Node-RED のインストールなど

```
npm install -g --unsafe-perm node-red
```

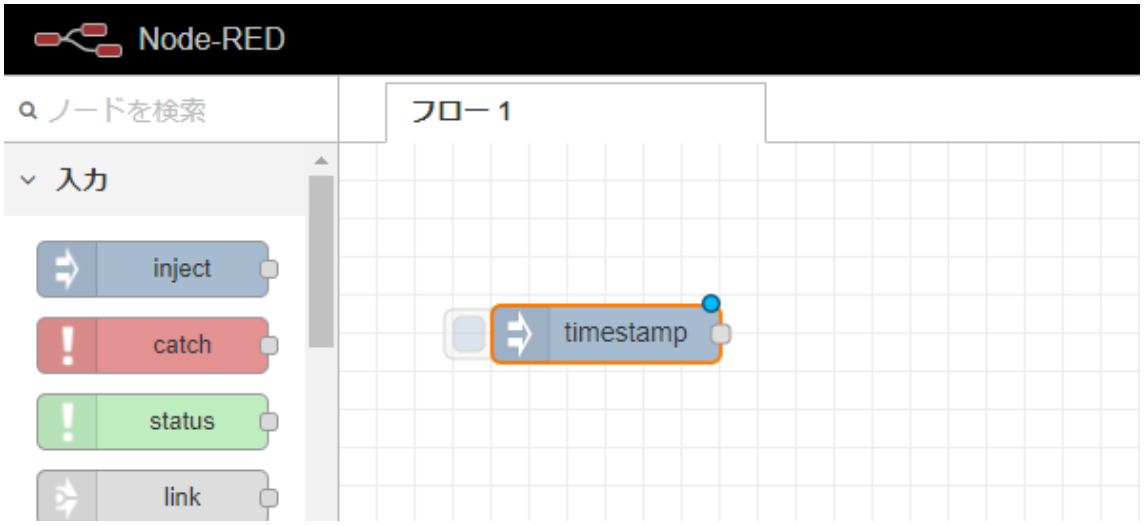
起動

```
$ node-red
```

ブラウザで　http://127.0.0.1:1880/　にアクセス

インジェクトをフローにドロップする

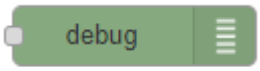
←このブロックです



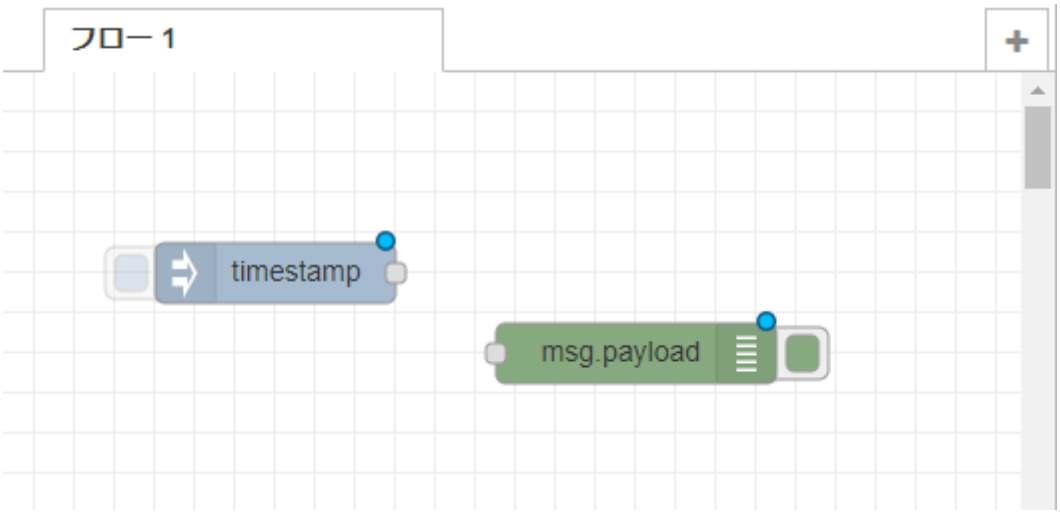
ドロップすると inject の表示は timestamp に代わる

（デフォルトで表示されるのがタイムスタンプなのでこうなる）

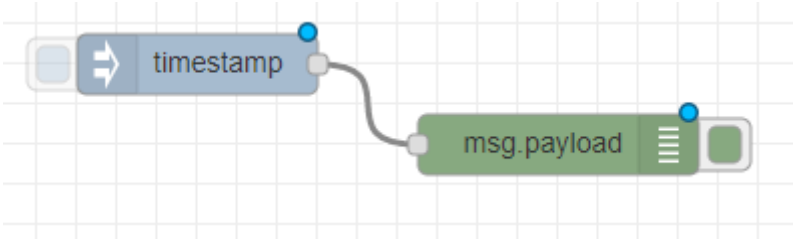
debug タグをドロップする（msg.payload に表示が変わる）



←このブロックです



□と□を順にクリックすると線でつながる



画面右上のデプロイボタンを押す

デプロイボタンの下にある虫のアイコンをクリックすると

 タグに表示されるものがデバッグに切り替わる

ここでプログラムの動作確認を表示することができるようになる

フロー上のタイムスタンプの左側のボタンを押すと
デバッグ画面上に表示される

タイムスタンプを文字列に変える

インジェクトタブをダブルクリックすると以下の画面となる

ペイロード　の　日時　の部分をクリックすると、ドロップダウンリストになっている
ここで文字列を選び、“ごきげんよう”などと入れて完了ボタンを押す

デプロイ後、インジェクトの横のボタンを押すと、出力されるメッセージが
変更される

ここまでが簡単なフローの操作説明です

いったん画面上のフローを選択して削除キーでけしておきます

