[3일차 - 강의 예제]

● 실습파일 제출 요령
1) 실습파일들은 1개 파일로 압축해서 첨부 (파일명은 [이름].zip)
2) 수업공유 드라이브에 제출
   - 날짜별 폴더가 있음

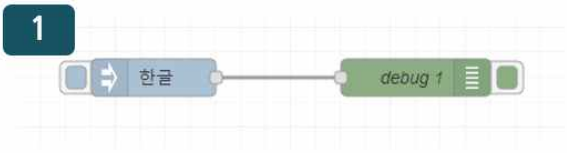
● Node JS, Node-Red 설치
1. Node JS 설치
   https://nodejs.org/ko/ 에서 설치파일 다운로드 후 설치
   . 설치 확인 : 커맨드 창에서
    node --version && npm --version

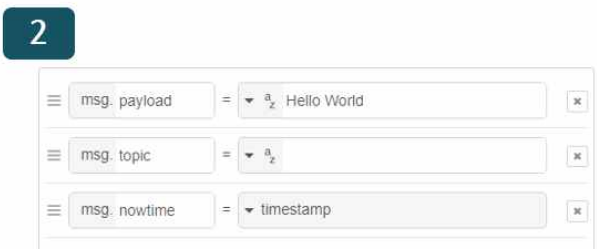
2. Node-Red 설치
   (커맨드 창에서)
   npm install -g --unsafe-perm node-red

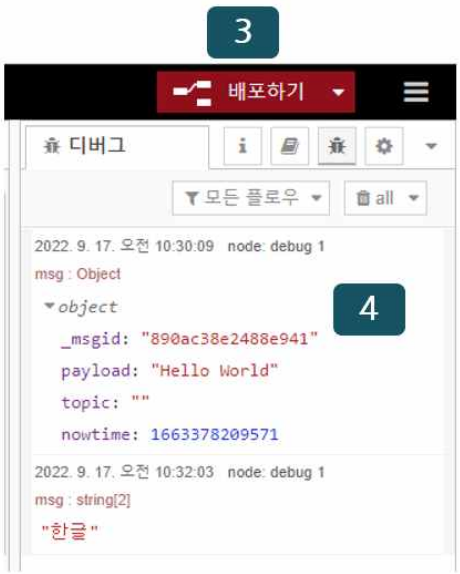
3. Node-Red 실행
   (커맨드 창에서)
   node-red

4. 브라우저에서 접속
   http://127.0.0.1:1880

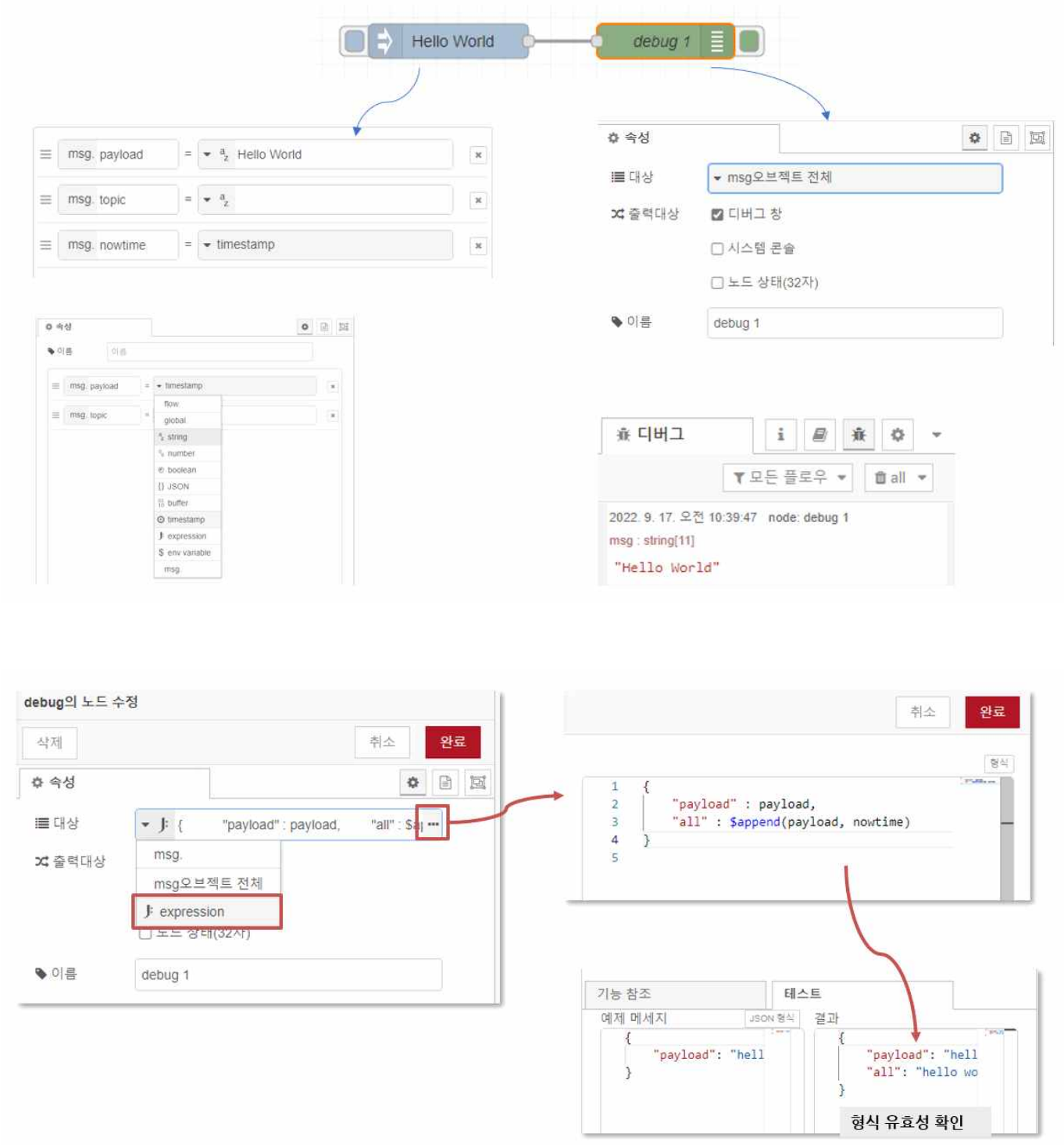
1. 노드 배치 + 노드 연결
2. 노드 속성 편집
3. '배포하기' 클릭
4. 디버그 창에서 확인
*** API 확인 시에는 브라우저에서 확인**



