

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

000000 ms

Web/Node APIs

Stack

warmup.js

Task Queue

Run Script

Console

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

000001 ms

Web/Node APIs

Stack

warmup.js

Task Queue

Run Script

Console

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

000002 ms

Web/Node APIs

Stack

Date.now

warmup.js

Task Queue

Run Script

Console

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

000003 ms

Web/Node APIs

Stack

warmup.js

Task Queue

Run Script

Console

```
01: function waitSync(delayMs) {  
02:   const waitStart = Date.now();  
03:   while (Date.now() - waitStart < delayMs) {}  
04: }  
05:  
06: const start = Date.now();  
07: function elapsedTimeLog(...args) {  
08:   const elapsedMs = Date.now() - start;  
09:   const sec = Math.floor(elapsedMs / 1000);  
10:   console.log(sec, ...args);  
11: }  
12:  
13: function handlerA() {  
14:   waitSync(6000);  
15:   elapsedTimeLog("A");  
16: }  
17:  
18: function handlerB() {  
19:   elapsedTimeLog("B");  
20: }  
21:  
22: setTimeout(handlerA, 1000);  
23: setTimeout(handlerB, 2000);  
24:  
25: elapsedTimeLog("C");
```

000004 ms

Web/Node APIs

Stack

warmup.js

Task Queue

Run Script

Console

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

000005 ms

Web/Node APIs

Stack

warmup.js

Task Queue

Run Script

Console

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

000006 ms

Web/Node APIs

Stack

warmup.js

Task Queue

Run Script

Console

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

000006 ms

Web/Node APIs

timeout

exp: 001006 ms
cb: handlerA

Stack

warmup.js

Task Queue

Run Script

Console


```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

000007 ms

Web/Node APIs

timeout

exp: 001006 ms
cb: handlerA

timeout

exp: 002007 ms
cb: handlerB

Stack

warmup.js

Task Queue

Run Script

Console

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

000008 ms

Web/Node APIs

timeout

exp: 001006 ms
cb: handlerA

timeout

exp: 002007 ms
cb: handlerB

Stack

elapsedTimeLog

warmup.js

Task Queue

Run Script

Console

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

000009 ms

Web/Node APIs

timeout

exp: 001006 ms
cb: handlerA

timeout

exp: 002007 ms
cb: handlerB

Stack

Date.now

elapsedTimeLog

warmup.js

Task Queue

Run Script

Console

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

000010 ms

Web/Node APIs

timeout

exp: 001006 ms
cb: handlerA

timeout

exp: 002007 ms
cb: handlerB

Stack

Math.floor

elapsedTimeLog

warmup.js

Task Queue

Run Script

Console

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

000011 ms

Web/Node APIs

timeout

exp: 001006 ms
cb: handlerA

timeout

exp: 002007 ms
cb: handlerB

Stack

console.log

elapsedTimeLog

warmup.js

Task Queue

Run Script

Console

0 C

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

000012 ms

Web/Node APIs

timeout

exp: 001006 ms
cb: handlerA

timeout

exp: 002007 ms
cb: handlerB

Stack

elapsedTimeLog

warmup.js

Task Queue

Run Script

Console

0 C

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

000013 ms

Web/Node APIs

timeout

exp: 001006 ms
cb: handlerA

timeout

exp: 002007 ms
cb: handlerB

Stack

warmup.js

Task Queue

Run Script

Console

0 C

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

000014 ms

Web/Node APIs

timeout

exp: 001006 ms
cb: handlerA

timeout

exp: 002007 ms
cb: handlerB

Stack

Task Queue

Run Script

Console

0 C


```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

000015 ms

Web/Node APIs

timeout	exp: 001006 ms cb: handlerA	timeout	exp: 002007 ms cb: handlerB
---------	--------------------------------	---------	--------------------------------

