

物聯網實務第五周作業

電機四乙10828241 陳大荃

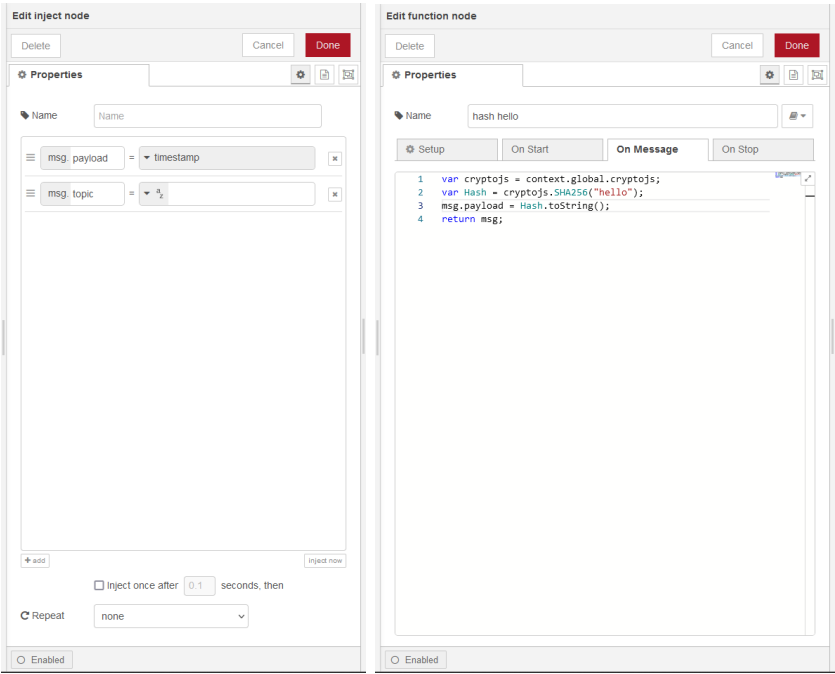
October 12, 2022

Exercise 5-1 Hash "hello".



