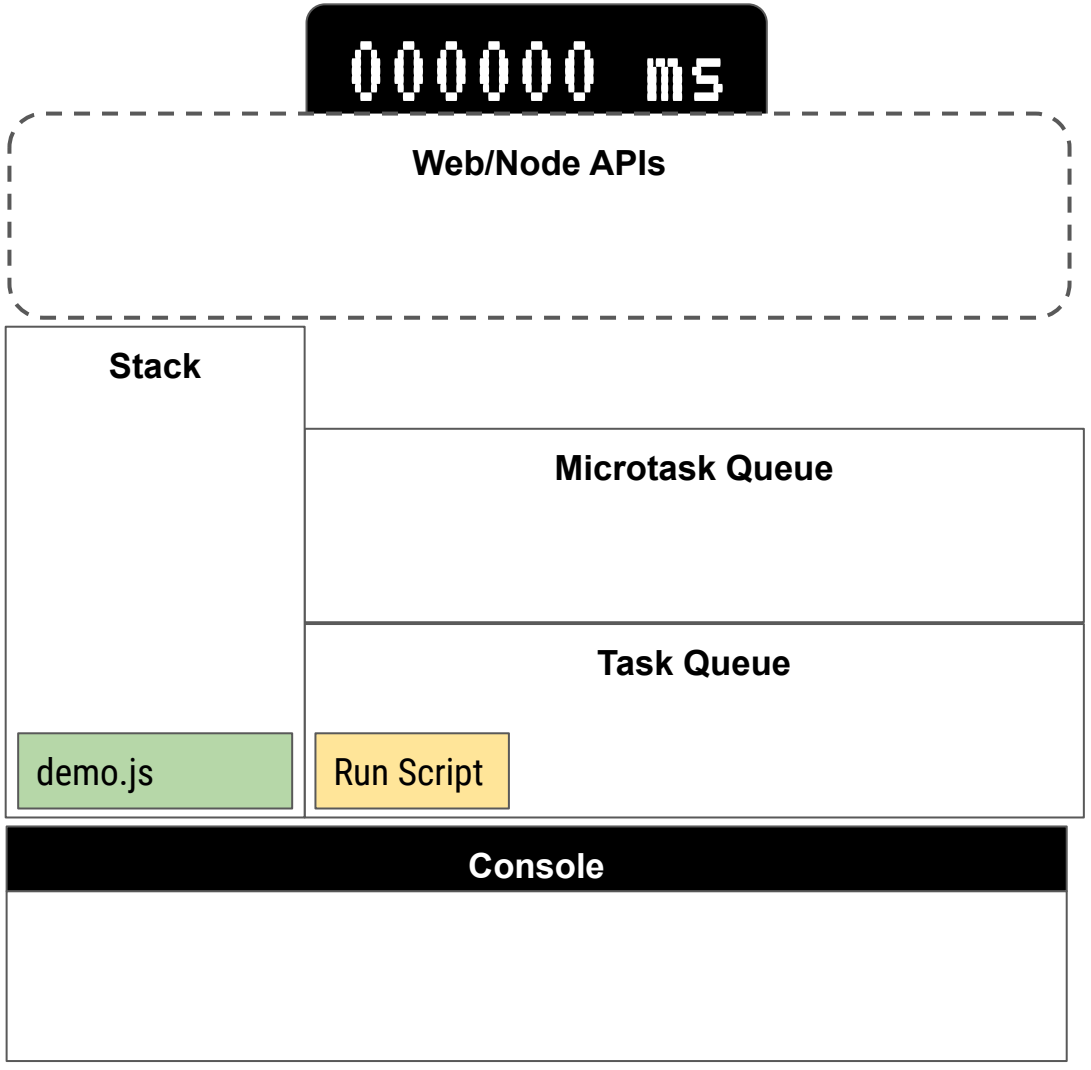
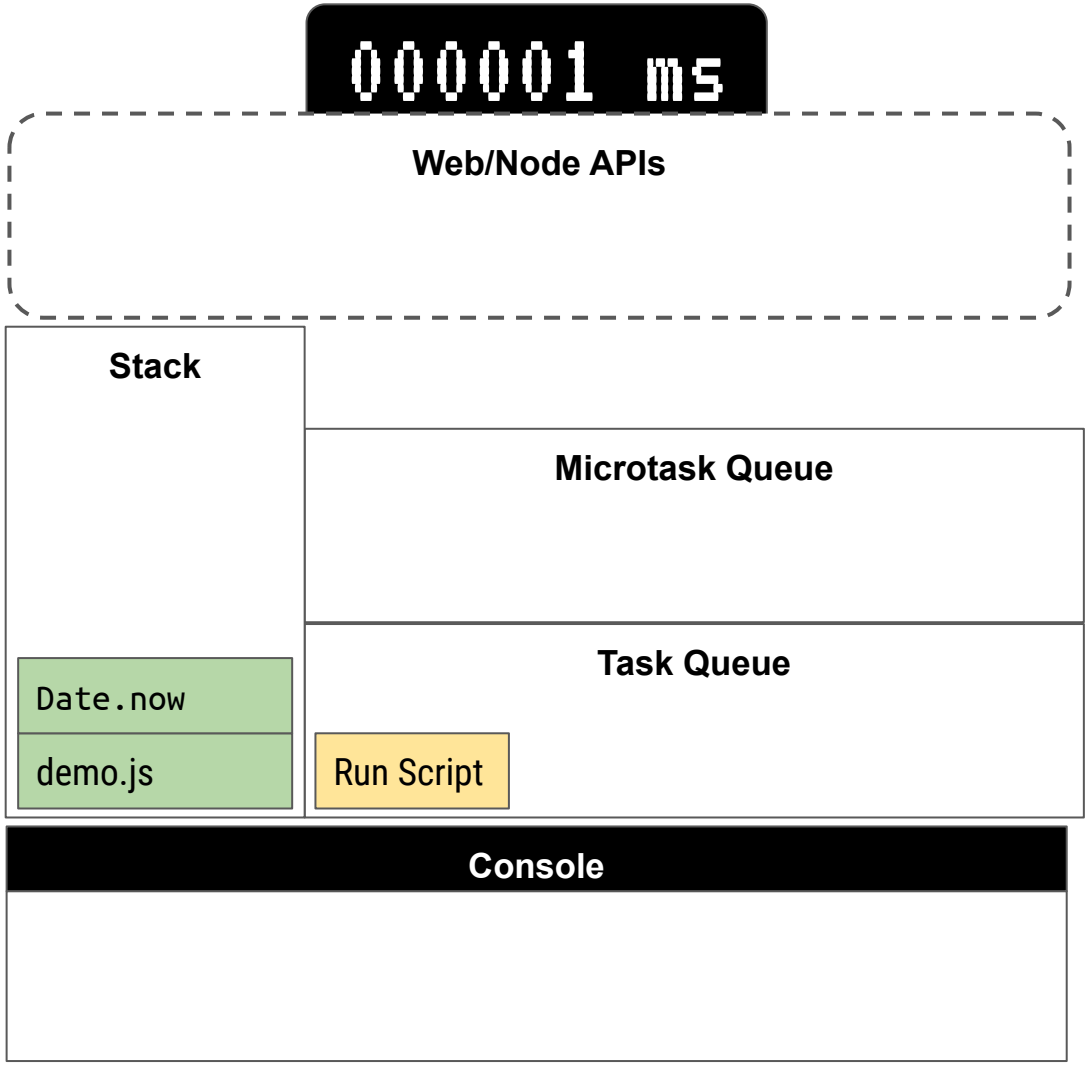
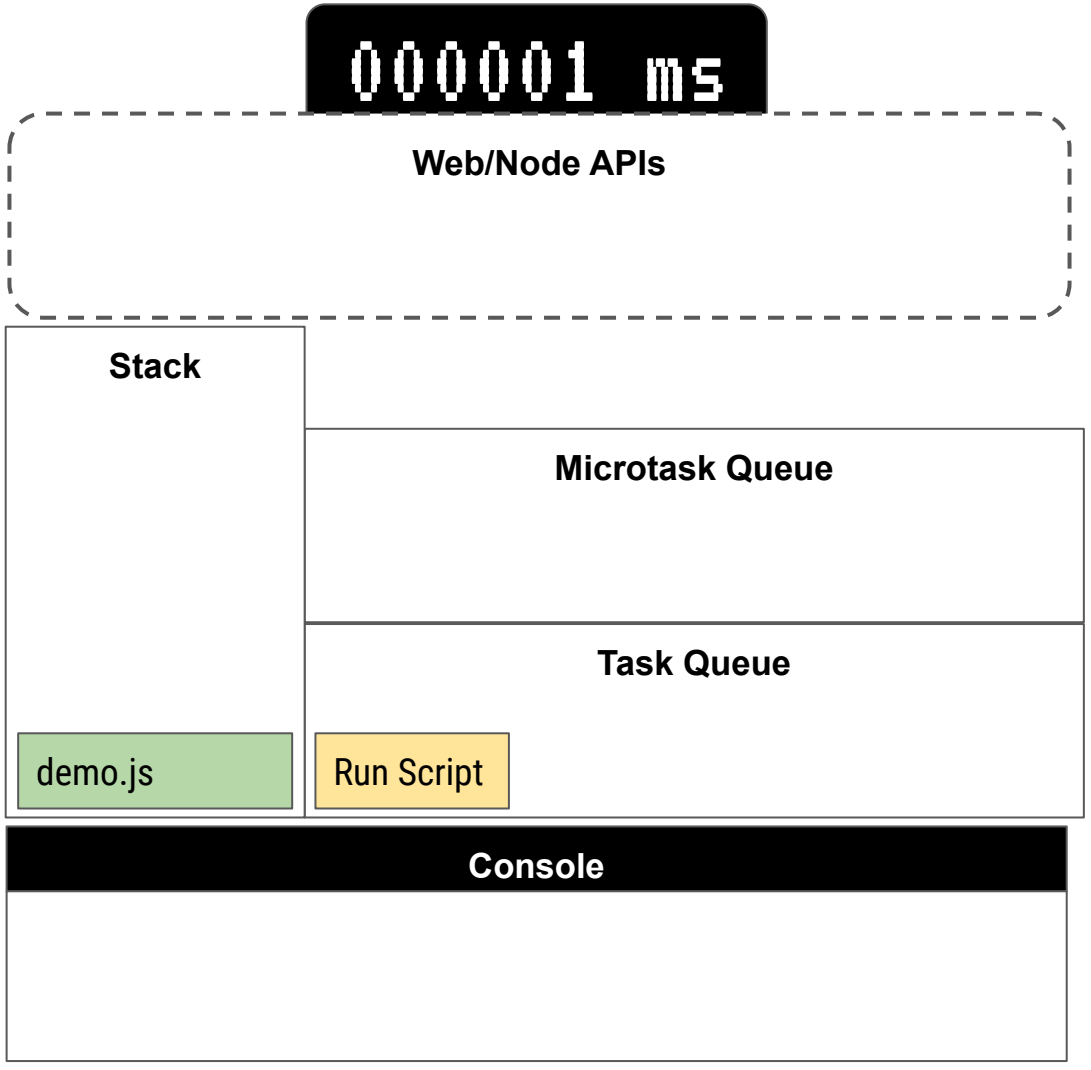
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



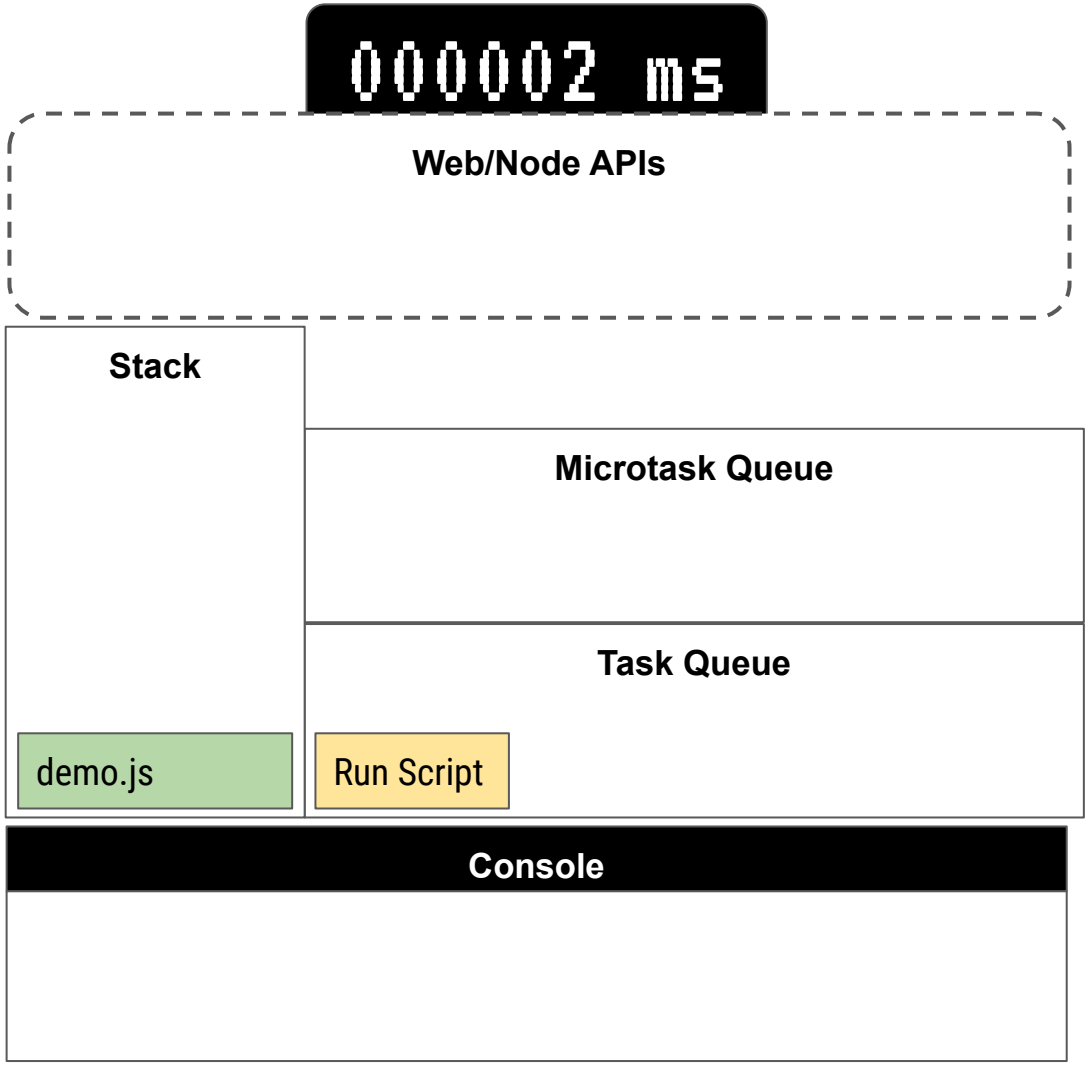
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



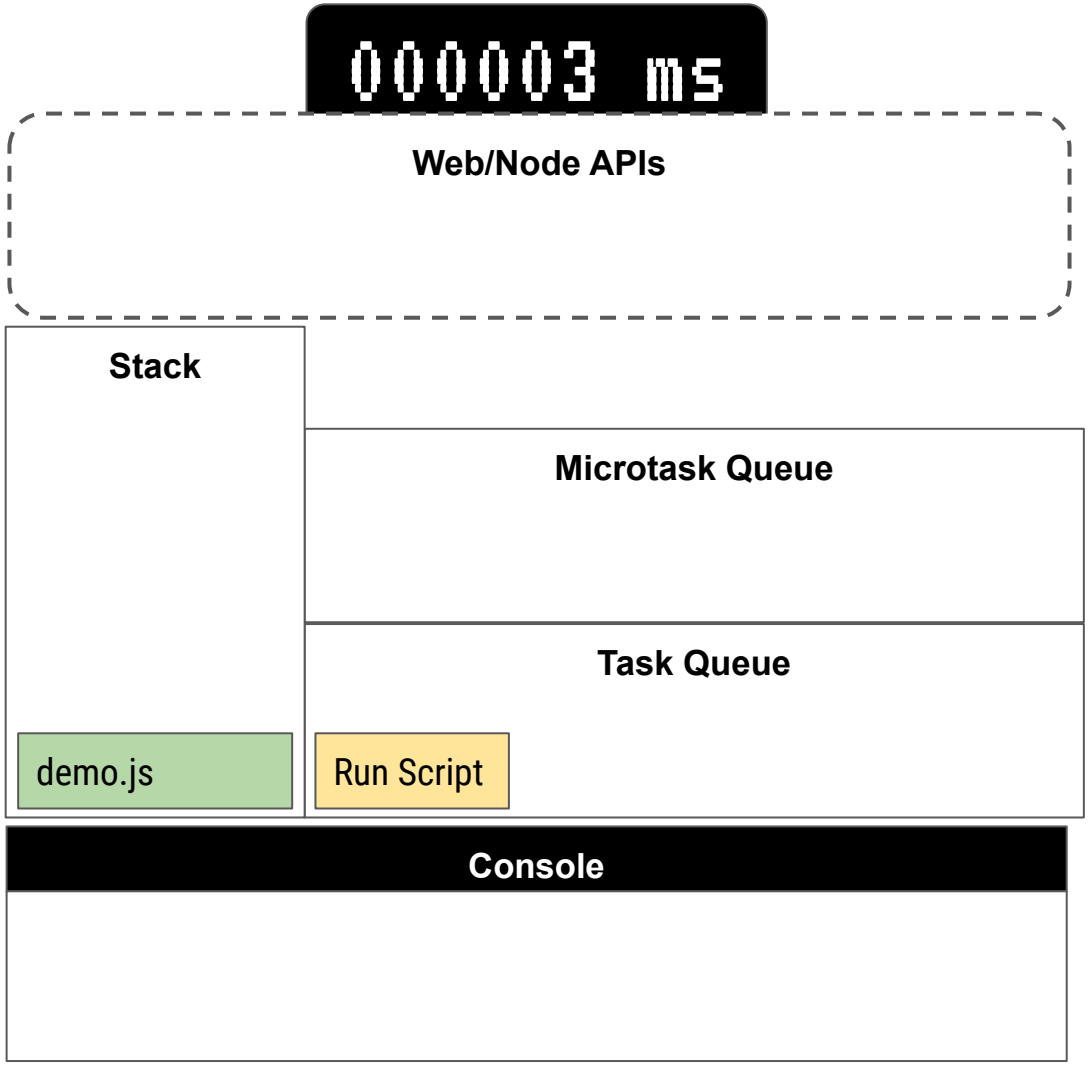
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



