



JFrog Connect Demo

Nick Ristuccia
Developer Advocate at JFrog

<https://jfrog.com/connect/>



Trickle-Up Scenario: Agriculture

- Nurture and engage individual developers
- Developers influence company decisions
- What industries and use-cases?
- Offer fun demos, be prepared for industry perspective

Request from Sonnetopolis



Sonnetopolis sends a desperate greeting. Our humble culture finds its fate fleeting. Man does not live on bread alone. But we cannot feed our people with a poem. We need **potash** to make our land fertile. Please find a trading nation to help with this hurdle.

About Sonnetopolis:

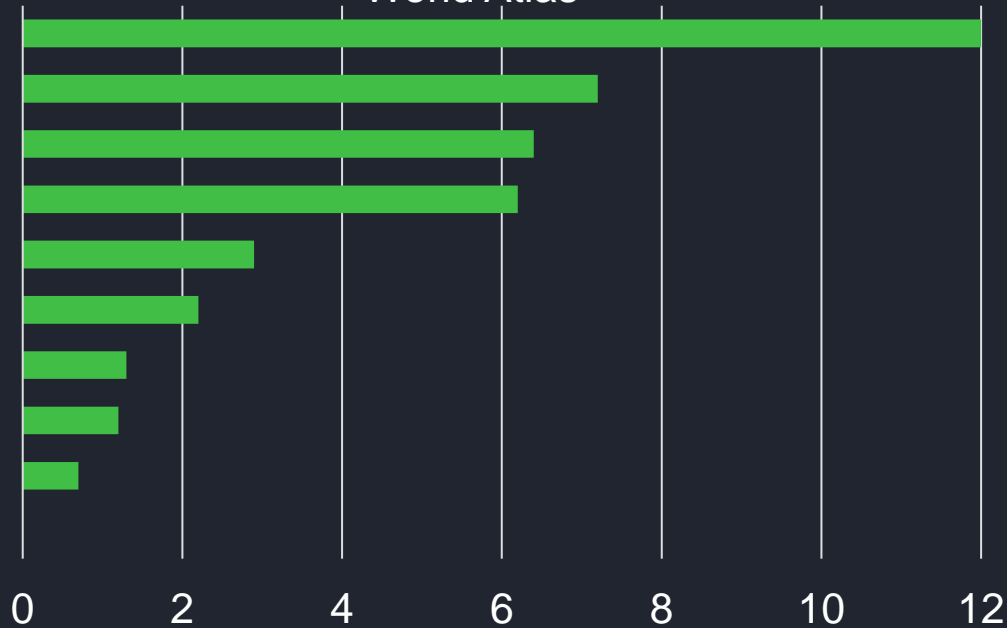
- Population: 4,000,000
- Exports: Poetry
- Natural Resources: Very limited
- Agriculture: Grows its own food but relies heavily on imports to make fertilizer

About Potash:

- Potassium salt compounds (KCl , K_2SO_4 , KNO_3)

Who is a Good Trading Partner?

Million Tons of Potash Produced in 2017,
World Atlas



Top Potash Producing Countries:

Belarus

Canada

Chile

China

Germany

Israel

Jordan

Russia

Spain

Canada

Everyone loves Canadian potash! Other countries are outbidding us and driving up the price. We can only get 25% of the potash we need.

Russia, Belarus

Pariah states created a war zone. Other trading partners will punish us.

China

Phosphate exports were banned. Potash is next.

Germany

They won't trade for poems... only liquified natural gas or other forms of energy.

