Bringing Innovation to Life

Markus van Kempen

Executive Architect &

Venture Capitalist

IBM SPEED

E: mvk@ca.ibm.com

T: @markusvankempen

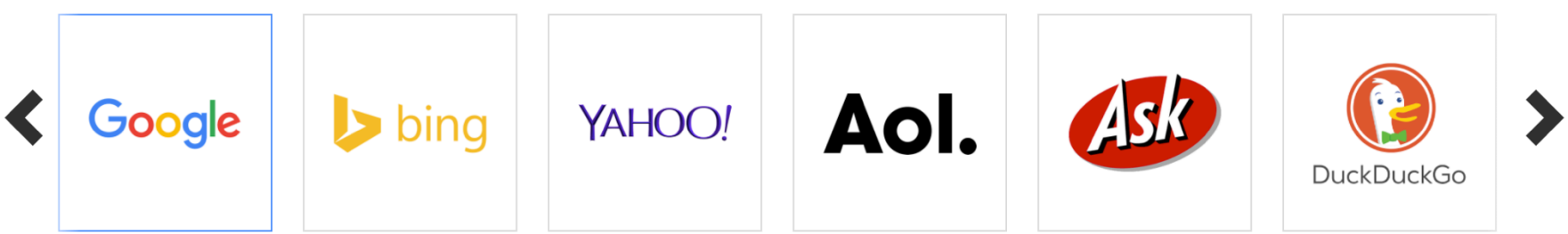
IBM

Agenda

- Review last exercises
- JavaScript Basics
- Code Backup/Restore
- Exercises
  - Weather /Candle/eMail
- Homework
  - Iss/Weather /Candle/eMail