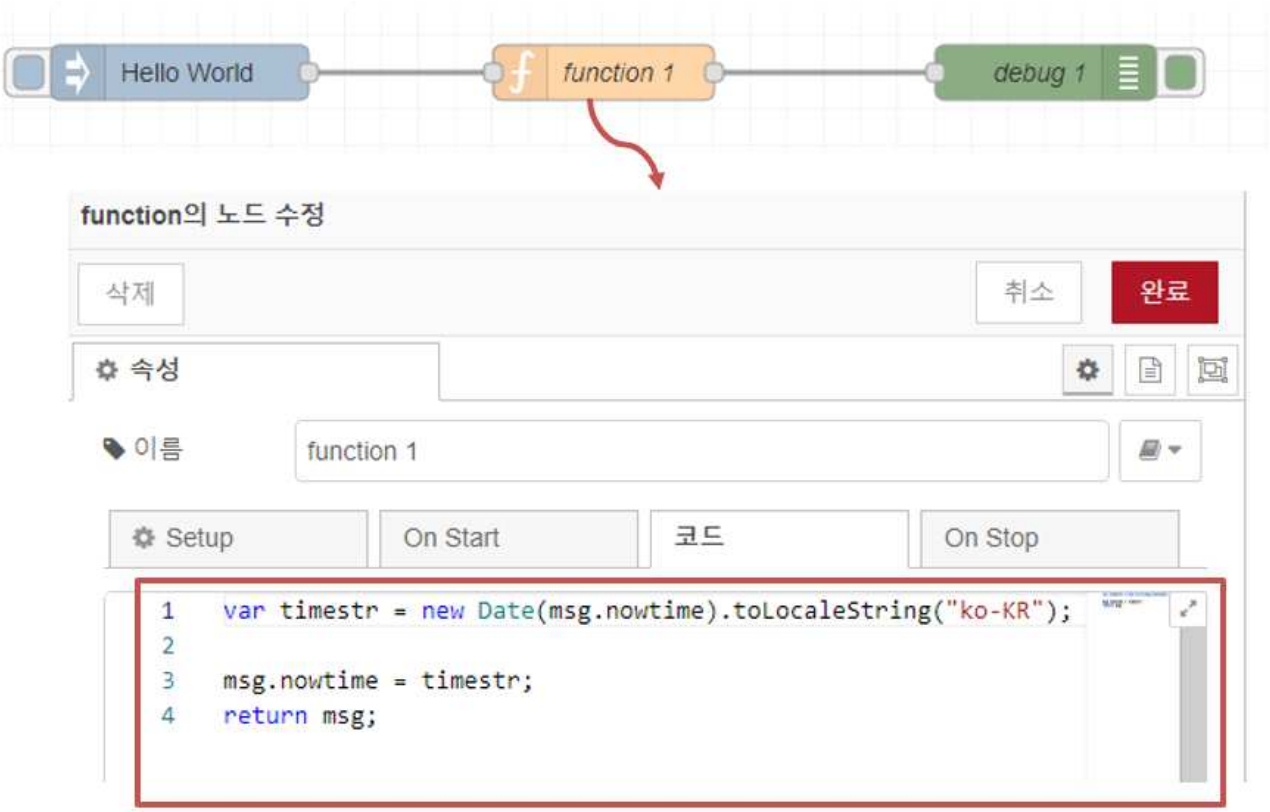


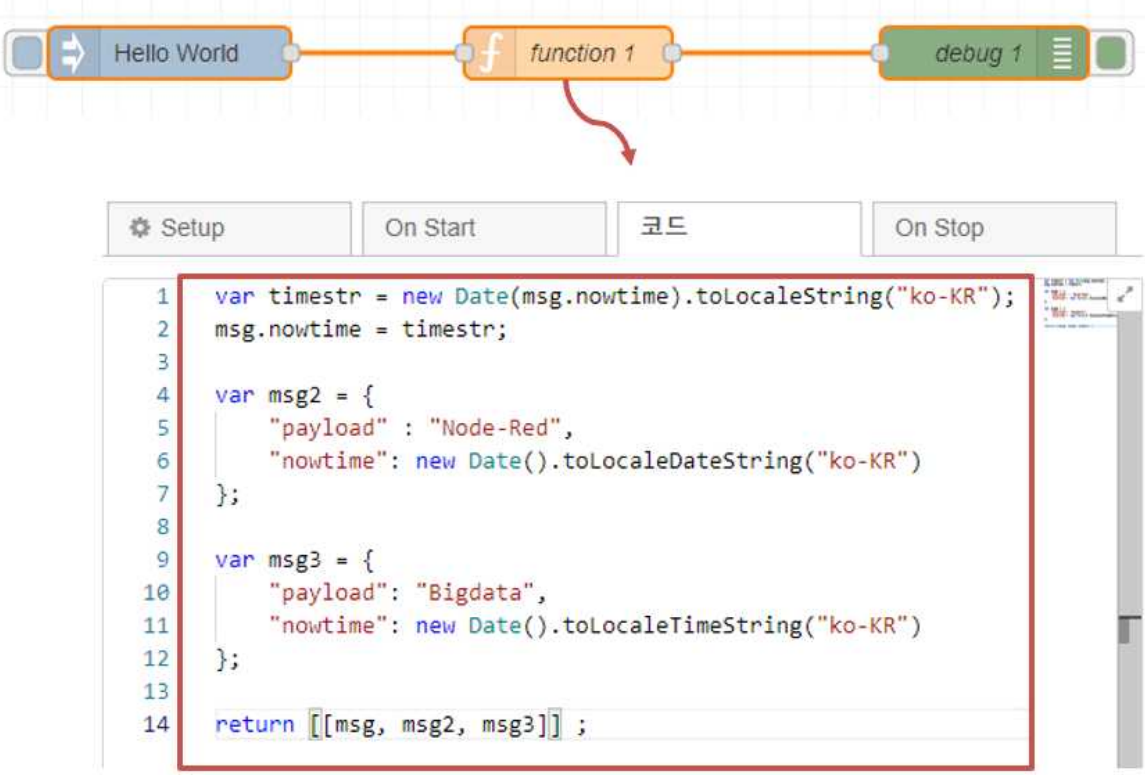


ex1 //

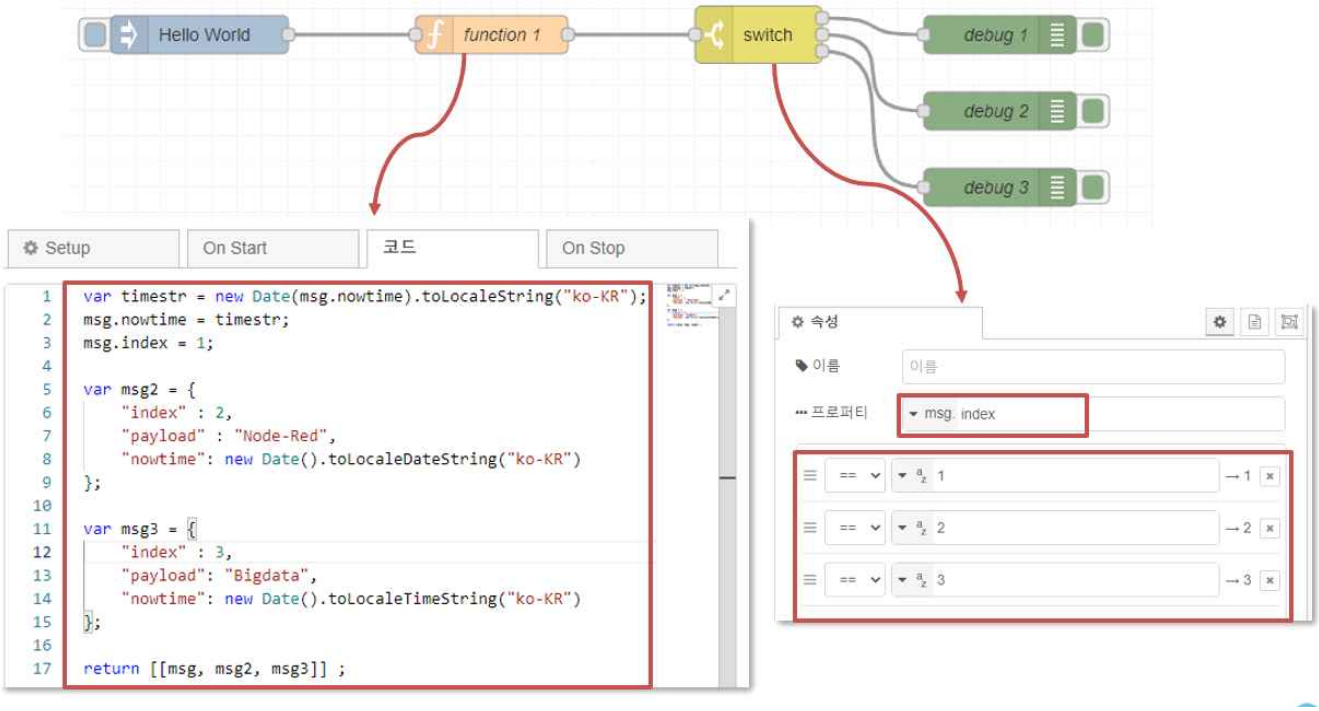

Hello World
debug 1
msg. payload = Hello World
msg. topic =
msg. nowtime = timestamp
속성
대상 msg오브젝트 전체
출력대상 디버그 창
시스템 콘솔
노드 상태(32자)
이름 debug 1
디버그
모든 플로우
all
2022. 9. 17. 오전 10:39:47 node: debug 1
msg : string[11]
"Hello World"
debug의 노드 수정
삭제
취소
완료
속성
대상 { "payload" : payload, "all" : $a
출력대상
msg.
msg오브젝트 전체
expression
이름 debug 1
취소
완료
{
"payload" : payload,
"all" : $append(payload, nowtime)
}
기능 참조
테스트
예제 메세지
JSON 형식
결과
{
"payload": "hell
}
{
"payload": "hell
"all": "hello wo
}
형식 유효성 확인