Israel

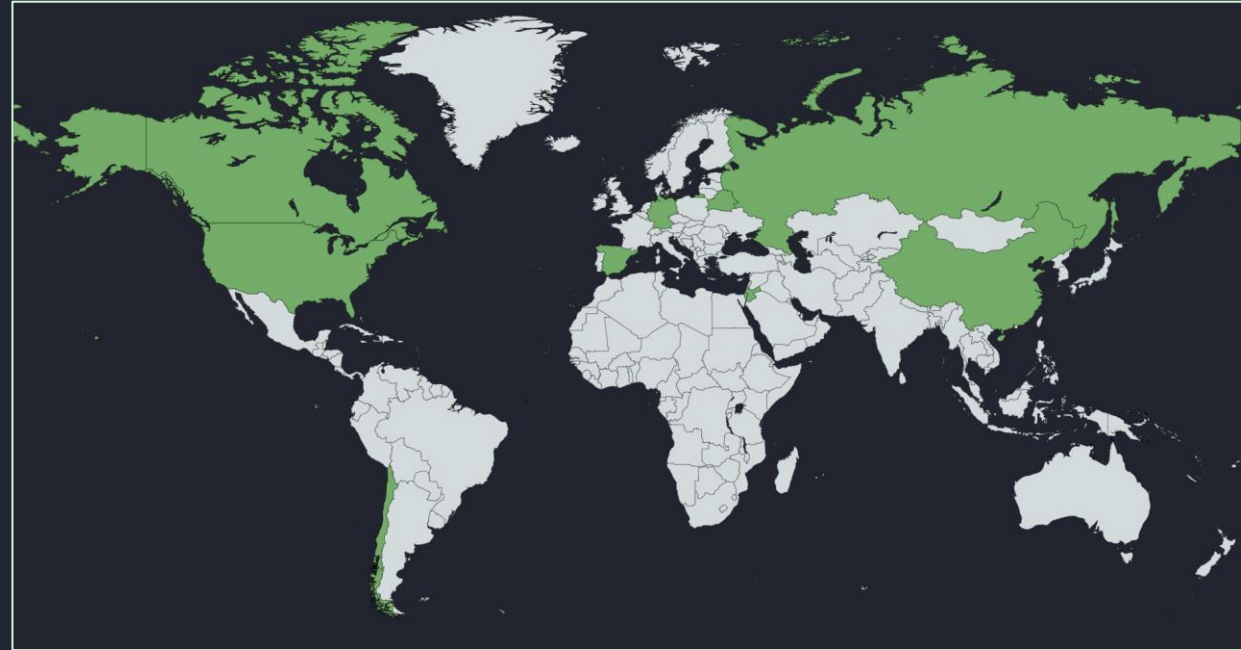
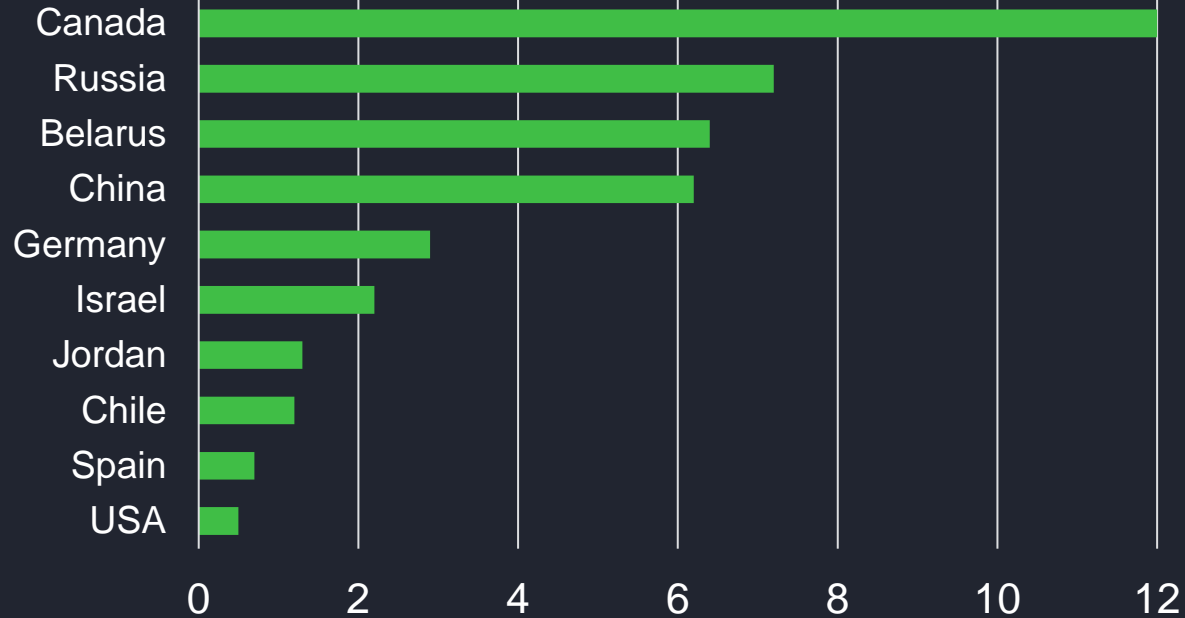
They can offer 10% of what we need.

Jordan, Chile, Spain

None to spare. What's produced is consumed.

The Agriculture Takeaway

Million Tons of Potash Produced in 2017,
World Atlas



Fertilizer and its ingredients come from a small number of countries. They're susceptible to supply chain disruptions and geopolitics (<https://www.cnbc.com/2022/03/22/fertilizer-prices-are-at-record-highs-heres-what-that-means.html>)

Help From **AI** and **IoT Farming**

- **Double crop yields**
- Consume only **a quarter of the inputs**
- Sensors monitor crop rows and detect the optimal time to introduce inputs
- According to geopolitical strategist Peter Zeihan
(<https://www.youtube.com/watch?v=LzipwDQBUyc>, @51:20)



John, a Sonnetopolis arable farmer, uses smart devices in his fields which monitor soil moisture, analyze this data, and use it to regulate irrigation. However, the application that has been shipped with those devices has a fault that sporadically stops the irrigation on hot days. The consequences of this software bug are potentially devastating: his entire crop could be lost.

Updating Devices

Manual updates:

- Take an SD card or flash drive to each device
- Are devices in inconvenient locations or inaccessible?
- Did you miss a device?
- This process does not scale well



OTA updates:

- Reach many devices at once
- Fail-safe (Rollback if necessary)
- Division between different types and models
- Monitor devices' health

Device requirements:

- Linux
- Network connection
- Install the Connect Agent





Readying a Raspberry Pi

Required Hardware

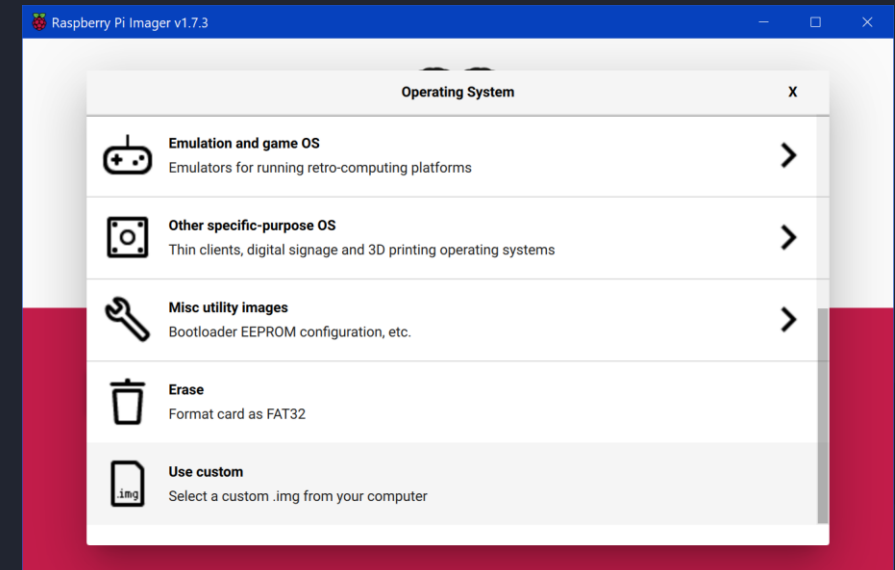
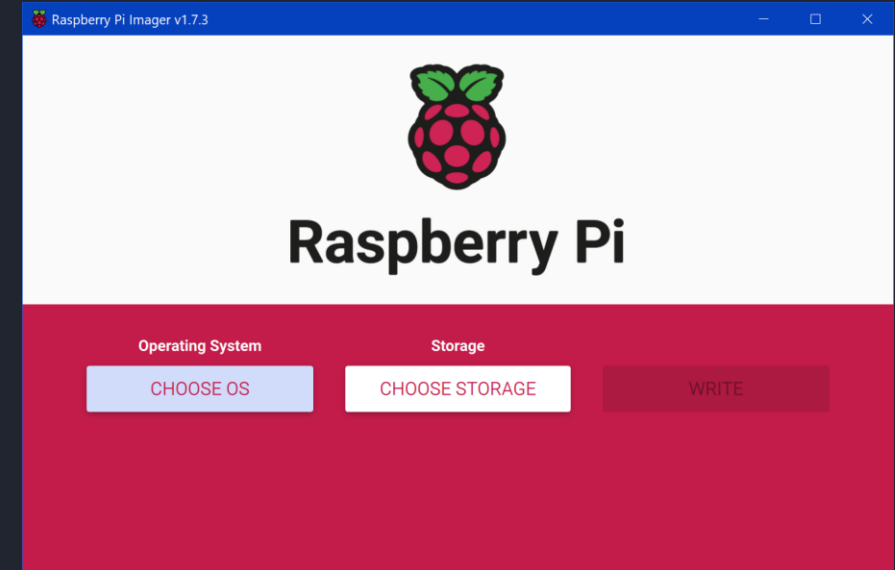
1. Raspberry Pi
2. Power adapter
3. SD Card
4. HDMI Cable
5. Monitor
6. USB keyboard
7. Wifi adapter or ethernet connection



Our setup includes a Raspberry Pi 1 Model B, 4GB SD card, and NetGear wifi adapter A6150. Older RPi models like this don't include built-in wifi support.

Flash a Linux Image

1. Download and run the Raspberry Pi Imager:
<https://www.raspberrypi.com/software/>
2. Choose a Raspberry Pi OS. These are all Debian Linux variants. Your choice will depend on your SD card size and Raspberry Pi model:
 - For smaller SD cards, choose options with no desktop environment
 - For older RPi models with no built-in wifi, choose an older OS. These are more likely to have available wifi adapter drivers.
 - We chose "Use custom" and downloaded the 2020-05-28 Debian Buster image:
https://downloads.raspberrypi.org/raspios_lite_armhf/images/
3. Choose the SD card and write the image



Power On

4. Insert the SD card in the RPi
5. Power on the RPi
6. Login

For older OS images, the default login may be:

- Username: pi
- Password: raspberry



Download the Connect Agent

9. Log in to your JFrog Connect account or choose Start for Free: <https://jfrog.com/connect/>
 10. Once you're logged in, select **Start with your device now** and name your project
 11. Select **Register Device**
 12. Enter either of the two statements shown into the RPi terminal.
- The RPi should be connected in a matter of seconds.

Device Registration

To register a new device to the project **Demo01**, run the following command:

```
Root User Regular User
sudo wget -O - "https://connect.jfrog.io/v2/install_connect" | sudo sh -s ccmZcKb5BFCsfWcveGL2a2ReZGBrh9diqA Demo01
```

Copy to clipboard

Optional Parameters

You can add optional parameters at the end of the register command.



<code>-n=<device_name></code>	Set a device name at registration (for example, you can use "\$HOSTNAME" as your device name)
<code>-g=<group_name></code>	Assign the device to a specific group at registration (group must exist).
<code>-s=<software_version></code>	When defined, it will create an application called default_app with the specified version at registration.

Example Command

```
sudo wget -O - "https://connect.jfrog.io/v2/install_connect" | sudo sh -s ccmZcKb5BFCsfWcveGL2a2ReZGBrh9diqA Demo01 -n=my_device_name -g=Test -s=1.0
```


Test the Connection (Optional)

13. Click the button for remote control access to the device. Within a few seconds, a reverse SSH tunnel is created. Click **Connect**.

ID	DEVICE NAME	DEVICE STATE	REMOTE CONTROL	LAST UPDATE STATUS	APPLICATIONS	MONITORED PROCESSES	TAGS
d-a7bf-5b74	Raspberry Pi 001			Success	No apps configured	+	+

14. Enter **sudo reboot now** in the remote terminal. The actual RPi will reboot, and the remote terminal should close automatically. Rebooting the RPi isn't necessary. It's just an obvious action you can use to prove the connection works.