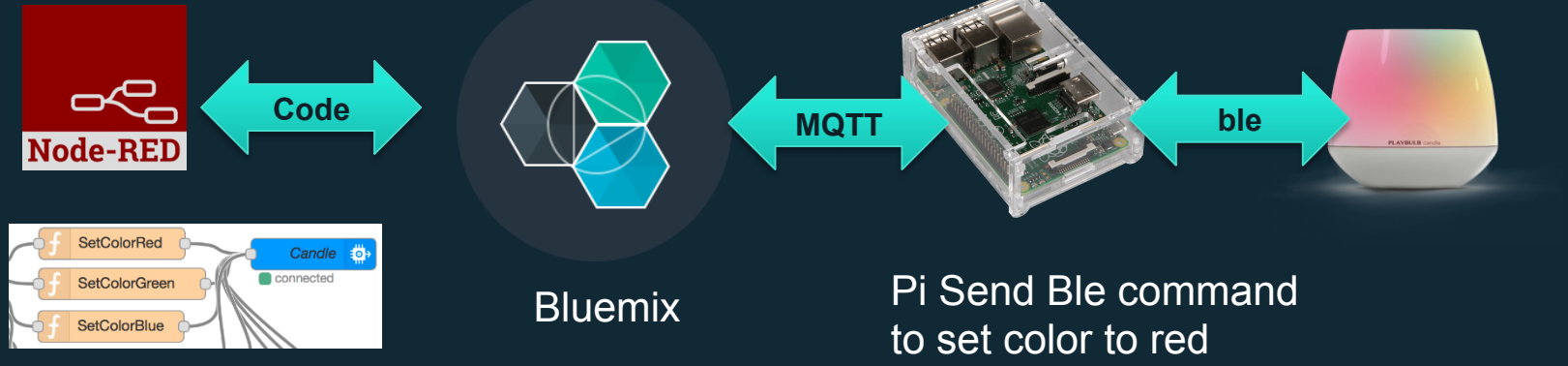

Learning by
Doing

Search Stuff

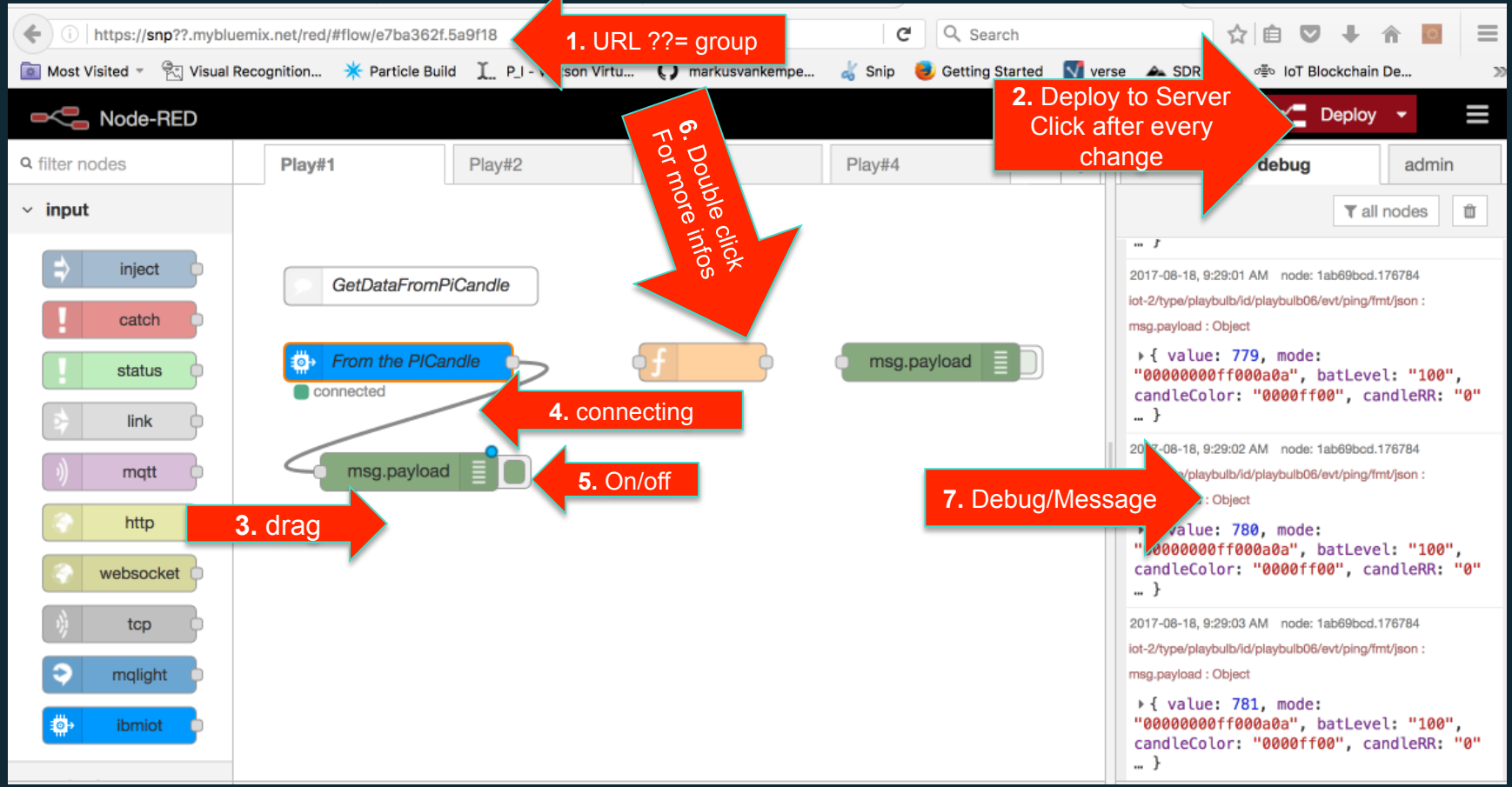


Recap from Last session

Objective – Controlling a Light wireless via a PI



Node-red tool – Drag/Drop - Javascript coding tool

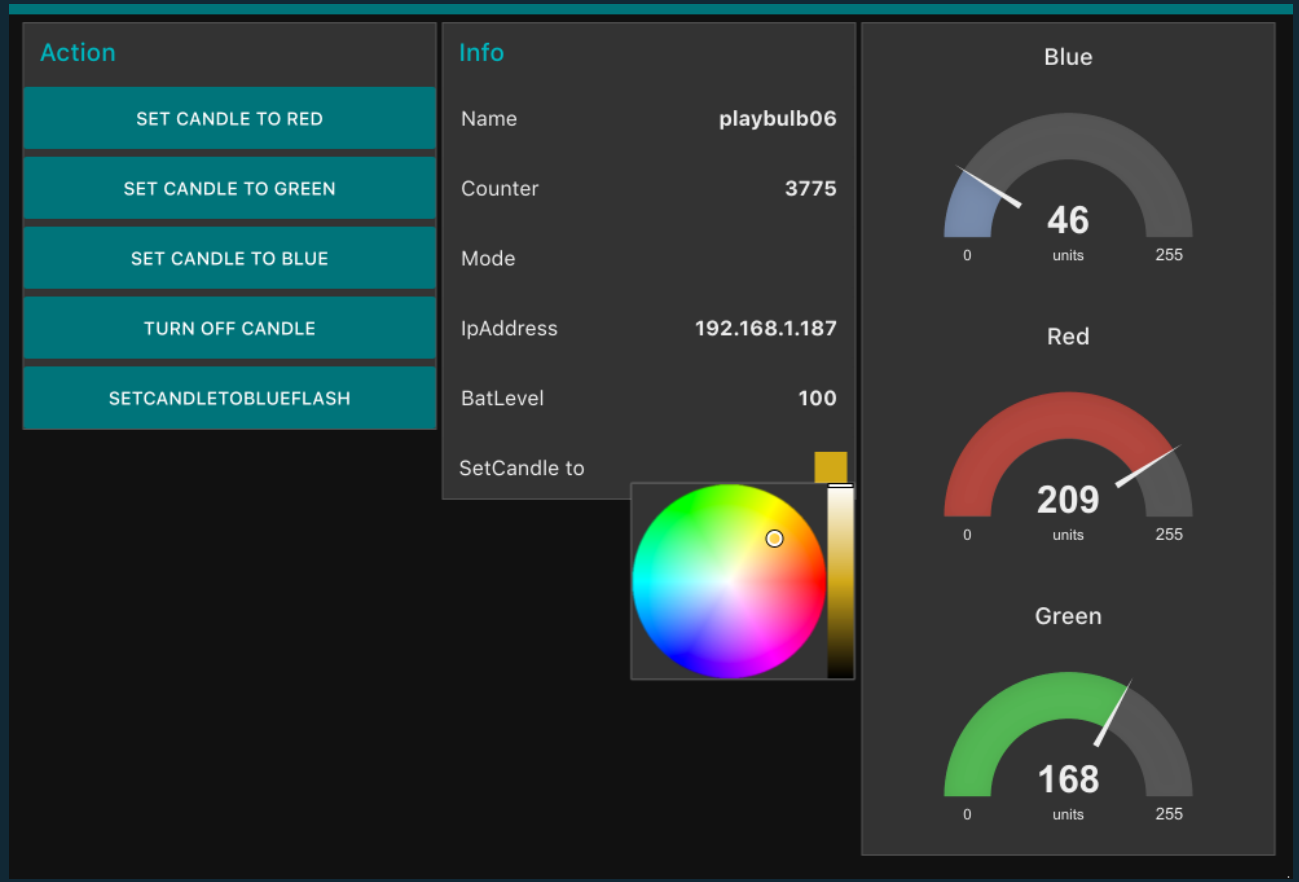


Exercise #4

- Create a dashboard to web/mobile-enable the candle function
- Open a new browser tab with the URL

- https://snp??.mybluemix.net/ui (Where ?? is your group number)
- E.g. http://???.mybluemix.net

Result (code is in Github)