(a) Node-RED Flow

(b) debug

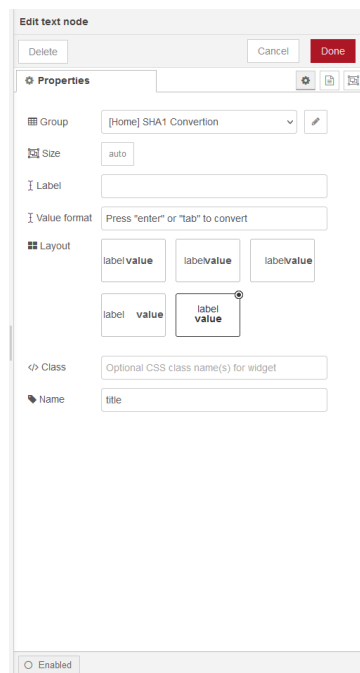
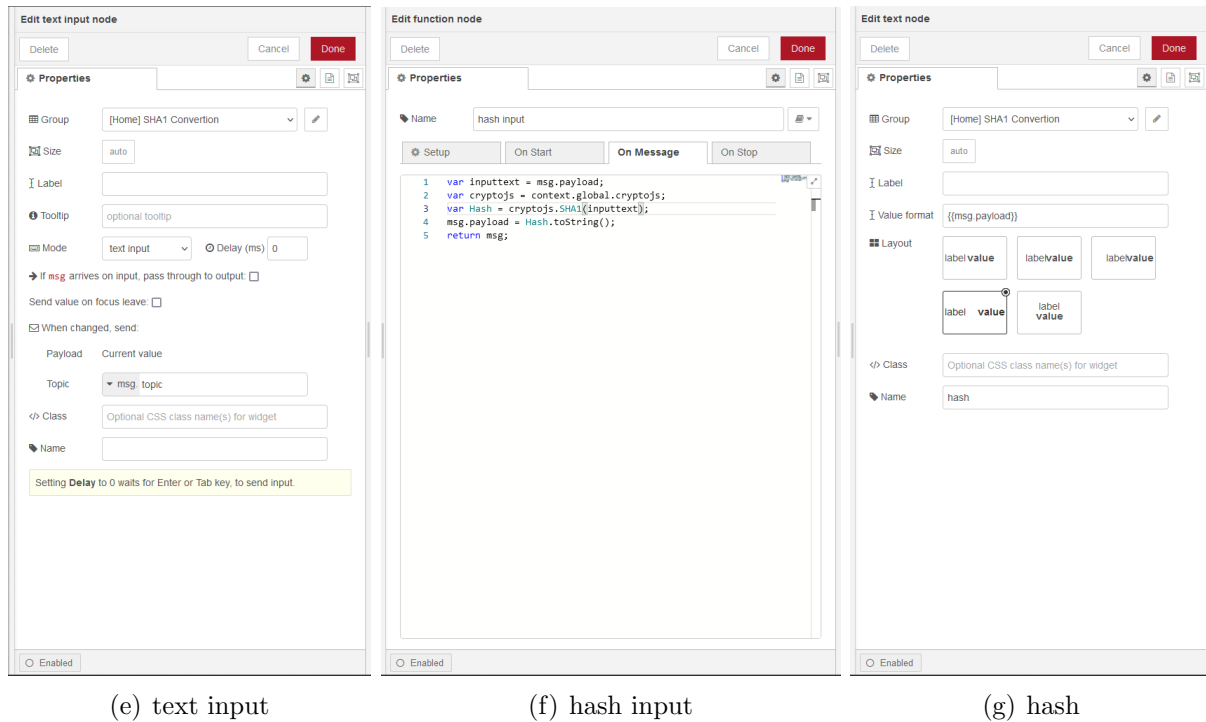
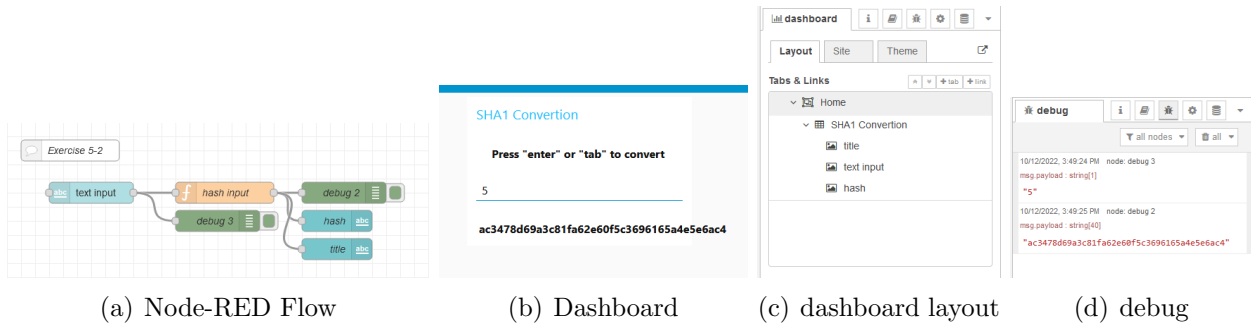


(c) timestamp

(d) hash hello

Figure 1: Exercise 5-1.

Exercise 5-2 Hash input in dashboard.



(h) title

Figure 2: Exercise 5-2.

Exercise 5-3 Create genesis block.