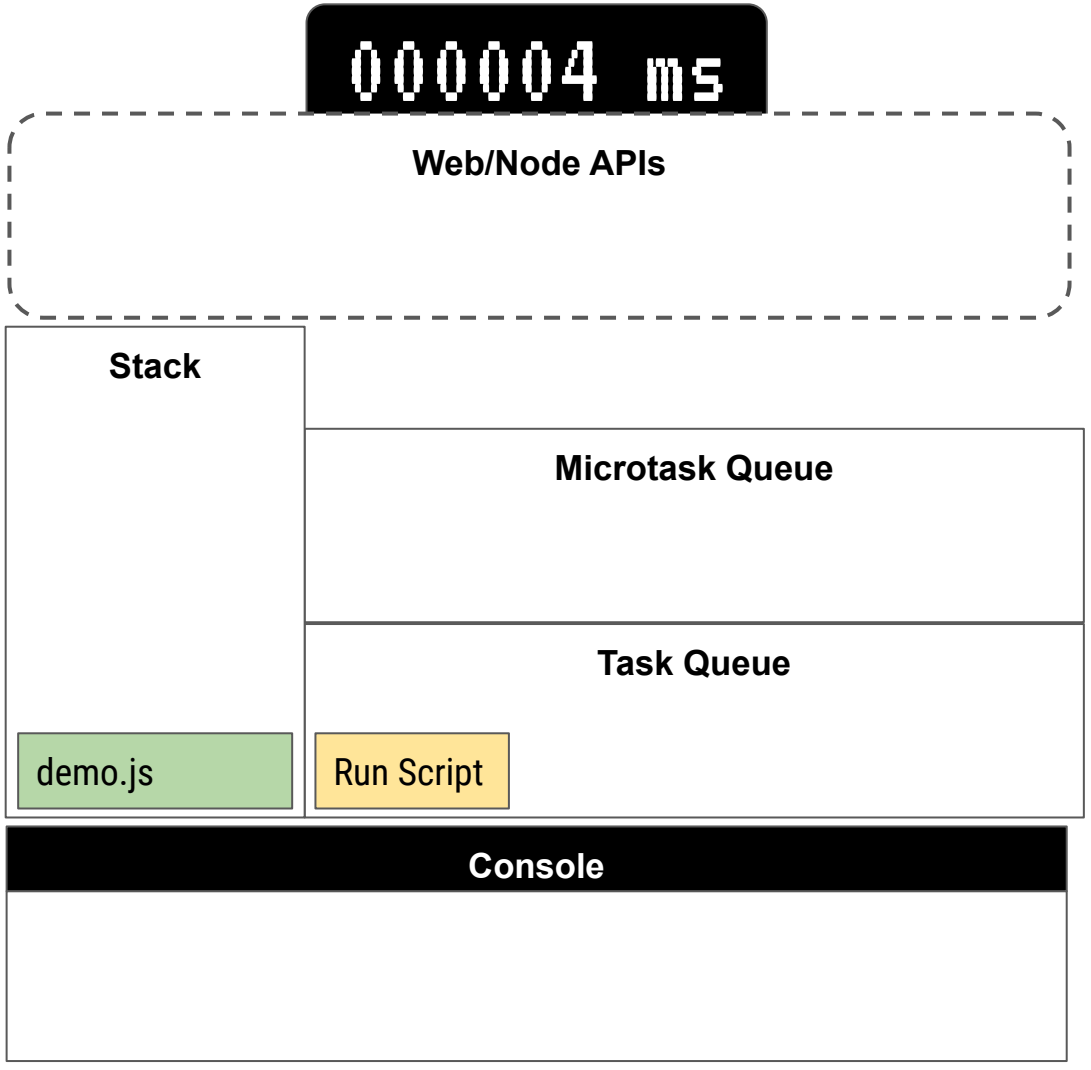
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



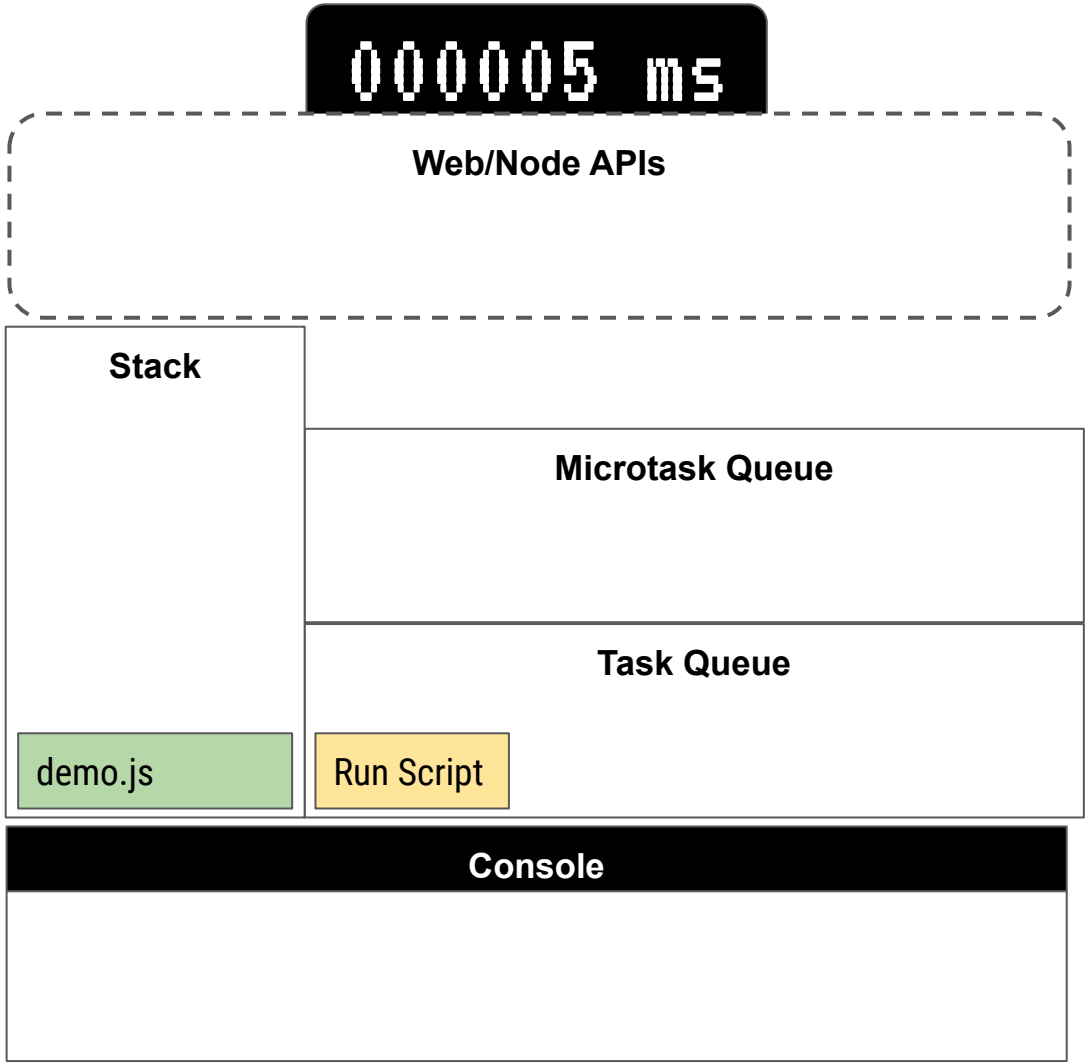
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



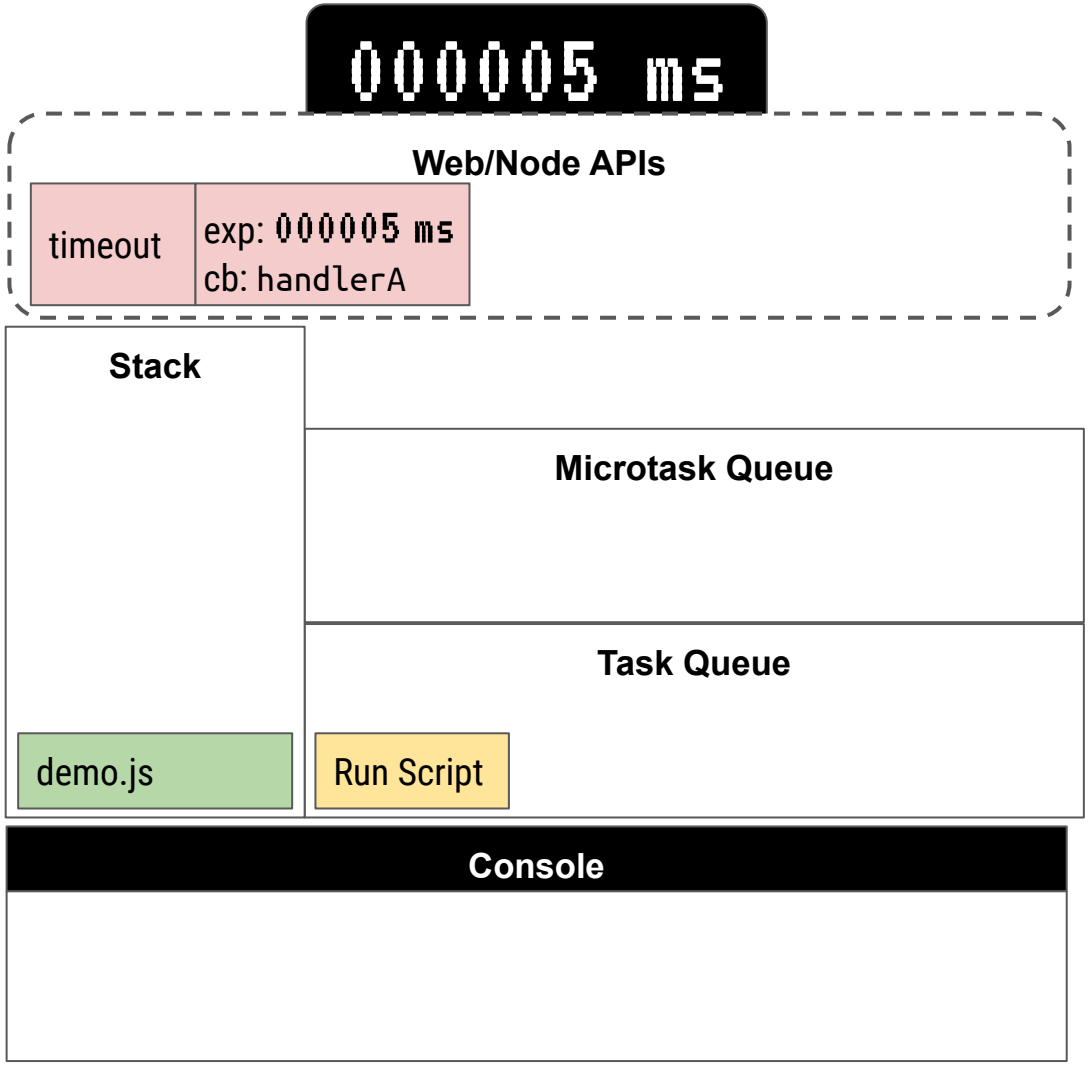
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



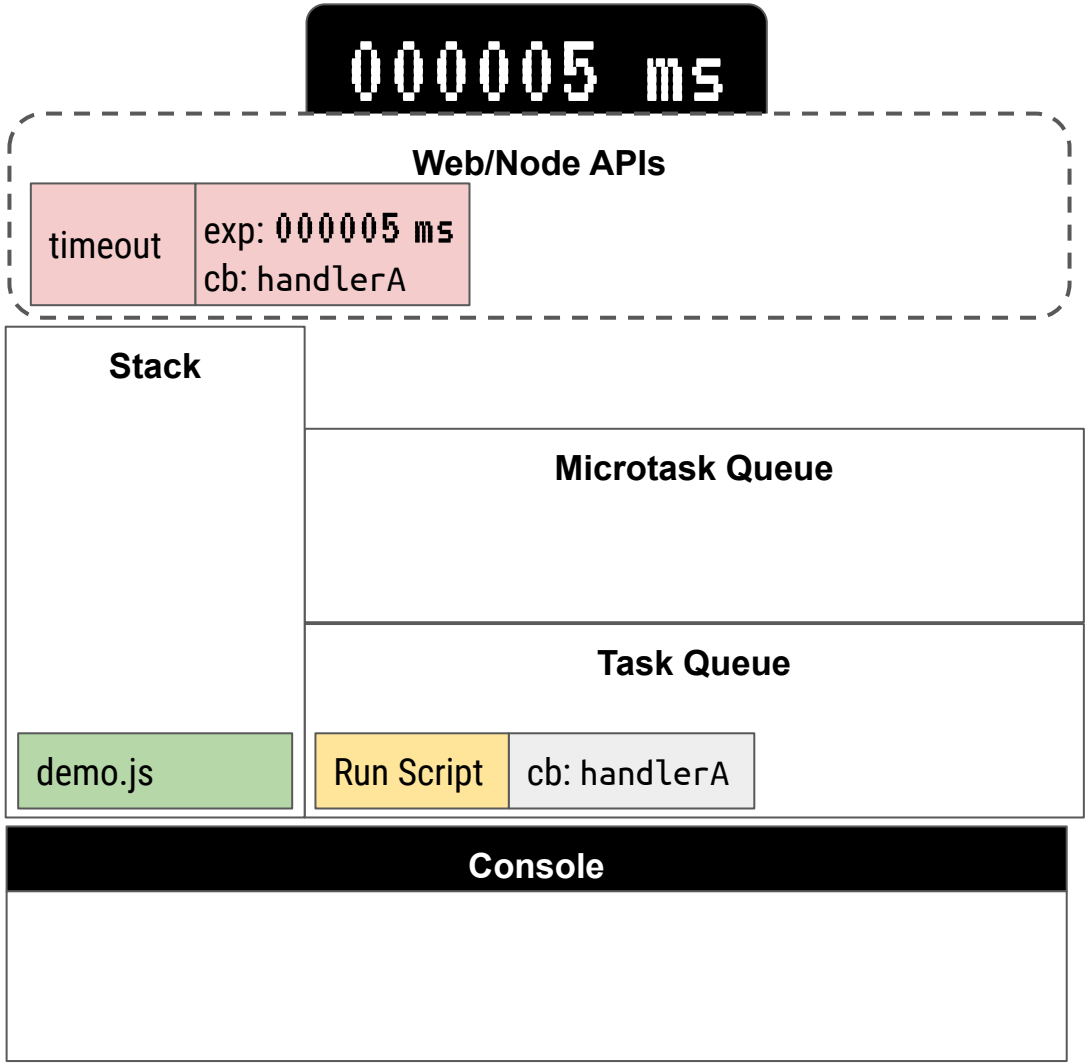
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