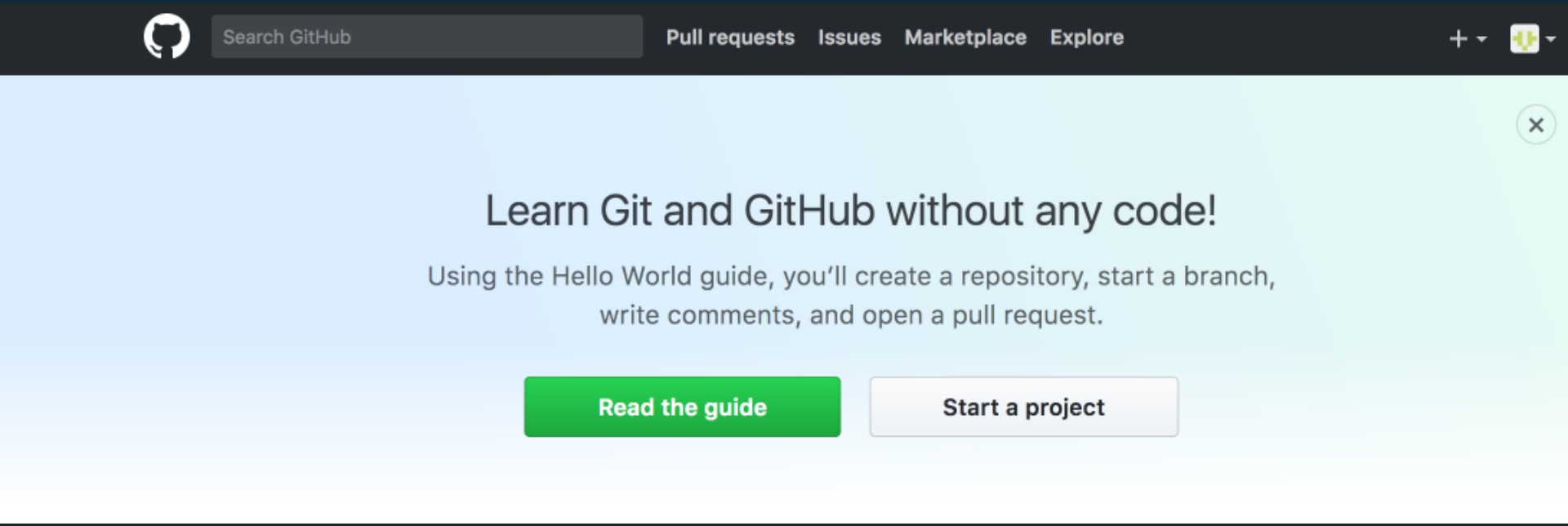
Tips

Undo and Redo - Ctrl+z , Ctrl+y
Find and Replace- Ctrl+f, Ctrl+h

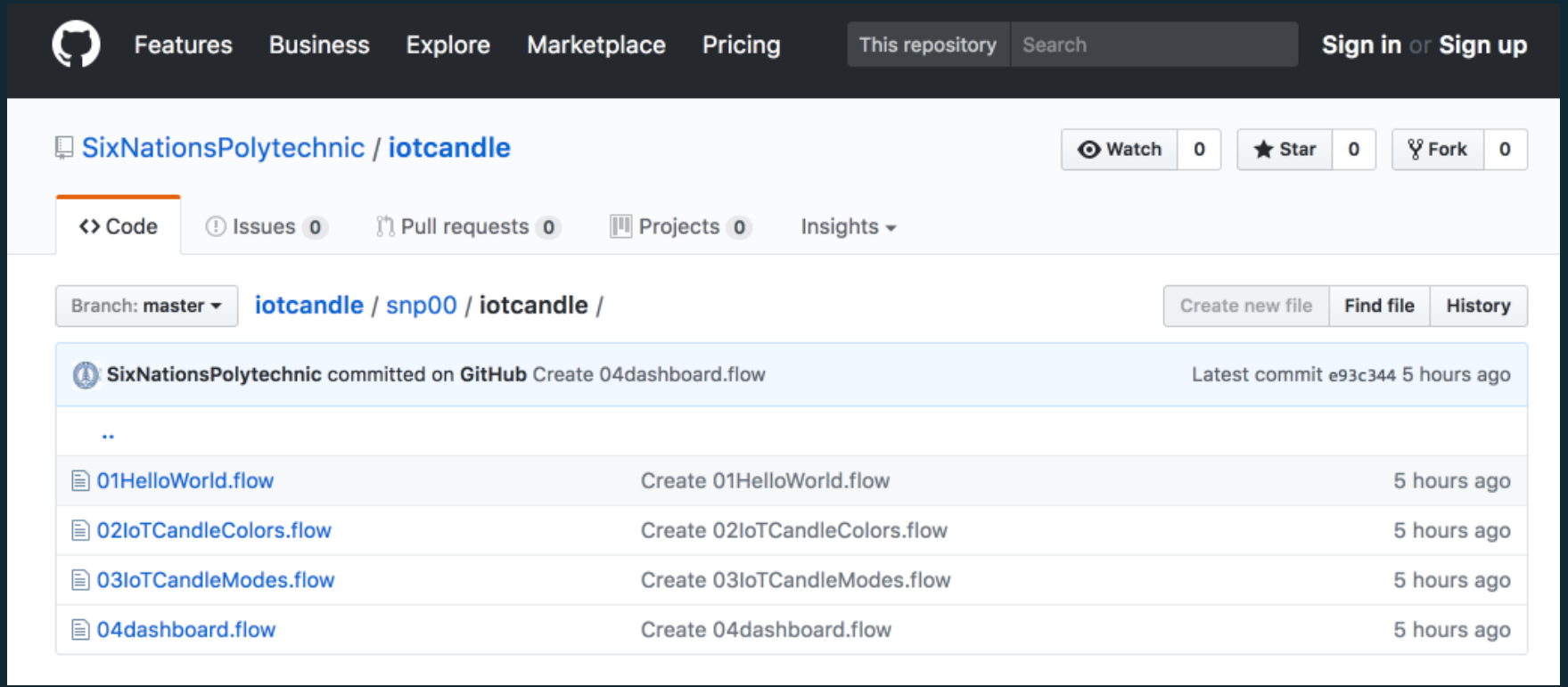
cut (delete and copies) Ctrl+x,

select all = Ctrl-a
copy = Ctrl+c
Paste = Ctrl+p

Code Backup and Restore / storage and share
https://github.com/



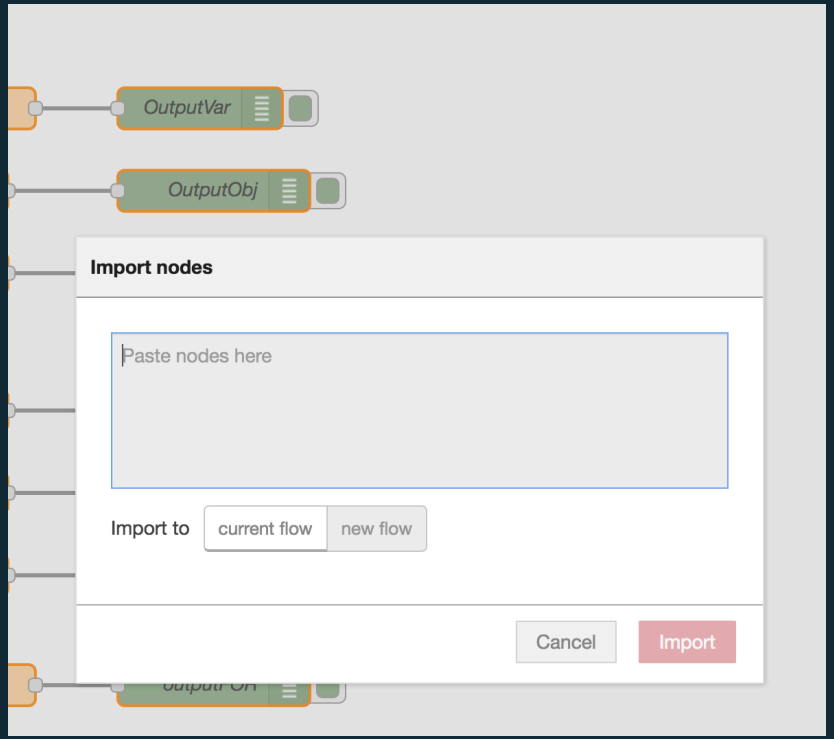
https://github.com/SixNationsPolytechnic/iotcandle/tree/master/snp00/iotcandle

