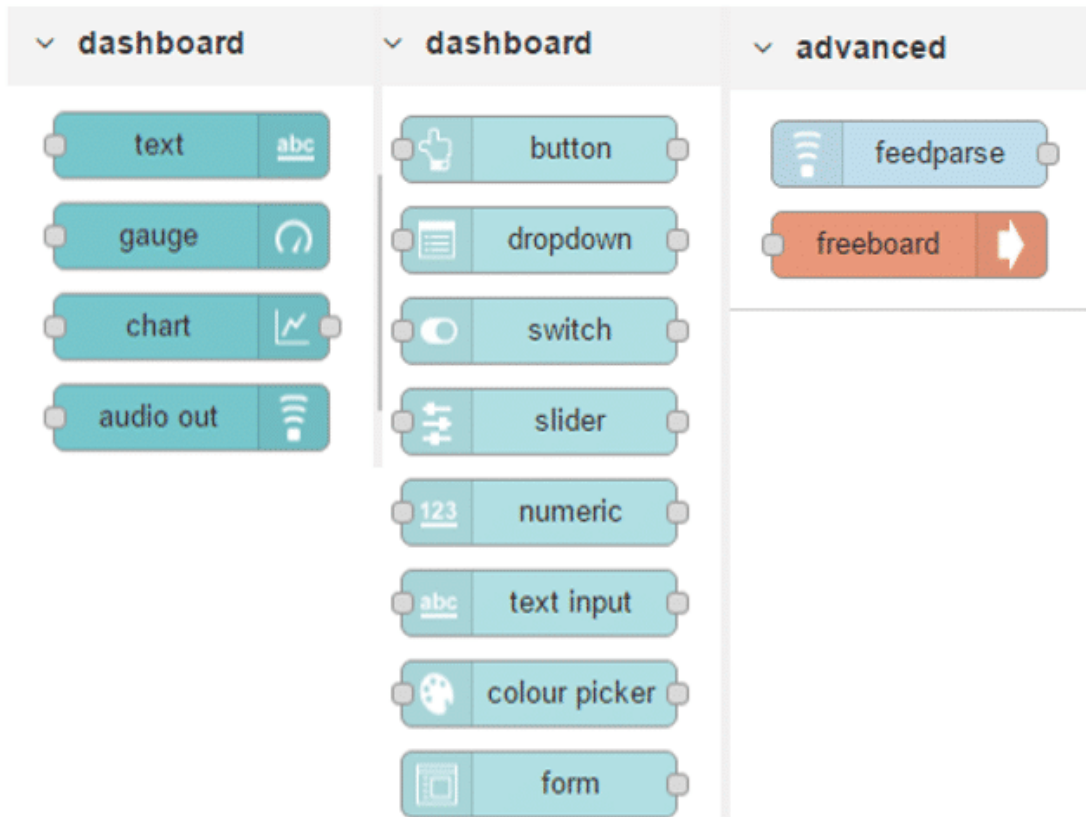




Node-Red Dashboards وواجهات المستخدم في Node-Red

في هذا الدرس سوف نستعرض بعض التقنيات التي تسمح لنا بعمل عرض تصوّري للبيانات (visualize data) التي تمر خلال التدفقات. سوف نركز على ثلاث أساليب:

- استخدام الطرف الثالث من أداة Dashboard وهي FreeBoard (الجزء الأول)
- استخدام العقد الافتراضية لواجهة المستخدم (Dashboard) المقدمة بشكل افتراضي في Node-RED (الجزء الثاني)
- استخدام أداة الرسم التخطيطي للجافا سكريبت القياسية (الجزء 3).



بنهاية هذا الدرس ستكون لديك معرفة كافية لتحديد الطريقة المناسبة لتحقيق احتياجاتك في الحصول على عرض تصوّري للبيانات على صفحة ويب.

استخدام خدمة لوحة التحكم FreeBoard :

تعد هذه الطريقة مثالا بسيطا لقراءة وعرض البيانات تصويريا باستخدام عقدة FreeBoard من عقد الـ Node-Red . سوف نستخدم خدمة FRED القائمة على السحابة كمحرك للـ Node-Red ونستعرض البيانات تصويريا من خلال خدمة الطقس على الانترنت يمكن تقسيم هذا المثال إلى جزئين:

الأول: إنشاء عقدة الطقس openweathermap في FRED الموجودة ضمن العقد المدمجة. كما قمنا به في درس بناء التدفقات: تنبيهات الطقس

الثاني: عرض البيانات من عقدة openweathermap باستخدام عقدة FreeBoard

أولا لإنشاء عقدة الطقس قم بإنشاء حساب أو تسجيل الدخول إلى <http://openweathermap.org/appid> للحصول على مفتاح API الخاص بك.

API key

a8f237f405fca445e4e878d3d7f8aba2

Reset APPID

قم بسحب عقدة الطقس openweathermap إلى مساحة العمل والنقر عليها مرتين لإدخال البيانات: مفتاح API وإحداثيات مدينتك.

نقوم بعد ذلك بربطها بعقدة الإخراج debug والضغط على نشر Deploy

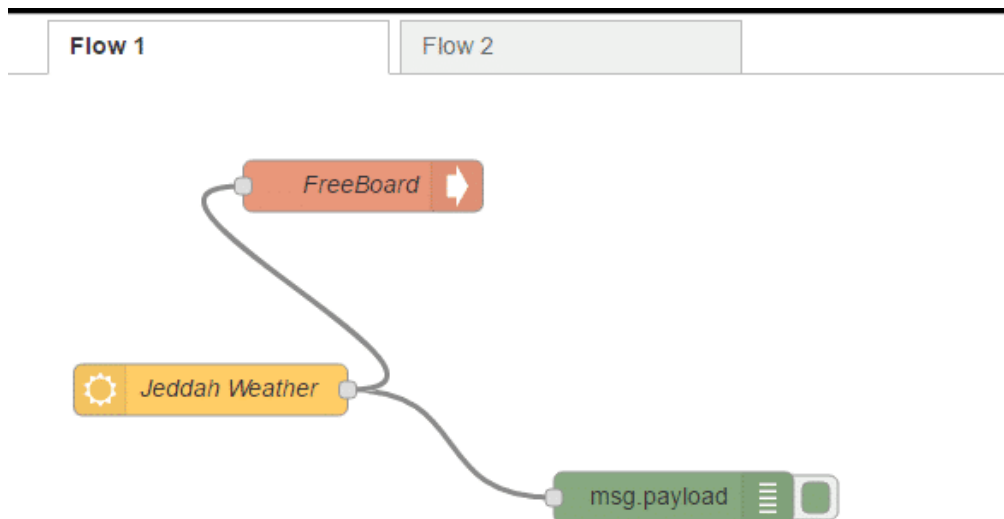
The screenshot shows the Node-RED web interface. On the left, a flow named 'Flow 1' contains two nodes: 'Jeddah Weather' (an orange node with a gear icon) and 'msg.payload' (a green node with a list icon). They are connected by a wire. On the right, the 'debug' tab is active, displaying a log entry for 3/16/2017, 1:08:28 PM. The log shows the message payload as an object with the following data:

```
{  "weather": "Clouds",  "detail": "scattered clouds",  "tempk": 297.9,  "tempc": 24.6,  "humidity": 100,  "maxtemp": 297.9,  "mintemp": 297.9,  "windspeed": 9.68,  "winddirection": 343.501,  "location": "Şumaymah",  "sunrise": 1489635022,  "sunset": 1489678376,  "clouds": 32,  "description": "The weather in Şumaymah at coordinates: 21.29, 39.24 is Clouds (scattered clouds)."} 
```