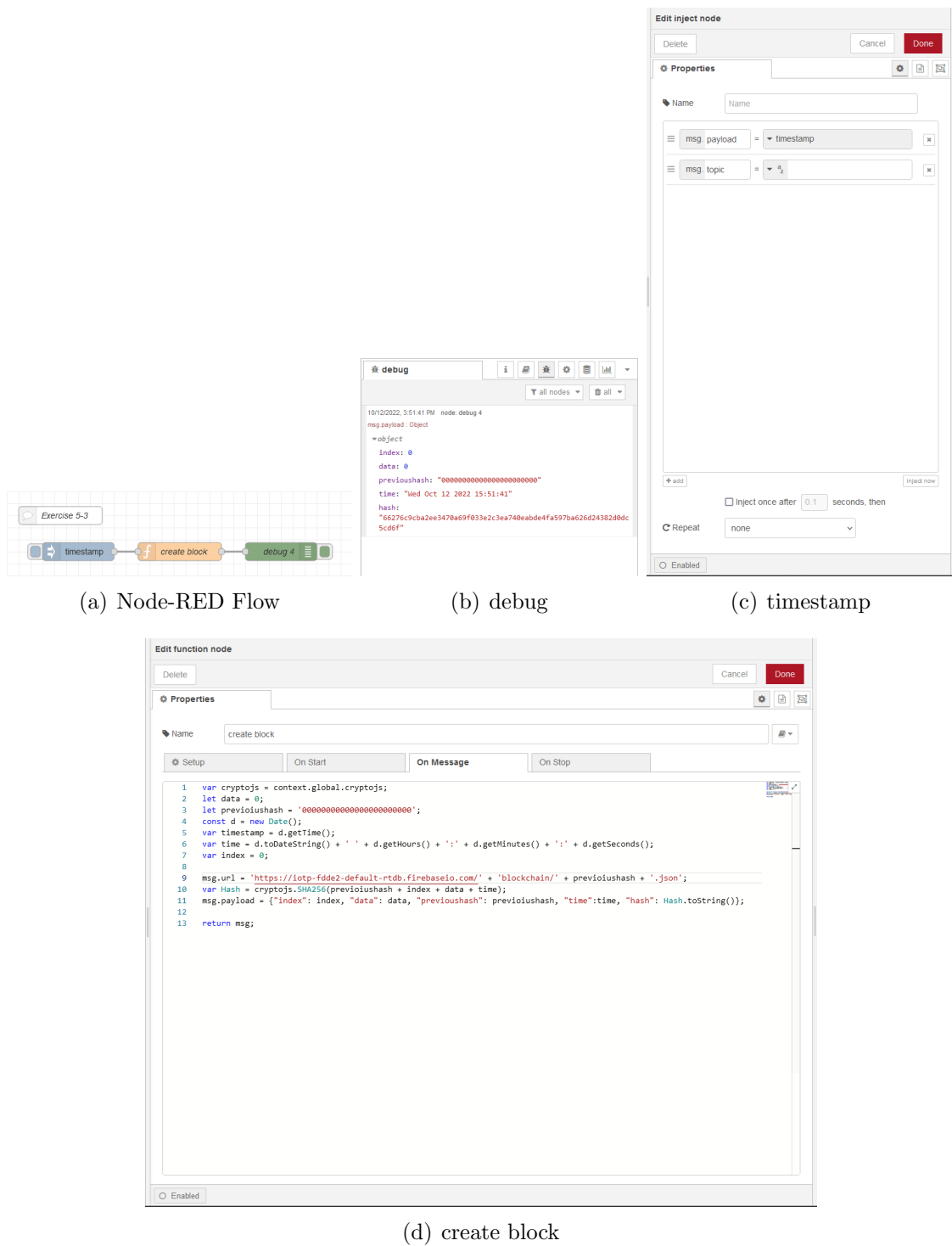
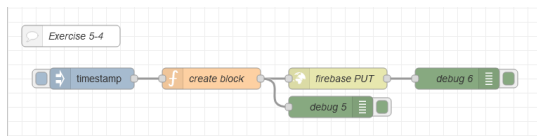


Figure 3: Exercise 5-3.

Exercise 5-4 Create and store genesis block.