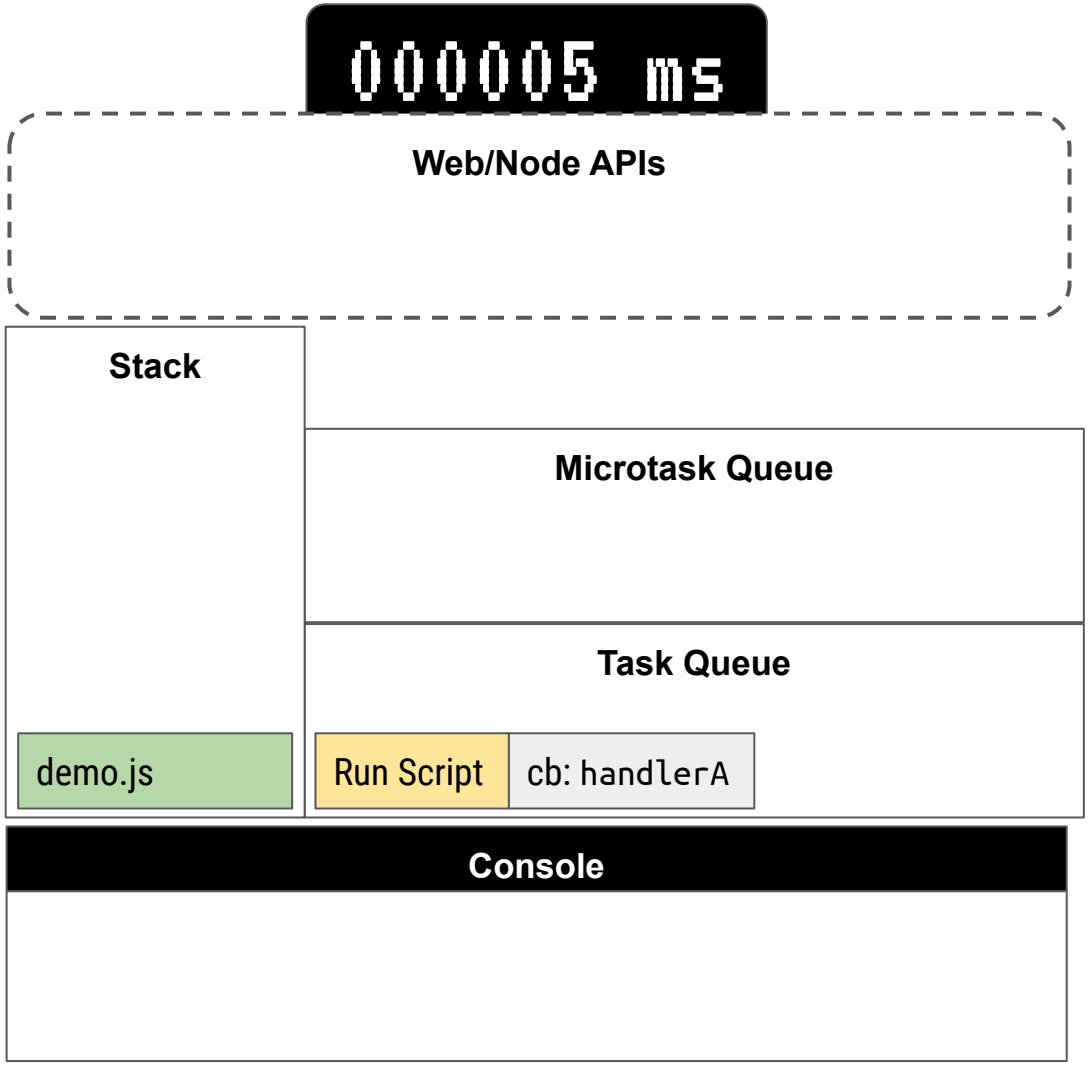
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



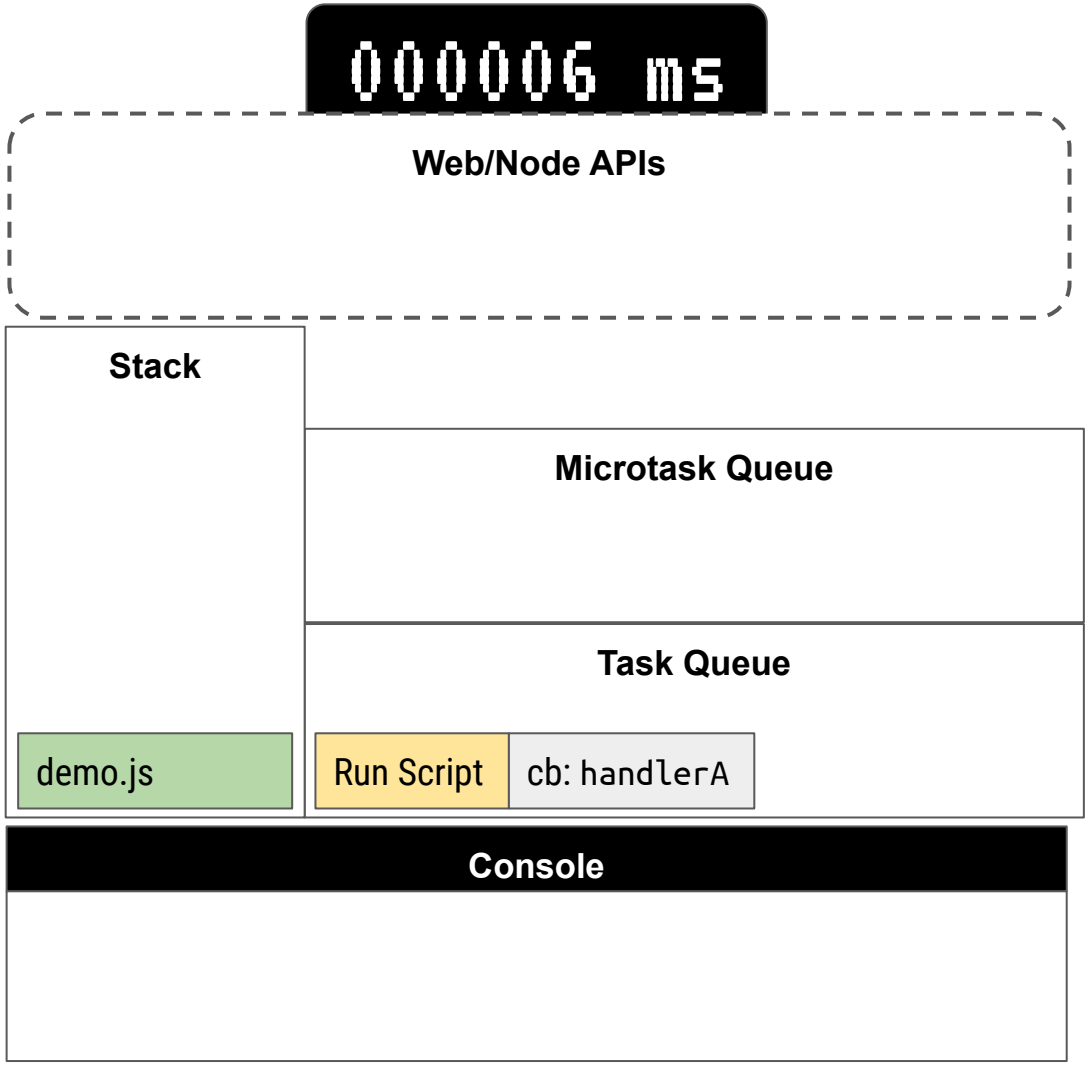
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



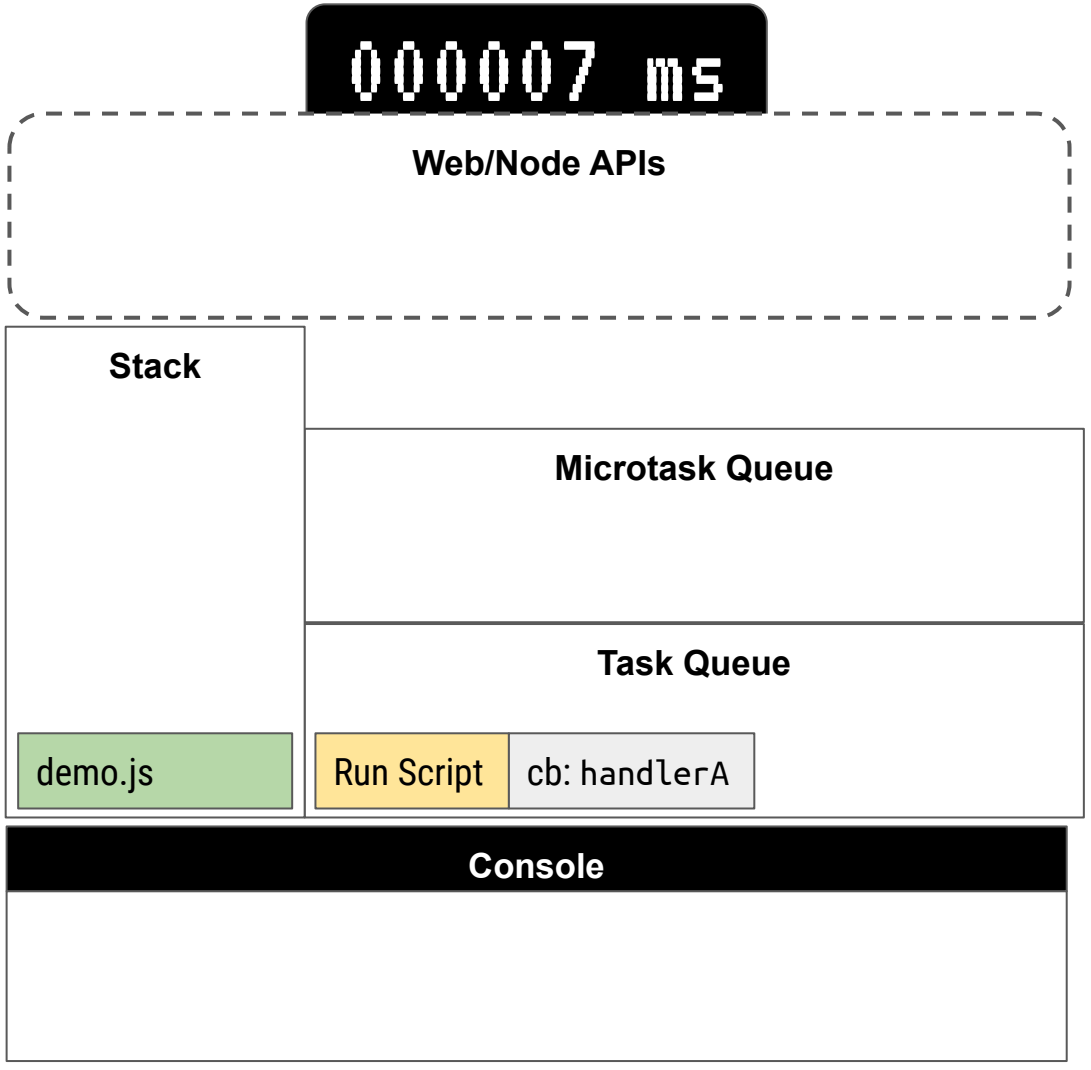
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



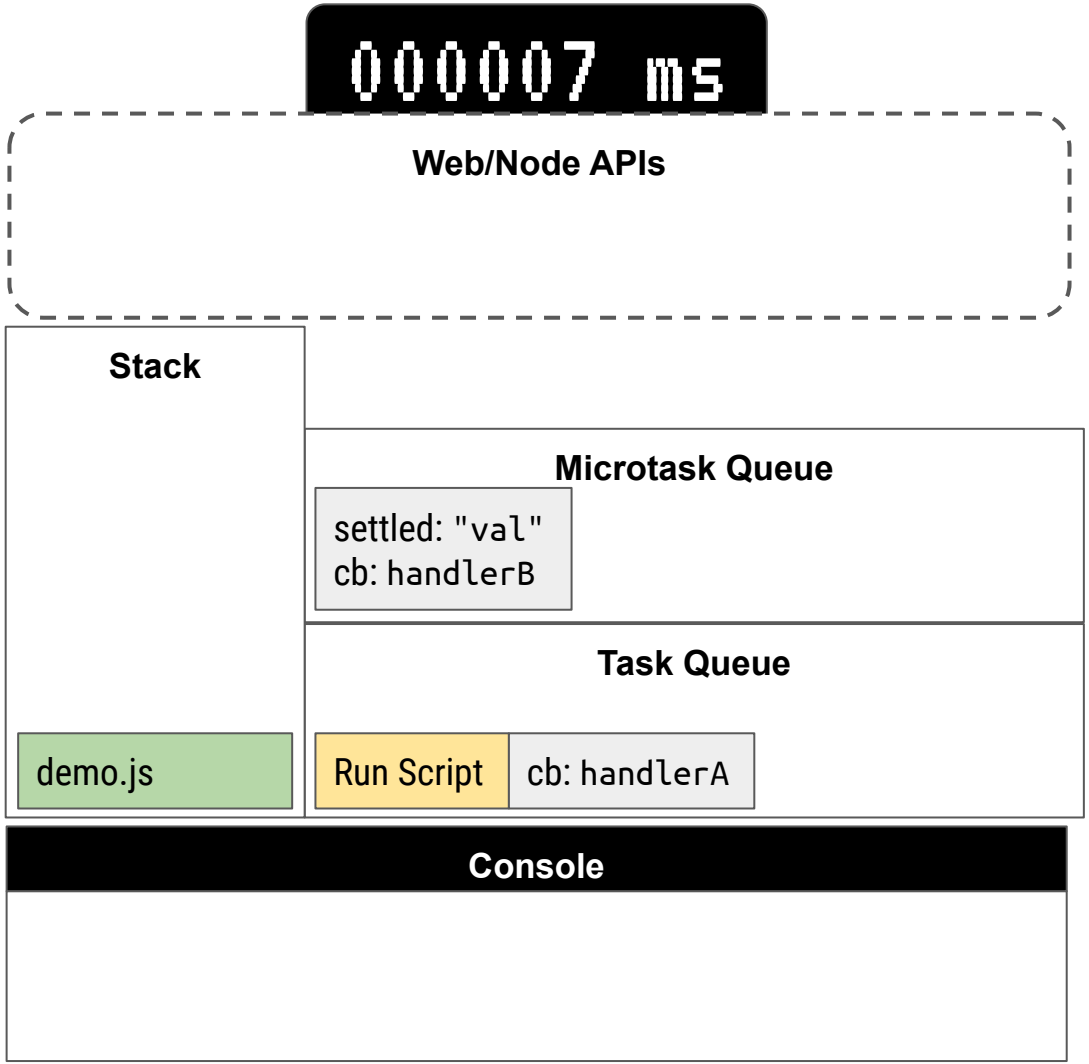
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



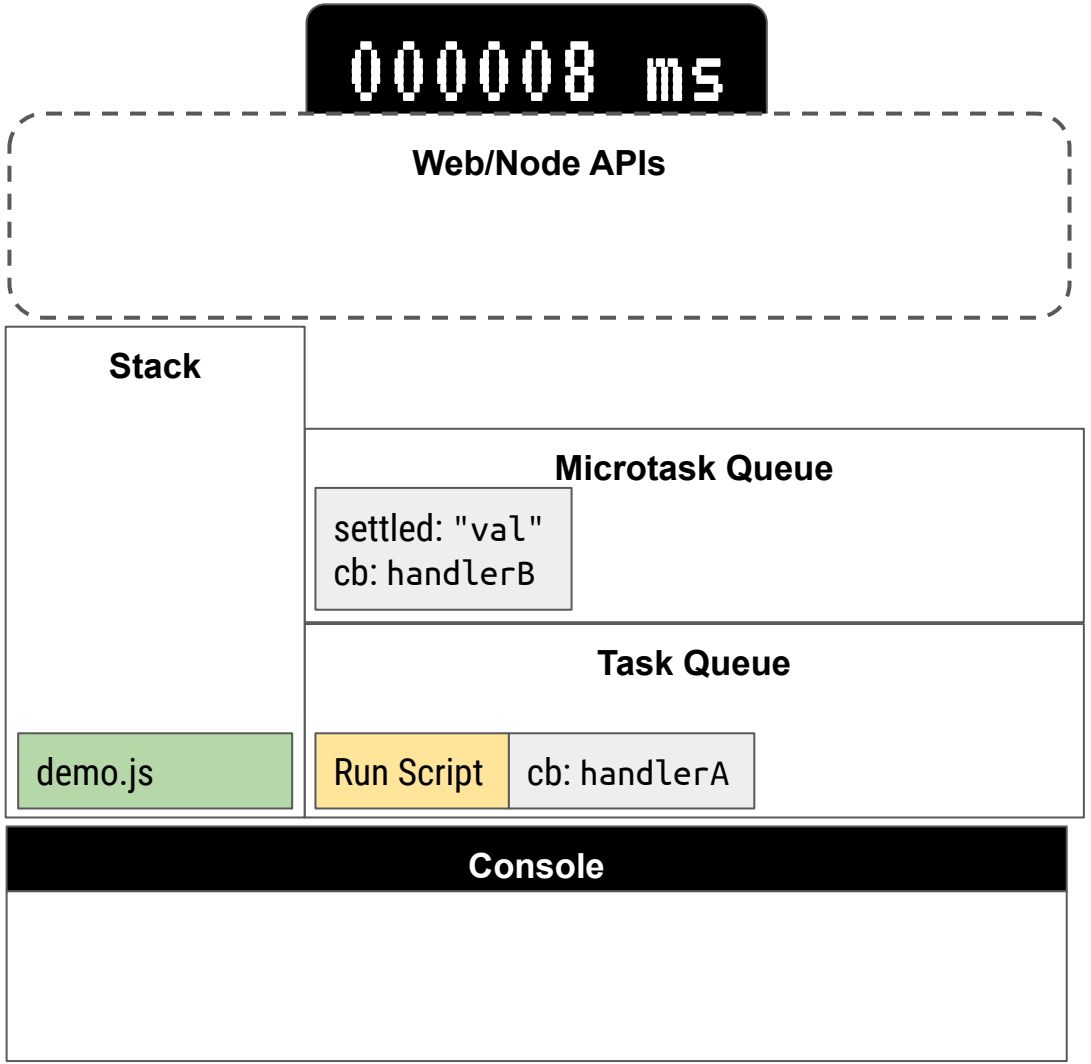
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