15. Scroll down in Connect. You should find modifiable general details about the device, as well as see technical details.

General details

Change

ID

d-a7bf-5b74

Name

Raspberry Pi 001

Group

Production

Last seen

~14 seconds ago

Location

United States

Description

Update Trigger

Off

JFrog Connect agent version

6.0

Technical details

Monitored Processes

+

Mac addresses

b8:27:eb:a6:05:89
08:36:c9:7f:26:44

IP address

100.0.48.126

CPU usage

1%

RAM usage

9%

Disk usage

1.325 GB used (36%) / 3.583 GB

About the Agent

- Stealthy memory footprint:
 - ~4MB of disk space
 - ~11MB of RAM
- Always on
 - Periodically checks with Connect servers
 - Every 15 seconds by default (in the trial version of Connect)
 - Period is changeable in **Settings**
 - Communicates client-side via port 443 (HTTPS) or 80 (HTTP)
 - Independent



Production Benefits

- Designed to support complex network environments
<https://docs.connect.jfrog.io/overview/architecture/agent>
 - No Public IP
 - Behind double-nat
 - Under a firewall, cellular modem
 - Unstable wifi connection
- Register devices at scale
<https://docs.connect.jfrog.io/register/register-linux-devices-at-scale>
 - Option A: Insert Connect service into a build of an image
 - Option B: Freeze image



OTA Updates

Create an Update Flow

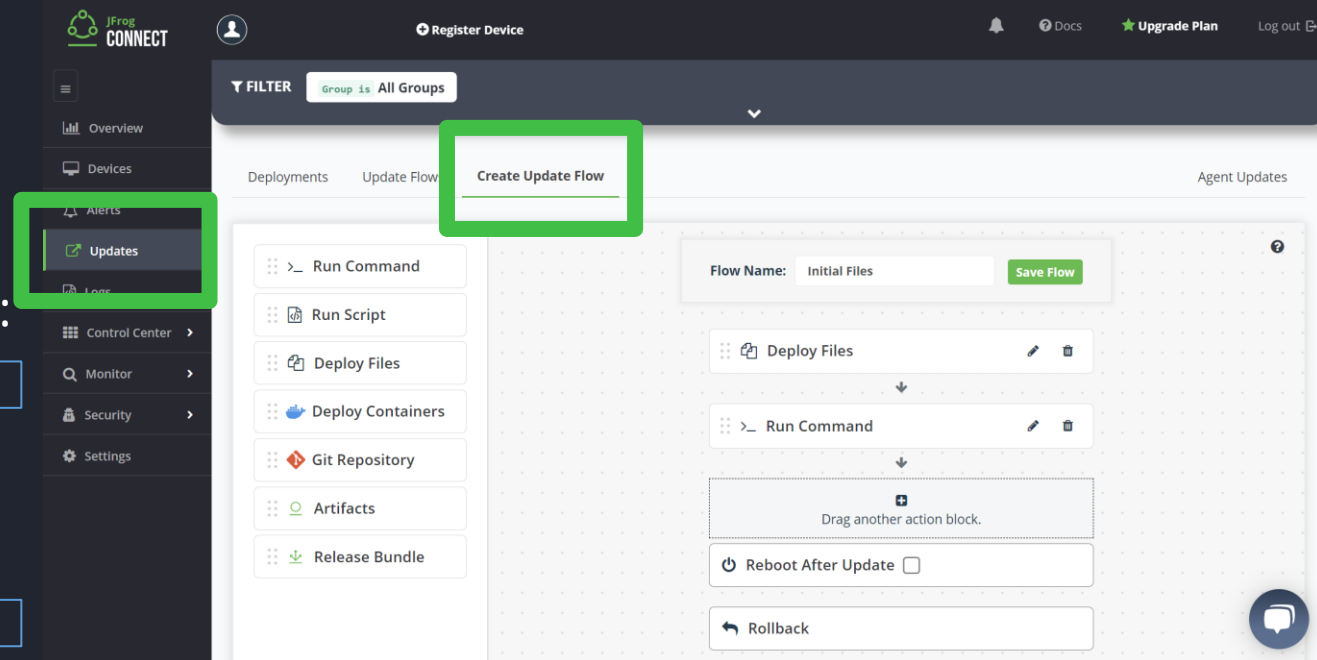
1. Navigate to **Updates** > **Create Update Flow**
2. Name the flow **Initial Files**
3. Drag the **Deploy Files** block into the flow and click the **pencil icon** to edit the block
 - Destination: /home/pi
 - Sample project files are available here:

```
git clone https://github.com/NickR2600/ConnectDemo01.git
```

Drag in the Run Command block and edit by adding this command:

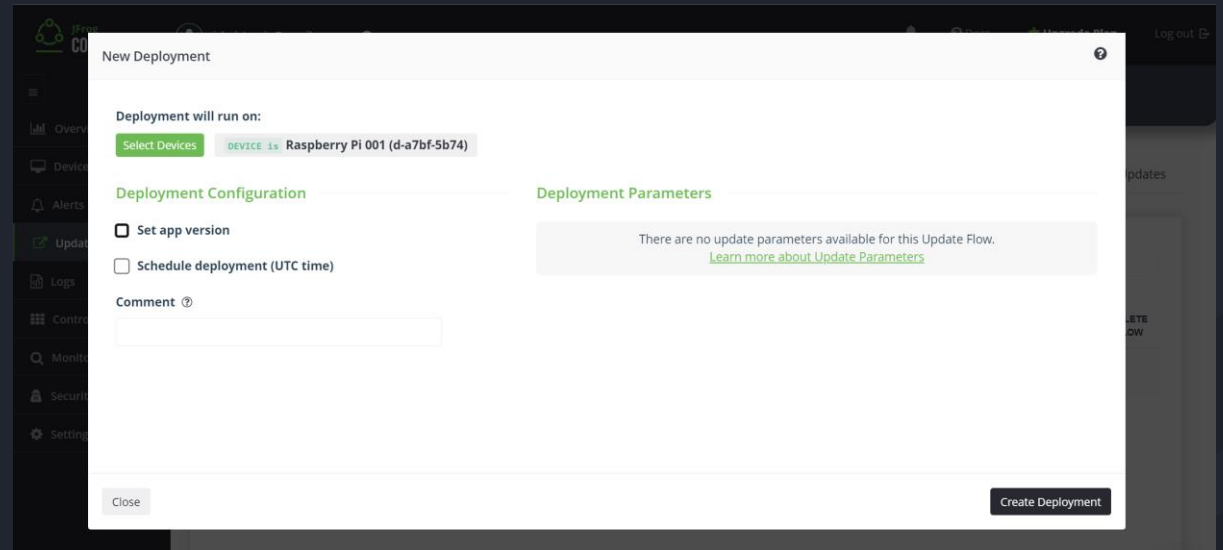
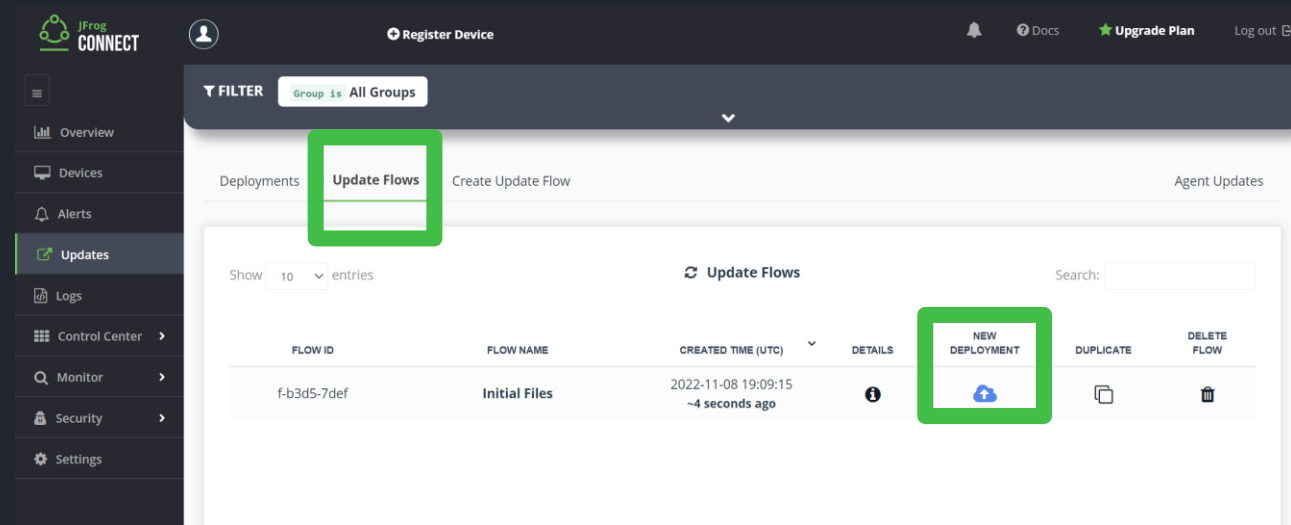
```
runuser -l pi -c 'make && ls' > /dev/tty1
```

4. Click **Save Flow**
5. When asked about the General Rollback, select **Continue Anyway**



Deploy the Update Flow

7. Navigate to **Updates > Update Flows**
8. Click the **New Deployment** cloud icon for the Initial Files update
9. Click the **Select Devices** button
 - Filter for, and select your specific device
 - Select **Apply, Next, Finish**
10. In this case, unselect **Set app version**
11. Click **Create Deployment** and run the deployment



Observe the Update

12. Navigate to **Updates > Deployments**

13. For the update you just created, click the **progress** icon

14. Observe the progress in the dialog box and the monitor attached to your RPi

15. After the deployment succeeds, enter **./demo** on the keyboard connected to the RPi to run the program.

- If you don't have a RPi, do this in a remote terminal for the device created by Connect

The screenshot shows the JFrog Connect interface. On the left is a sidebar with navigation options: Overview, Devices, Alerts, Updates (selected), Logs, Control Center, Monitor, Security, and Settings. The main area is titled 'Deployments' and contains a table of deployment records. The table has columns: APPLICATION VERSION, CREATED TIME (UTC), DEPLOYMENT TYPE, FLOW NAME, COMMENT, XRAY STATUS, and PROGRESS SUMMARY. The first row shows 'No apps updated' for 'Initial Files' with a status of 'NOT APPLICABLE'. A green box highlights the 'PROGRESS' icon in the 'PROGRESS SUMMARY' column. A green success message overlay reads 'Success Deployment created.'