ex2 //

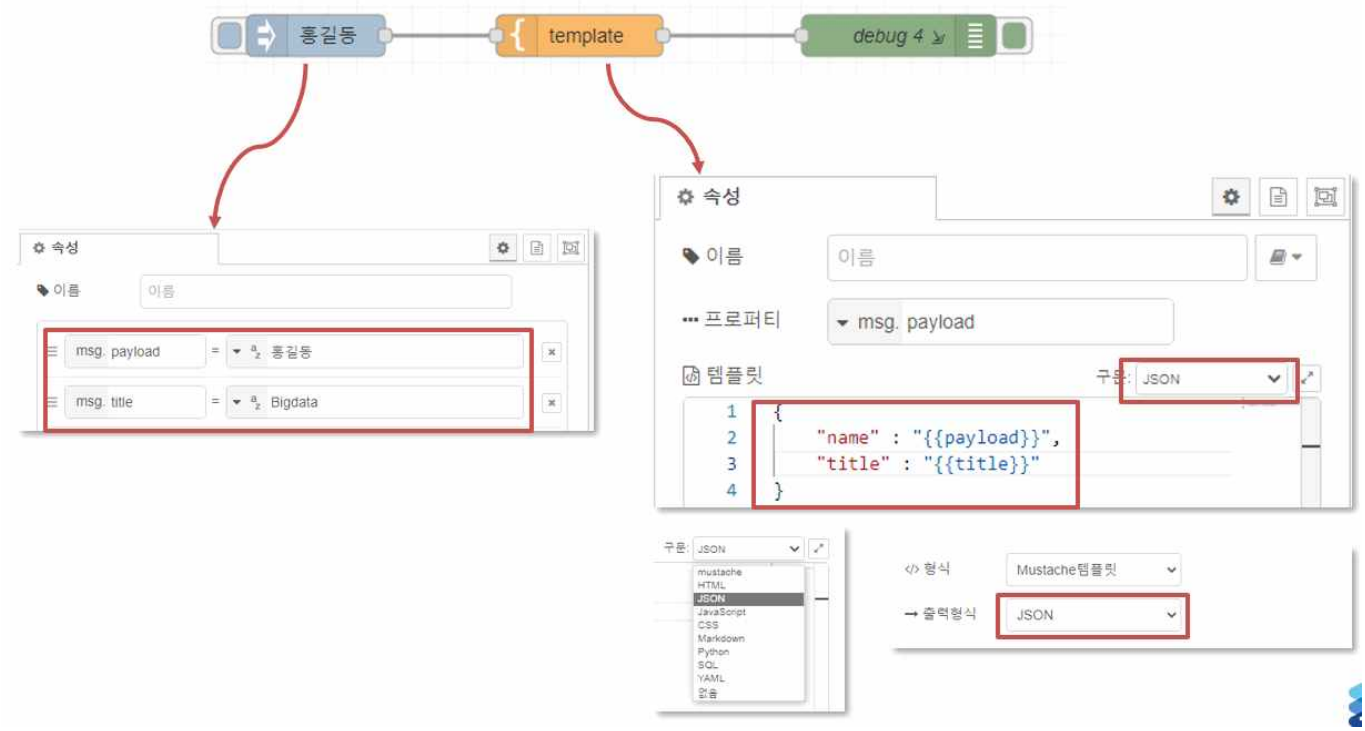

Hello World
function 1
debug 1
function의 노드 수정
삭제
취소
완료
속성
이름
function 1
Setup
On Start
코드
On Stop
1 var timestr = new Date(msg.nowtime).toLocaleString("ko-KR");
2
3 msg.nowtime = timestr;
4 return msg;


Hello World
function 1
debug 1
Setup
On Start
코드
On Stop
1 var timestr = new Date(msg.nowtime).toLocaleString("ko-KR");
2 msg.nowtime = timestr;
3
4 var msg2 = {
5 "payload" : "Node-Red",
6 "nowtime": new Date().toLocaleDateString("ko-KR")
7 };
8
9 var msg3 = {
10 "payload": "Bigdata",
11 "nowtime": new Date().toLocaleTimeString("ko-KR")
12 };
13
14 return [[msg, msg2, msg3]] ;