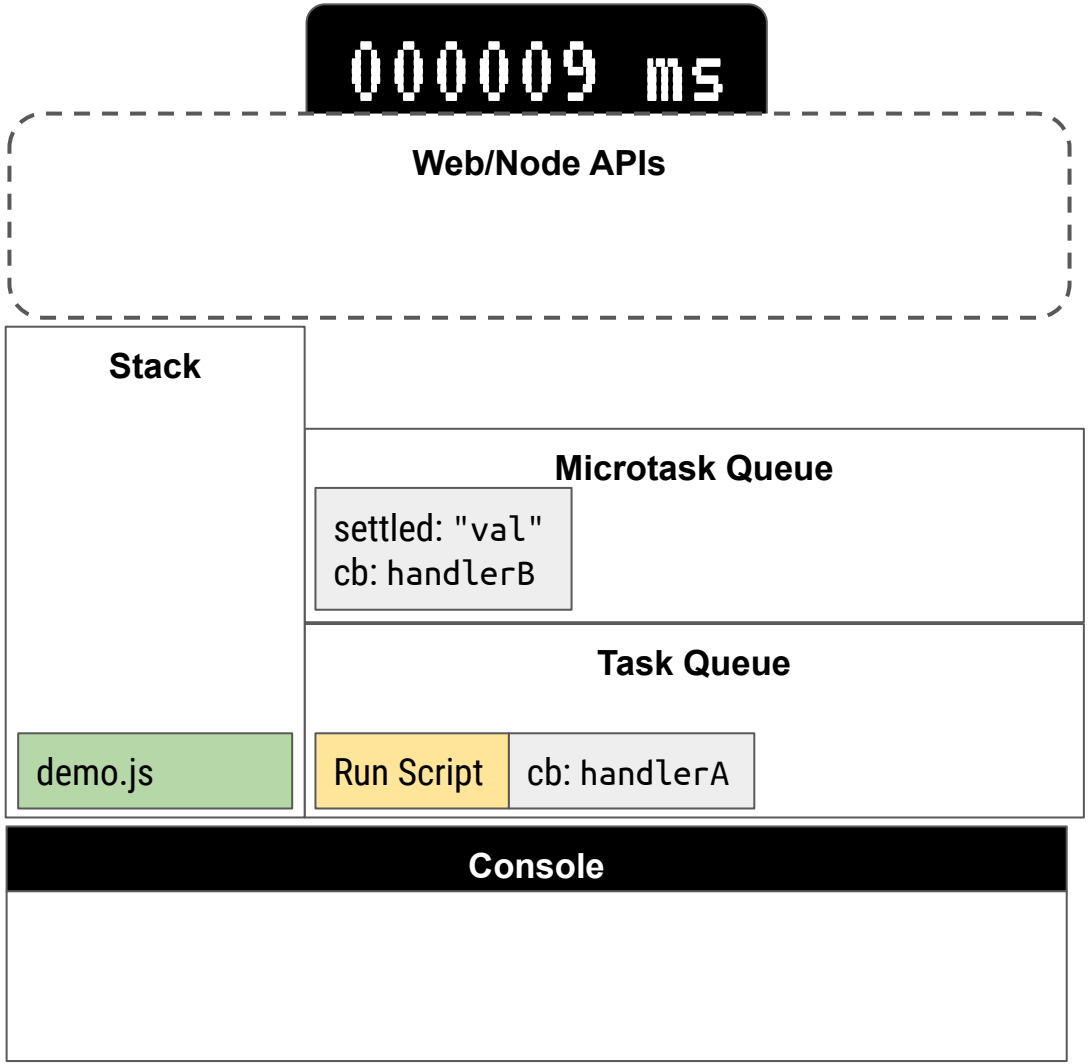
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



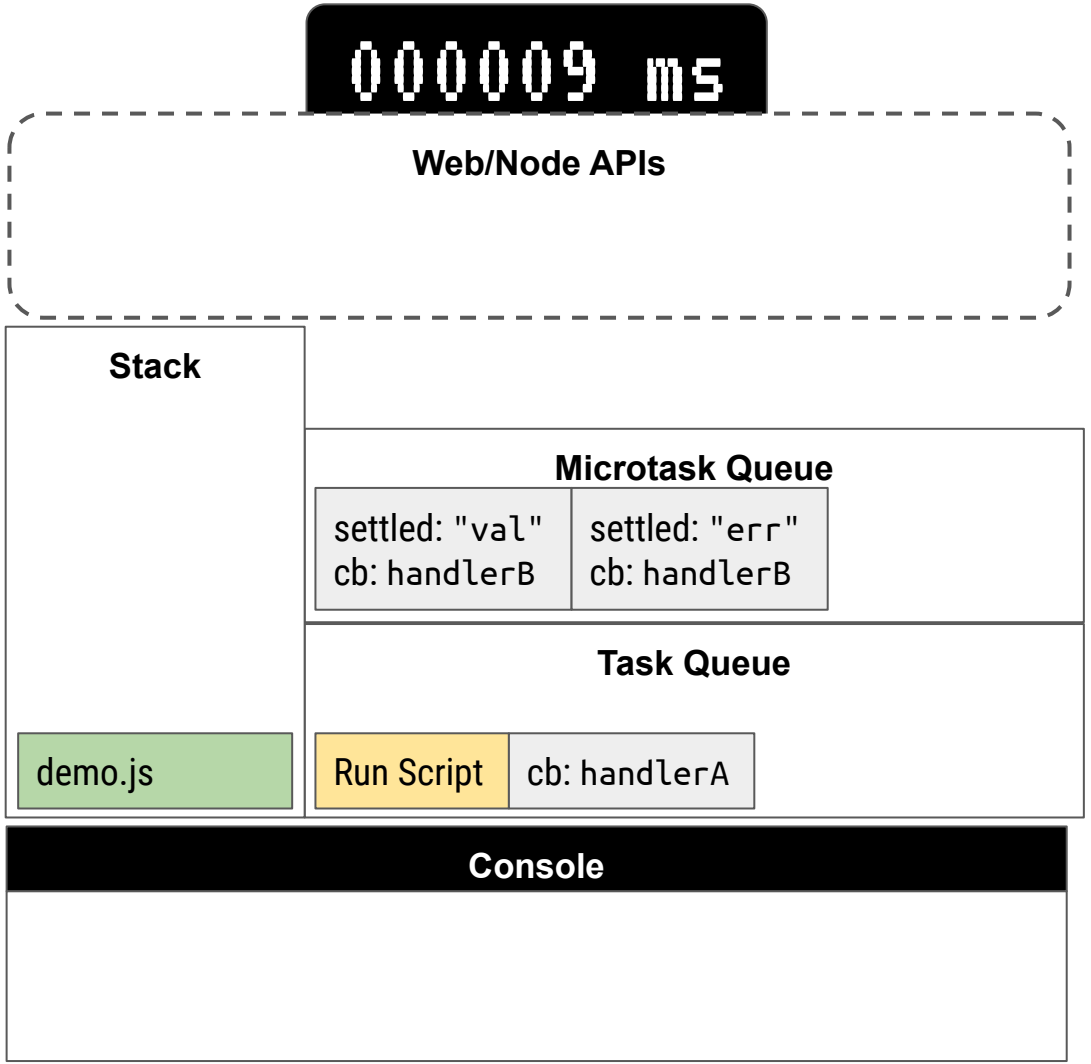
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



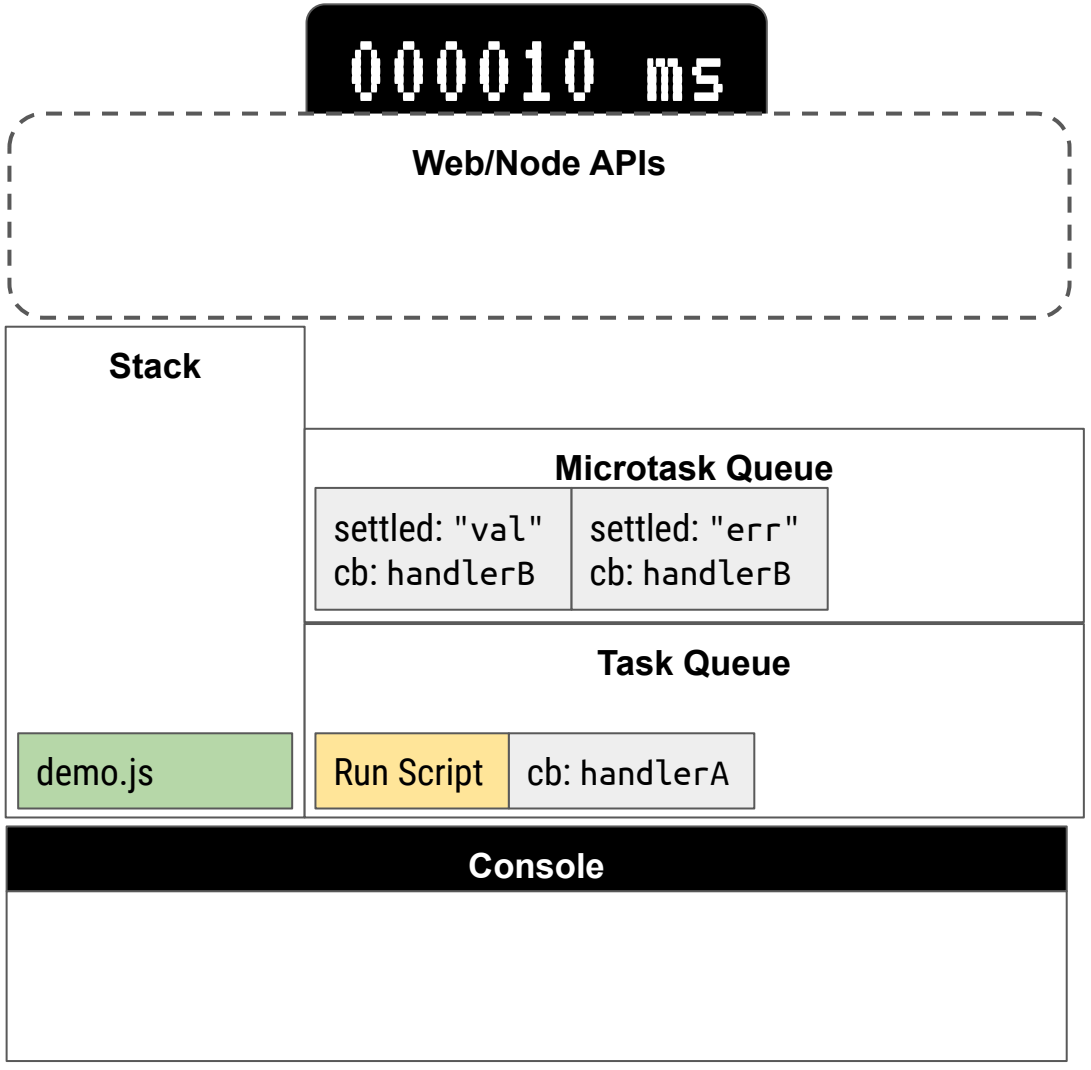
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



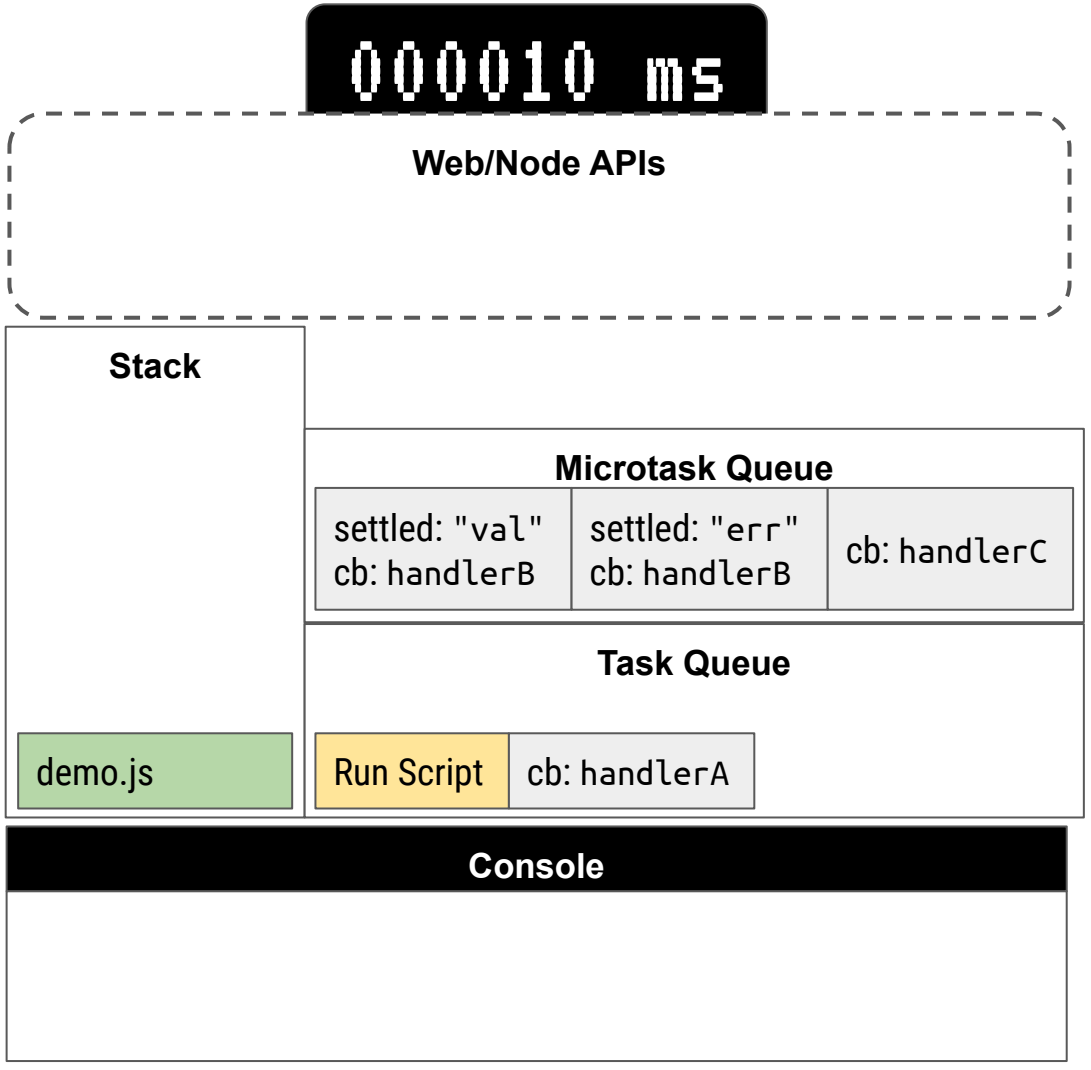
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



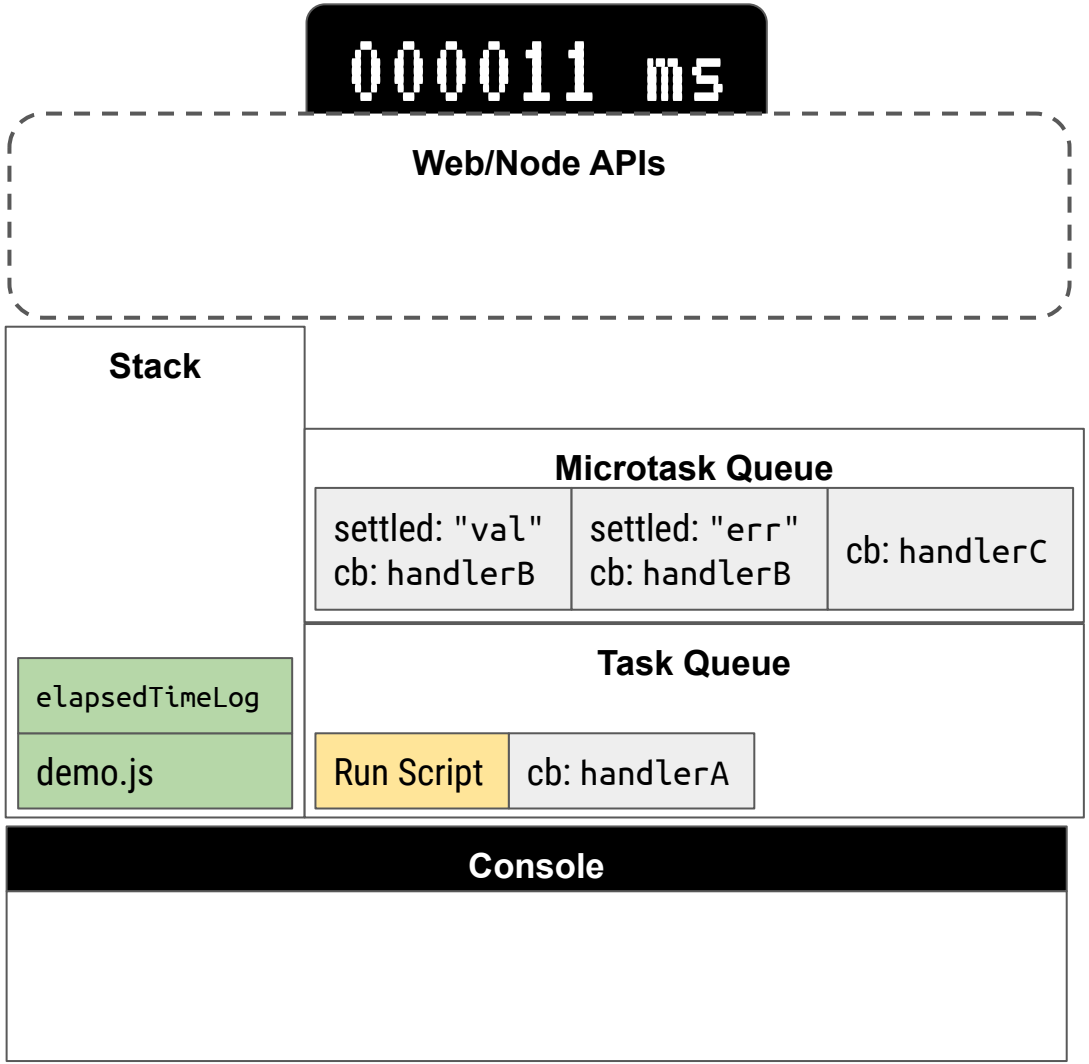
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