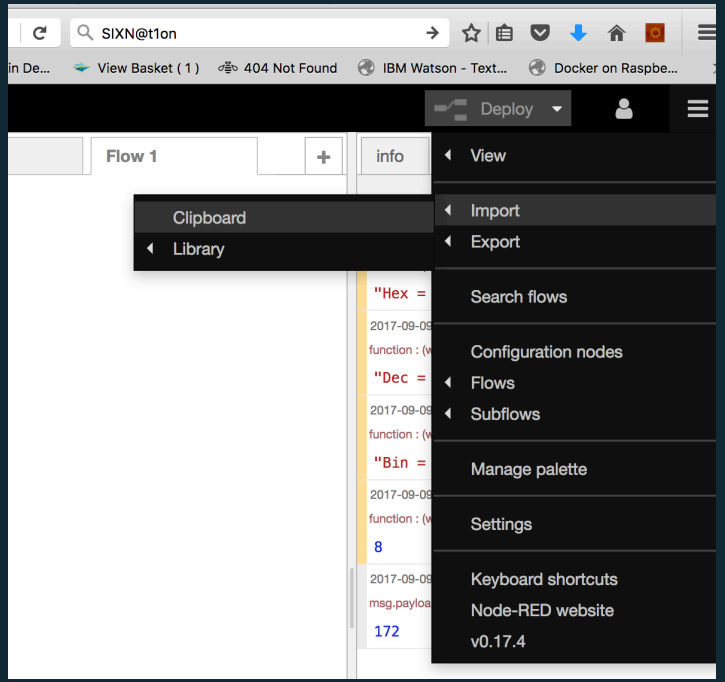


Import some code into your node-red space

- go to gitbub.com
- Sign in with @gmail.com and password
- Open a dashboard.flow
- Copy (CTRL a, CTRL c)
- Go to your node-red and click import
- Paste the code in the text box (CTRL p)
- Click ok and Deploy
- Adjust input/output to your group#3
- Test the code