(a) Node-RED Flow

```

https://iotp-fdde2-default-rtdb.firebaseio.com/
└─ blockchain
   └─ 00000000000000000000000000000000
      └─ data: 0
         hash: "aa610afe53a47d877c6c06362930bb60cdf0dc1000275aa9e84d0c6ec908e074"
         index: 0
         previoushash: "00000000000000000000000000000000"
         time: "Wed Oct 12 2022 15:53:46"
  
```

(b) Firebase

```

10/12/2022, 3:53:46 PM node: debug 5
msg.payload: Object
  > object
    index: 0
    data: 0
    previoushash: "00000000000000000000000000000000"
    time: "Wed Oct 12 2022 15:53:46"
    hash: "aa610afe53a47d877c6c06362930bb60cdf0dc1000275aa9e84d0c6ec908e074"

10/12/2022, 3:53:46 PM node: debug 6
msg.payload: string[169]
"["data":0,"hash":"aa610afe53a47d877c6c06362930bb60cdf0dc1000275aa9e84d0c6ec908e074","index":0,"previoushash":"00000000000000000000000000000000","time":"Wed Oct 12 2022 15:53:46"]"
  
```

(c) debug

Inject node configuration:

- Name:
- msg.payload:
- msg.topic:
- Inject once after: seconds, then
- Repeat:
- Enabled: ☐

(d) timestamp

Function node configuration for 'create block':

```

1 var cryptojs = context.global.cryptojs;
2 let data = 0;
3 let previoushash = '00000000000000000000000000000000';
4 const d = new Date();
5 var timestamp = d.getTime();
6 var time = d.toString() + ' ' + d.getHours() + ':' + d.getMinutes() + ':' + d.getSeconds();
7 var index = 0;
8
9 msg.url = 'https://iotp-fdde2-default-rtdb.firebaseio.com/' + 'blockchain/' + previoushash + '.json';
10 var Hash = cryptojs.SHA256(previoushash + index + data + time);
11 msg.payload = {"index": index, "data": data, "previoushash": previoushash, "time": time, "hash": Hash.toString()};
12
13 return msg;
  
```

(e) create block

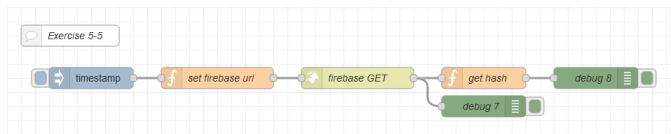
HTTP request node configuration for 'firebase PUT':

- Method:
- URL:
- Enable secure (SSL/TLS) connection: ☐
- Use authentication: ☐
- Enable connection keep-alive: ☐
- Use proxy: ☐
- Only send non-2xx responses to Catch node: ☐
- Disable strict HTTP parsing: ☐
- Return:
- Headers:
- Name:
- Enabled: ☐

(f) firebase PUT

Figure 4: Exercise 5-4.

Exercise 5-5 Get genesis block's hash.



(a) Node-RED Flow

```

10/12/2022, 3:56:58 PM node: debug 7
msg.payload: string[197]
{"data":0,"hash":"aa618afe53a47d877c6c86362938b68cdf8dc1000275aa9e84d8c6ec980e074","index":0,"previoushash":"00000000000000000000000000000000","time":"Wed Oct 12 2022 15:53:46"}

10/12/2022, 3:56:58 PM node: debug 8
msg.payload: string[64]
"aa618afe53a47d877c6c86362938b68cdf8dc1000275aa9e84d8c6ec980e074"
  
```

(b) debug

(c) timestamp

(d) set firebase url

(e) firebase GET

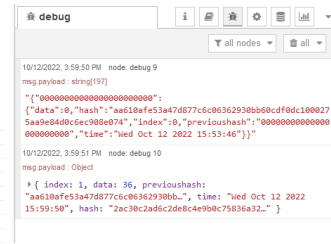
(f) get hash

Figure 5: Exercise 5-5.