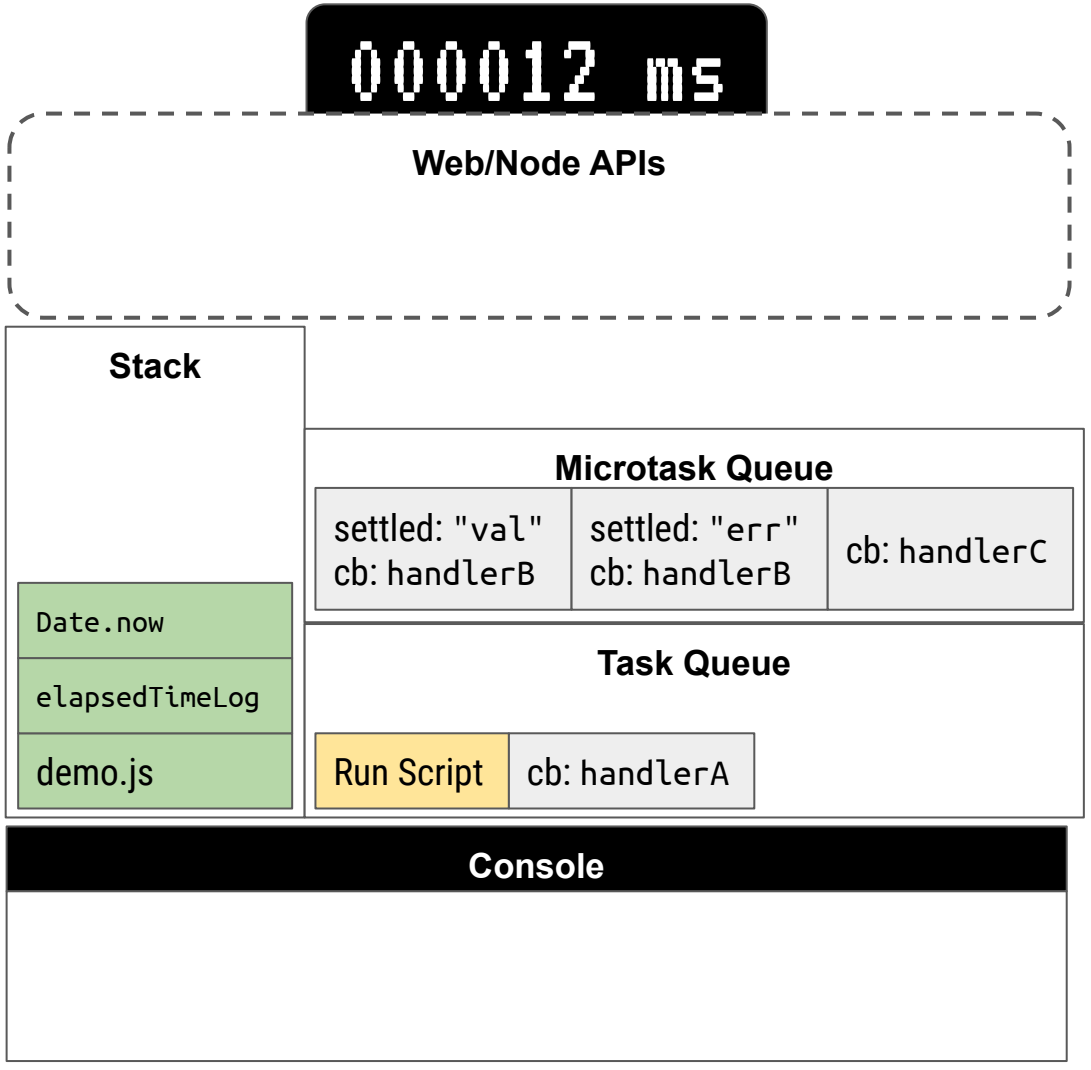
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



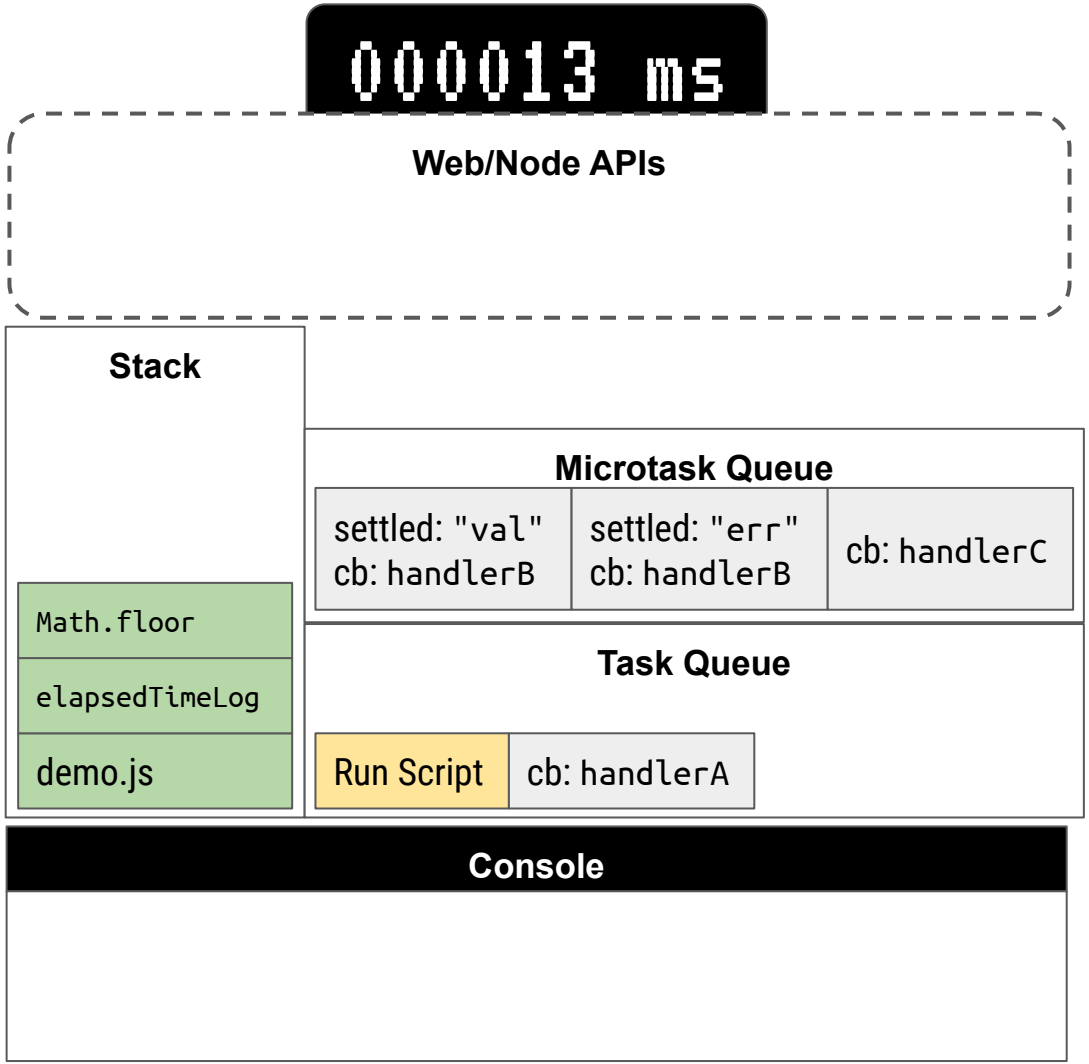
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



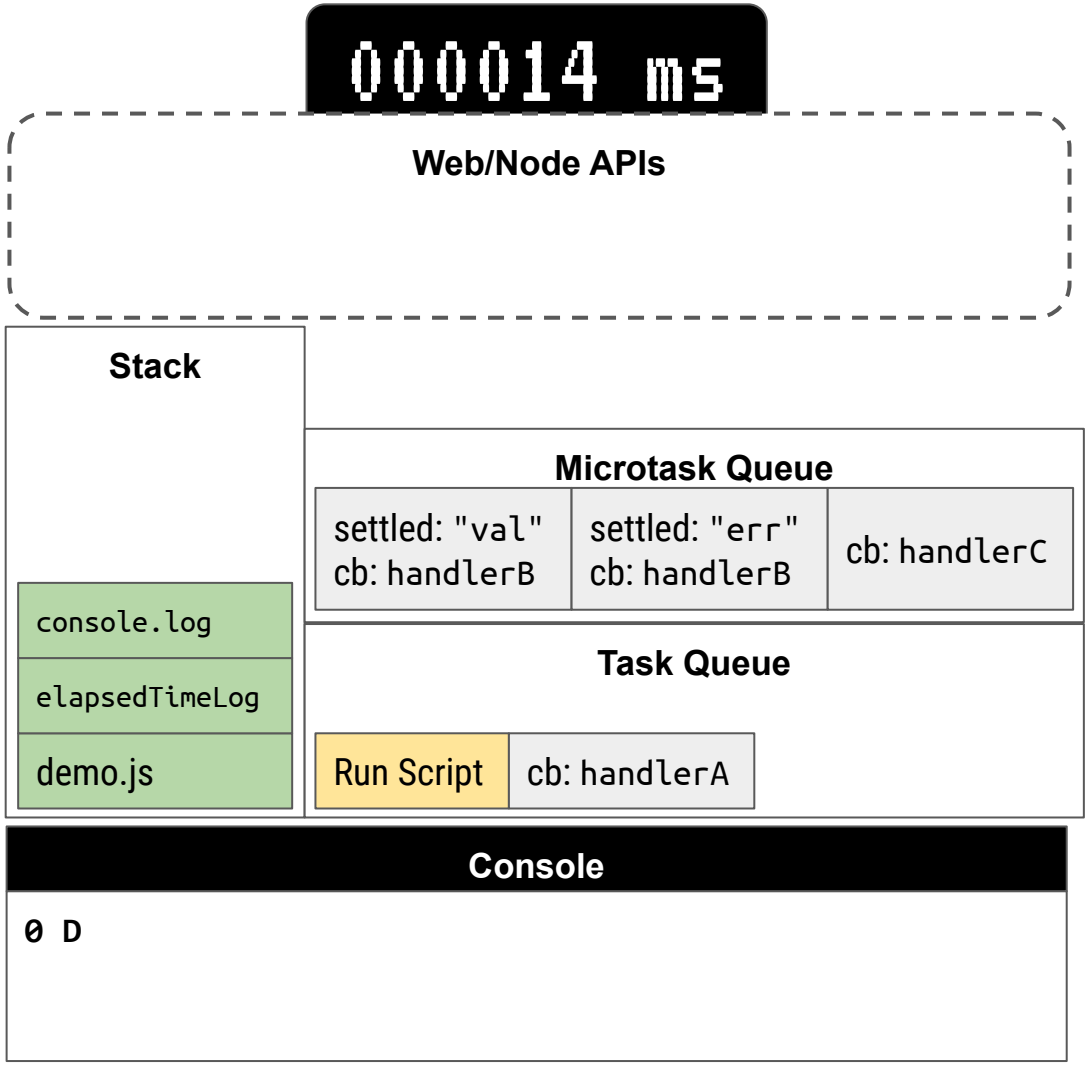
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



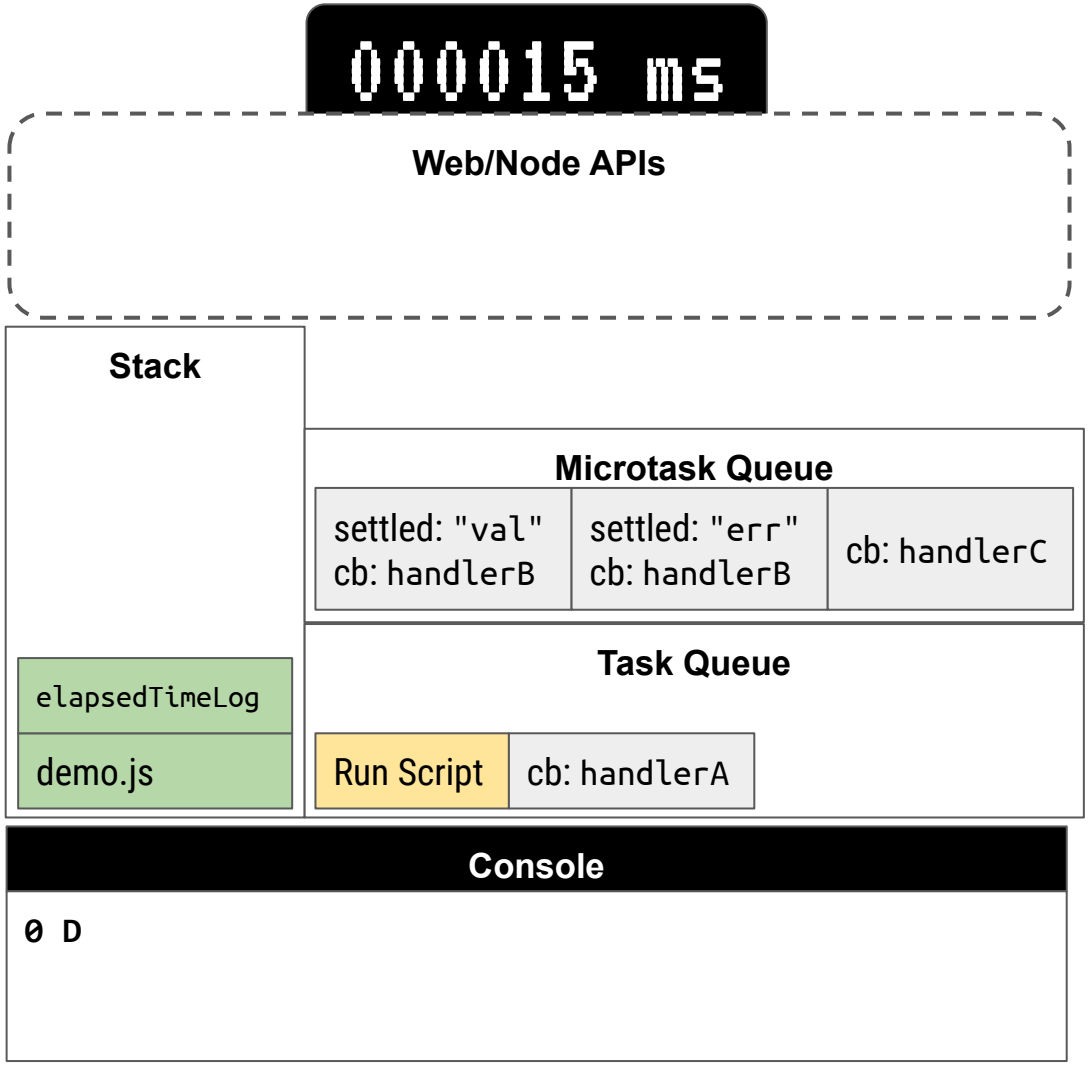
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



