

Getting Started with Octoblu



Octoblu enables companies to create IoT services with secure real-time exchange of data. The services are built on an open communications and management platform that supports a variety of protocols for physical devices to communicate seamlessly with each other, people, legacy applications, and cloud services. Through public, private, or hybrid clouds users can connect, design, process, and analyze the flow of information. All services have been designed through a robust security and right management architecture.

Using Octoblu APIs you can easily integrate existing or new applications on to the Octoblu IoT platform. In this way you can add exponential value to your product via the power of being connected to a myriad of platforms, devices, and web-services through one messaging standard.

We've provided resources to get started with our API, applications, visual designer, and open plugin architecture. Our goal is to help you harness the power of the Octoblu ecosystem to meet your IoT goals.

<http://www.citrix.com/go/citrix-developer/octoblu-developer-community.html>

Open the Design Page

The screenshot shows the Octoblu Design Page. On the left is a sidebar with the Octoblu logo and a menu with icons and labels: Home, Connect, Design, Process, Analyze, Templates, Admin, Profile, and Sign out. The main area has a header with 'Demo Flow' and a dropdown arrow. Below the header is a white box with a '+' icon and a 'Demo Flow' button. The central workspace is a blue grid containing a flow diagram. The flow starts with a 'Trigger' icon (a black cylinder with a blue top), which connects to two parallel paths. The top path goes through a 'StockPrice' icon (a red bar chart with a green arrow) and then to a 'debug' icon (a magnifying glass with a gear). The bottom path goes through a 'Weather' icon (a sun behind a cloud with blue rain) and then to another 'debug' icon. Each debug icon has a text box that says 'click this debug to see the output of StockPrice' and 'click this debug to see the output of Weather' respectively.

Deploy and Test Flow

The screenshot shows the 'Deploy and Test Flow' interface. At the top is a blue toolbar with icons for delete, share, undo, redo, and a green play button. Below the toolbar is a blue grid with a flow diagram. The flow starts with a 'trigger' icon (a black cylinder with a blue top), which connects to a 'debug' icon (a magnifying glass with a gear). Below the debug icon is a text box that says 'click the trigger to test out your flow'.

Open Debug Console

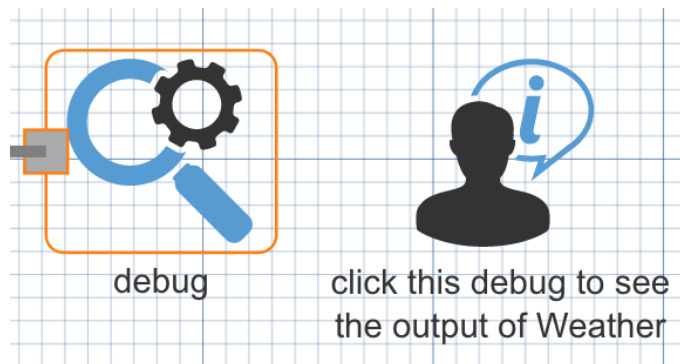
The screenshot shows the Debug Console interface. It consists of a horizontal row of five buttons with icons: a plug, a USB symbol, a plus sign, a bug, and a downward arrow.

1/28/15 12:18 PM - debug

```
{
  "date": "2015-01-28T19:18:17.476Z",
  "message": {
    "node": "Zeb29667-a722-11e4-81b1-19e62bd918bd",
    "name": "debug",
    "msgType": "input",
    "msg": {
      "temperature": 68.59
    }
  }
}
```

You'll see output like this = `"msg" : { "temperature" : 68.59 }`

This is the **JSON message** that was sent from the **Weather** node to the **debug** node.



You can tie the output of any node to a **Debug**.
You can also flip a **debug flag**.

—  Debug

Select a channel node to edit it. You can search for and select any available API endpoint and the parameters will appear.

Edit > StockPrice ? x

Name

StockPrice

Filter Endpoints

Endpoint

Get Last Trade Price

GET http://stock.octoblu.com/last-trade-price/:symbol

Symbol

GOOG

Debug

Edit > Weather ? x

Name

Weather

Filter Endpoints

Endpoint

Get Temperature In Fahrenheit

GET http://weather.octoblu.com/temperature/fahrenheit

City

Chandler

State

Arizona

Country

USA

Debug

We're going to edit the **Stock price node**.

Symbol
{{msg.payload}}

Using **{{double curly braces}}** you can **reference an incoming message**. Earlier we saw that the message the Weather node outputs is a **JSON** object of [msg.temperature](#).