نجد أن البيانات التي تظهر في لوحة الإخراج debug هيكل JSON مما يعني أنه يمكننا استخدام أي من هذه القيم ببساطة.

سنقوم بإنشاء صفحة تحكم لعرض البيانات مرئياً باستخدام عقدة FreeBoard .

نقوم بسحب عقدة FreeBoard إلى ساحة العمل والنقر عليها مرتين لتسميتها



بالنقر على نشر Deploy ستحصل عقدة الطقس على البيانات وتقوم بإرسالها إلى كلا من عقدة الإخراج debug وعقدة freeboard. عقدة FreeBoard ذكية جداً فهي تقوم باستقبال البيانات وتحليلها ومعرفة كيفية جعلها متاحة باستخدام واجهة المستخدم FreeBoard للحصول على مزيد من المعلومات عن هذه العقدة قم بالنقر على علامة التبويب info في لوحة الإخراج

Deploy ▼
☰

info

debug

dashboards

Node

Name	FreeBoard
Type	freeboard
ID	49a102e8.ece9dc

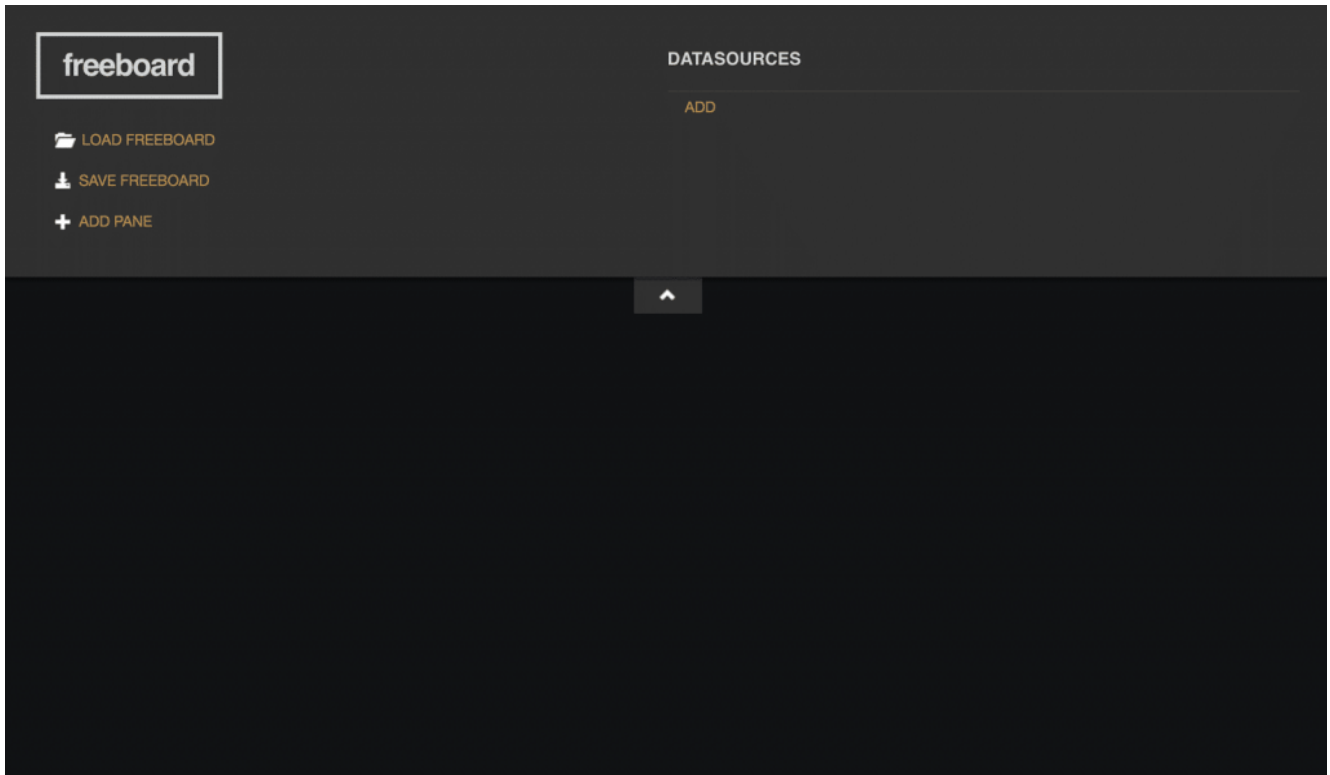
► **Properties**

Using this node you can transmit information to freeboard. Just send some JSON msgs (for example using a function node) to this node and receive them in freeboard.

Open your local Freeboard and add the Datasource that is named like the Freeboard Node in Node-RED (see "name" below "Properties" in the table above).

قم بالضغط على الرابط في لوحة المعلومات أو الرابط التالي:

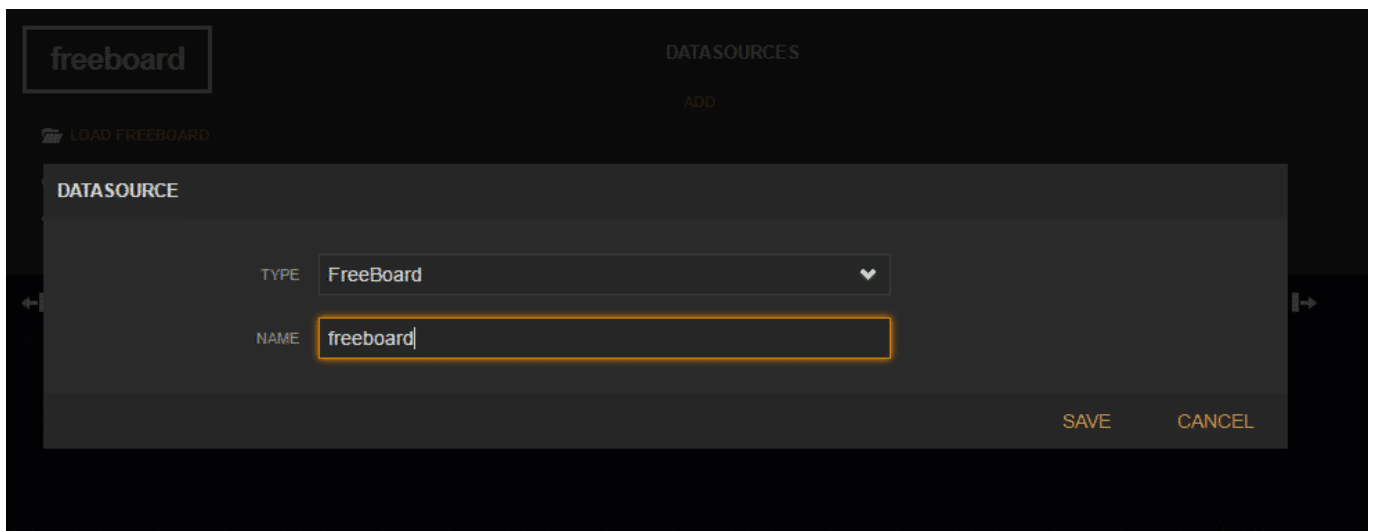
[/https://{username}.fred.sensetecnic.com/api/freeboard](https://{username}.fred.sensetecnic.com/api/freeboard)



هذا التبويب يسمح لك بعمل عرض تصويري للبيانات في FreeBoard وحفظها وتحميلها.

سنقوم الآن بعمل عرض تصويري لبيانات الطقس لدينا، نحتاج أولاً إلى إضافة مصدر للبيانات في freeboard . انقر على "ADD" تحت عنوان "DATASOURCES"

وتحت عنوان "TYPE" حدد اسم العقدة "FreeBoard". في حالتنا أطلقنا عليها اسم "freeboard" مما سيسمح لنا بالوصول إلى أي بيانات تتصل بعقدة "freeboard" في Node-Red.



سنقوم الآن بإضافة pane انقر على "ADD PANE" سوف يظهر لنا جزء جديد فارغ. ولإضافة "Widget" قم بالنقر على علامة (+) في الجزء الجديد واختر "Gauge"

تحت "DATASOURCE" قم باختيار عقدة "Freeboard". كما تشاهد فإن حقول البيانات المختلفة في عقدة الطقس openweathermap هيكل JSON متاحة لتصويرها مرئياً.

إذا لم تظهر لك الحقول في القائمة المنسدلة قم بالرجوع إلى صفحة Node-Red وانقر على نشر deploy مرة أخرى ليتمكن freeboard من استقبال البيانات وتخزينها

في هذا المثال قمنا باختيار "tempC"

WIDGET

TYPE: Gauge

TITLE: Temperature in Celsius

VALUE: datasources["freeboard"]["tempC"] + DATASOURCE .JS EDITOR

UNITS: C

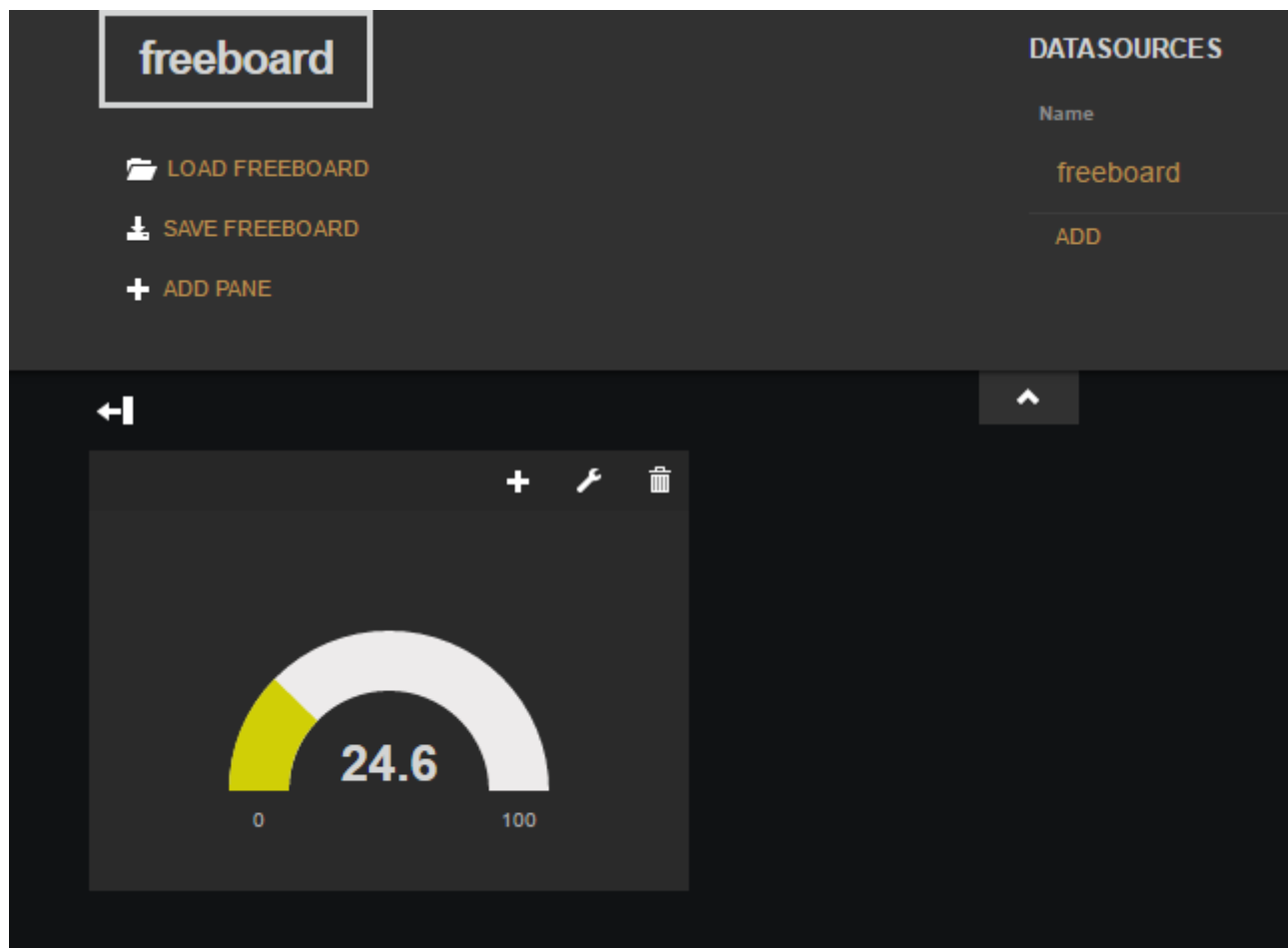
MINIMUM: 0

MAXIMUM: 100

SAVE CANCEL

FreeBoard يأخذ البيانات بشكل أساسي هيكل JSON لينتج الحقول الأخرى

العرض التصوري لبياناتك ينبغي أن يظهر بهذا الشكل:

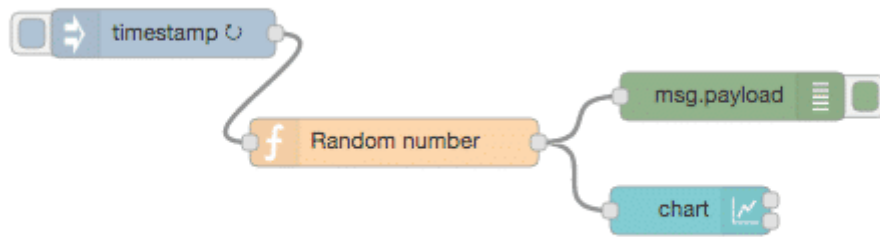


مقدمة لعقد واجهة المستخدم من Node-RED-dashboar :

في هذا المثال سنستخدم العقد المضمنة في Node-Red. إذا كنت تستخدم FRED فتتحقق من إضافة هذه العقدة من خلال زر إضافة/إزالة العقد في لوحة الإدارة وأنت لا تستخدم واجهة المستخدم القديمة للعقد.

في البداية سنقوم ببناء تدفق بسيط يرسل أرقامًا عشوائية بين 0 و 99 إلى رسم بياني بسيط. سنحتاج إلى:

- عقدة إدخال لتكرار إطلاق البيانات مرارا كل بضعة ثوان
- عقدة معالجة function لتوليد أرقام عشوائية
- وواحدة من عقد node-red-dashboar وفي هذه المرة سنختار عقدة الرسم البياني Chart node



قبل أن ننظر في كيفية عمل عقدة الرسم البياني، سنقوم بتكوين عقدة إدخال تقوم بإرسال طابع زمني كل 5 ثوان عن طريق وضع الحمولة على الطابع الزمني ونختار الفترة 5 ثوان في حقل التكرار.

Edit Inject node

✉ Payload ▼ timestamp

📄 Topic

🔄 Repeat interval

every 5 seconds

☐ Inject once at start?

📁 Name Name

Note: "interval between times" and "at a specific time" will use cron. See info box for details.

Ok Cancel

سيمثل هذا جزء الإدخال في التدفق الذي نريد بناءه

لإعداد عقدة المعالجة function لتوليد الأرقام العشوائية سنقوم بكتابة دالة رياضية بسيطة بواسطة جافا سكربت

```
msg.payload = Math.round(Math.random()*100);
return msg;
```

لمحة عن الكود :

msg المتغير الذي يحتوي على نص الرسالة. استخدمنا الدالة Math.round() للحصول على أعداد مقربة إلى أقرب عدد صحيح

والدالة Math.random()*100 لتوليد أرقام عشوائية بين 0 و99

ستولد عقدة المعالجة رقم عشوائي بين 0 – 99 وتمررها إلى عقدة الرسم البياني

بالنقر مرتين على عقدة الرسم البياني ستظهر لنا خيارات إعدادها

Edit chart node

Delete

Cancel

Done

Group

Add new ui_group...

Size

auto

Label

chart

Type

Line chart

X-axis

last

1

hours

OR

1000

points

X-axis Label

HH:mm:ss

Y-axis

min

max

Legend

None

Interpolate

linear

Blank label

display this text before valid data arrives

Name

إذا قمت بالنقر على زر حقل Group، سيطلب منك إعداد علامات التبويب لواجهة المستخدم

Tab

Home

Width

6

Name

Default

☒ Display group name

يسمح خيار التبويب بتحديد علامات التبويب التي تريدها في صفحة واجهة المستخدم وسترى عنصر واجهة المستخدم (في مثالنا هذا الرسم البياني). علامة التبويب الافتراضية هي الصفحة الرئيسية Home التي نستخدمها هنا. إذا قمت بالنقر على زر تعديل edit في يمين حقل التبويب ستتمكن من إنشاء تبويب جديد واختياره.

الآن سنستخدم الافتراضي Home

حقل الاسم هو الاسم الأساسي لعقدة Node-Red - الافتراضي هنا هو الرسم البياني ويمكنك تسميته كما شئت

حقل المجموعة Group يسمح لك بتجميع عناصر واجهة المستخدم - سيأتي فيما بعد كيفية عمله عند إضافة عنصر واجهة مستخدم آخر -

يمكنك تسميته بأي نص الآن سنستخدم الافتراضي Home

حقل محور x : يتيح لك تحديد مقدار البيانات التي تريد من الرسم البياني تخزينها وعرضها. كلما زادت القيمة في حقل last يعني أن بيانات أكثر سيتم تخزينها وعرضها على الرسم

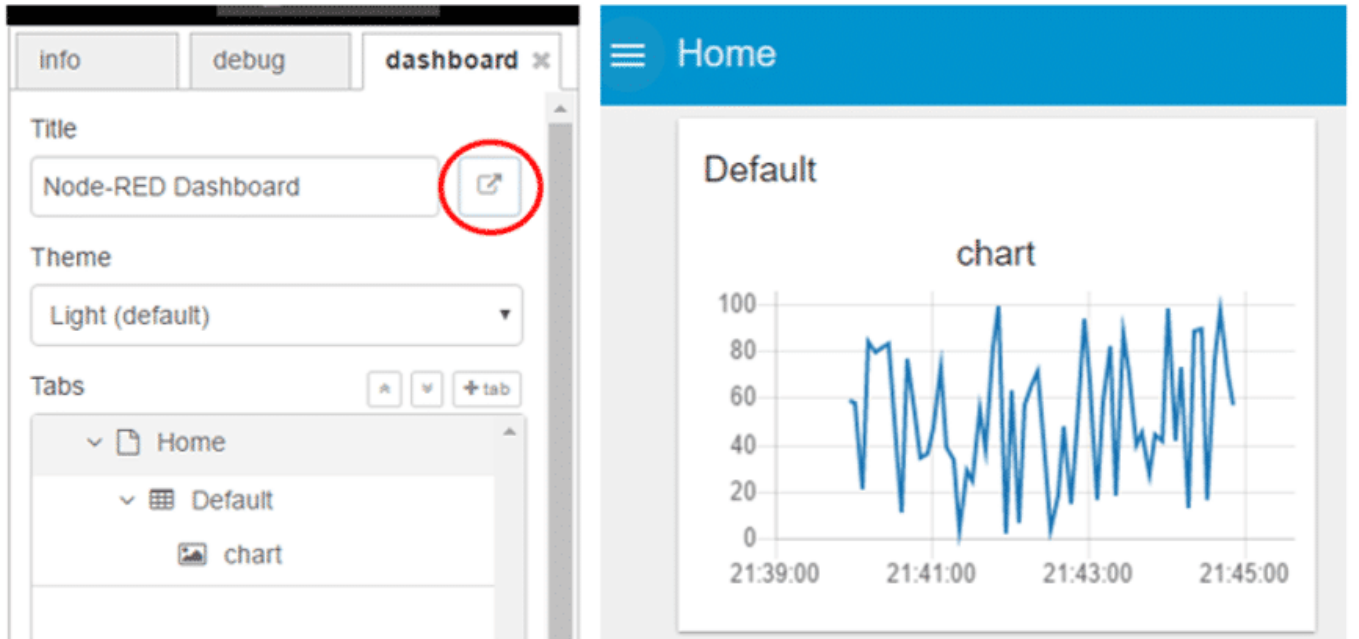
سنختار مدى قصير وهو 5 دقائق أي أن الرسم سيتضمن البيانات الواردة خلال 5 دقائق.