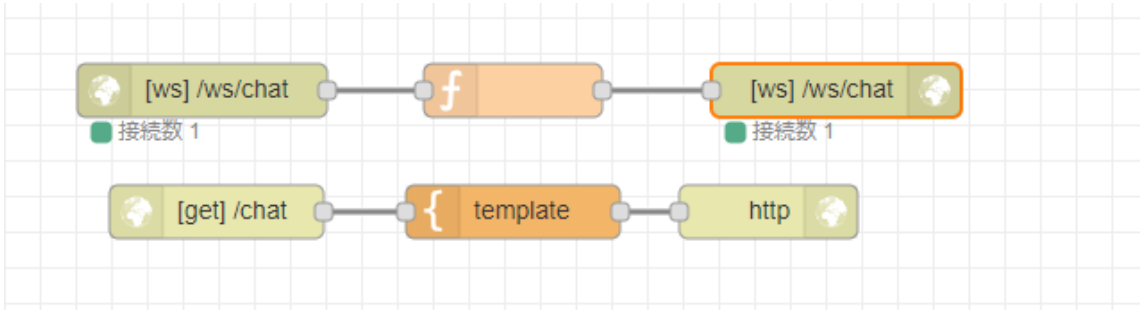
既存のフローをインポートする

一度作ったフローはテキスト（JSON）で保存することができる。

（添付した　chat.txt を開いてクリップボードにコピーしてください）
ハンバーガーメニューをクリックするとドロップダウンリストが表示される
読み込み・・・クリップボードを選び

下のボックスにコピーしておいたものを貼り付けて、「読み込み」ボタンを押します。

すると以下の用なフローが表示されます



上のほうが Websocket で
したが UI をコントロールする（HTML と JS）がテンプレートに書き込んであります

「デプロイ」ボタンを押します。

http://127.0.0.1:1880/chat
にアクセスすると　チャットＵＩページが表示されます

名前とメッセージに何か入れて send すると同じ画面を見ているメンバー全員に
Websocket 経由でメッセージが飛びます

●Node-Red の設定ファイル

C:¥Users¥ken.yuasa¥.node-red¥settings.js

109 行目あたりにある行をコピーして以下のように定義したあと node-red を再起動すると以下を起点にした WEB サーバーのようにふるまうようになります

```
    httpStatic: '/Users/ken.yuasa/node_modules/node-red/public/',
```

※httpStatic を指定すると、ここで指定した path を静的な Web コンテンツとして表示できます。
例えば/home/username/.node-red/と指定した場合、
/home/username/.node-red/index.html を作成すると/へアクセスすると作成した index.html が表示されます。