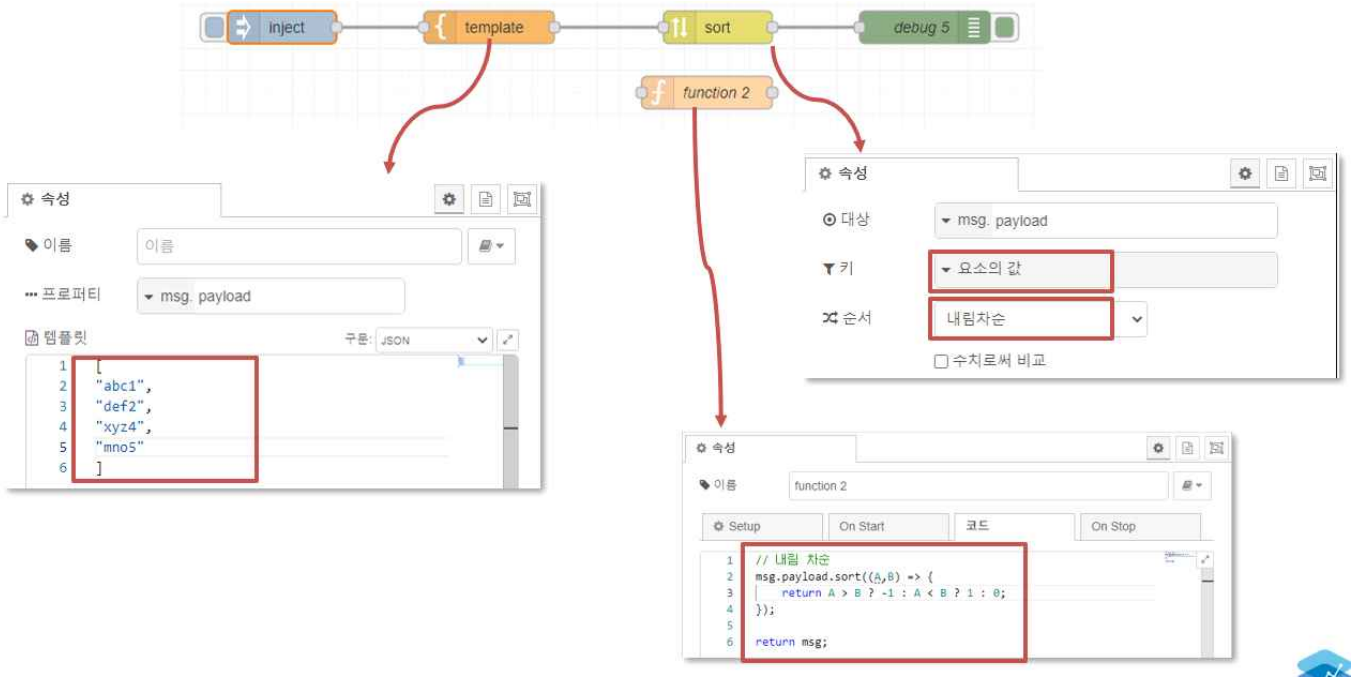
ex3.//


Hello World
function 1
switch
debug 1
debug 2
debug 3
Setup
On Start
코드
On Stop
var timestr = new Date(msg.nowtime).toLocaleString("ko-KR");
msg.nowtime = timestr;
msg.index = 1;
var msg2 = {
"index" : 2,
"payload" : "Node-Red",
"nowtime": new Date().toLocaleDateString("ko-KR")
};
var msg3 = {
"index" : 3,
"payload": "Bigdata",
"nowtime": new Date().toLocaleTimeString("ko-KR")
};
return [[msg, msg2, msg3]] ;
속성
이름
프로퍼티
msg. index
== 1 → 1
== 2 → 2
== 3 → 3

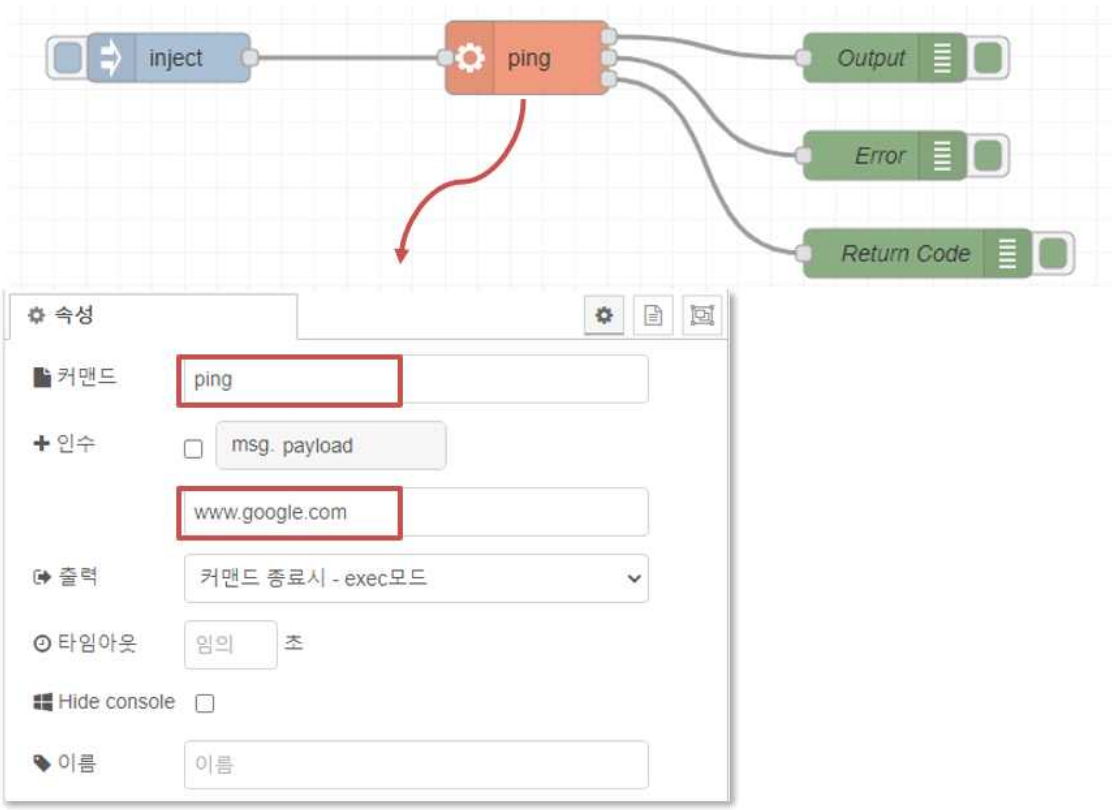
ex4.//


홍길동
template
debug 4
속성
이름
msg. payload = 홍길동
msg. title = Bigdata
속성
이름
이름
프로퍼티
msg. payload
템플릿
구문: JSON
{
"name" : "{{payload}}",
"title" : "{{title}}"
}
구문: JSON
mustache
HTML
JSON
JavaScript
CSS
Markdown
Python
SQL
YAML
없음
형식
Mustache템플릿
출력형식
JSON