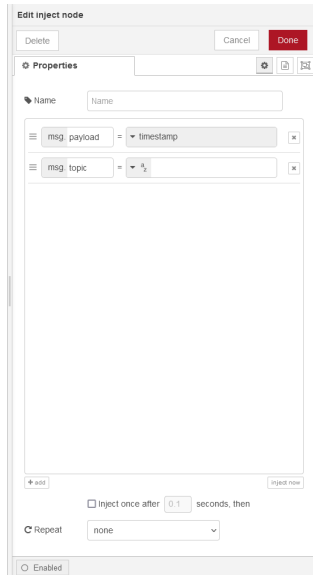
Exercise 5-6 Create the following blocks.



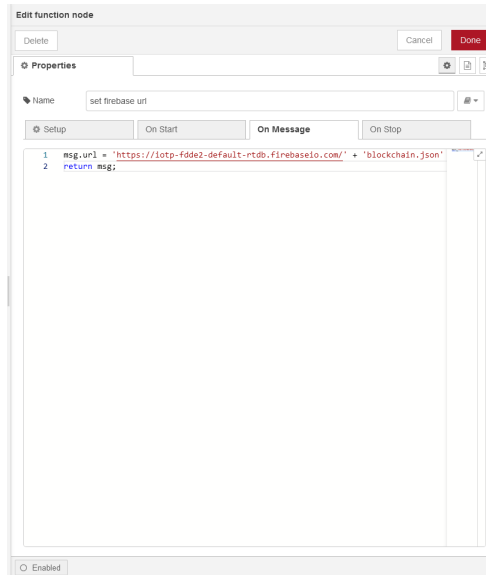
(a) Node-RED Flow



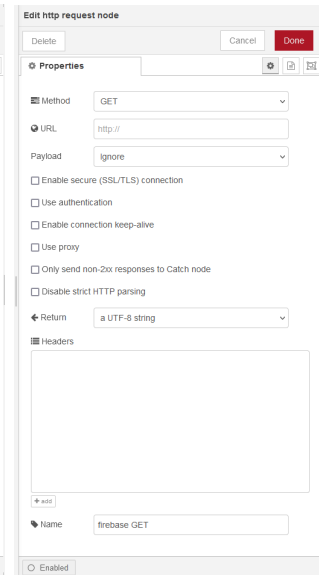
(b) debug



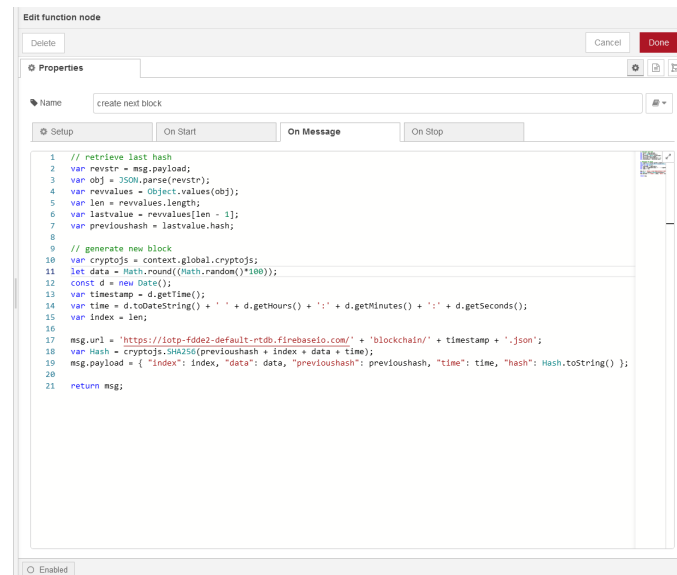
(c) timestamp



(d) set firebase url



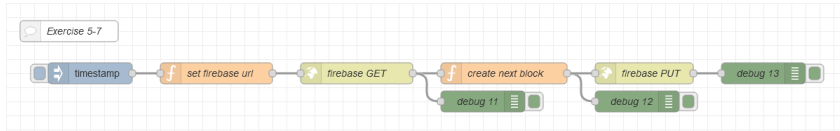
(e) firebase GET



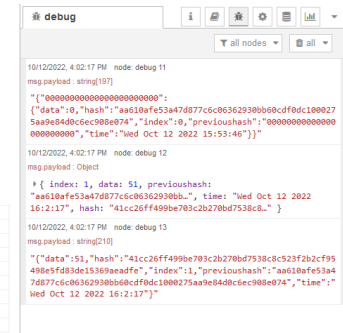
(f) create next block

Figure 6: Exercise 5-6.