وأخيرا حقل Interpolate سيحدد كيف سيقوم الرسم البياني بإدراج القيم الفعلية التي يستقبلها في الرسم ، سنستخدم الافتراضي الخطي linear

قم بتوصيل العقد واضغط على زر النشر Deploy

تأكد أن لوحة الإخراج debug تُظهر أرقاماً عشوائية

ثم توجه إلى تبويب dashboard الافتراضية الخاصة بك لرؤية النتيجة عند العمل في FRED ستجد واجهة الاستخدام الخاصة بك كالتالي:



أو بزيارة الرابط التالي

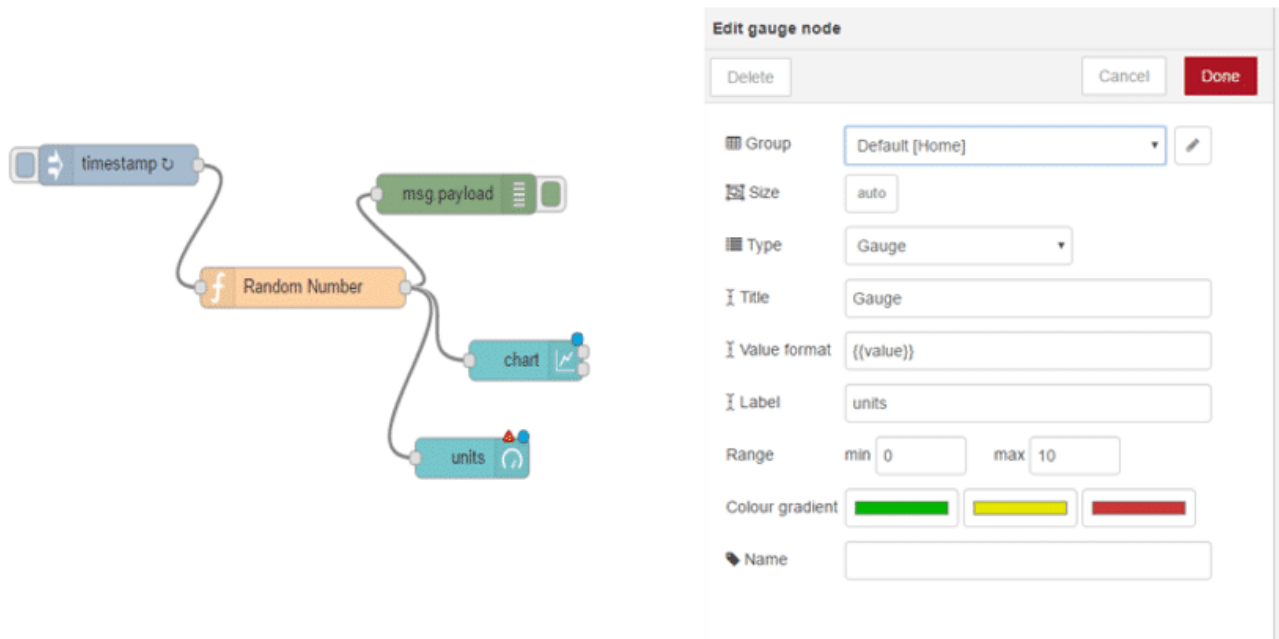
[/https://{your username}.fred.sensetecnic.com/api/ui](https://{your username}.fred.sensetecnic.com/api/ui)

يمكن الحصول بهذه الطريقة على رسوم بيانية رائعة وفقاً لمدى البيانات التي تختارها (محور Y) مع الزمن (محور X)

ويظهر اسم الرسم Default كما تم اختياره خلال تهيئة عقدة الرسم البياني

إذا كنت قد أنشأت علامات تبويب خاصة بك فستجد بالنظر إلى الزاوية العلوية اليسرى لصفحة الويب علامة التبويب الرئيسية Home وبالنقر عليها ستظهر لك قائمة منسدلة لعلامات التبويب التي أنشأتها.

سنقوم الآن بإضافة عناصر واجهة المستخدم أخرى إلى Dashboard. قم بإضافة عقدة Gauge إلى مساحة العمل وربطها بعقدة function. وبالنقر عليها مرتين لفتح الإعدادات الخاصة بها.