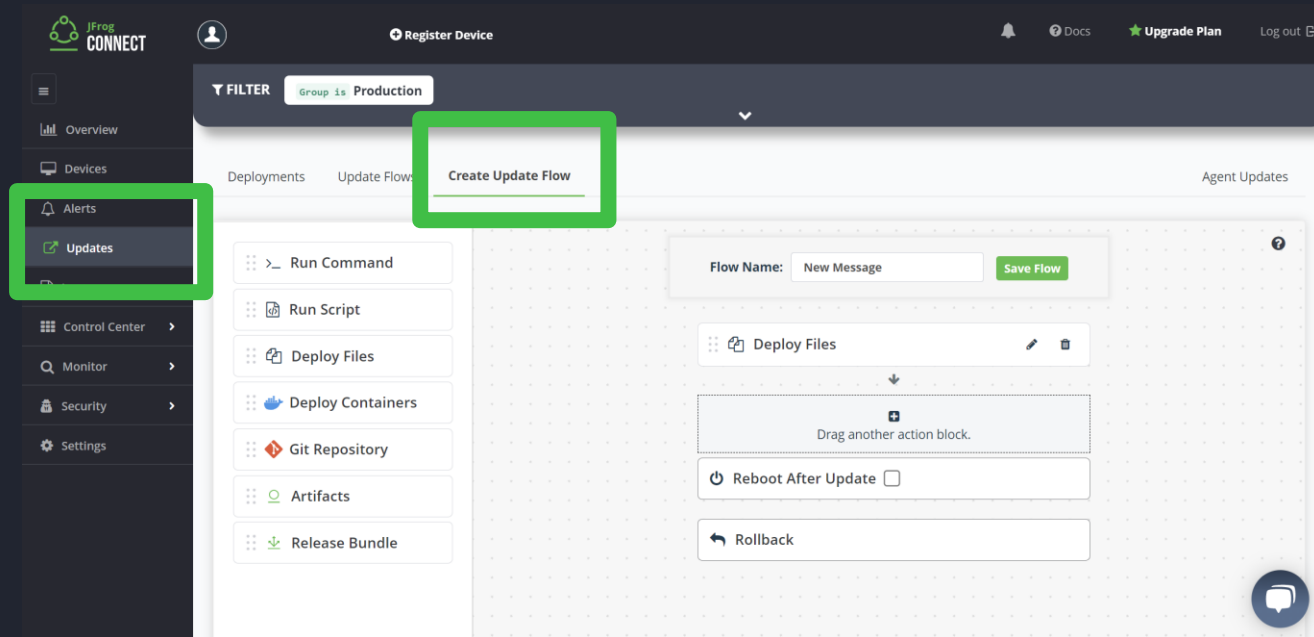
The screenshot shows the 'Deployment Configuration' dialog box. It has a left sidebar with 'Deployment Configuration' and 'Deployment Progress'. The 'Deployment Configuration' section shows: Application: No apps updated, Comment: N/A, Deployment Type: manual, Update Flow: Initial Files, and an 'Abort Deployment' button. The 'Deployment Progress' section shows a progress bar for 'Raspberry Pi 001 (d-a7bf-5b74)' which is 'In Progress'. Below the progress bar is a table with columns: DEVICE, UPDATE STATUS, UPDATE START TIME, and ABORT UPDATE. The table shows one entry for 'Raspberry Pi 001 (d-a7bf-5b74)' with status 'In Progress' and start time 'Deployment started at 2022-11-08 19:44:12 ~11 seconds ago'. There is an 'Abort' button next to the entry. The dialog also shows 'Showing 1 to 1 of 1 entries' and a 'Close' button at the bottom.

Notes about the Update

- Step 15 is necessary this time because this is a TTY program. Other programs, like those that drive a kiosk, can be started from step 4
- Other blocks exist to accommodate other actions, sources, and types:
 - Run a script
 - Deploy artifacts from Github
 - Deploy artifacts from Artifactory
 - Deploy a Docker Container
- Rollbacks prevent bad updates from bricking devices
 - Rollback to the previous stable state
 - Ability to specify actions when a particular step fail