Demo - Import

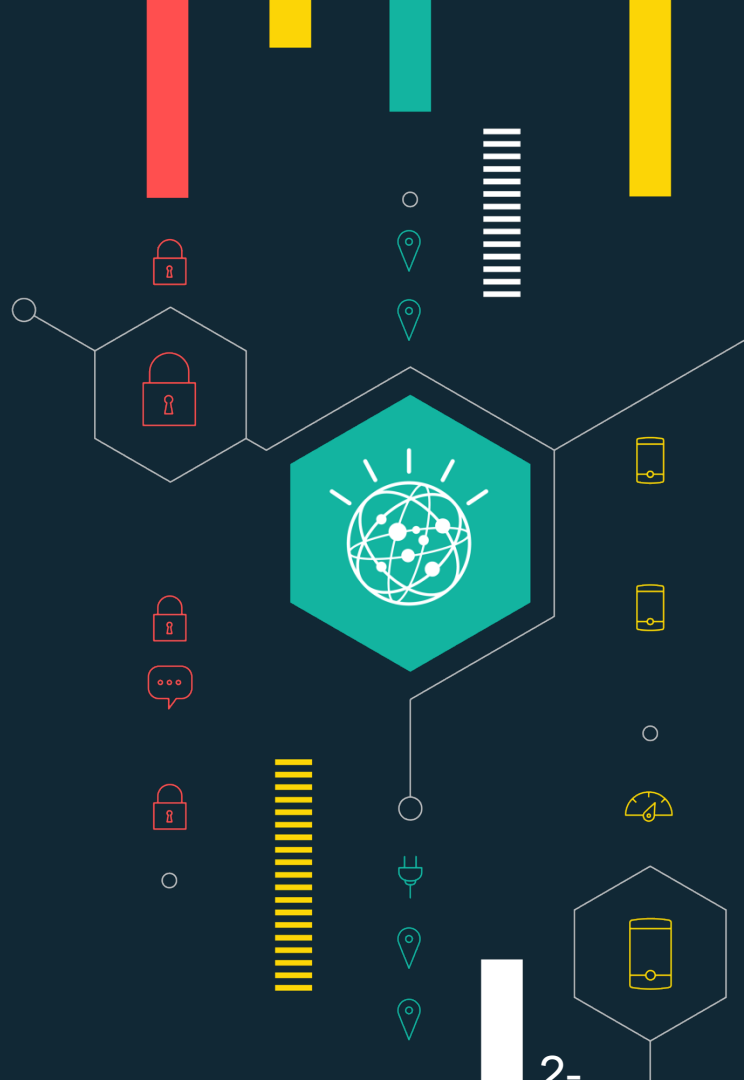

OutputVar
OutputObj
Import nodes
Paste nodes here
Import to
current flow
new flow
Cancel
Import


SIXN@t1on
View Basket (1)
404 Not Found
IBM Watson - Text...
Docker on Raspbe...
Deploy
Flow 1
info
Clipboard
Library
View
Import
Export
Search flows
Configuration nodes
Flows
Subflows
Manage palette
Settings
Keyboard shortcuts
Node-RED website
v0.17.4

IBM Watson IoT

JavaScript

Language Fundamentals I



IBM

Data types

JavaScript has three types: number, string, and boolean

- Everything else is an object

Numbers are always stored as floating-point values

- Hexadecimal numbers begin with 0x

Strings may be enclosed in single quotes or double quotes

- Strings can contains \n (newline), \" (double quote), etc.

Booleans are either true or false

- 0, "0", empty strings, undefined, null, and NaN are false , other values are true

Variables

Variables are declared with a var statement:

- var pi = 3.1416, x, y, name = "Dr. Dave" ;
- Variables names must begin with a letter or underscore
- Variable names are case-sensitive
- Variables are *untyped* (they can hold values of any type)
- The word var is optional (but it's good style to use it)

Variables declared within a function are local to that function (accessible only within that function)
Variables declared outside a function are global (accessible from anywhere on the page)

Comments

Comments are as in C or Java:

- Between `//` and the end of the line
- Between `/*` and `*/`

`/** … */`, are treated just the same as `/* … */` comments;

Examples

var pi = 3.1416; // Number

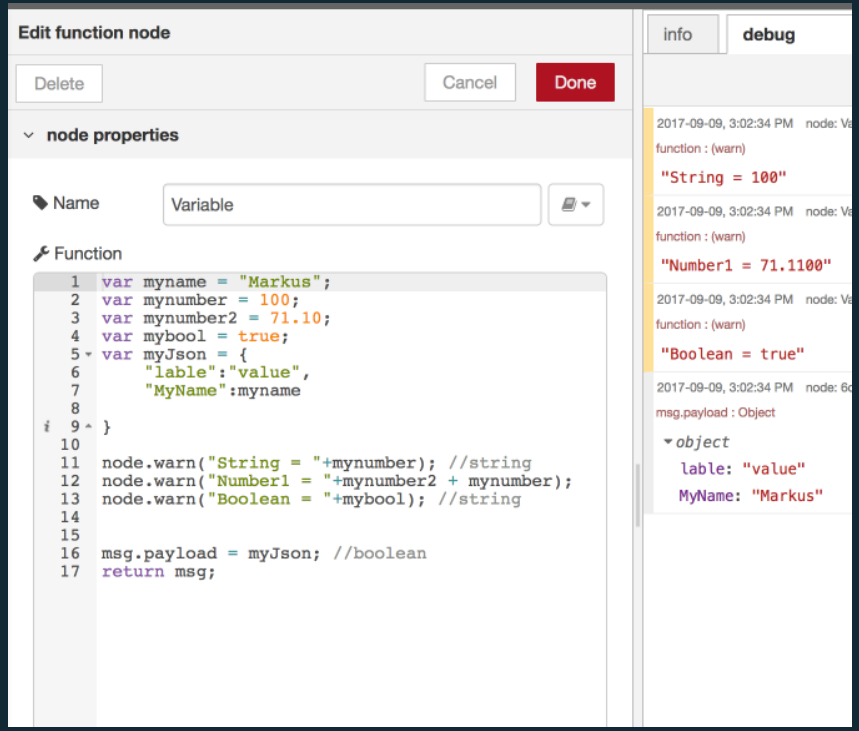
var name = “Markus”; //String

var b = false; // Boolean

var myemptyo = {} //json object

var myemptyo = []; // array object

console.log (myname); // browser to server debug
node.warn (myname); // Node Red debug



Import code into Node-Red

- Grab 02-02-Variable01.flow from github (iotcandle)
- Import and deploy
- Test and play with code.