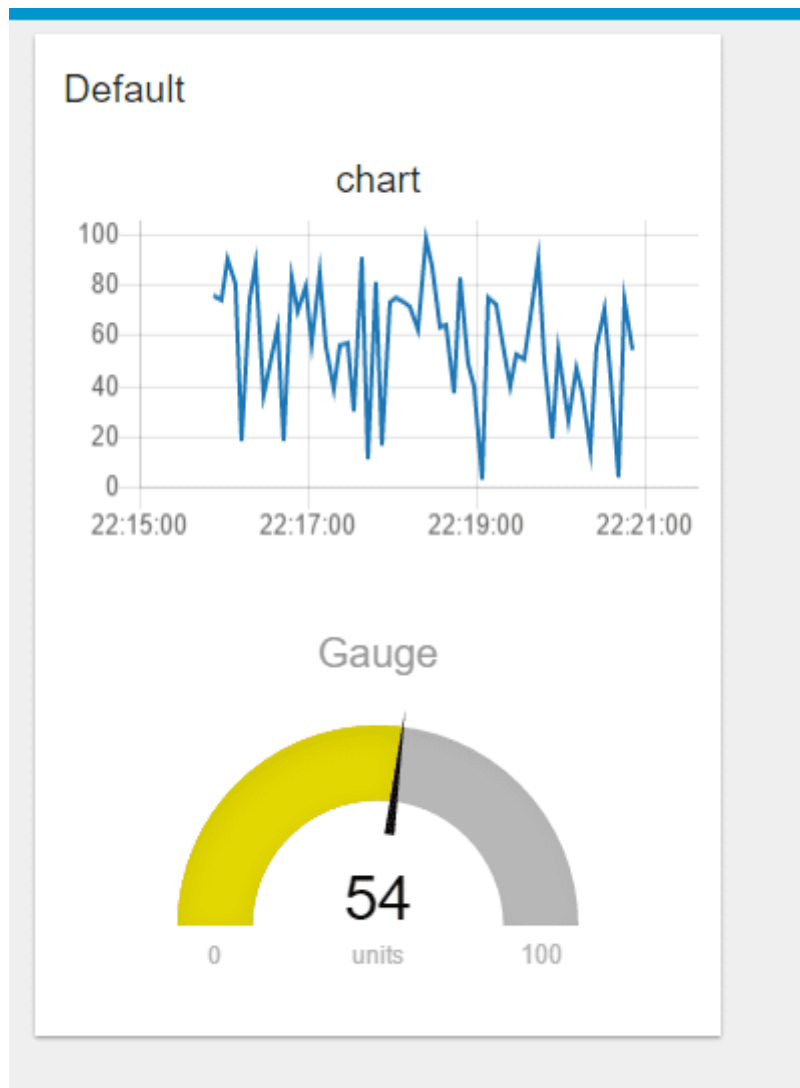
سوف نستخدم هنا نفس التبويب السابق Home ونفس المجموعة Default [Home]

حقل Min و Max يسمحان لنا بتحديد الحد الأدنى والأقصى في المقياس الذي سيظهر. تأكد من تعيين الحد الأقصى 100 ليعطى أعلى قيمة من الأعداد العشوائية التي ستولدها عقدة function

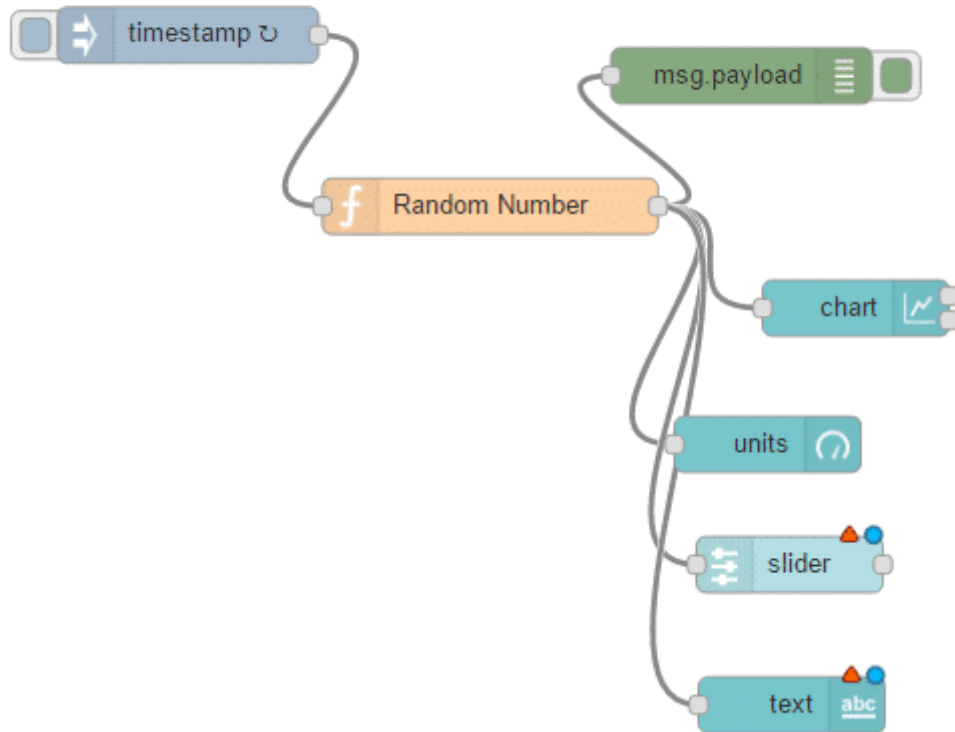
في حقل التدرج اللوني يمكنك تغيير الألوان التي تظهر في المقياس

قم بالضغط على نشر deploy وتوجه إلى الصفحة الخاصة بك Dashboard لمشاهدة النتيجة

سيُظهر الرسم القيم العشوائية التي تم توليدها في الخمس دقائق الأخيرة وسيمثل المقياس القيمة الأخيرة



كمثال أخير سنستخدم عقدتين أخرى من عقد واجهة المستخدم وهما عقدة Slider وعقدة النص text لإظهار نفس البيانات على شريط تمرير وكسلسلة نصية. قم بسحبهما إلى مساحة العمل وربطهما بعقدة function



سنضع هاتين العقدتين تحت نفس التبويب Home لكن سنختار اسم مجموعة آخر وليكن "authorWidget" ستحتاج إلى النقر على إضافة مجموعة واجهة استخدام جديدة (add new_ui group) من القائمة المنسدلة لحقل Group ستحتاج أيضا لتعديل الحد الأدنى والأقصى للقيم (max: 100) في عقدة slider لإظهار الموقع الصحيح في شريط التمرير

Edit slider node

Delete
Cancel
Done

Group
anotherWidget [Home]