Stack

Task Queue

Console

0 C

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

000500 ms

Web/Node APIs

timeout

exp: 001006 ms
cb: handlerA

timeout

exp: 002007 ms
cb: handlerB

Stack

Task Queue

Console

0 C

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

001006 ms

Web/Node APIs

timeout

exp: 001006 ms
cb: handlerA

timeout

exp: 002007 ms
cb: handlerB

Stack

Task Queue

Console

0 C

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

001006 ms

Web/Node APIs

timeout

exp: 001006 ms
cb: handlerA

timeout

exp: 002007 ms
cb: handlerB

Stack

Task Queue

cb: handlerA

Console

0 C

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

001006 ms

Web/Node APIs

timeout

exp: 002007 ms
cb: handlerB

Stack

Task Queue

cb: handlerA

Console

0 C

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

001007 ms

Web/Node APIs

timeout

exp: 002007 ms
cb: handlerB

Stack

handlerA

Task Queue

cb: handlerA

Console

0 C

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

001008 ms

Web/Node APIs

timeout

exp: 002007 ms
cb: handlerB

Stack

waitSync

handlerA

Task Queue

cb: handlerA

Console

0 C

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

001009 ms

Web/Node APIs

timeout

exp: 002007 ms
cb: handlerB

Stack

waitSync

handlerA

Task Queue

cb: handlerA

Console

0 C


```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

001509 ms

Web/Node APIs

timeout

exp: 002007 ms
cb: handlerB

Stack

waitSync

handlerA

Task Queue

cb: handlerA

Console

0 C

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

002007 ms

Web/Node APIs

timeout

exp: 002007 ms
cb: handlerB

Stack

waitSync

handlerA

Task Queue

cb: handlerA

Console

0 C

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

002007 ms

Web/Node APIs

timeout

exp: 002007 ms
cb: handlerB

Stack

waitSync

handlerA

Task Queue

cb: handlerA

cb: handlerB

Console

0 C

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

002007 ms

Web/Node APIs

Stack

waitSync

handlerA

Task Queue

cb: handlerA

cb: handlerB

Console

0 C

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

002008 ms

Web/Node APIs

Stack

waitSync

handlerA

Task Queue

cb: handlerA

cb: handlerB

Console

0 C

```
01: function waitSync(delayMs) {  
02:   const waitStart = Date.now();  
03:   while (Date.now() - waitStart < delayMs) {}  
04: }  
05:  
06: const start = Date.now();  
07: function elapsedTimeLog(...args) {  
08:   const elapsedMs = Date.now() - start;  
09:   const sec = Math.floor(elapsedMs / 1000);  
10:   console.log(sec, ...args);  
11: }  
12:  
13: function handlerA() {  
14:   waitSync(6000);  
15:   elapsedTimeLog("A");  
16: }  
17:  
18: function handlerB() {  
19:   elapsedTimeLog("B");  
20: }  
21:  
22: setTimeout(handlerA, 1000);  
23: setTimeout(handlerB, 2000);  
24:  
25: elapsedTimeLog("C");
```

007008 ms

Web/Node APIs

Stack

waitSync

handlerA

Task Queue

cb: handlerA

cb: handlerB

Console

0 C

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

007009 ms

Web/Node APIs

Stack

elapsedTimeLog

handlerA

Task Queue

cb: handlerA

cb: handlerB

Console

0 C

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

007010 ms

Web/Node APIs

Stack

Date.now

elapsedTimeLog

handlerA

Task Queue

cb: handlerA

cb: handlerB

Console

0 C


```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

007011 ms

Web/Node APIs

Stack

Math.floor

elapsedTimeLog

handlerA

Task Queue

cb: handlerA

cb: handlerB

Console

0 C

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

007012 ms

Web/Node APIs

Stack

console.log

elapsedTimeLog

handlerA

Task Queue

cb: handlerA

cb: handlerB

Console

0 C

7 A

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

007013 ms

Web/Node APIs

Stack

elapsedTimeLog

handlerA

Task Queue

cb: handlerA

cb: handlerB

Console

0 C

7 A

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

007014 ms

Web/Node APIs

Stack

handlerA

Task Queue

cb: handlerA

cb: handlerB

Console

0 C

7 A

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

007015 ms

Web/Node APIs

Stack

Task Queue

cb: handlerA

cb: handlerB

Console

0 C

7 A

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

007016 ms

Web/Node APIs

Stack

Task Queue

cb: handlerB

Console

0 C

7 A

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

007017 ms

Web/Node APIs

Stack

Task Queue

cb: handlerB

Console

0 C

7 A

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

007018 ms

Web/Node APIs

Stack

handlerB

Task Queue

cb: handlerB

Console

0 C

7 A


```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

007019 ms

Web/Node APIs

Stack

elapsedTimeLog

handlerB

Task Queue

cb: handlerB

Console

0 C

7 A

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

007020 ms

Web/Node APIs

Stack

Date.now

elapsedTimeLog

handlerB

Task Queue

cb: handlerB

Console

0 C

7 A

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

007021 ms

Web/Node APIs

Stack

Math.floor

elapsedTimeLog

handlerB

Task Queue

cb: handlerB

Console

0 C

7 A

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

007022 ms

Web/Node APIs

Stack

console.log

elapsedTimeLog

handlerB

Task Queue

cb: handlerB

Console

0 C

7 A

7 B

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

007023 ms

Web/Node APIs

Stack

elapsedTimeLog

handlerB

Task Queue

cb: handlerB

Console

0 C

7 A

7 B

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

007024 ms

Web/Node APIs

Stack

handlerB

Task Queue

cb: handlerB

Console

0 C

7 A

7 B

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

007025 ms

Web/Node APIs

Stack

Task Queue

cb: handlerB

Console

0 C

7 A

7 B

```
01: function waitSync(delayMs) {
02:   const waitStart = Date.now();
03:   while (Date.now() - waitStart < delayMs) {}
04: }
05:
06: const start = Date.now();
07: function elapsedTimeLog(...args) {
08:   const elapsedMs = Date.now() - start;
09:   const sec = Math.floor(elapsedMs / 1000);
10:   console.log(sec, ...args);
11: }
12:
13: function handlerA() {
14:   waitSync(6000);
15:   elapsedTimeLog("A");
16: }
17:
18: function handlerB() {
19:   elapsedTimeLog("B");
20: }
21:
22: setTimeout(handlerA, 1000);
23: setTimeout(handlerB, 2000);
24:
25: elapsedTimeLog("C");
```

007026 ms

Web/Node APIs

Stack

Task Queue

Console

0 C

7 A

7 B