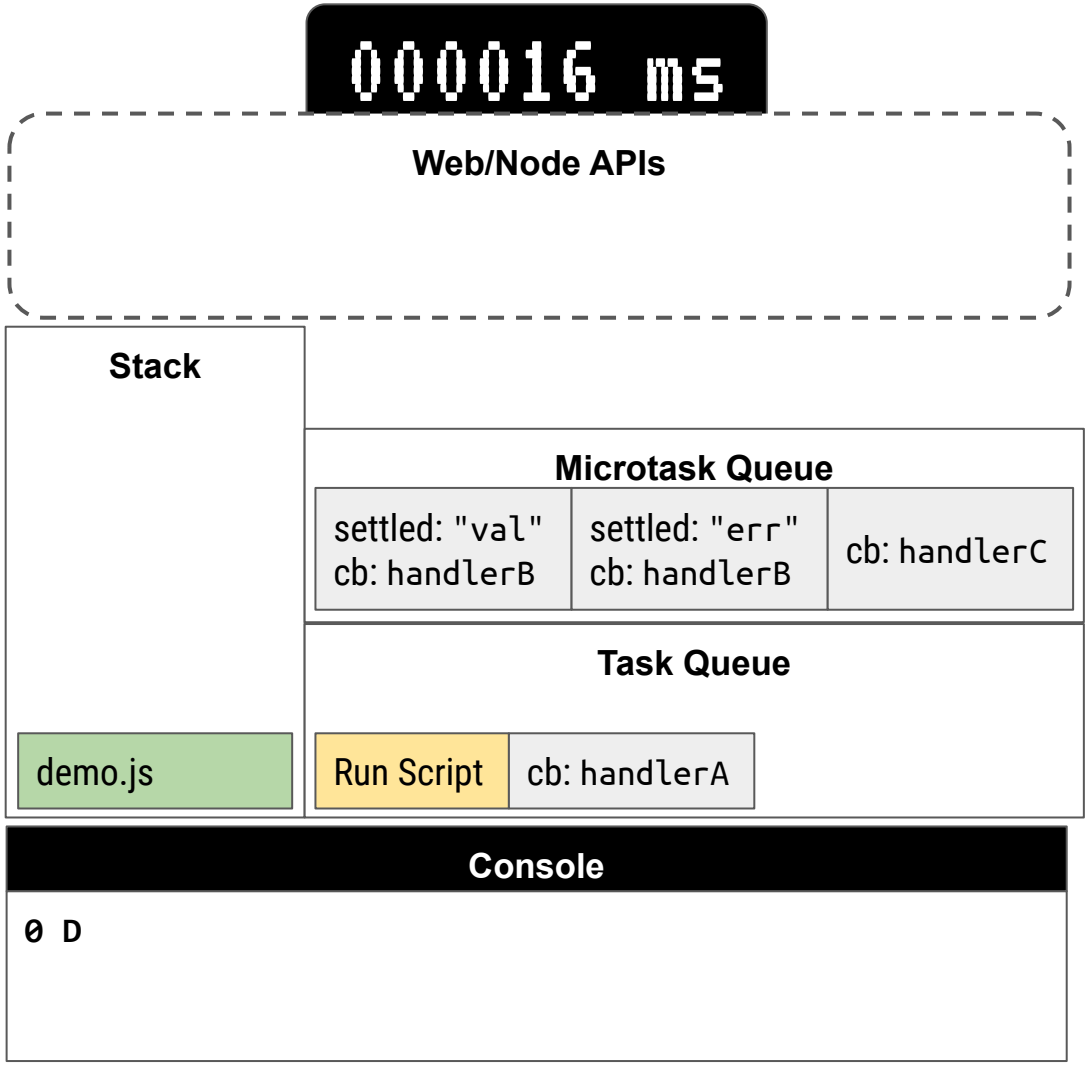
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



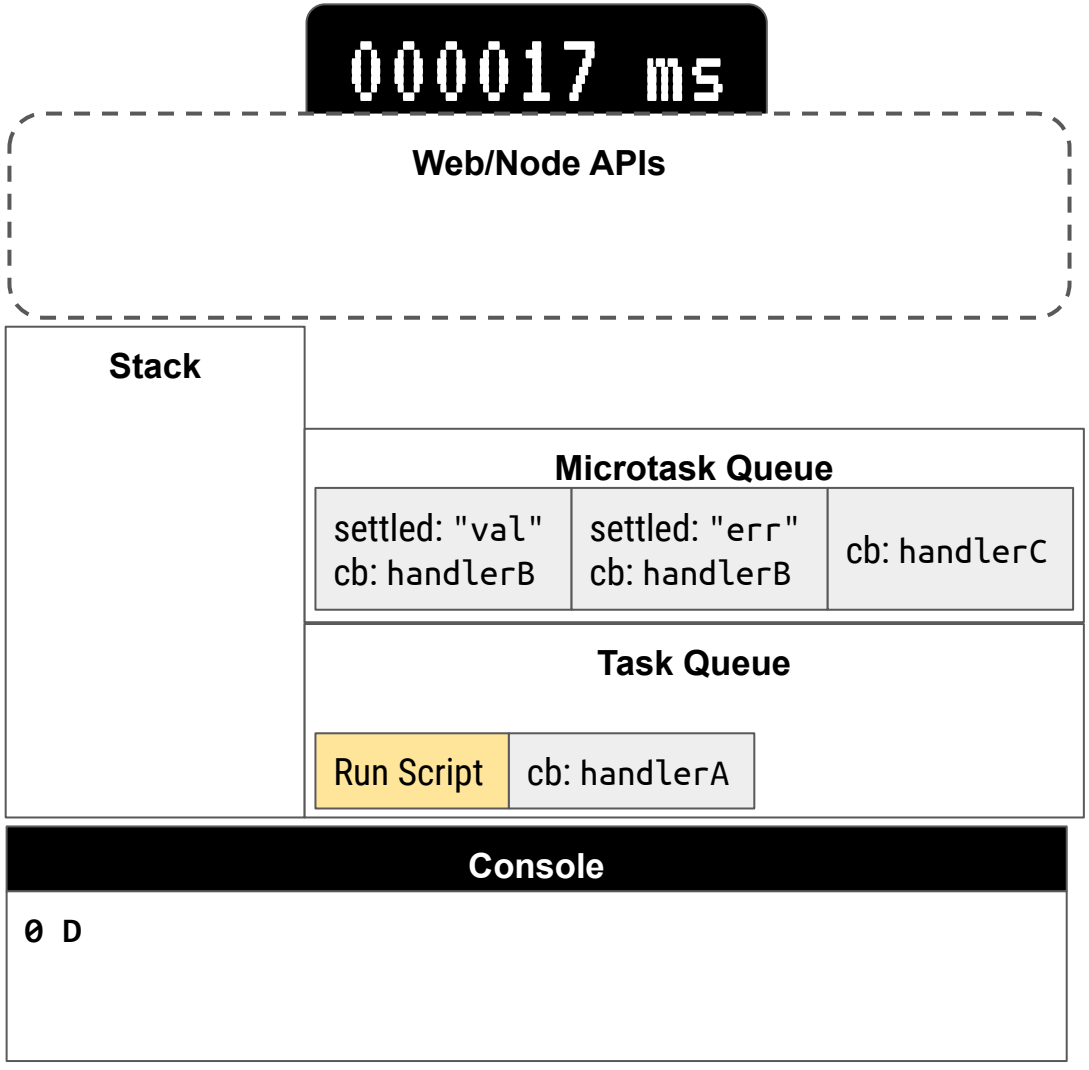
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



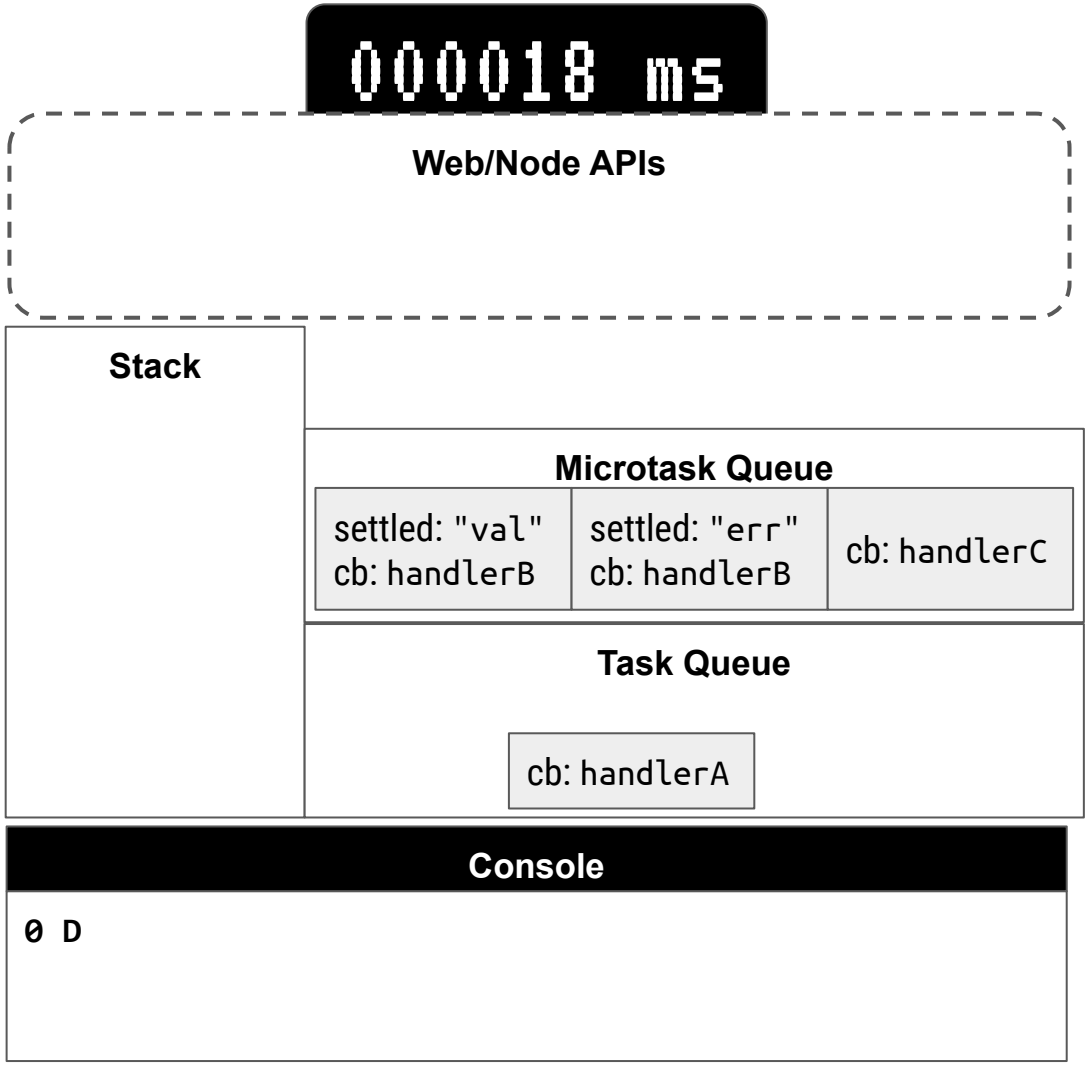
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



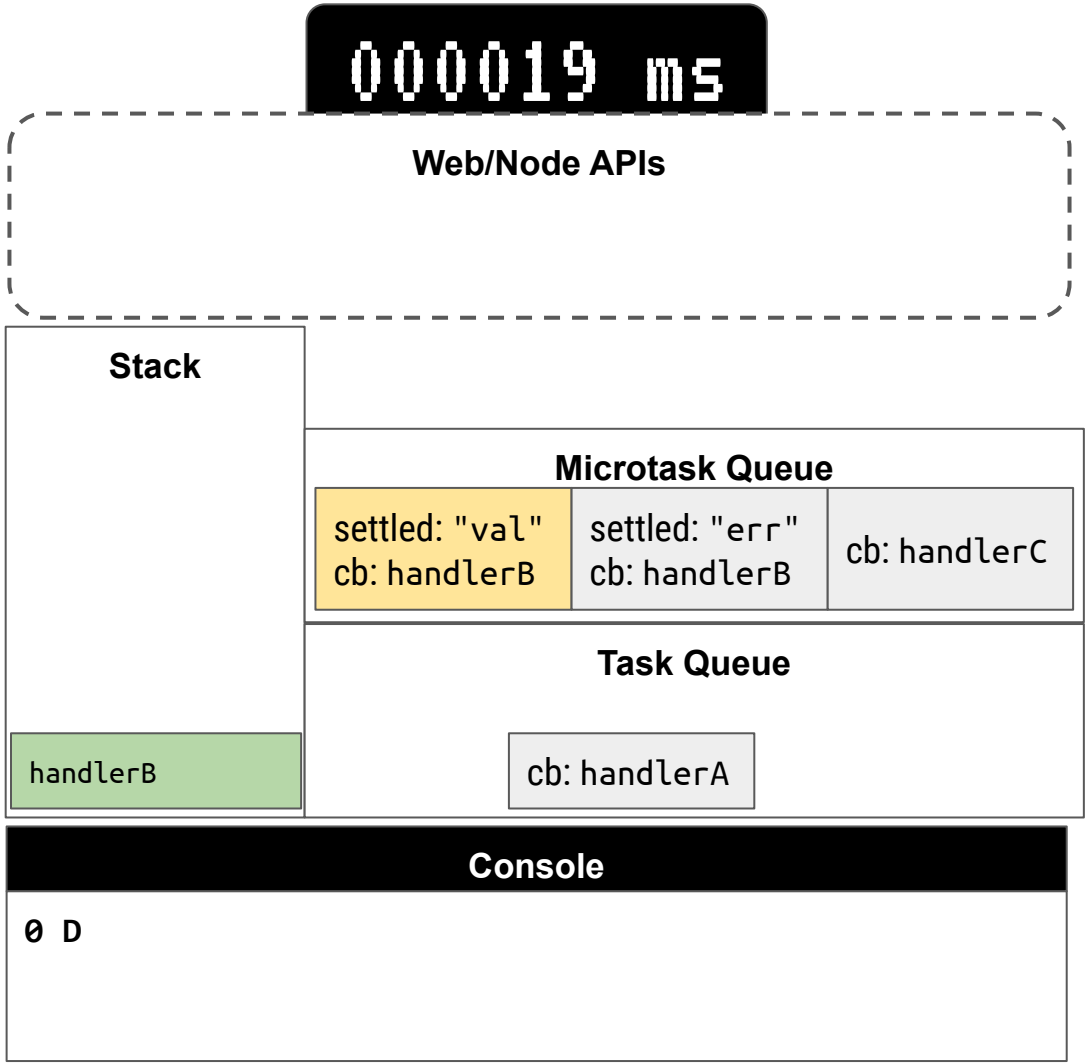
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



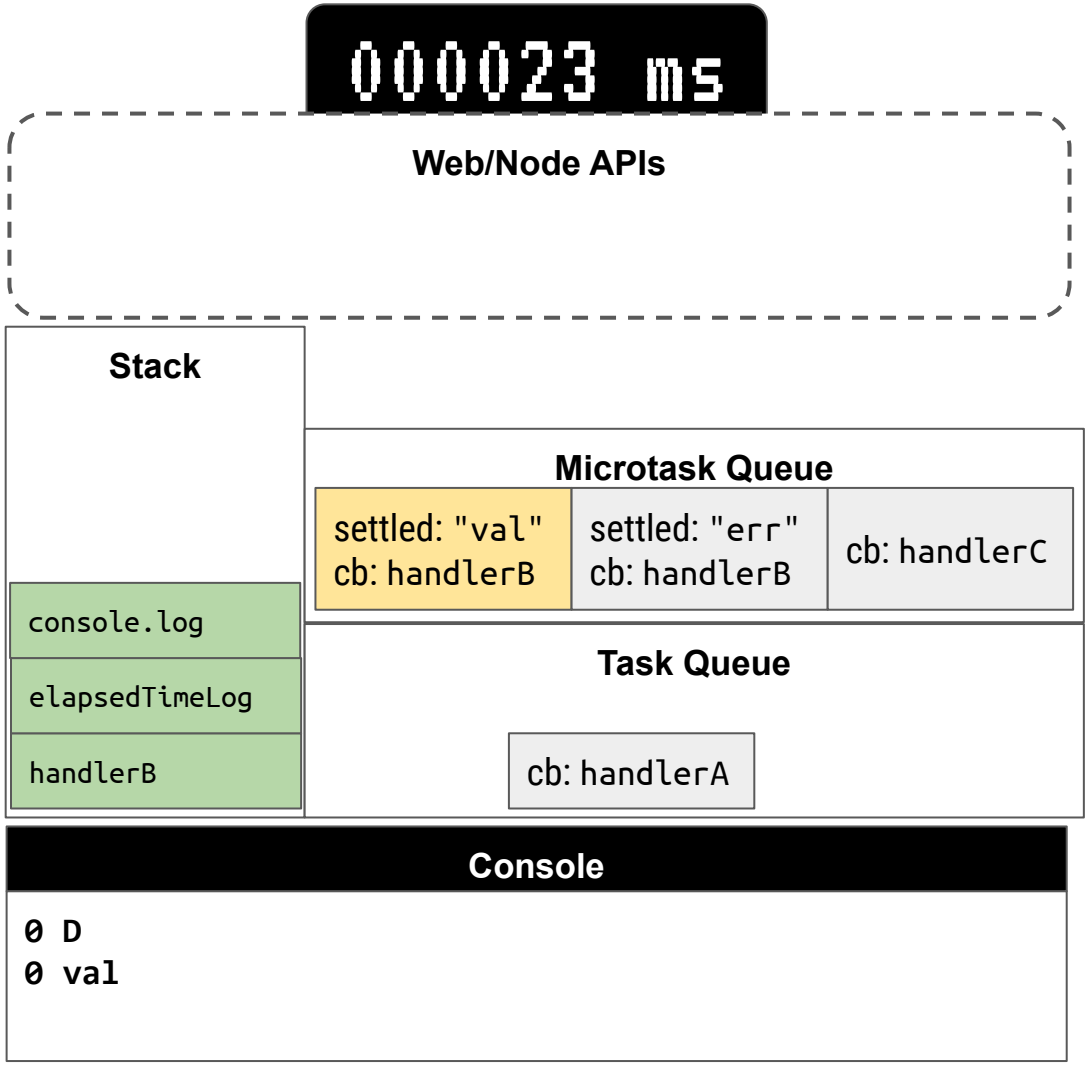
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