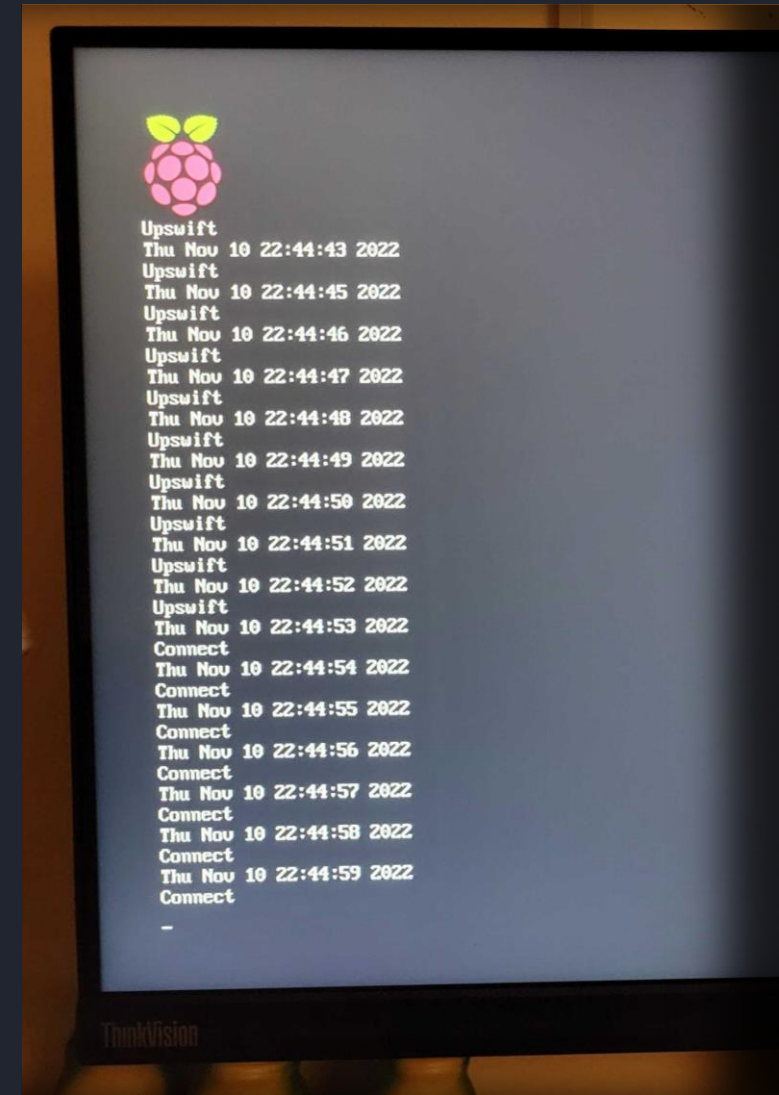
Update the Message

1. Ensure the program is still running on the RPi
2. Modify message.txt to read **Connect** instead of **Upswift**
3. Navigate to **Updates > Create Update Flow**
4. Name the flow **New Message**
5. Drag the **Deploy Files** block into the flow and click the **pencil icon** to edit the block
 - Destination: /home/pi
 - Specify the updated message.txt
6. Click **Save Flow**
7. When asked about the General Rollback, select **Continue Anyway**



Deploy the Update Flow

8. Navigate to **Updates > Update Flows**
9. Click the **New Deployment** cloud icon for the New Message update
10. Click the **Select Devices** button
 - Filter for, and select your specific device
 - Select **Apply, Next, Finish**
11. In this case, unselect **Set app version**
12. Click **Create Deployment** and run the deployment
13. Observe the new message in the running program!
 - If you don't have a RPi, open a remote terminal and observe the new output when you run `./demo` again





At Scale

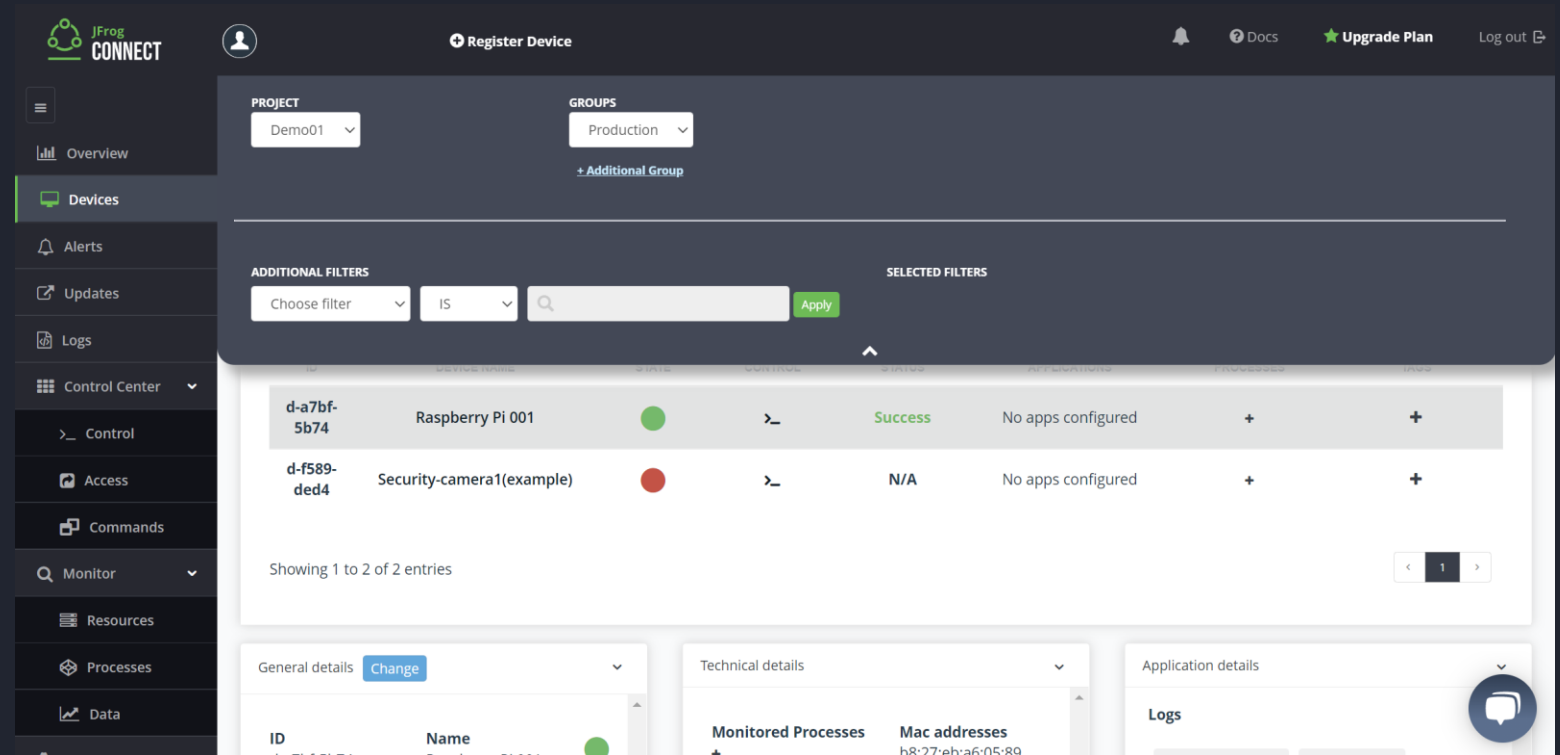
Tips for Managing Devices at Scale

Filtering through devices:

- Use the **Filter Bar**
- Create groups
- Tag devices
- Create app/version numbers
 - When Deploying a Flow
 - In the Devices Tab

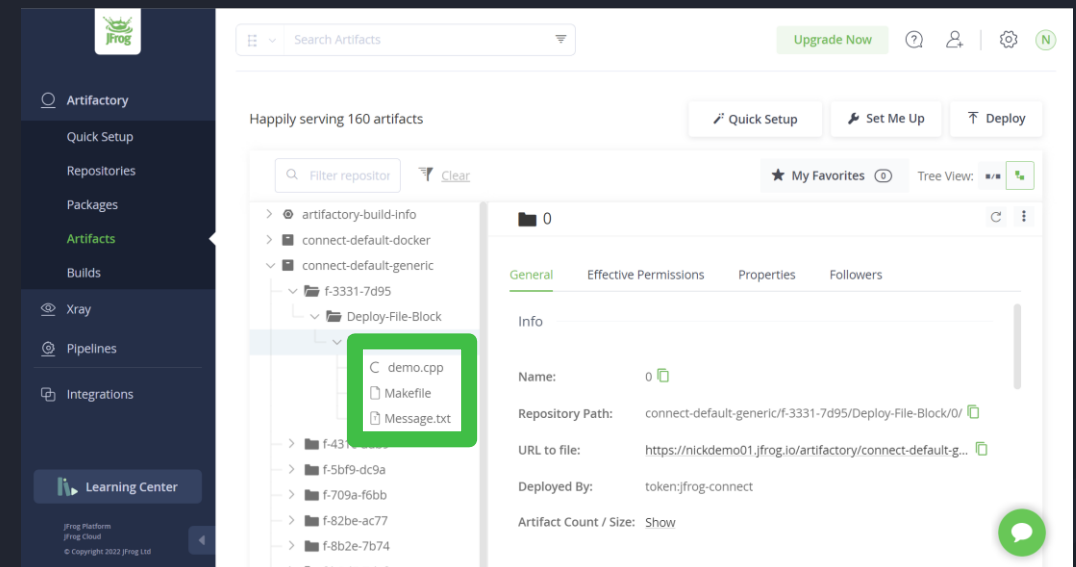
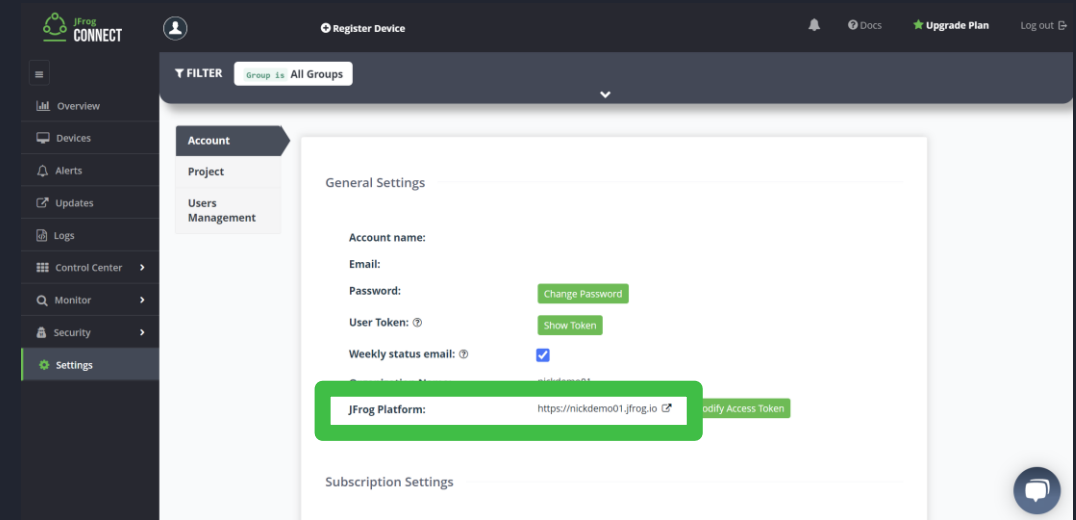
Recommendation:

- Get it right on one device before rolling out to entire fleet



Things You May Not Have Noticed

- C++ is platform-dependent
 - We generated a device-specific binary (the executable)
 - That binary and message.txt are all we need to push to devices
- A new Connect instance comes with a new instance of Artifactory
 - Files you upload via Connect are stored automatically in your Artifactory instance
 - Find your Artifactory/JFrog Platform instance under the Connect **Settings** tab
 - You can specify another Artifactory instance in the Artifacts block of Create Update Flow





Thank you!

Nick Ristuccia
Developer Advocate at JFrog

<https://jfrog.com/connect/>