The **trigger node** can be told to send out a specified message as [msg.payload](#)

Edit > trigger ? x

Name

trigger

Topic

Payload Type

Timestamp

Blank

☒ String

Payload

Debug

Payload Type

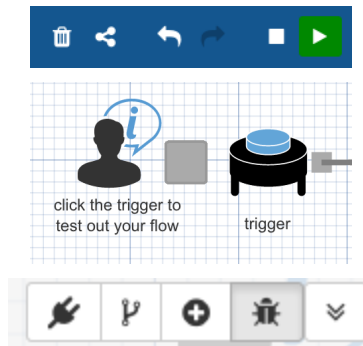
- ☐ Timestamp
- ☐ Blank
- ☒ String

Payload

GOOG

We're going to send out a Stock Symbol

Re-Deploy your flow, hit the trigger, and open the debug console. You'll see that the "GOOG" stock price is shown. The **Stock node** referenced the message that was passed from the trigger and used that in an **API call** to the Stock Price channel.

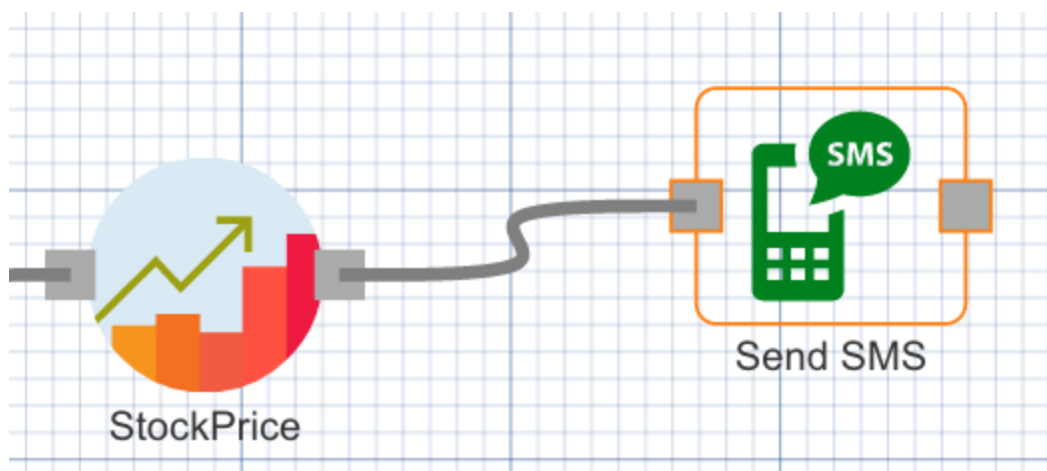


Lets edit our flow to send the Stock price as an SMS!

Open the Configured nodes tab.



Select the SMS node and drop it into your flow.



Wire your Stock Price node to the SMS node.

Edit your SMS node as such.

- Add your number and include a Country code, use “1” for the US.
- Reference [msg.price](#) , the expected output of the Stock Price node

Edit > Send SMS ? x

Name

Send SMS

Filter Endpoints

Endpoint

/message

POST http://sms.octoblu.com/message

Destination Number(s)

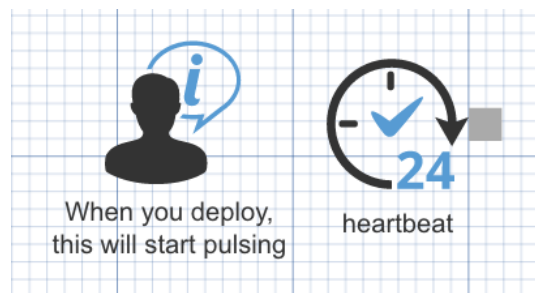
1800YOUFLOW

Message (Add data to text using {{payload.values}})

{{msg.price}}

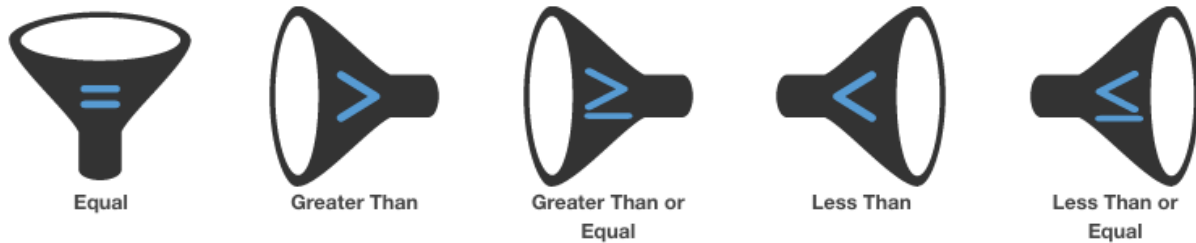
☐ Debug

Re-Deploy, and press the trigger! You should receive a text message in a few seconds with the current **GOOG Stock Price!**



You can use the **interval node** in place of a **trigger**. It will behave the same way except at an **interval in milliseconds!**

There is more functionality that you can add from the **Logic Operators** tab. For example, you can use these comparator nodes to control the flow of messages based on different conditions.