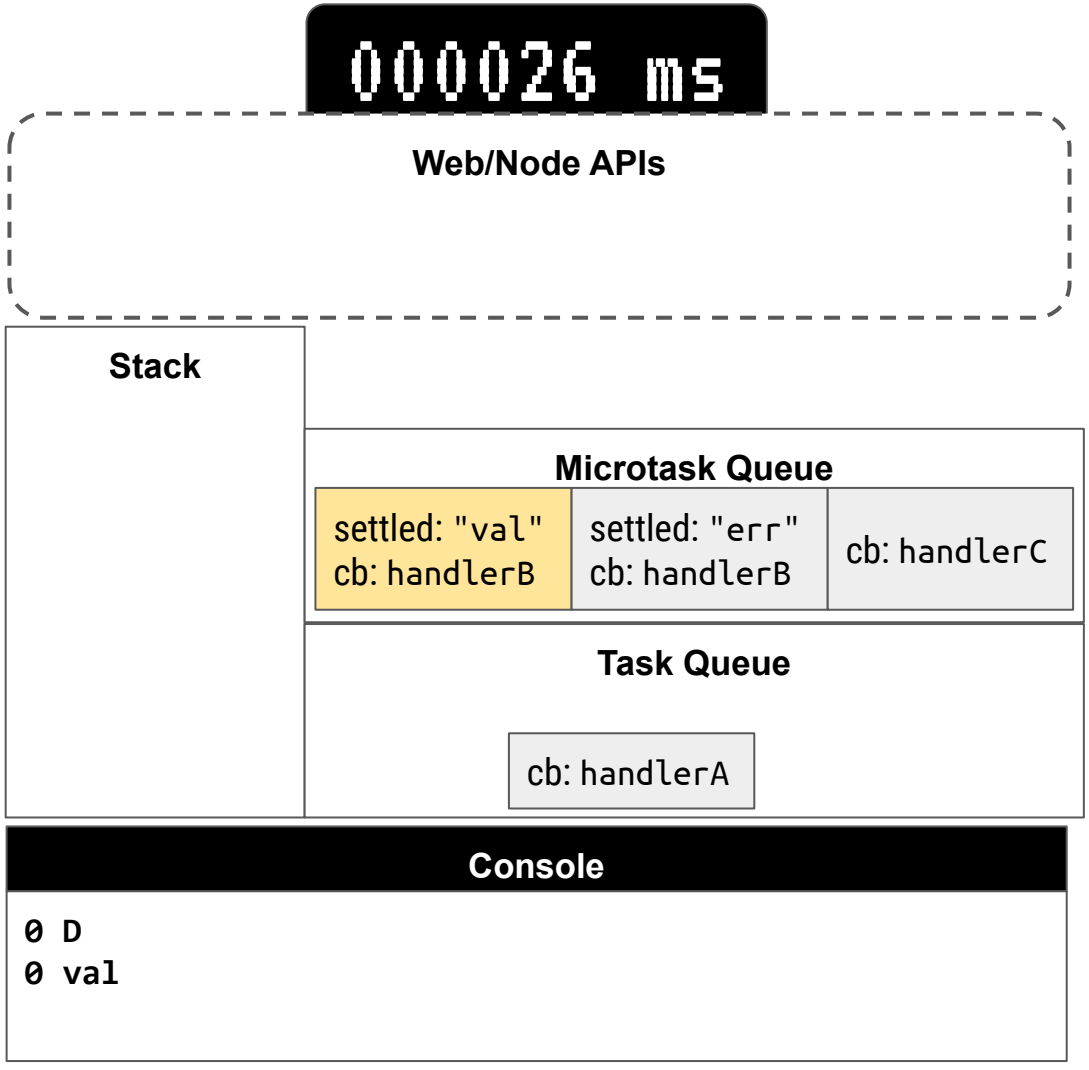
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



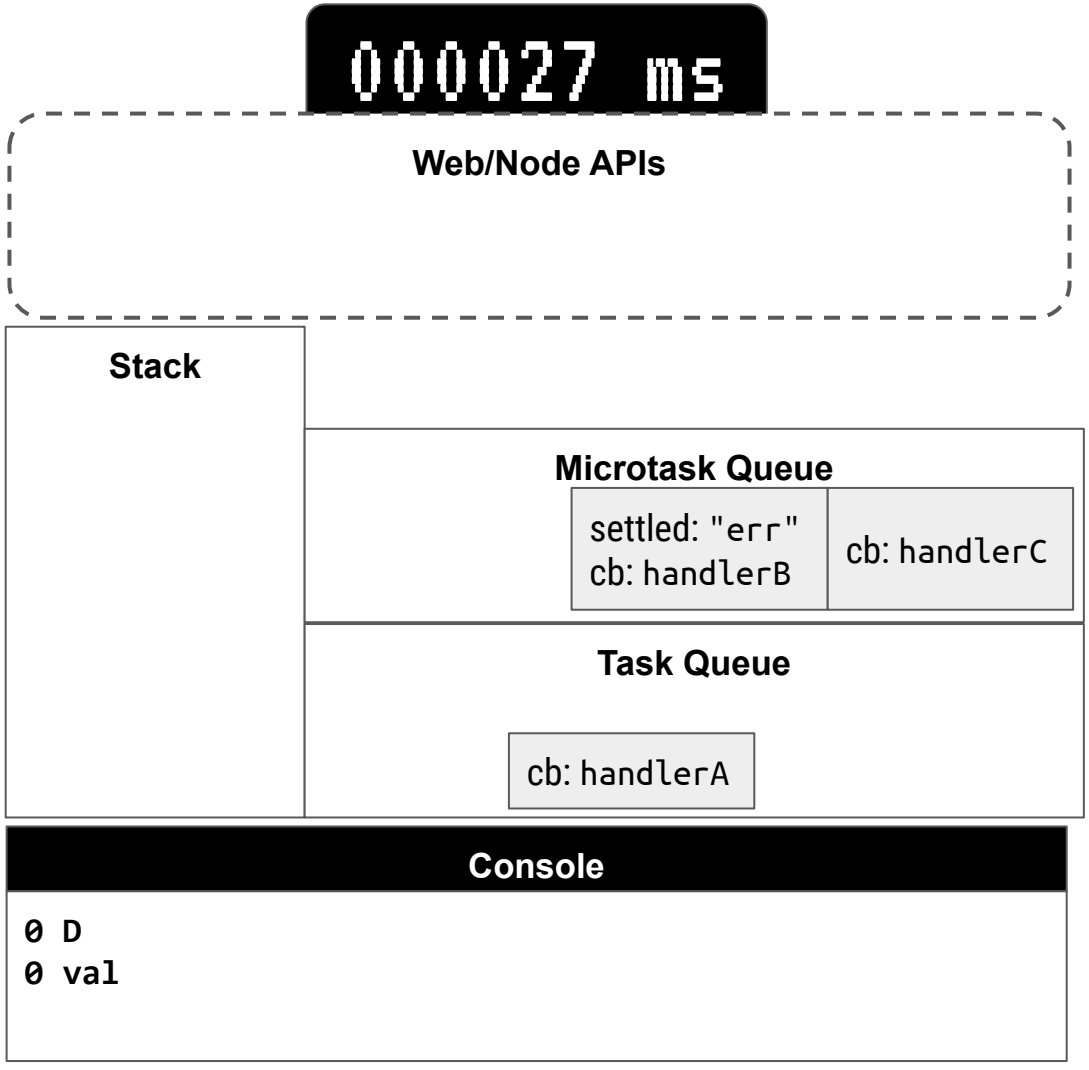
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



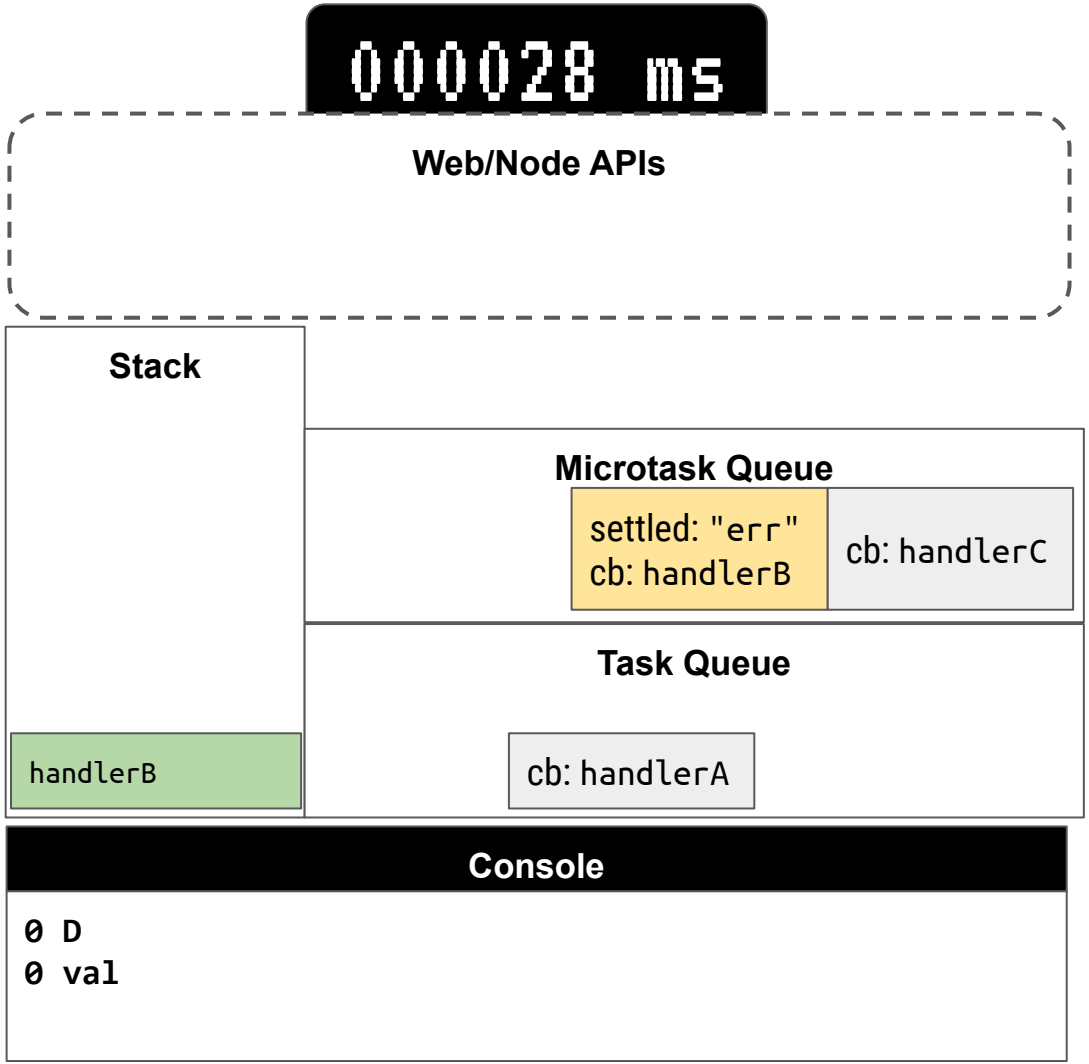
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



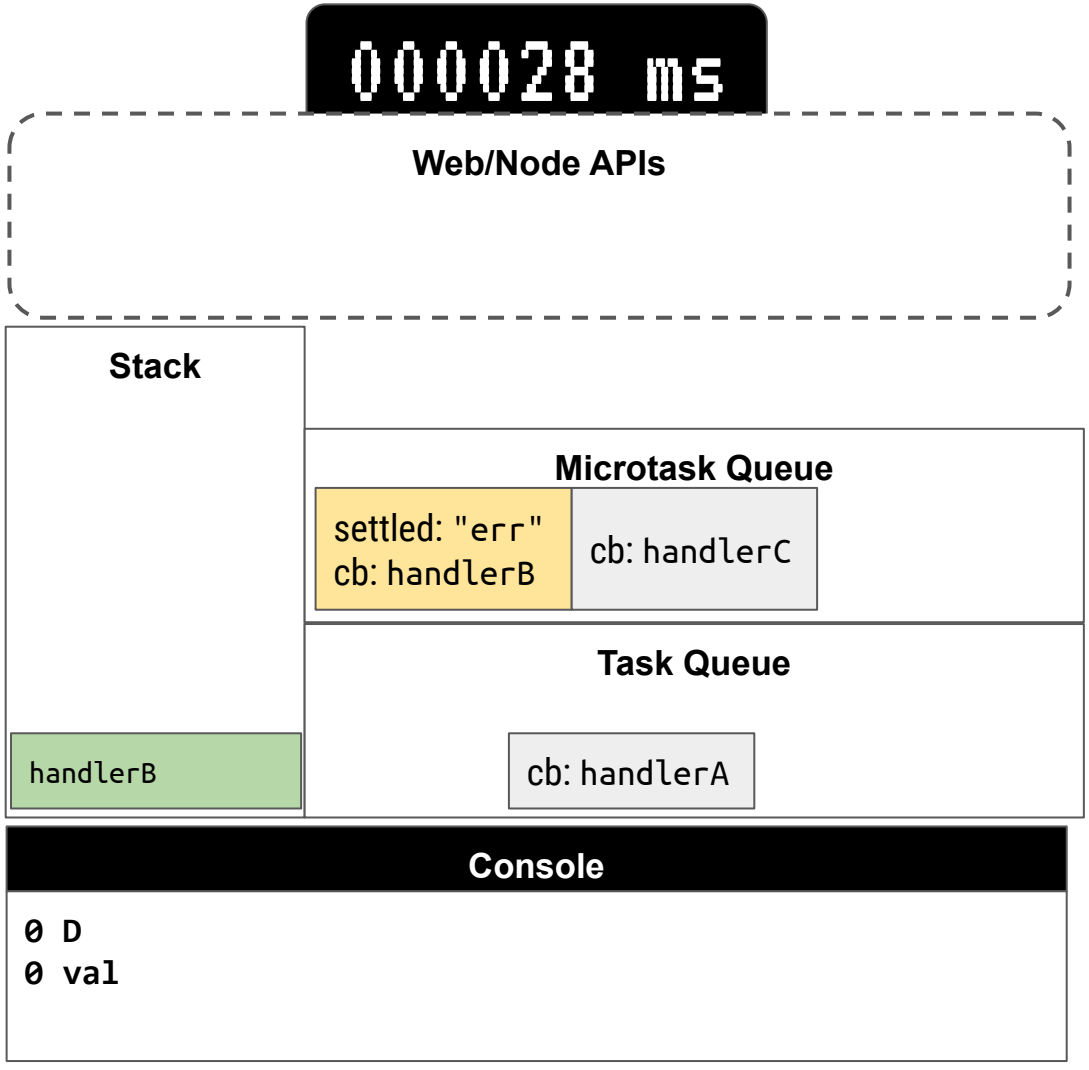
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