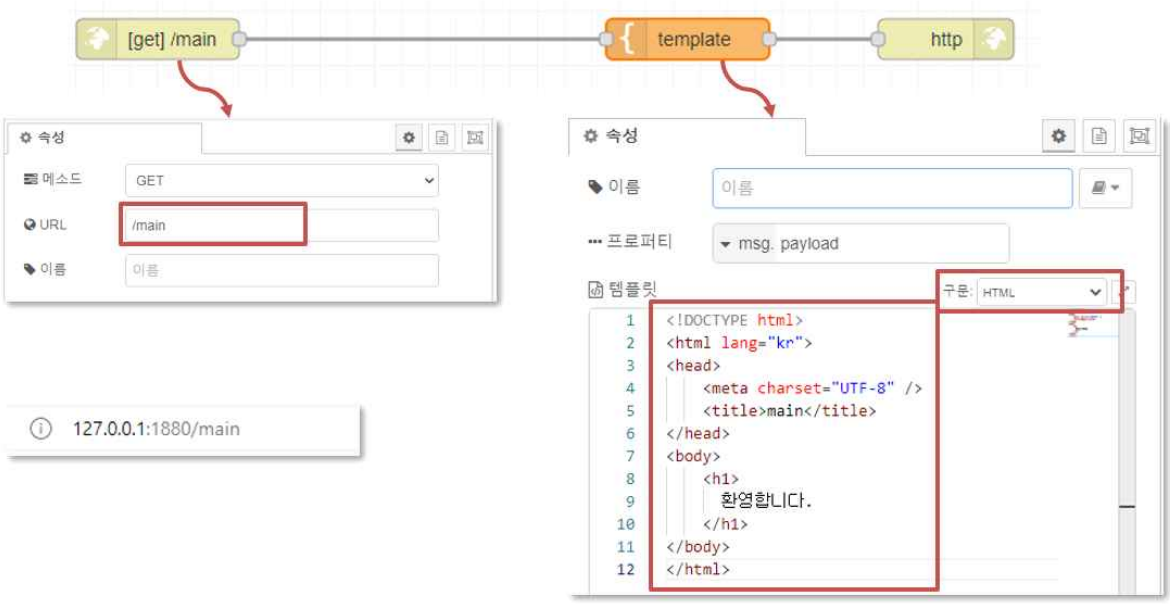
ex5 //

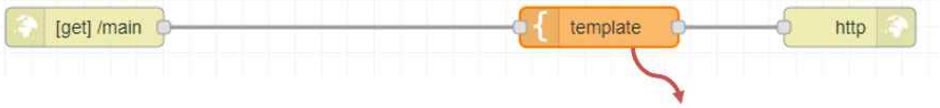

inject
template
sort
debug 5
function 2
속성
이름
프로퍼티
msg. payload
템플릿
구문: JSON
[
"abc1",
"def2",
"xyz4",
"mno5"
]
속성
대상
msg. payload
키
요소의 값
순서
내림차순
수치로써 비교
속성
이름
function 2
Setup
On Start
코드
On Stop
// 내림 차순
msg.payload.sort((A,B) => {
return A > B ? -1 : A < B ? 1 : 0;
});
return msg;

ex6 //


inject
ping
Output
Error
Return Code
속성
커맨드
ping
인수
msg. payload
www.google.com
출력
커맨드 종료시 - exec모드
타임아웃
임의
초
Hide console
이름
이름

ex7 //



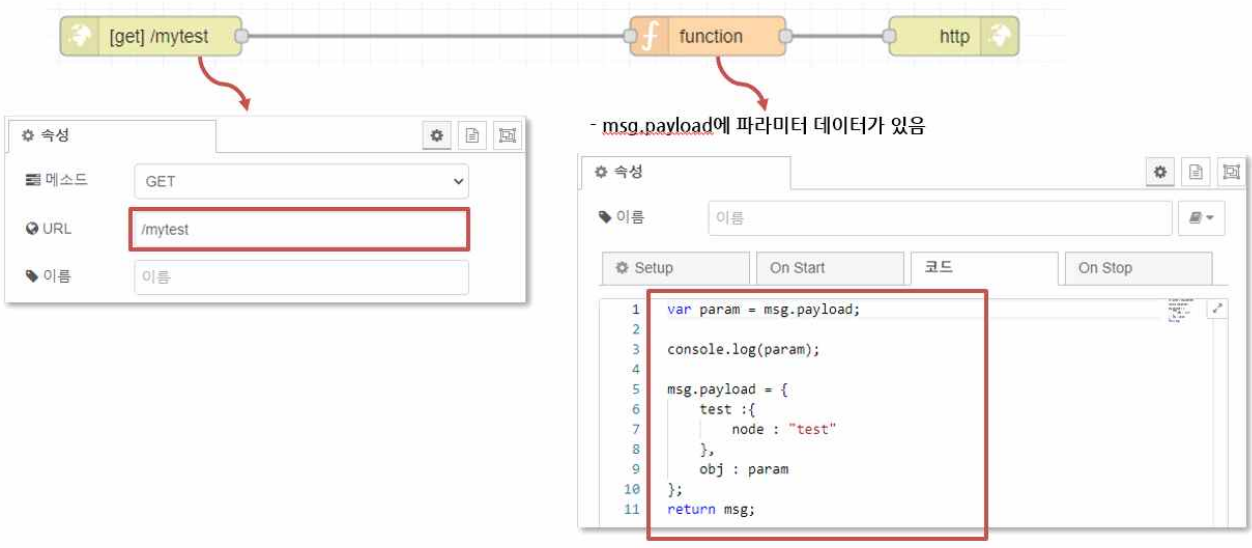


honggildong 님 환영합니다.

