memory:

   a. Turn off your swap memory:

   ```
   sudo /sbin/dphys-swapfile swapoff
   ```

   b. Open your swap configuration file:

   ```
   sudo nano /etc/dphys-swapfile
   ```

   c. Set *CONF_SWAPFACTOR* to 2 by changing the line that says `#CONF_SWAPFACTOR=2` to `CONF_SWAPFACTOR=2`, that is by deleting the `#` character to uncomment the line.

   d. Save and exit `nano` by hitting Ctrl+x, then typing 'y', and then hitting Enter twice to confirm things.

   e. Turn the swap file back on:

   ```
   sudo /sbin/dphys-swapfile swapon
   ```

3. Clone our repositories and build them:

   a. Go to the source code directory in the catkin workspace:

   ```
   cd ~/catkin_ws/src
   ```

   b. Clone our repositories:

   ```
   git clone -b master https://github.com/robotpt/cordial
   git clone -b master https://github.com/robotpt/qt-robot
   ```

   c. Build our workspace:

   ```
   cd ~/catkin_ws
   catkin_make
   ```

   Note: It takes around five minutes for this command to finish. You can setup QT's body computer at the same time as it runs, if you like.

4. Setup our code to run when QT's head computer turns on.

   a. Copy the autostart script into the correct directory:

```
roscp qt_robot_pi start_usc.sh /home/qtrobot/robot/autostart/
```

b. Enable the autostart script:

   i. Open a webbrowser on QT (e.g., Firefox) and go to http://192.168.100.1:8080/.