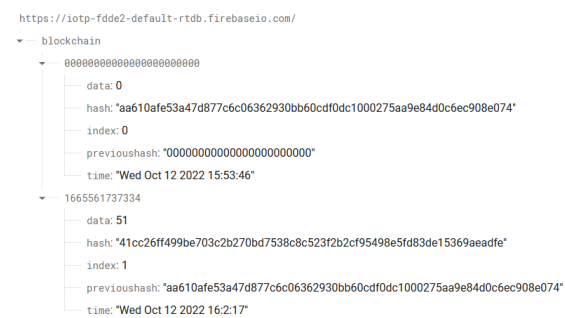
Exercise 5-7 Create and store the following blocks.



(a) Node-RED Flow



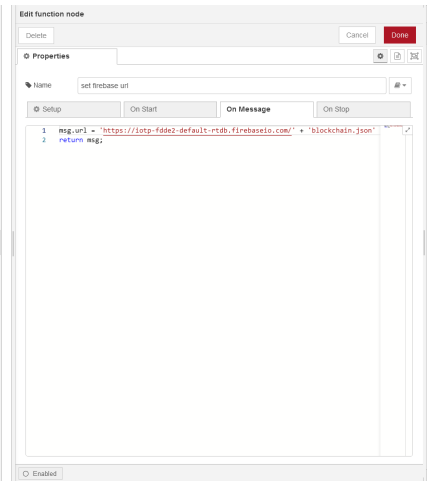
(b) debug



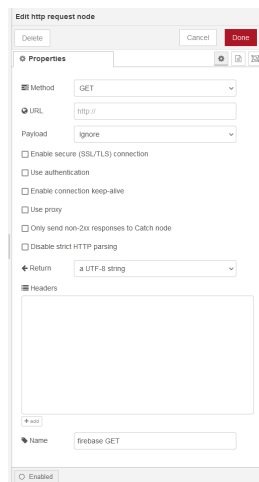
(c) Firebase



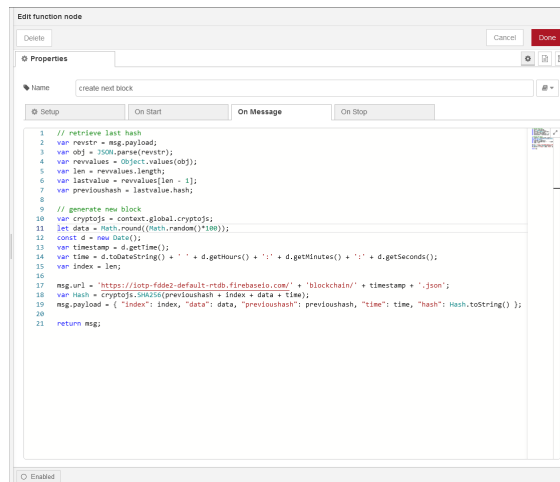
(d) timestamp



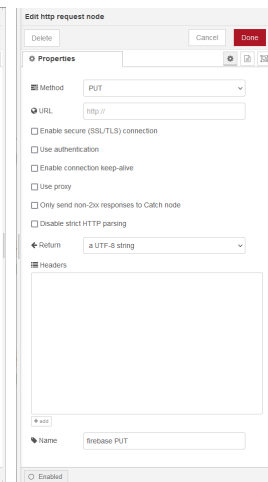
(e) set firebase url



(f) firebase GET



(g) create next block



(h) firebasePUT

Figure 7: Exercise 5-7.