Operators

Arithmetic operators:

`+` `-` `*` `/` `%` `++` `--`

Comparison operators:

`<` `<=` `==` `!=` `>=` `>`

Logical operators:

`&&` `||` `!` (`&&` and `||` are *short-circuit* operators)

Bitwise operators:

`&` `|` `^` `~` `<<` `>>` `>>>`

Assignment operators:

`+=` `-=` `*=` `/=` `%=` `<<=` `>>=` `>>>=` `&=` `^=` `|=`

Example

```
var i = 1;
var a = 5;

 i = i +1;
 i> 1; // true
 i != 2; //false

 a > 0 && i >0 //true

 a > 10 || i >1 //true
```

Statements

- If statements:

  if (***condition***) ***statement***;

  if (***condition***) ***statement***; else ***statement***;

- Familiar loop statements:

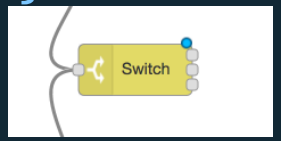
  while (***condition***) ***statement***;

  do ***statement*** while (***condition***);

  for (***initialization***; ***condition***; ***increment***) ***statement***;

Example

```
// if

var i = 1;
var a = 5;

If(a > i){

}else{

}
```




```
//  while

var i = 1;
var a = 5;

while (a > i)
{
 i++;
}
```

```
//  for

for(i=0;i<100;i++)
{
a = i;
}
```

Import code into Node-Red

- Grab 02-02-Variable02.flow from github (iotcandle)
- Import and deploy
- Test and play with code.
- Take a look at IF and WHILE

Test Imported flow and create a for loop

Create For loop;

```
var i = 1;
var a = 5;

for(i=0;i<10;i++)
{
 i++;
 node.warn(" i = "+i);
}
msg.payload = i;
return msg;
```

Import code into Node-Red

- Grab 02-02-Variable03.flow from github (iotcandle)
- Import and deploy
- Test and play with code.
- Take a look at IF and WHILE

Functions

Functions should be defined in the <head> of an HTML page, to ensure that they are loaded first

The syntax for defining a function is:

function ***name***(***arg1***, ***…***, ***argN***) { ***statements*** }

- The function may contain return ***value***; statements
- Any variables declared within the function are local to it

The syntax for calling a function is just

name(***arg1***, ***…***, ***argN***)

Simple parameters are passed *by value,* objects are passed *by reference*

Create Function and a loop

```
/* comment / describe your program */
function hello(name)
{
        node.warn("Hello "+name);
}

for(i=0;i<10;i++)
{
  hello("Markus "+i);
}

msg.payload = i;
return msg;
```

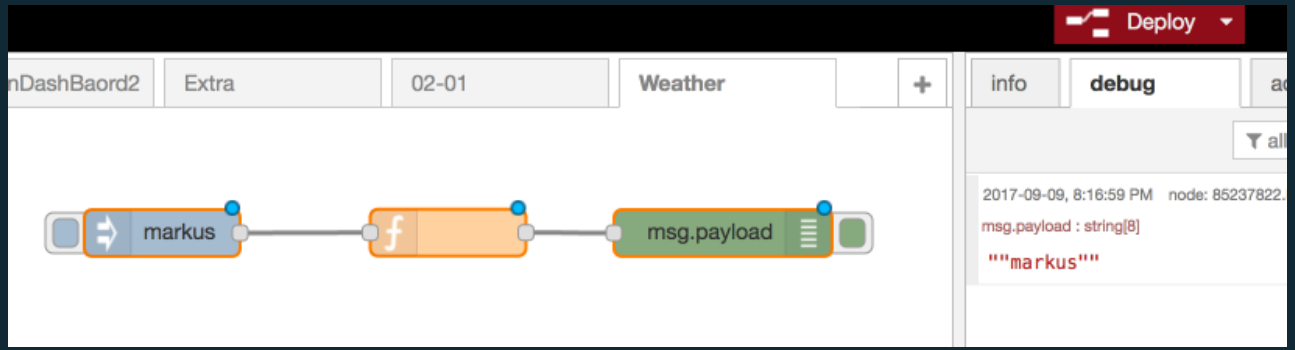
Create JSON object

```
/* comment / describe your program */
var myname = "Markus"
var myJson = {
    "lable":"value",
    "string":myname

}

myJson.age = 48;
myJson.firstName = "Markus";
myJson.lastName = "van Kempen";

msg.payload = myJson;
return msg;
```

Node-RED

msg = JSON Object

msg.playload
= also a JSON (sub) object used to transport data from one node to the other



Numbering Systems		
System	Base	Digits
Binary	2	0 1
Octal	8	0 1 2 3 4 5 6 7
Decimal	10	0 1 2 3 4 5 6 7 8 9
Hexadecimal	16	0 1 2 3 4 5 6 7 8 9 A B C D E F

Decimal vs. Hexadecimal vs. Binary

DEC	HEX	BIN
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

DEC	HEX	BIN
16	10	0001 0000
17	11	0001 0001
18	12	0001 0010
19	13	0001 0011
20	14	0001 0100
21	15	0001 0101
22	16	0001 0110
23	17	0001 0111
24	18	0001 1000
25	19	0001 1001
26	1A	0001 1010
27	1B	0001 1011
28	1C	0001 1100
29	1D	0001 1101
30	1E	0001 1110
31	1F	0001 1111