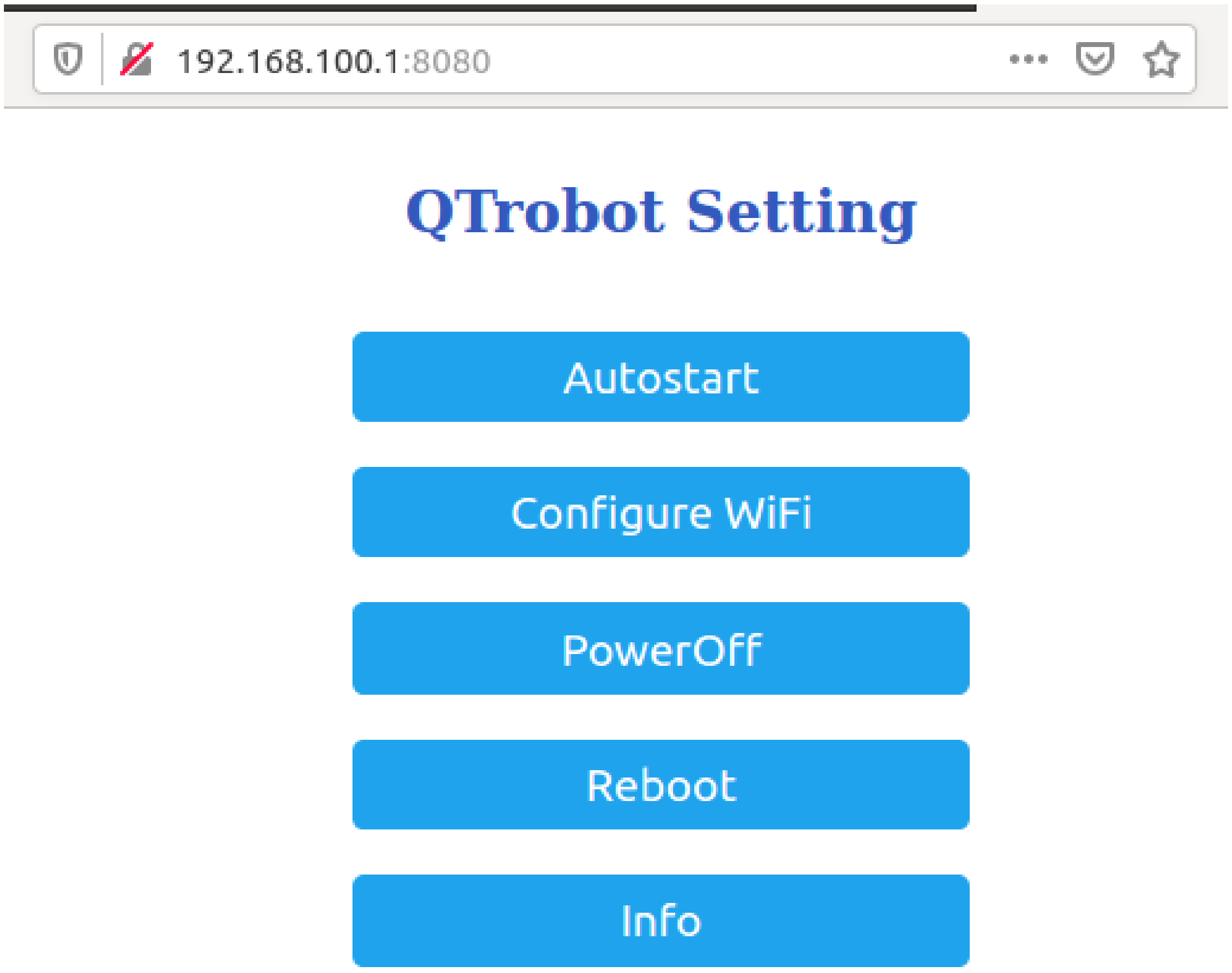


Fig. 1: QT's configuration menu.

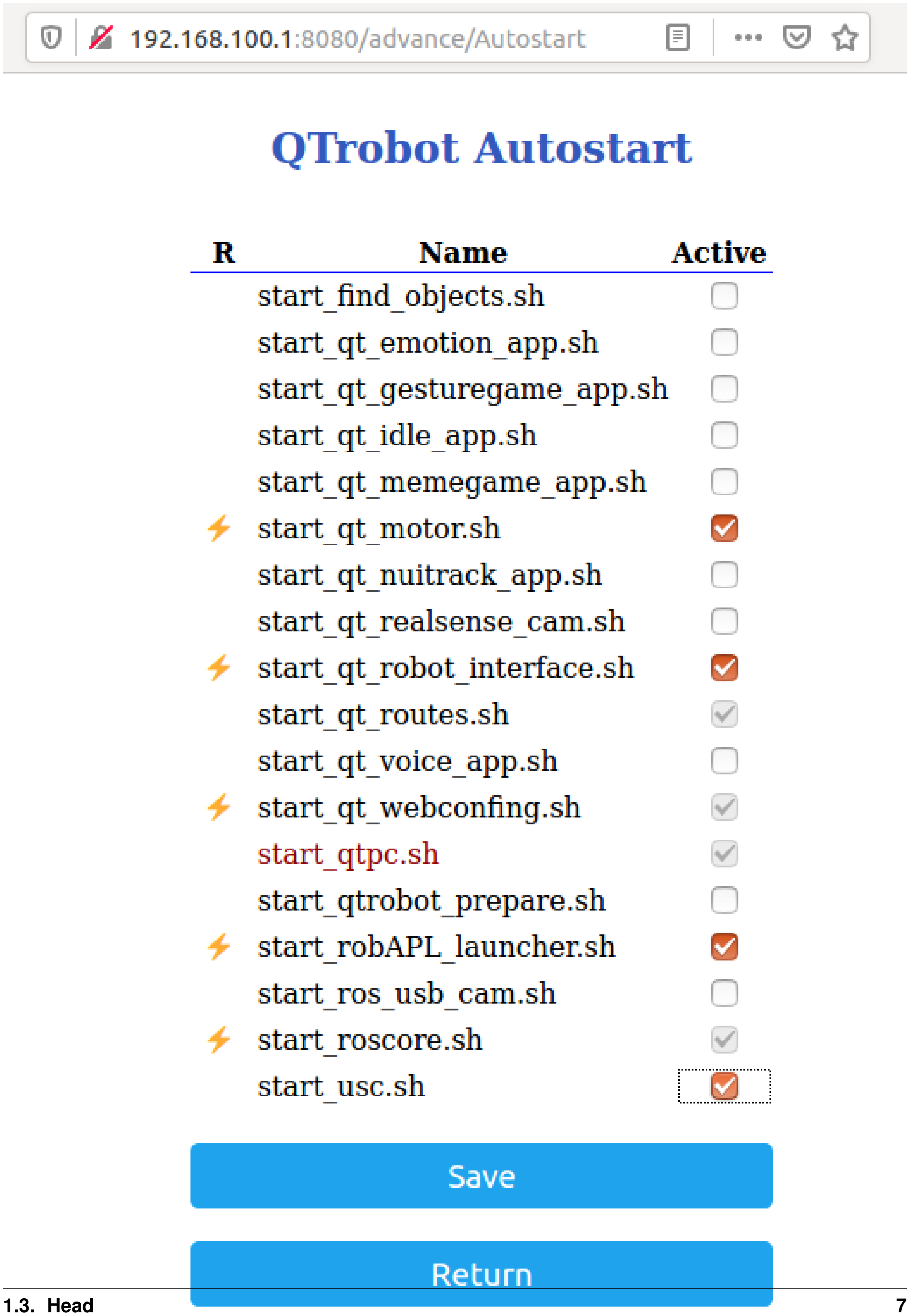
   ii. Click 'Autostart'. You'll be prompted for a username and password. Enter `qtrobot` for both.