したがって httpStatic を指定する場合は httpAdminRoot は/より下層の path にする必要があります。

つまり、画像とか、ｊｓ、css なんかを配置して呼び出せるようになります
Bootstrap などはあらかじめ入っているようです

```
C:¥Users¥ken.yuasa¥node_modules¥node-red¥public>tree
├──icons
├──red
│   └──images
│       └──typedInput
└──vendor
    ├──ace
    │   └──snippets
    ├──bootstrap
    │   ├──css
    │   └──img
    ├──font-awesome
    │   ├──css
    │   └──fonts
    ├──jquery
```

```
|   └──css
|       └──smoothness
|           └──images
└──jsonata
```

なのでここにスピーチバルーン js を載せて　サーバーが何か応答するような作りを考えたいです。
例えば chatUI の上にスフィンクスの画像をおき、なぞなぞ吹き出し問答がつくれないかなと考えています

スフィンクス：「朝は四本足、昼は二本足、夕は三本足。この生き物は何か？」

参加者：おばけ？

スフィンクス：はずれ

Adding ComicBubbles to a web page
http://comicbubbles.com/tutorial

以下のものを
C:¥Users¥ken.yuasa¥node_modules¥node-red¥public>に配置して
```
|   comicbubbles.min.js
|   comicbubbles_editor.min.js
|   index.html
|   lions.jpg
```

テンプレート上の HTTP.JS から参照させるということ

メモ
node-red をネットワーク環境がない PC にインストール？するには

ユーザー直下にある C:¥Users¥ken.yuasa¥AppData¥Roaming¥npm¥node_modules
のアーカイブを作ってコピーすればよさそう

cd nodewk¥simple-bot

mkdir mongodb

mongod --dbpath ./mongodb

cd C:¥Users¥ken.yuasa¥nodewk¥simple-bot

node make-dic.js

node chat-server.js

Add ComicBubbles to the picture | Remove ComicBubbles

'speak', tailLocation: 'nw', tailX: 521, tailY: 239}
});

[Connected]