Size
auto

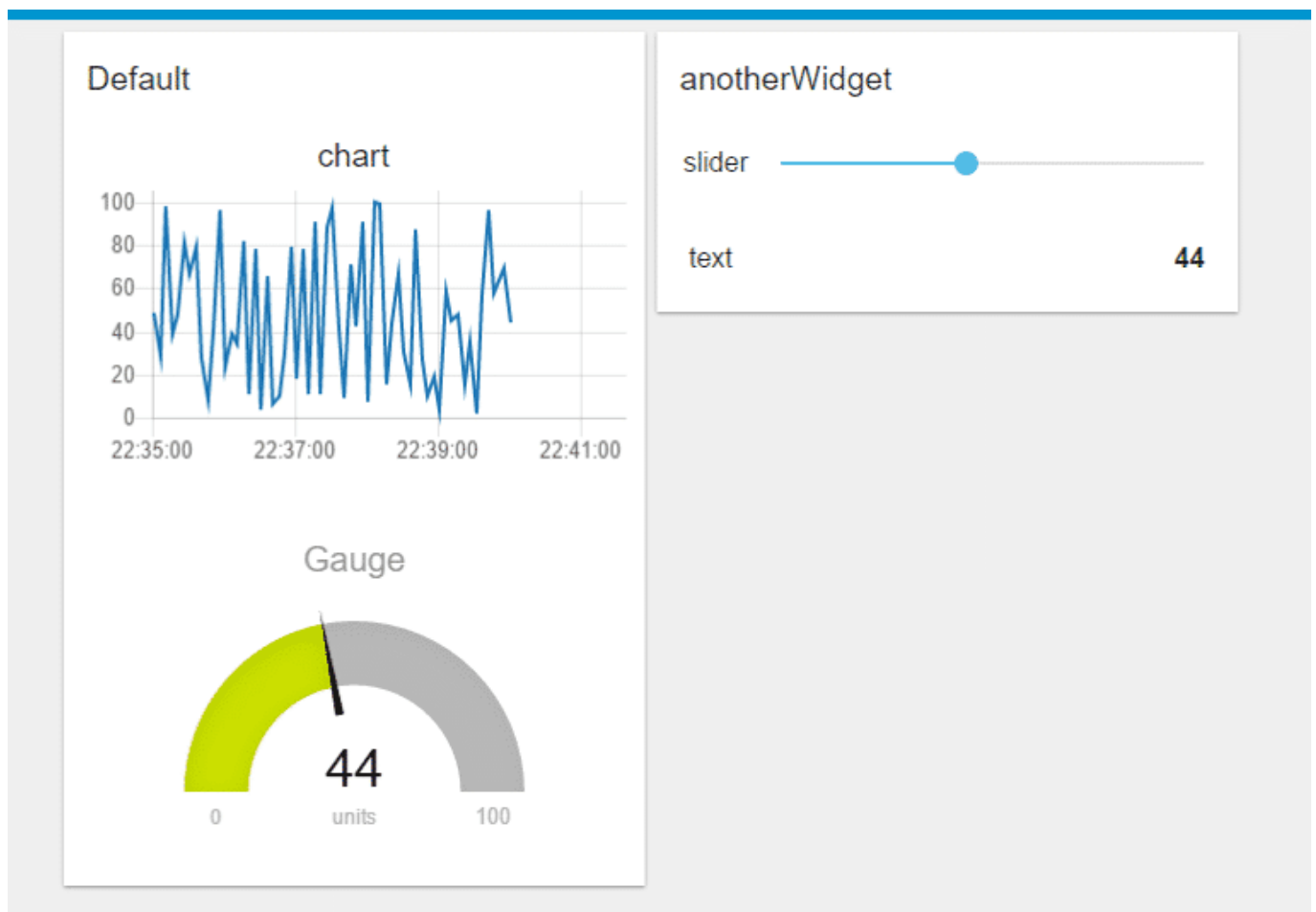
Label
slider

Range
min 0 max 100 step 1

→ If `msg` arrives on input, pass through to output: ☒

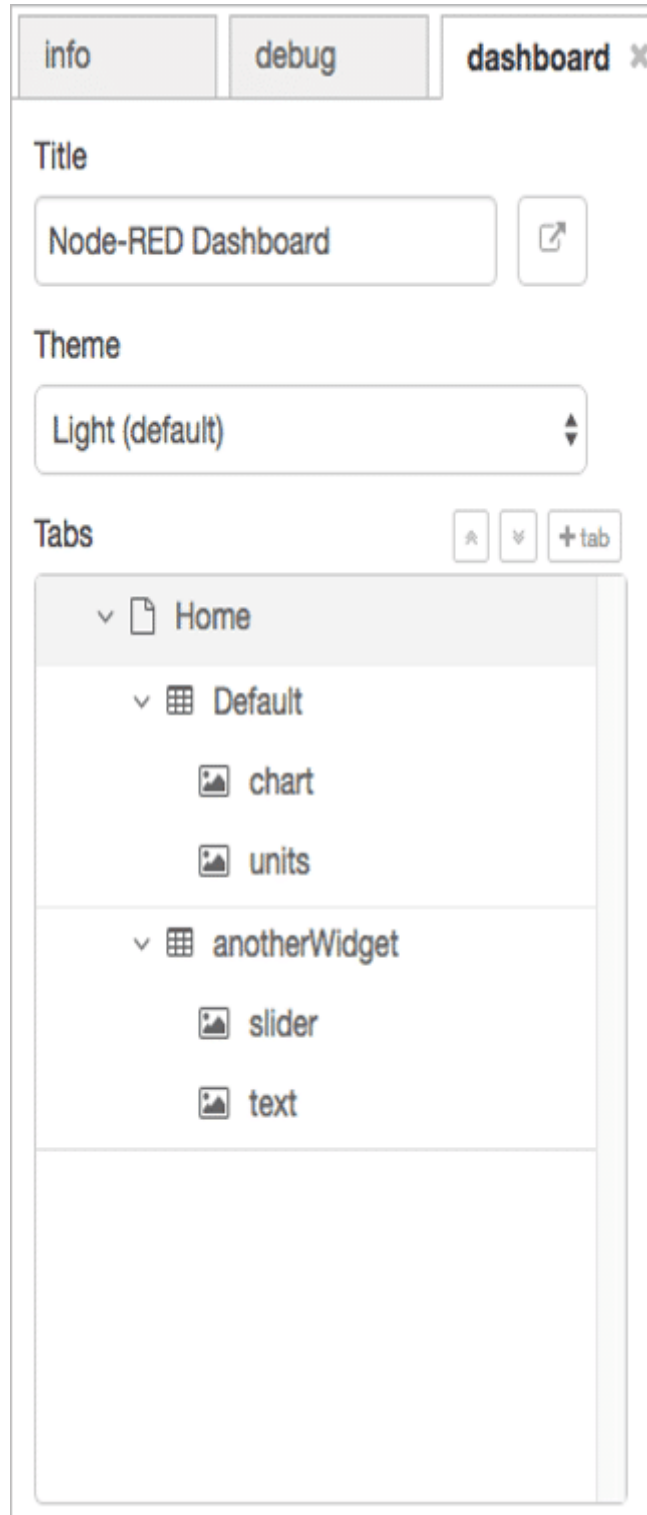
☒ When changed, send:

قم بالضغط على نشر deploy ومشاهدة النتيجة في صفحتك Dashboard



في علامة تبويب dashboard في FRED يمكنك إعادة ترتيب العناصر في الصفحة.

إذا لم يكن تبويب dashboard ظاهرا لك فانقر على زر القائمة في أعلى الزاوية اليمنى من الصفحة واختر dashboard < view



لديك الآن الأساسيات اللازمة لبناء صفحتك الخاصة dashboard باستخدام بيانات العالم الحقيقي وربطها بعقد أخرى.