   iii. Click the 'Active' checkbox next to `start_usc.sh`.

   iv. Click 'Save' and then 'Return' twice.

Note: You can reboot to see these changes take effect, or continue on and we'll reboot eventually.

If you'd like, you can confirm that things are running after a reboot by opening a terminal and running the following command. You should see both `/sound_listener` and `/start_face_server`:

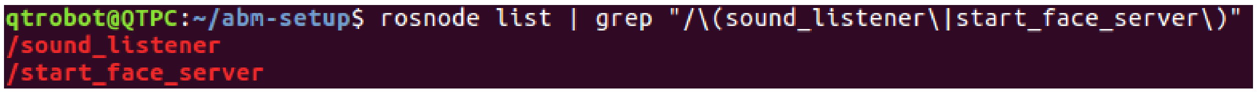


192.168.100.1:8080/advance/Autostart

QTrobot Autostart

R	Name	Active
	start_find_objects.sh	
	start_qt_emotion_app.sh	
	start_qt_gesturegame_app.sh	
	start_qt_idle_app.sh	
	start_qt_memegame_app.sh	
⚡	start_qt_motor.sh	✓
	start_qt_nuitrack_app.sh	
	start_qt_realsense_cam.sh	
⚡	start_qt_robot_interface.sh	✓
	start_qt_routes.sh	✓
	start_qt_voice_app.sh	
⚡	start_qt_webconfing.sh	✓
	start_qtpc.sh	✓
	start_qtrobot_prepare.sh	
⚡	start_robAPL_launcher.sh	✓
	start_ros_usb_cam.sh	
⚡	start_roscore.sh	✓
	start_usc.sh	✓

```
rosnode list | grep "/\(sound_listener\|start_face_server\)"
```



```
qtrobot@QTPC:~/abm-setup$ rosnode list | grep "/\(sound_listener\|start_face_server\)"
/sound_listener
/start_face_server
```

Fig. 3: What you should see if the head nodes are running correctly.

1.4 Body

1.4.1 Getting your Amazon Web Service credentials

For QT to speak, we use Amazon Polly, which requires an Amazon Web Services account. At our current usage, using Amazon Polly is free up to a certain level), but you will need a credit card to create an account.

1. Create an Amazon Web Services account.
2. Once you sign in, in the top right of the page, click your account name (mine says "Audrow"), then in the drop-down menu click "My Security Credentials," then click "Create New Access Key."
3. Record your access key and keep it somewhere safe. You can do this by downloading this or just viewing it and copy-pasting it to somewhere for later reference.

Note: It is best practice to create separate accounts with less access than your root account and use those access keys, see Amazon's security best practices.