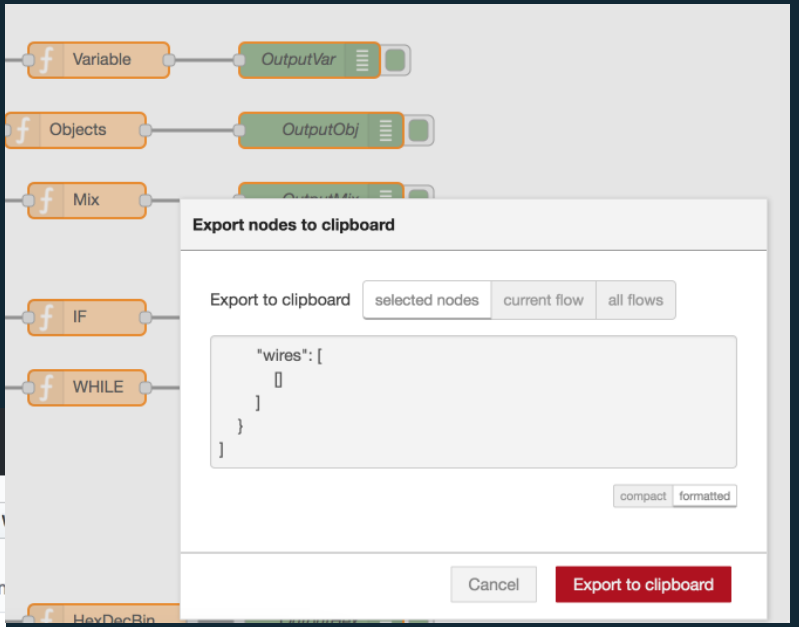
Binary/Dec/Hex

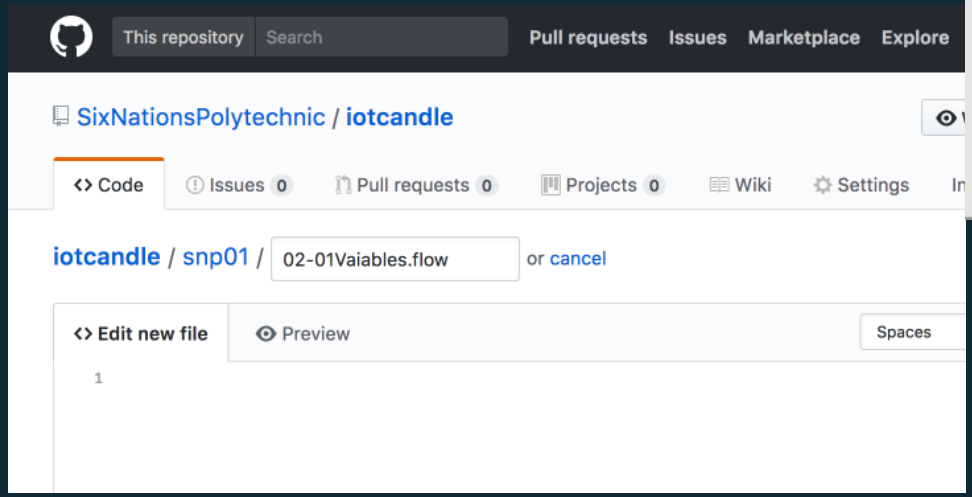
```
var hex = 0xac;

var dec = 100;

var bin = 0b1000;

// conversion via toString(base) base = 10,2,16 etc

bin.toString(10);
```

Backup/save your code in github.com

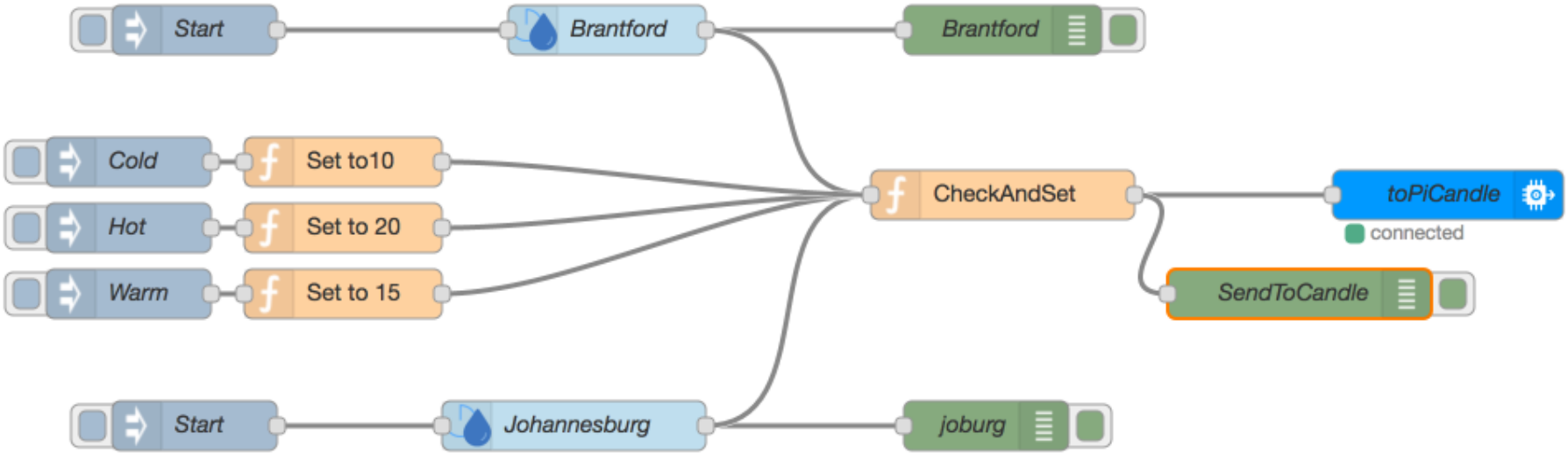
- go to gitbub.com
- Sign in with snpkids01@gmail.com and password
- Export your flow (copy to the clipboard)
- Create a file in github with your group number
  - snp??/ 02-01Variable.flow
- Paste your code in (CTRL p)
- Click commit

Demo

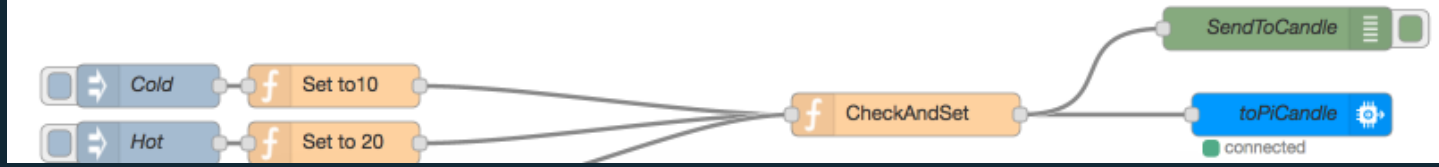

Variable
OutputVar
Objects
OutputObj
Mix
IF
WHILE
Export nodes to clipboard
Export to clipboard
selected nodes
current flow
all flows
"wires": [
[]
]
}
]
compact
formatted
Cancel
Export to clipboard


