

```
git log --pretty=format:"%h%x09%an%x09%ad%x09%s"
860f8a3 mentally24      Wed Apr 27 15:19:49 2022 +0800    W01~W09
f2ef347 mentally24      Tue Apr 26 20:05:08 2022 +0800    W01~W08
4640076 mentally24      Wed Apr 20 21:18:37 2022 +0800    midprep p3_answer
e777c1c mentally24      Wed Apr 20 19:34:28 2022 +0800    middrep p2_answer
3fa2e10 mentally24      Wed Apr 20 19:09:48 2022 +0800    middprep p1_91 answer
feb284d mentally24      Wed Apr 20 18:54:01 2022 +0800    w09 midprep starter
ffb9acf mentally24      Thu Apr 14 16:17:45 2022 +0800    p1 :getuser() changebtn()
for week6
14fc5a8 mentally24      Thu Apr 14 16:03:26 2022 +0800    p1 :getuser() changebtn()
for week6
be3a50e mentally24      Thu Apr 14 16:00:07 2022 +0800    p1 :getuser() changebtn()
for week6
74e6145 mentally24      Thu Apr 14 15:40:30 2022 +0800    Initial commit
```

w12 p1

The screenshot displays a development environment with a code editor on the left and a browser window on the right. The code editor shows a JavaScript file named `boilingWater.js` with the following content:

```
// Make Soup
// boil water 10 min
// chop carrots
// add carrots boil for 1 min
// chop onion
// add onion boil for 1 min
// BROWSER!!!!!! Fetch Data, Get Geolocation, setTimeo

boilingWater();
console.log('chop carrots');

function boilingWater(time){
  console.log('boiling...');

  setTimeout(()=>{
    console.log('boiling done');
    console.log('add carrots');
    setTimeout(()=>{
      console.log('carrots boiling done');
      console.log('add onion');
    },2000);

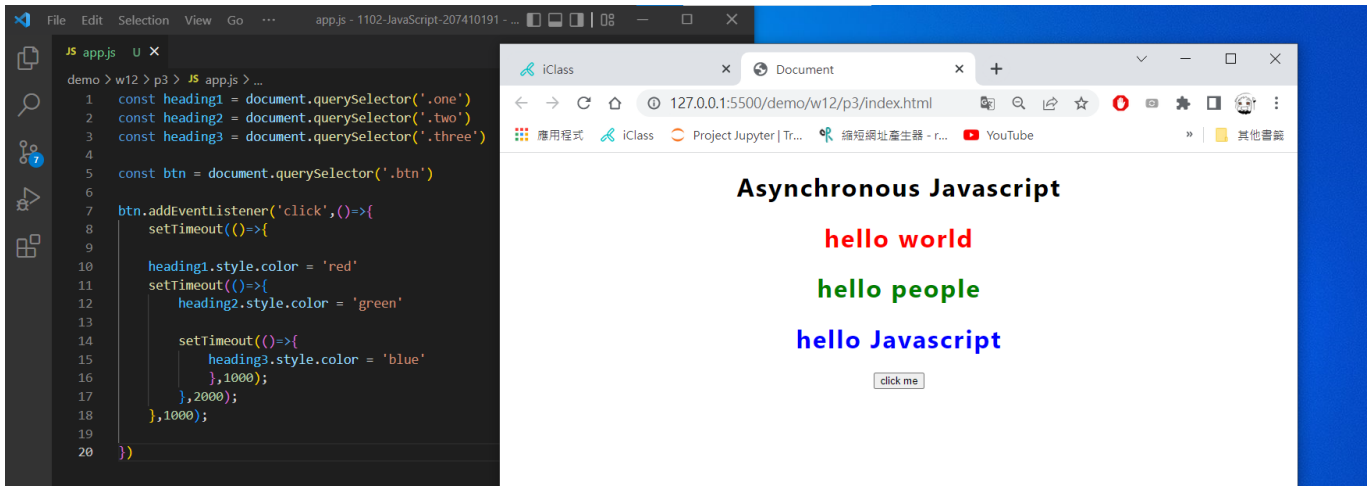
    setTimeout(()=>{
      console.log('onion boiling done');
    },2000);
  },3000);
}
```

The browser window shows the page `127.0.0.1:5500/index.html` with the title `Asynchronous Javascript`. The DevTools console is open, showing the following log messages:

- boiling... (app.js:31)
- chop carrots (app.js:28)
- boiling done (app.js:34)
- add carrots (app.js:35)
- carrots boiling done (app.js:37)
- add onion (app.js:38)
- onion boiling done (app.js:42)

```
git log --pretty=format:"%h%x09%an%x09%ad%x09%s" --after="2022-0510"
3a854a7 mentally24      Wed May 11 19:14:09 2022 +0800    W12-P1: Making soup demo
for Async JavaScript
```

w12 p2

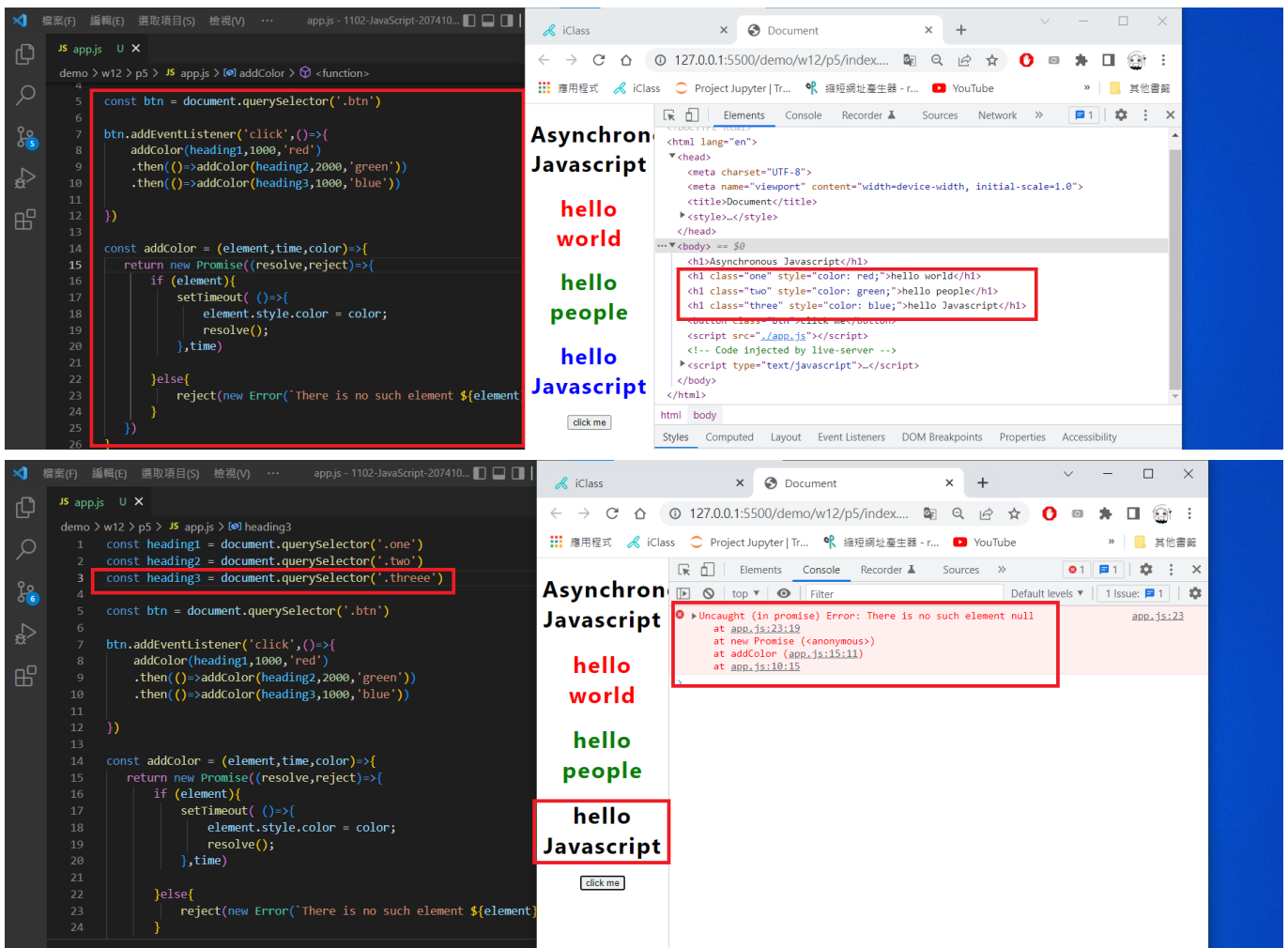


```

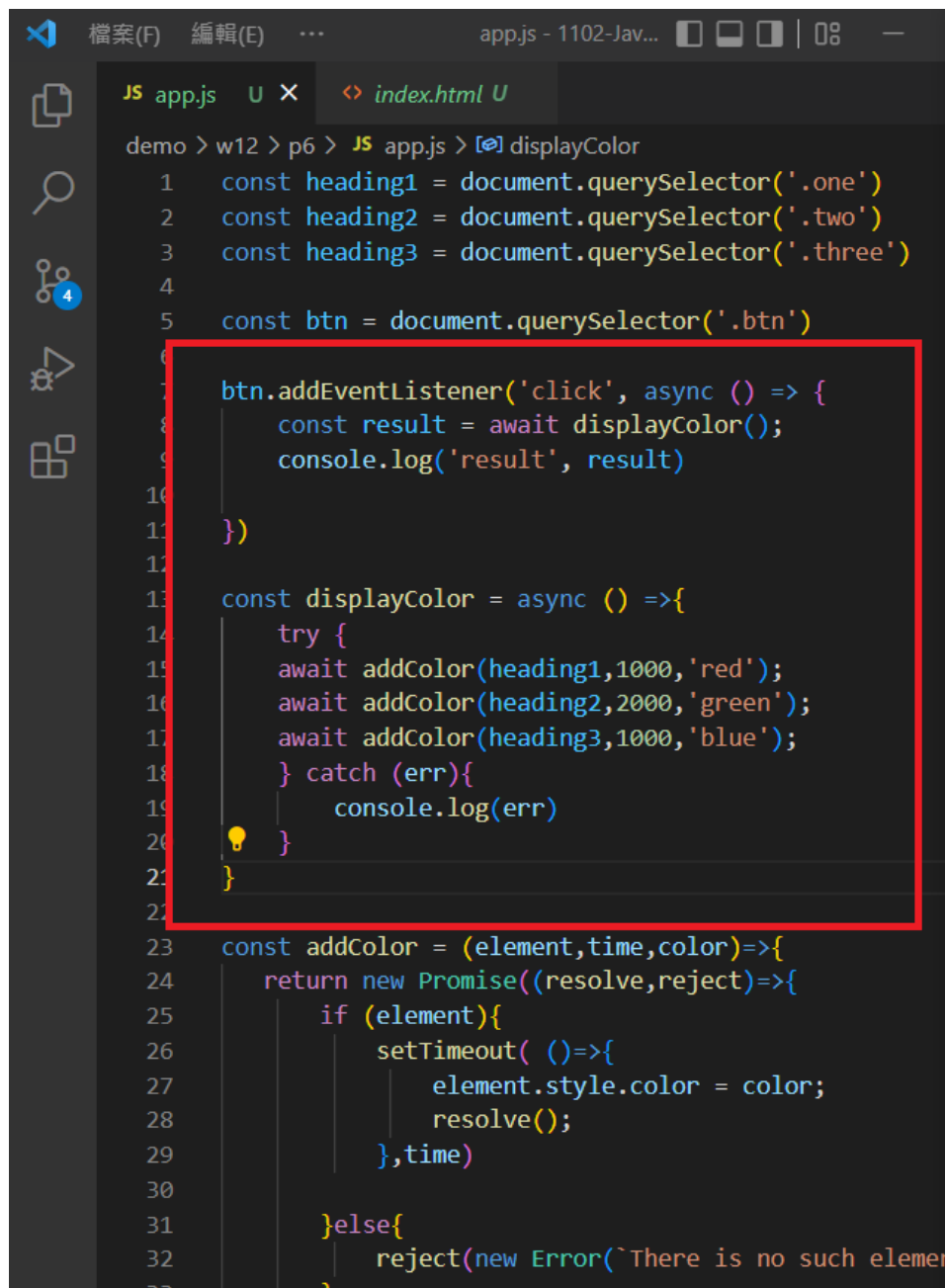
git log --pretty=format:"%h%x09%an%x09%ad%x09%s" --after="2022-05-10"
59d1261 mentally24      Wed May 11 19:43:42 2022 +0800  w12-P2 DOM call-back
functions demo -- colors change from red (1s), green (2s), blue (1s)