Edit > Greater Than

?

×

Name

Greater Than

Left

Right

{{msg.payload}}

>

10

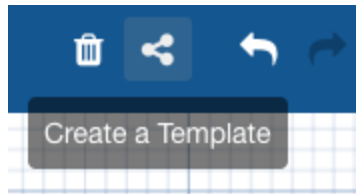
☐

Debug

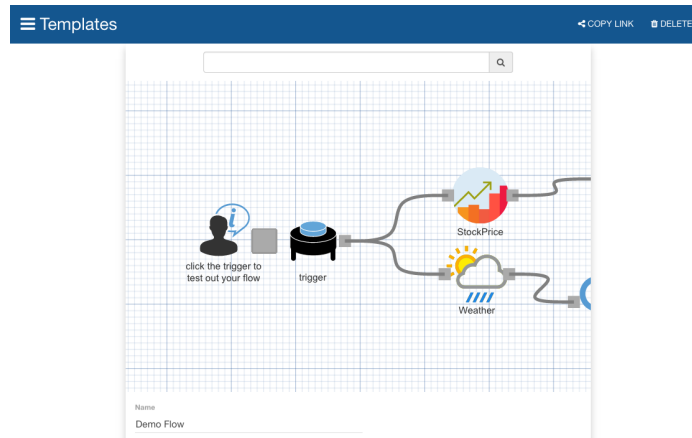
For example, using the **Greater Than** node to check the referenced [msg.payload](#) to see if it is greater than 10. If the condition is met, `msg.payload` will be passed to the output of the Greater Than node. Anything wired on that other side will receive the original [msg.payload](#).

Sharing and Importing Flows

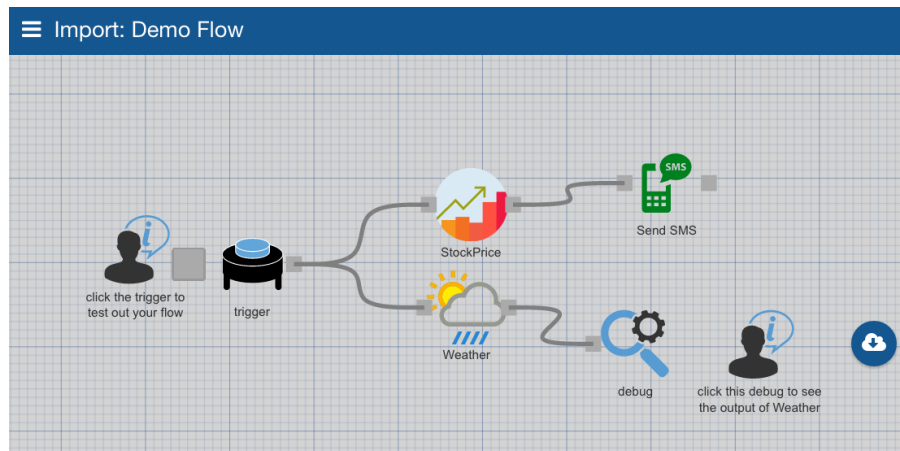
Click the **Create a Template** icon.



In the follow page, name your flow and then click **copy link**




That link can be shared and will open to an Import flow page. Press the **cloud icon** to add it to your account.



Congratulations!

You should now have a basic mastery of Octoblu usage! The best way to get better at creating amazing IoT applications is to play and experiment. Use triggers and the debug node to test and see the output from different nodes to help yourself understand more. We'll be posting more tutorials and documentation. Also be sure to checkout www.hackster.io/octoblu for tutorials on adding hardware and devices to your project using Gateblu and Microblu!



Octoblu - Web app for configuring and securing connections, designing flows, and querying analytics

Meshblu - Multi-protocol & API powered open source IoT platform that can be deployed as a cloud or mesh network.

Gateblu & Mobiblue - Fixed and mobile gateways for connecting home/office/field devices and BLE wearable devices to **Meshblu** and **Octoblu**

Microblu - Open source microcontroller OS for connecting Arduino, Spark, Pinoccio, Raspberry Pi, Galileo, Edison, etc to **Meshblu** and **Octoblu**