アントワン: こんにちは

アントワン | ここにメッセージをかいてね? | 送る

```
<head>
  <meta name="viewport" content="width=320, initial-scale=1">
  <title>Chat</title>
</head>

<body>
  <div id="wrapper">
    <div>
      <button id="add-button" class="show" type="button" onclick="addComicBubbles()">Add ComicBubbles to the picture</button>
      <button id="destroy-button" type="button" onclick="destroy()">Remove ComicBubbles</button>
    </div>
    <div id="moai_box" class="image">
      <img id="Moais" src="Moai.jpg">
    </div>
    <div id="Moais-comic-bubbles-output"></div>
    <div id="chat_box" class="content">
    </div>
  <br>
    <div id="footer">
      <div class="content">
        <input type="text" id="user" placeholder="お名前は?" />
        <input type="text" id="message" placeholder="ここにメッセージをかいてね?" />
        <input type="button" id="send_btn" value="送る" onclick="sendMessage()">
      </div>
    </div>
  </div>
</body>
<script src="comicbubbles.min.js"></script>
<script src="comicbubbles_editor.min.js"></script>
    <script>
      var MoaiBubbles;

      function addComicBubbles() {
        MoaiBubbles = new ComicBubbles("Moais", { canvas: { readonly: false } }, {
```

```
            bubble: [
                { id: 'b1', text: "おはようございます", x: 105, y: 45, width: 'auto', height: 'auto', fontSize: '30px', background: '#8b0000', color: '#ffffff', bubbleStyle: 'speak', tailLocation: 'nw', tailX: 521, tailY: 239, visible: false }
            ]
        });


        MoaiBubbles.onBubbleStateChange(function (bubbles) {
            for (var i = 0; i < bubbles.length; i++) {
                var bubble = bubbles[i];
                //onBubbleStateChange can help save data (bubble.x, bubble.y, bubble.text, etc.) with ajax to a database
            }
        });

        //Responsiveness
        //To adjust bubbles position: use canvas{responsive: true} or setResponsive(true)
        //To resize bubbles:
        //1. Set bubble properties: width: 'auto', height: 'auto', className: 'myClass' (setWidth, setHeight, setClassName)
        //2. Use 'myClass' in your media query to override bubble fontSize setting
        //    e.g. @media screen and (max-width: 600px) { .myClass {font-size: 17px !important;} }

        MoaiBubbles.onCanvasLoad(function () {

            this.setResponsive(true); //'this' refers to MoaiBubbles

            var Bubble1 = this.getBubbleById('b1');

            // bubbles appear and disappear with callback functions

Bubble1.setClassName('Moais').delay(1000).setWidth('auto').setHeight('auto').pumpUp(
            function () {
```

```
                    Bubble1.setText("なぞなぞを出すぞ");

                });

                document.getElementById("add-button").className = "";
                document.getElementById("destroy-button").className = "show";
            });

        }


        function destroy() {
            DestroyComicBubbles(MoaiBubbles);
            document.getElementById("add-button").className = "show";
            document.getElementById("destroy-button").className = "";
        }
    </script>

<script type="text/javascript">
  var hadr = location.host;
  var wsUri = "ws://127.0.0.1:1880/ws/chat";
  var ws = new WebSocket(wsUri);

  function createSystemMessage(message) {
    var message = document.createTextNode(message);

    var messageBox = document.createElement('p');
    messageBox.className = 'system';

    messageBox.appendChild(message);

    var chat = document.getElementById('chat_box');
    chat.appendChild(messageBox);
  }

  function createUserMessage(user, message) {
```

```
    var user = document.createTextNode(user + ': ');

    var userBox = document.createElement('span');
    userBox.className = 'username';
    userBox.appendChild(user);

    var message = document.createTextNode(message);

    var messageBox = document.createElement('p');
    messageBox.appendChild(userBox);
    messageBox.appendChild(message);

    var chat = document.getElementById('chat_box');
    chat.appendChild(messageBox);
}

ws.onopen = function (ev) {
    createSystemMessage('[Connected]');
};

ws.onclose = function (ev) {
    createSystemMessage('[Disconnected]');
}

ws.onmessage = function (ev) {
    var payload = JSON.parse(ev.data);
    createUserMessage(payload.user, payload.message);

    var chat = document.getElementById('chat_box');
    chat.scrollTop = chat.scrollHeight;
}

function sendMessage() {
    var user = document.getElementById('user');
    var message = document.getElementById('message');
```

```
    var payload = {
      message: message.value,
      user: user.value,
      ts: (new Date()).getTime()
    };

    ws.send(JSON.stringify(payload));
    message.value = "";
  };
</script>

<style type="text/css">
  * {
    font-family: "Palatino Linotype", "Book Antiqua", Palatino, serif;
    font-style: italic;
    font-size: 24px;
  }

  html,
  body,
  #wrapper {
    margin: 0;
    padding: 0;
    height: 90%;
  }

  #wrapper {
    background-color: #ecf0f1;
  }

  #moai_box {
    box-sizing: border-box;
    height: 50%;
  }

  #chat_box {
```

```
    box-sizing: border-box;
    height: 40%;
    overflow: auto;
}

#footer {
    box-sizing: border-box;
    position: fixed;
    bottom: 0;
    height: 50px;
    width: 100%;
    background-color: #2980b9;
}

#footer .content {
    padding-top: 4px;
    position: relative;
}

#user {
    width: 20%;
}

#message {
    width: 68%;
}

#send_btn {
    width: 10%;
    position: absolute;
    right: 0;
    bottom: 0;
    margin: 0;
}

.content {
```

```
    width: 70%;
    margin: 0 auto;
}

input[type="text"],
input[type="button"] {
    border: 0;
    color: #fff;
}

input[type="text"] {
    background-color: #146EA8;
    padding: 3px 10px;
}

input[type="button"] {
    background-color: #f39c12;
    border-right: 2px solid #e67e22;
    border-bottom: 2px solid #e67e22;
    min-width: 70px;
    display: inline-block;
}

input[type="button"]:hover {
    background-color: #e67e22;
    border-right: 2px solid #f39c12;
    border-bottom: 2px solid #f39c12;
    cursor: pointer;
}

.system,
.username {
    color: #aaa;
    font-style: italic;
    font-family: monospace;
    font-size: 16px;
```

```
    }

</style>
```