```

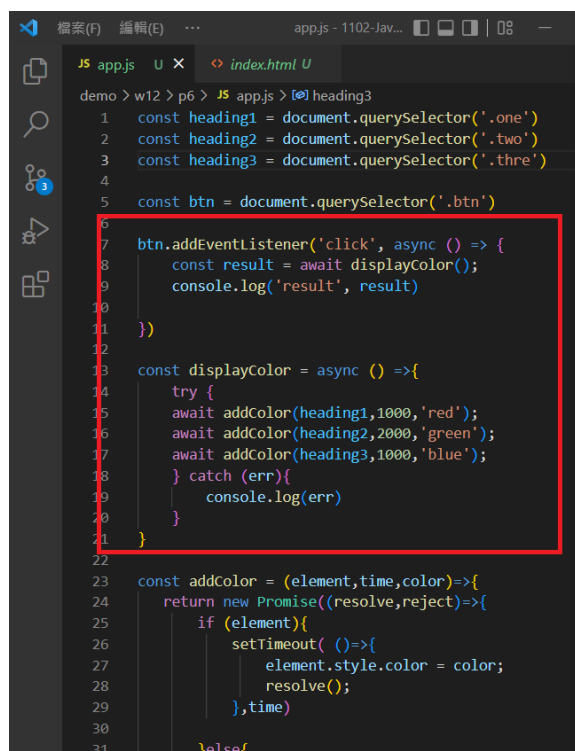
w12 p3 promise



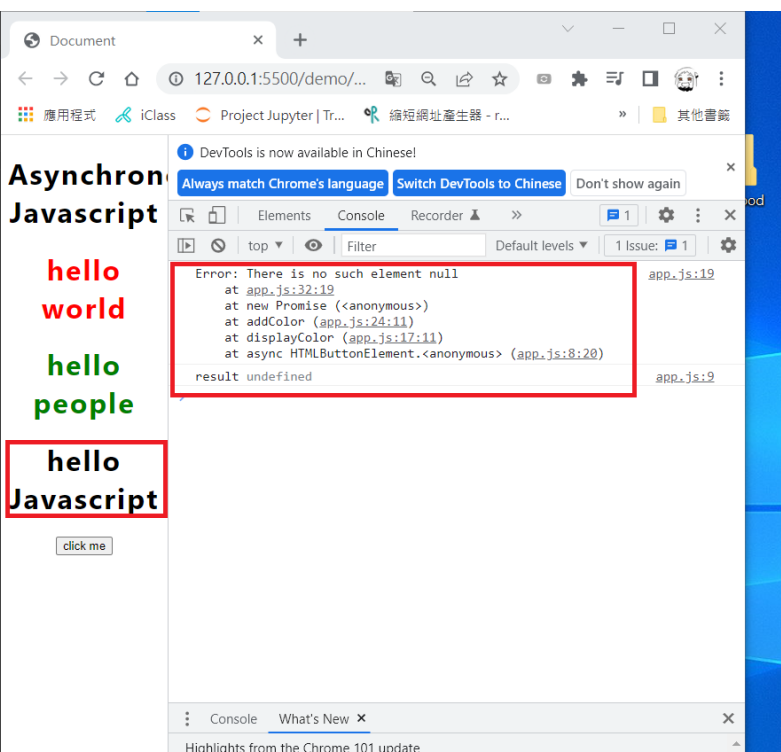
w12 p4 await



```
demo > w12 > p6 > JS app.js > displayColor
1  const heading1 = document.querySelector('.one')
2  const heading2 = document.querySelector('.two')
3  const heading3 = document.querySelector('.three')
4
5  const btn = document.querySelector('.btn')
6
7  btn.addEventListener('click', async () => {
8    const result = await displayColor();
9    console.log('result', result)
10  })
11
12
13  const displayColor = async () =>{
14    try {
15      await addColor(heading1,1000,'red');
16      await addColor(heading2,2000,'green');
17      await addColor(heading3,1000,'blue');
18    } catch (err){
19      console.log(err)
20    }
21  }
22
23  const addColor = (element,time,color)=>{
24    return new Promise((resolve,reject)=>{
25      if (element){
26        setTimeout( ()=>{
27          element.style.color = color;
28          resolve();
29        },time)
30      }else{
31        reject(new Error(`There is no such element`))
32      }
33    })
34  }
```



```
demo > w12 > p6 > JS app.js > heading3
1  const heading1 = document.querySelector('.one')
2  const heading2 = document.querySelector('.two')
3  const heading3 = document.querySelector('.three')
4
5  const btn = document.querySelector('.btn')
6
7  btn.addEventListener('click', async () => {
8    const result = await displayColor();
9    console.log('result', result)
10  })
11
12
13  const displayColor = async () =>{
14    try {
15      await addColor(heading1,1000,'red');
16      await addColor(heading2,2000,'green');
17      await addColor(heading3,1000,'blue');
18    } catch (err){
19      console.log(err)
20    }
21  }
22
23  const addColor = (element,time,color)=>{
24    return new Promise((resolve,reject)=>{
25      if (element){
26        setTimeout( ()=>{
27          element.style.color = color;
28          resolve();
29        },time)
30      }else{
31        reject(new Error(`There is no such element`))
32      }
33    })
34  }
```



```
32     reject(new Error(`There is no such element`));
33   }
34 }
```

Import and export
user flows as a JSON



w12 last

```
Jason@DESKTOP-8C7CC2C MINGW64 ~/Desktop/功課1/github/1102-JavaScript-207410191 (main)
$ git log --pretty=format:"%h%x09%an%x09%ad%x09%s" --after="2022-05-10"
1fa16dc mentally24      Wed May 11 21:03:51 2022 +0800  w12-P4
2e84761 mentally24      Wed May 11 20:30:02 2022 +0800  w12-P3 use promise addColor(element,time,color) to solve-call back hell in w12-P2
59d1261 mentally24      Wed May 11 19:43:42 2022 +0800  w12-P2 DOM call-back functions demo -- colors change from red (1s), green (2s), blue (1s)
3a854a7 mentally24      Wed May 11 19:14:09 2022 +0800  W12-P1: Making soup demo for Async JavaScript
```