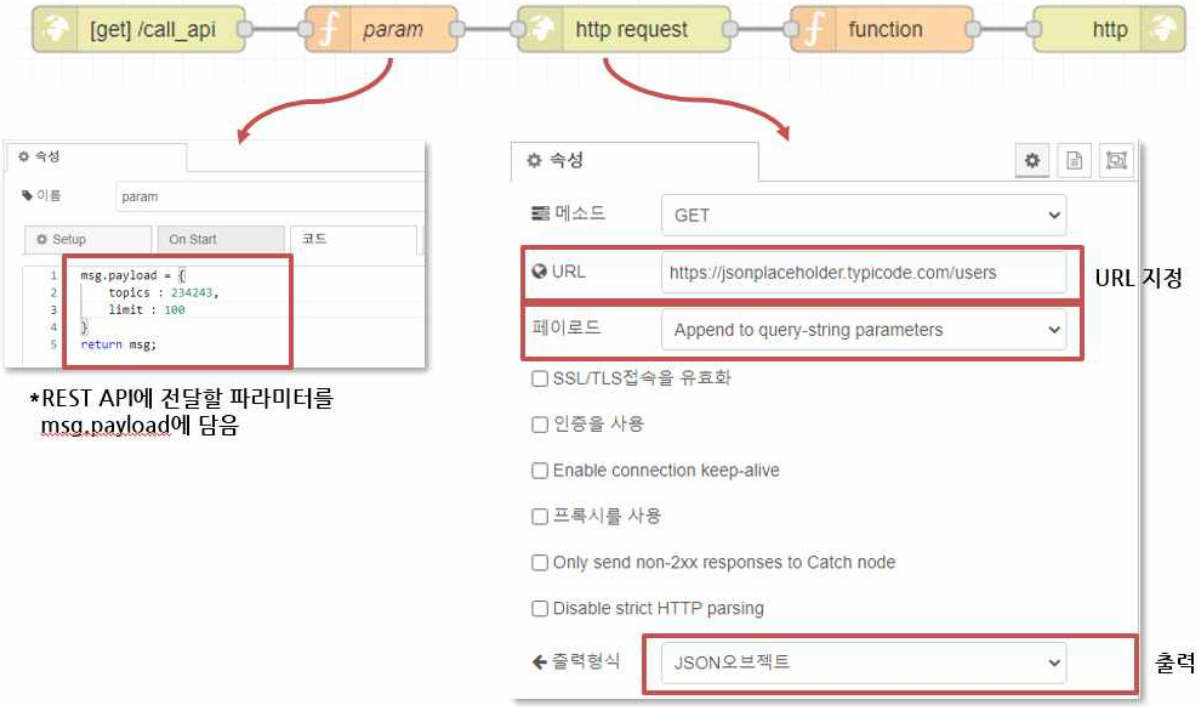
- {{payload.name}} 식으로 파라미터를 받을 수 있음

```
<!DOCTYPE html>
<html lang="kr">
<head>
   <meta charset="UTF-8" />
   <title>main</title>
</head>
<body>
   <h1>
     {{payload.name}} 님 환영합니다.
   </h1>
</body>
</html>
```