استخدام مكتبة الرسم البياني للجافا سكريبت لبناء dashboard مخصصة :

سنستخدم في هذا المثال من درس بناء Dashboards وواجهات المستخدم خدمة الويب التي تتمثل في عقدة HTTP حيث تسمح لنا باستضافة صفحات ويب عبر قبول طلبات HTTP

لقد استخدمنا بالفعل هذا النهج في مثالنا الأول دون شرح التفاصيل من خلال عقدة الطقس openweathermap

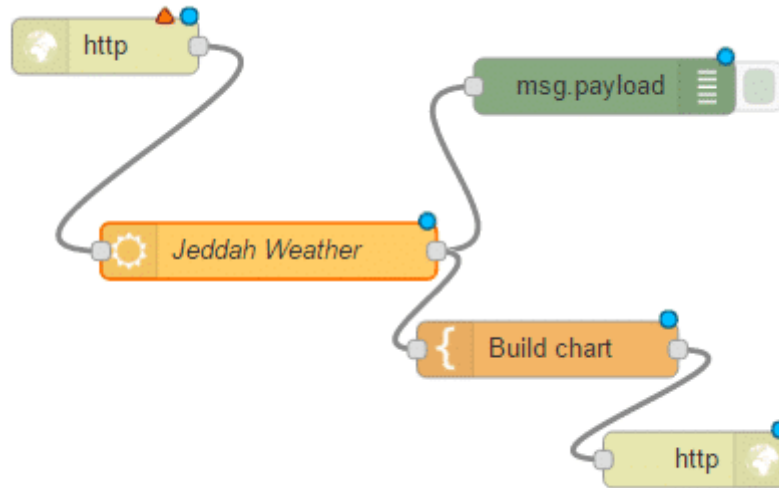
حيث كانت تولد البيانات في هيكل JSON ثم عرضها تصويريا باستخدام مكتبة Morris.JS للرسوم البيانية.

تدعم المكتبة 4 أنواع رئيسية من الرسوم: الرسم الخطي والمساحة والأعمدة bar chart والحلقي donut chart

سنستخدم في هذا المثال الرسم الحلقي لكن الطريقة ستكون نفسها بالنسبة لكل أنواع الرسوم البيانية.

سنبني تدفقاً يتكون من 4 عقد. العقدة الأولى والأخيرة هي عقدة HTTP كمدخل ومخرج تعاملان معا بحيث تستقبل طلبات HTTP وترسل ردود HTTP

يمثل التدفق مثالا بسيطا لخاصة ويب لرسم بيانات الطقس



يستقبل هذا التدفق طلبات HTTP من أي مصدر وليكن المتصفح، عند وصول الطلب يتم الاستعلام عن الطقس عبر عقدة openweathermap ثم يستخدم عقدة القالب template لبناء صفحة HTTP باستخدام بيانات الطقس ثم يمرر ذلك إلى عقدة الإخراج HTTP التي ترسل صفحة ويب للمتصفح.

بالنقر مرتين على عقدة الإدخال HTTP وهيئة مع موقعك في هذا المثال سيكون:

public/weather/

وللوصول إليها فيما بعد نستخدم الرابط التالي

<https://{user name}.fred.sensetecnic.com/api/public/weather>

عند وصول طلب إلى عقدة الإدخال HTTP تتكون رسالة لتحريك العقدة التالية في التدفق وهي عقدة الطقس

تحصل عقدة الطقس على البيانات وفق إحداثيات الموقع الذي تم إعدادها بالنسبة له وتقوم بإرسالها كهيكل JSON إلى عقدة القالب template .

عقدة HTML template هي عقدة أخرى مبنية في Node-RED ، و التي تشبه عقدة function ، حيث تسمح لك ببناء تعليمات برمجية . فبدلاً من استخدام الجافا سكريبت (JavaScript) مثل عقدة function ، فإن عقدة template تعمل مع نص مثل HTML .

Edit template node

Delete
Cancel
Done

Name
Build chart

Set property
msg.payload

Template
Syntax Highlight:
HTML

1 This is the payload: {{payload}} !

قم بكتابة كود html المستخدم في عقدة template, كما هو موضح أدناه :

```

<!doctype html>
<head>
  <title>A Node RED Example</title>
  <link rel="stylesheet"
href="//cdnjs.cloudflare.com/ajax/libs/morris.js/0.5.1/morris.css">
  <script src="//cdnjs.cloudflare.com/ajax/libs/raphael/2.1.0/raphael-
min.js"></script>
  <script src="//ajax.googleapis.com/ajax/libs/jquery/1.9.0/jquery.min.js"></script>
  <script
src="//cdnjs.cloudflare.com/ajax/libs/morris.js/0.5.1/morris.min.js"></script>
</head>
<html>
  <div id="chart-example" style="height: 250px;"></div>
  <script>
Morris.Donut({
  element: 'chart-example',
  data: [
    {label: "Temperature (Celcius)", value: {{payload.tempc}} },
    {label: "Humidity", value: {{payload.humidity}} },
    {label: "Wind Speed (knts)", value: {{payload.windspeed}} }
  ]
});
</script>
</html>

```

يبدأ الكود بهيكل <head>

لتعريف المكتبات الخارجية التي سيتم استخدامها وهنا قمنا بتعريف مكتبات *Morris.JS*

لاحظ بأننا قمنا بجمع جميع عناصر صفحة الويب في عقدة واحدة قد لا يكون هذا مناسباً في مشاريع أخرى.

في السطر العاشر <div> يتم تعريف الاسم والارتفاع

وفي السطر 11 يبدأ الكود المتعلق بالرسم البياني وفقاً لمكتبة *Morris.JS* وتحديد الرسم البياني الحلقي *donut chart*

في السطر 15 و 16 و 17 يتم تعريف عناصر الرسم البياني

وهنا تم اختيار 3 عناصر : درجة الحرارة والرطوبة وسرعة الرياح

القيم من عقدة الطقس هيكل JSON يمكن استخدامها بصورة مباشرة عبر تحميلها على الرسالة المتدفقة أي *payload.tempC* و *payload.humidity* و *payload.windspeed*

وبمجرد أن تولد عقدة *template* ملف HTTP تمرر الرسالة إلى العقدة الأخيرة عقدة الإخراج HTTP

وهي عقدة استجابة HTTP. هذه العقدة تحزم HTML كرد HTTP و التي ترسل إلى المتصفح .

الآن قم بالضغط على نشر *deploy* ثم توجه في المتصفح إلى الرابط الذي تم تكوينه في عقدة الإدخال HTTP

<https://{user name}.fred.sensetecnic.com/api/public/weather>

ستظهر النتيجة رسم بياني حلقي بسيط يظهر لك درجة الحرارة والرطوبة وسرعة الرياح بمجرد تمرير المؤشر عليه