1.4.2 Setting up our interaction in a Docker container

0. Change your system timezone to be in your current timezone. To do this, you can click the time in the upper-right of the desktop on QT and then click 'Time & Date settings. . . '
1. Open a terminal and clone this repository onto QT's body computer:

```
git clone -b master https://github.com/robotpt/cordial-docker ~/
↪cordial-docker
```

2. Run a script to allow for updates:

```
sudo bash ~/cordial-docker/scripts/nuc_setup.bash
```

Warning: If this step fails, try the following commands before rerunning:

```
sudo apt install --reinstall python3-six
sudo apt install --reinstall python3-chardet
```

Note: This step takes five minutes or so.

3. Setup Docker:

a. Install Docker:

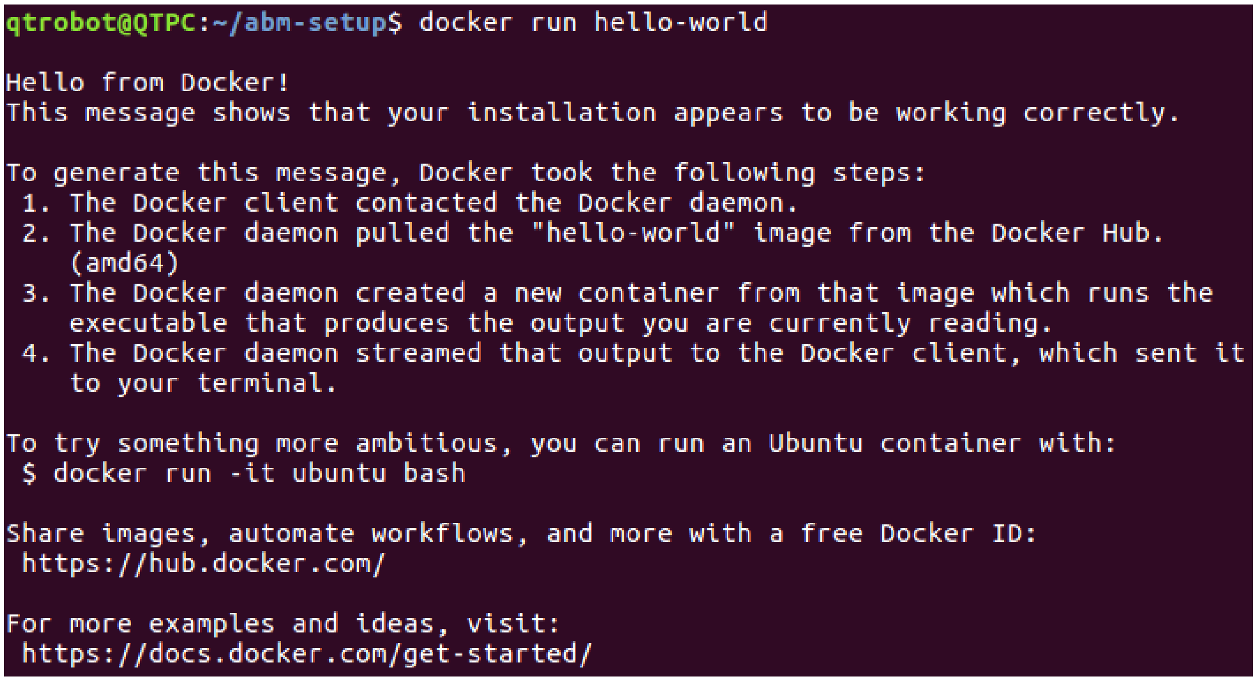
```
curl -fsSL https://get.docker.com -o get-docker.sh
sh get-docker.sh
```

b. Set Docker to run without `sudo`:

```
sudo groupadd docker
sudo gpasswd -a $USER docker
newgrp docker
```

c. Test that Docker is installed correctly and works without `sudo`:

```
docker run hello-world
```



```
qtrobot@QTPC:~/abm-setup$ docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

Fig. 4: What is printed from running the `hello-world` docker container.

4. Setup Docker-compose:

   a. Install Docker-compose:

   ```
   sudo curl -L "https://github.com/docker/compose/releases/download/1.25.3/
   ↪docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
   sudo chmod +x /usr/local/bin/docker-compose
   ```

   b. Check that docker compose is installed correctly:

   ```
   docker-compose version
   ```

5. Setup the docker container:

```
bash ~/cordial-docker/docker/run.sh
```

(continues on next page)

(continued from previous page)

```
.. note::

    The first time that you run the Docker script, it will take around 15 minutes␣
↪(depending on your internet) to setup the container.  After that, it will be␣
↪fast.  Feel free to take a break or go get coffee :-)

.. note::

    I did have an error occur during this command one of the times I was setting␣
↪it up.  It might have been a network issue.  I ran it again and it succeeded. ␣
↪If you have trouble here let me know.
```

6. Configure your AWS credentials by typing `aws configure` in the terminal that pops up.

1.4.3 Setting up remote access to QT

Get Dataplicity login credentials from Audrow and sign on. Go to the devices tab and then click "+ Add New Device". Copy or enter this command into a terminal on QT's body PC and enter QT's password 'qtrobot'. After that runs, remote access should be setup. You can confirm this by clicking the added device and confirming that you can explore the file system (e.g., `ls /home/qtrobot` and you should see familiar directories such as `cordial-docker`).

CHAPTER 2

Troubleshooting

2.1 Unable to contact ROS master at 192.168.100.1:11311

This means that the two computers in QT are not talking correctly because they did not boot in the right order. To fix this, restart QT by pressing down QT's power button for a second or two. QT should slump forward and the face screen should go off. If the face screen doesn't go off, power QT on and repeat the process. I haven't ever had to do this more than twice, as by the second time they are synced.

To test if the computers are talking correctly, you should be enter the following command and see a list of outputs. If you get an error, restart QT.

```
rostopic list
```

2.2 The tablet doesn't connect

This can be two things in my experience:

1. The tablet is on the wrong wifi network
2. The interaction isn't running (most commonly Fitbit credential errors)

2.2.1 Tablet is connecting to the wrong wifi

It is easiest to check that the tablet is on the correct wifi. When QT turns off, it turns off the wireless network that it is hosting. When this happens, most tablets will auto-join other networks that they have been connected to. What makes this worse is that web browsers often rememeber (cache) websites to make them quicker to load, which makes it look like you're connecting to the website hosted by QT, but you're not. To fix this, go into the tablet's settings and turn off autojoin for all wireless networks except for QT (or just the networks you expect the tablet to see while the interaction is running). Once this is set, you should be able to connect to and prompt the interaction on QT, including after QT restarts (if you've changed the container's restart policy).

2.3 QT's head computer doesn't have internet

If you can SSH into QT's head computer, but are unable to clone a repository or make an update, check if you have internet. The following command should show you that messages are being sent and responses are received, it just hangs there, you are not connected to the internet:

```
ping google.com
```

At the moment, just restart QT and hopefully the problem will be fixed. LuxAI has recognized this problem and given me steps to fix it but I haven't tested them.

If you would like to try, here is their email

> In our older setup of RPI/NUC network (same as your QT) we had enabled port forwarding (8080 -> 80) on both machine to facilitate accessing the QTrobot wev config. In some cases and time-to-time (also depending on the router) this causes problem for RPI to reach the internet properly.
>
> In this FAQ we explained it how to disable it: https://docs.luxai.com/FAQ/Network/
>
> Some more comments on this issue : https://github.com/luxai-qtrobot/QA/issues/3
>
> However all you need to do (as I imagine this can be the cause of the problem) is to disable the port forwarding on both machines QTPC and QTRP.
>
> For QTRP (RPI), just follow the simple instruction in the FAQ link and comment the corresponding line in :code‘start_qt_routes.sh‘. Double check that your `/etc/network/interfaces` on RPI has the following config:

```
auto lo eth0
iface lo inet loopback
auto eth0
iface eth0 inet static
address 192.168.100.1
netmask 255.255.255.0
gateway 192.168.100.2
dns-nameservers 192.168.100.2 8.8.8.8
```

> Regarding the QTPC, you can completely disable/remove the 'start_qt_routes.sh'.