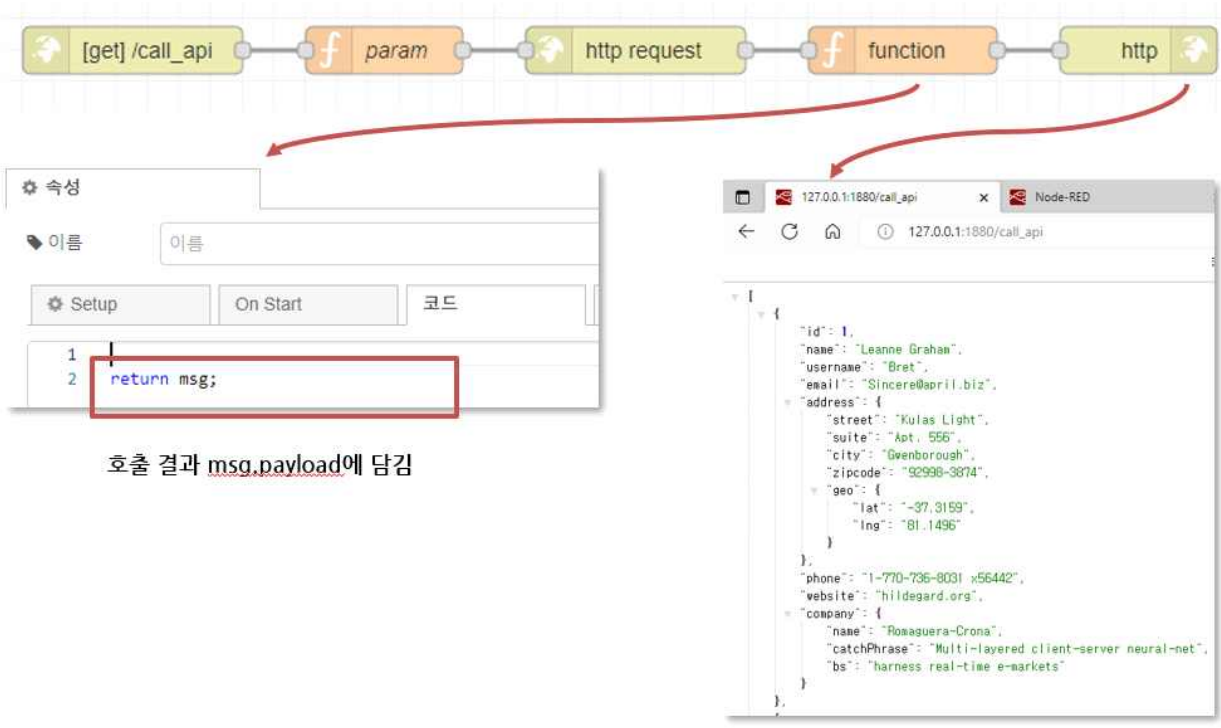
ex8 //



ex9 //

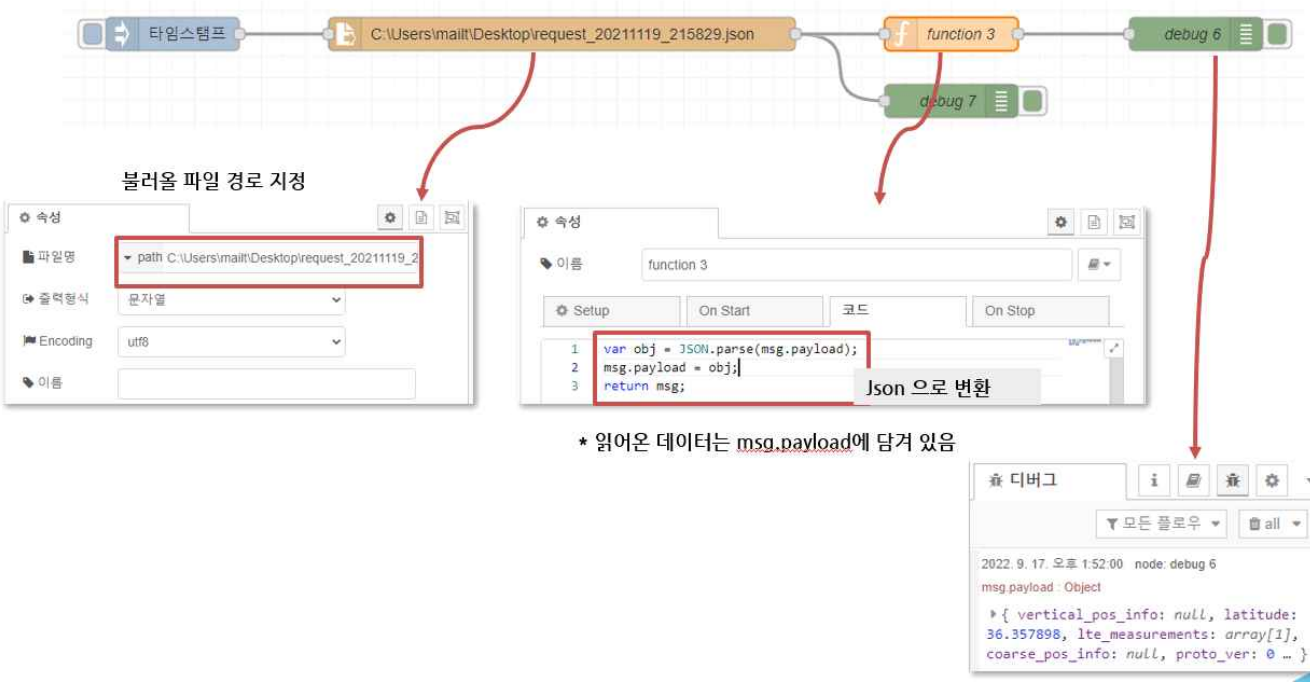


*REST API에 전달할 파라미터를 msg.payload에 담음



호출 결과 msg.payload에 담김

ex10 //


타임스탬프
C:\Users\mailt\Desktop\request_20211119_215829.json
function 3
debug 6
debug 7
불러올 파일 경로 지정
속성
파일명
path C:\Users\mailt\Desktop\request_20211119_2
출력형식
문자열
Encoding
utf8
이름
속성
이름
function 3
Setup
On Start
코드
On Stop
1 var obj = JSON.parse(msg.payload);
2 msg.payload = obj;
3 return msg;
Json 으로 변환
* 읽어온 데이터는 msg.payload에 담겨 있음
디버그
모든 플로우
all
2022. 9. 17. 오후 1:52:00 node: debug 6
msg.payload : Object
{ vertical_pos_info: null, latitude: 36.357898, lte_measurements: array[1], coarse_pos_info: null, proto_ver: 0 … }

[3일차 최종 예제]

로그인 웹페이지를 만들고 정보를 확인하여 로그인 결과를 알려주는 프로그램 작성

1. Node-Red로 로그인 결과를 확인해 주는 API를 만드시오
   - 파라미터로 id/pw를 입력 받고 이것을 조건으로 유저를 확인하는 쿼리 작성
     user : "bigdatauser"
     password : "!bigdata"
   - API 이름은 login, 타입은 'post'방식
   - 유효한 사용자가 있고 비번이 맞으면 { result : "OK"} 로 응답
     없거나 비번이 틀리면 { result : "NO"}로 응답
2. 로그인 페이지 작성 (Node-Red 템플릿 노드 활용)
   - id 입력, pw 입력, '로그인' 버튼 작성
   - id/pw가 비어있지 않은지 체크 하는 로직 필요
3. ajax를 통해 2번에서 작성한 API와 연결하여 로그인 결과를 받고 성공여부를 메시지 박스(alert)로 출력

- AJAX로 로그인 여부 확인후
  main 페이지로 이동

- 로그인 페이지 제공 (loginform)
- 로그인 확인 API (login)
  : 로그인 확인 결과를 http로 실어 보냄
- 로그인 성공 후 환영 페이지 제공 (main)