This repository
Search
Pull requests
Issues
Marketplace
Explore
SixNationsPolytechnic / iotcandle
Code
Issues 0
Pull requests 0
Projects 0
Wiki
Settings
iotcandle / snp01 /
02-01Vaiables.flow
or cancel
Edit new file
Preview
Spaces
1

Weather / temperature form Brantford / Change Candle Color and Mail



Exercise



- Create a new Tab
- Create a temperature simulator
- Check the temperature
  - if (msg.observation.temp <10)

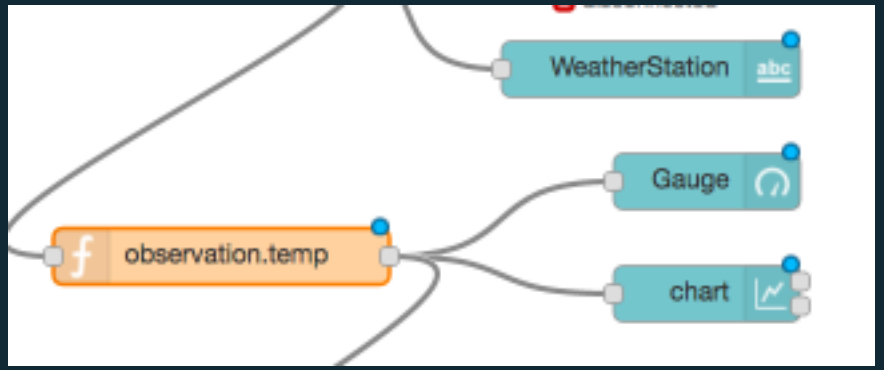
- Set color of candle json to red = hot , blue cold, orange warm
  - if (msg.observation.temp >20)
- r=255;
- Send message to Candle

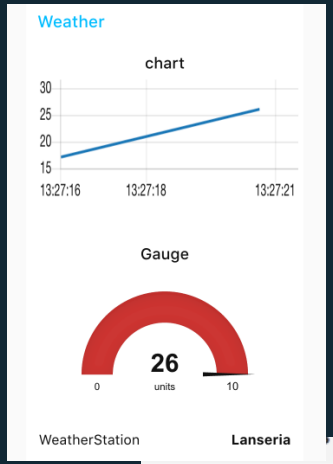
Exercise temperature simulator to set the candle

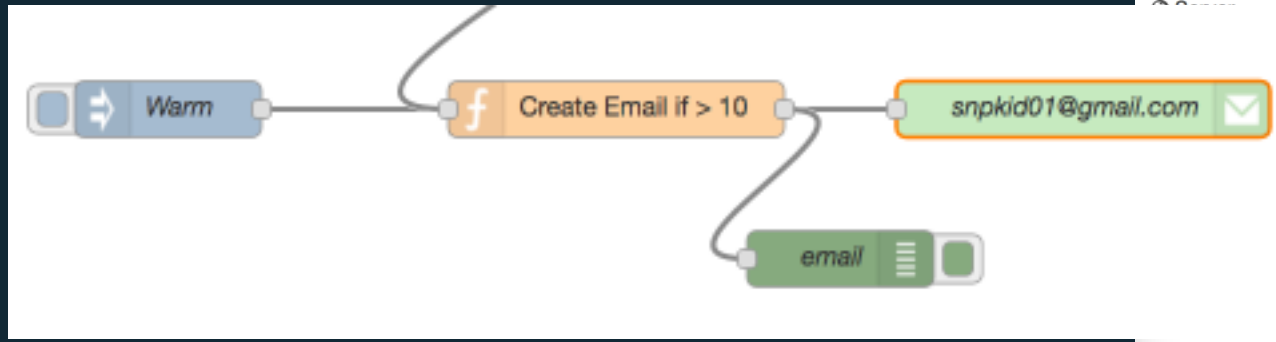

```
msg.observation={}
msg.observation.temp = 10
msg.payload = 10;
return msg;
```

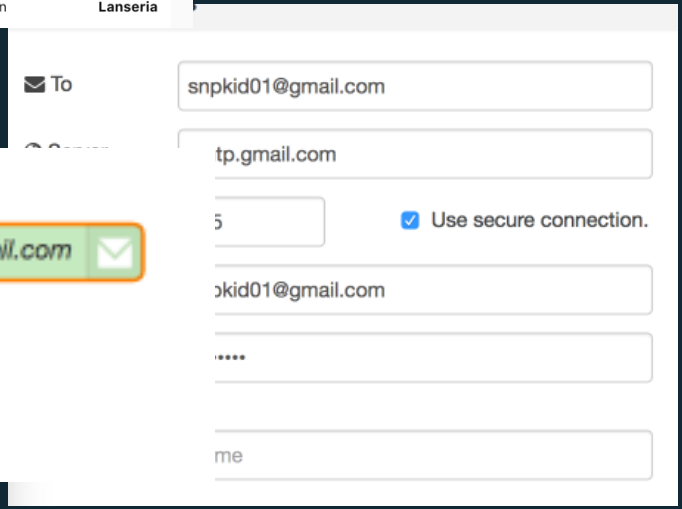

```
var r=0;
var g=0;
var b=0;

if (msg.observation.temp <=10)
      b=255;
if (msg.observation.temp >=20)
      r=255;
if (msg.observation.temp >10 && msg.observation.temp <20 )
{
  r=255;
  g=140;
}
var newmsg = {"cmd" : "setcolor",
"mode":"",
"speed":"",
"rr" : r,
"gg" : g,
"bb" : b
}
msg.payload=newmsg ;
return msg;
```

Add the UI and email functions










IBM
Markus van Kempen
Executive Architect & Venture Capitalist
IBM Corporate Strategy
Innovating with People and Technology
email: mvk@ ca.ibm.com
Twitter: @markusvankempen
Hashtag☺: #MVK


IBM