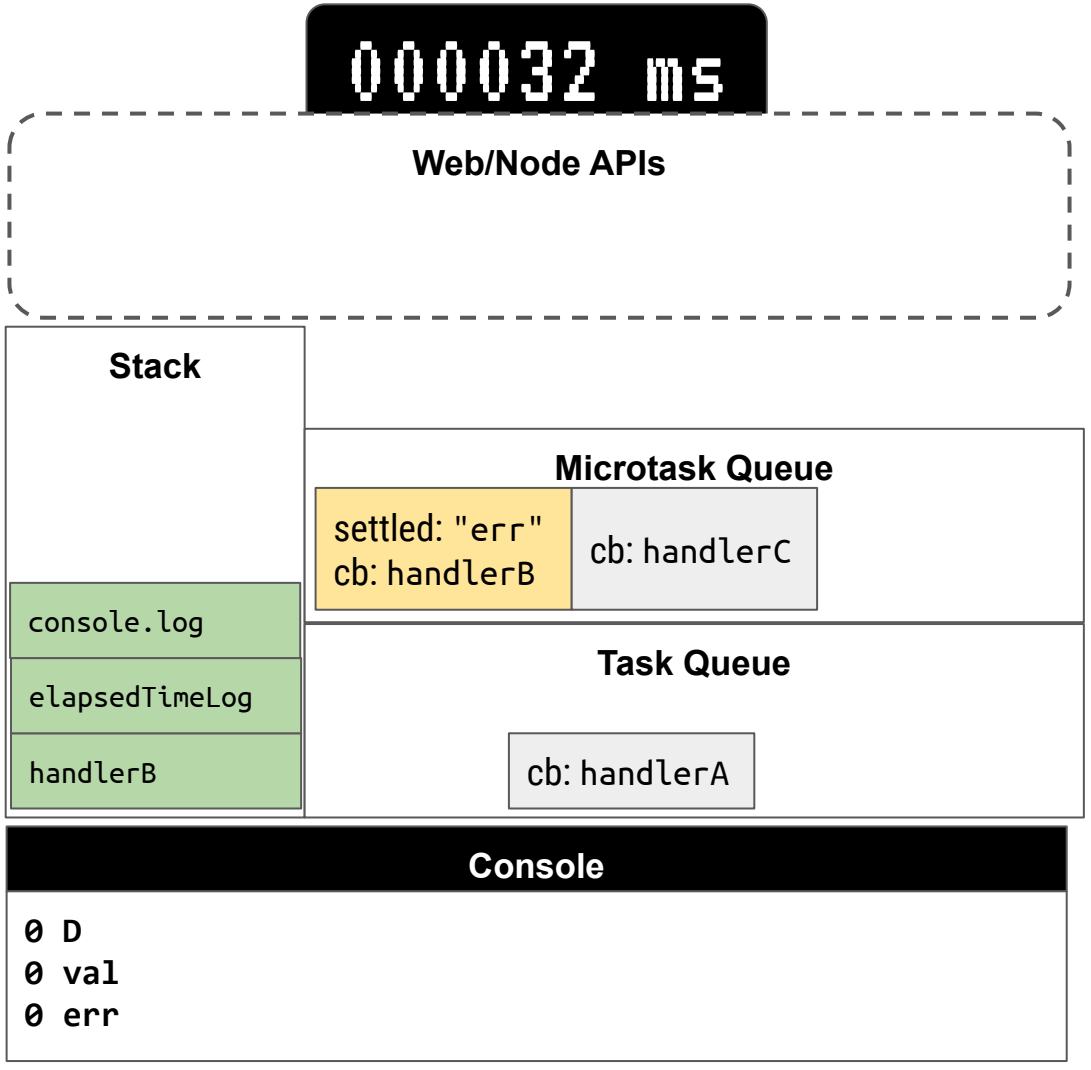
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



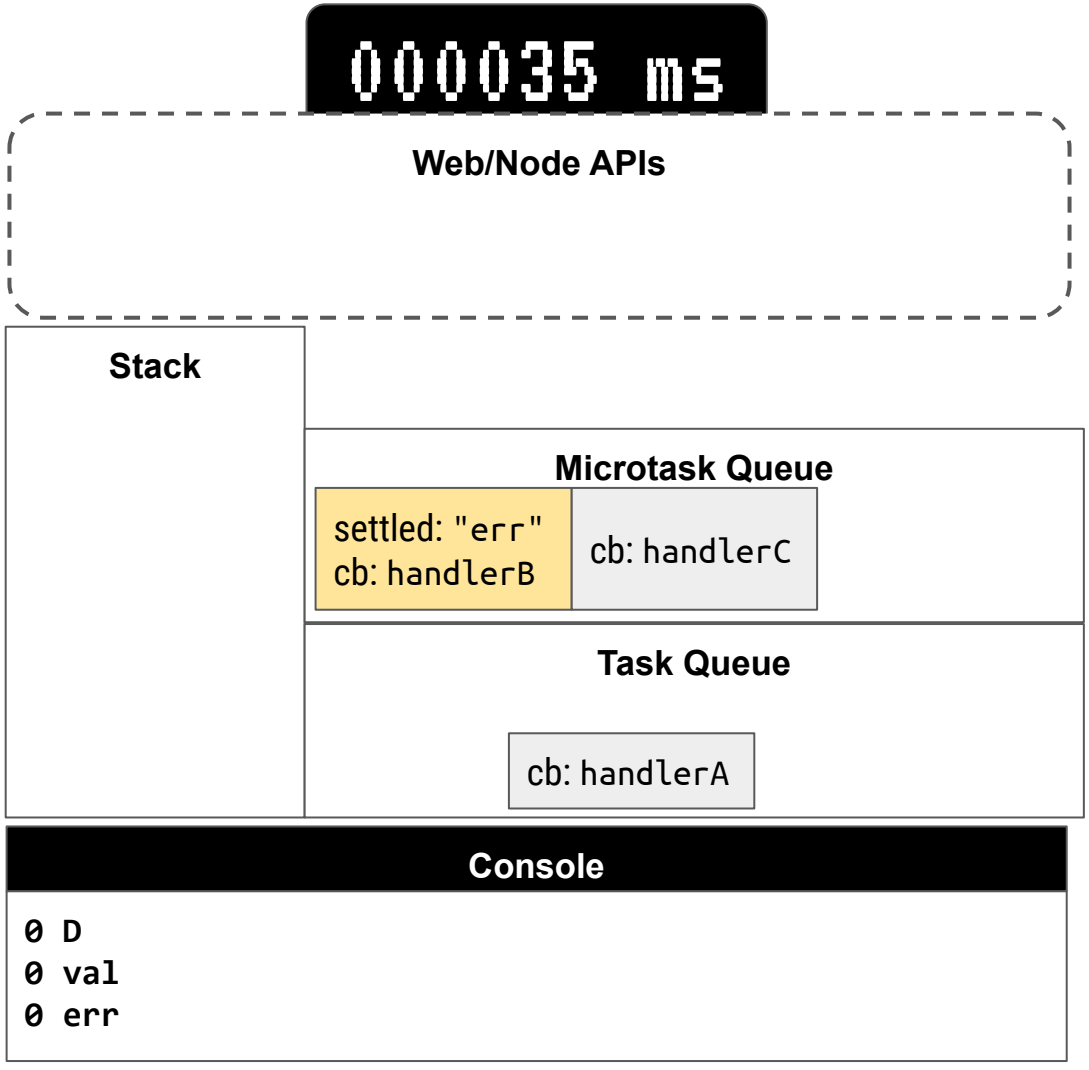
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



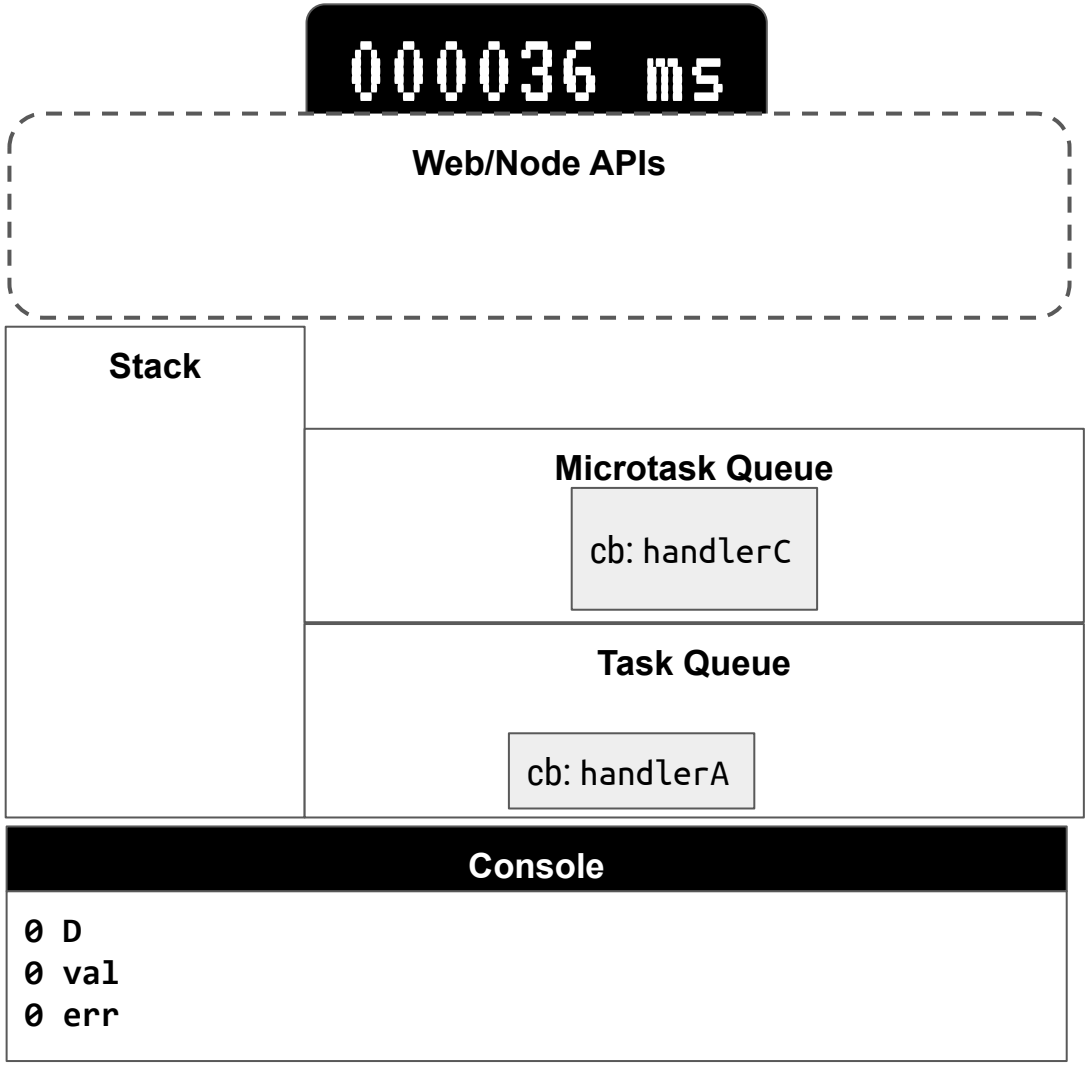
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



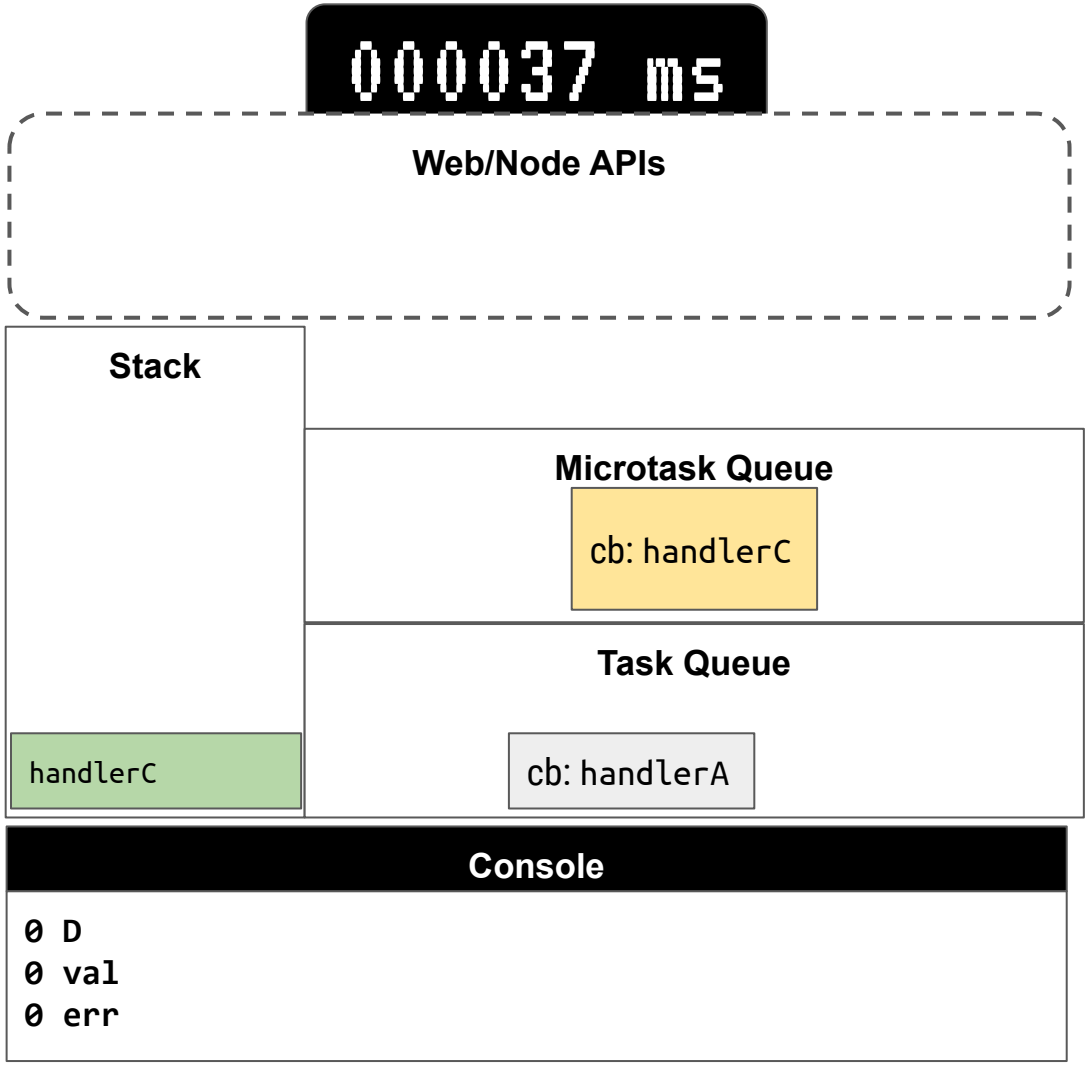
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



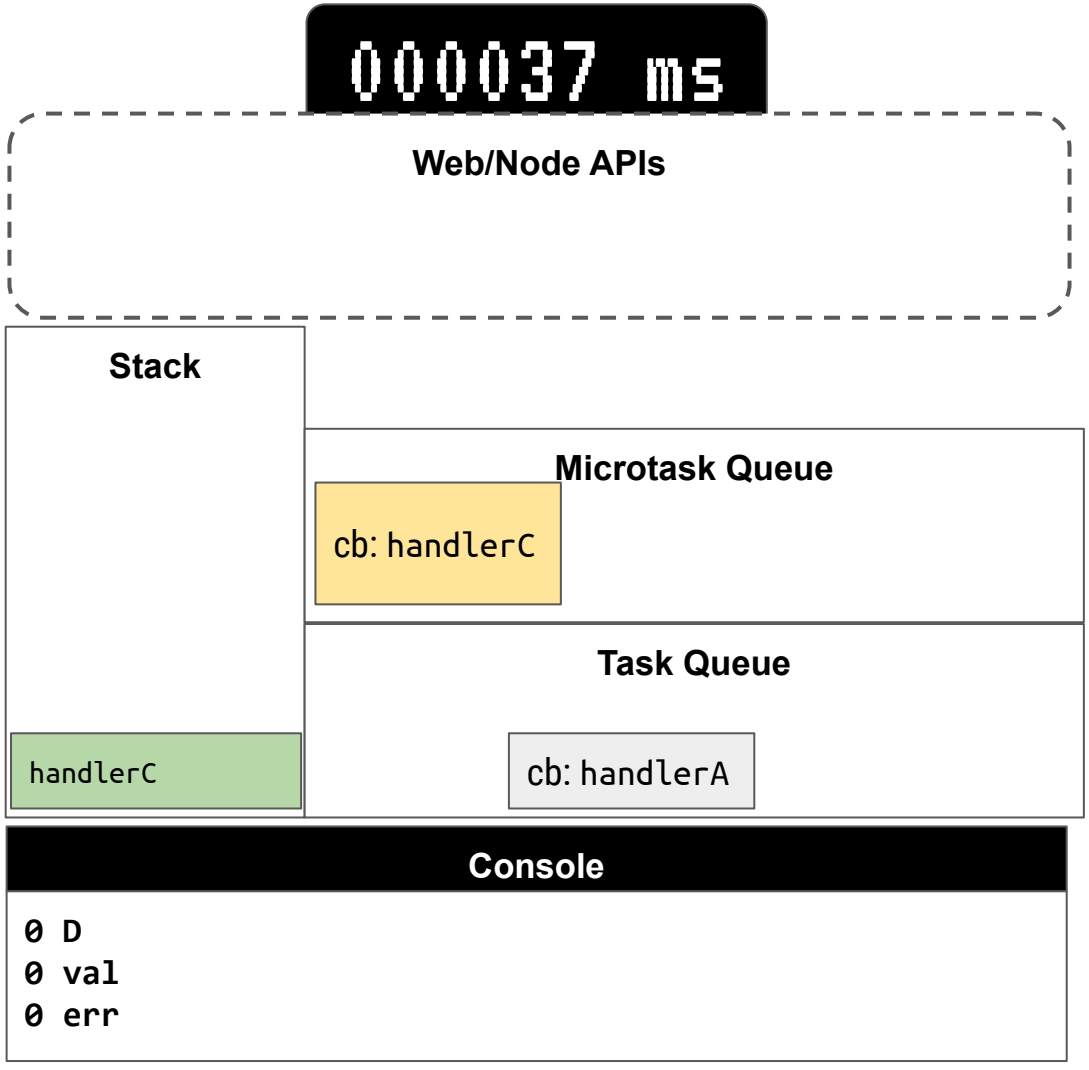
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



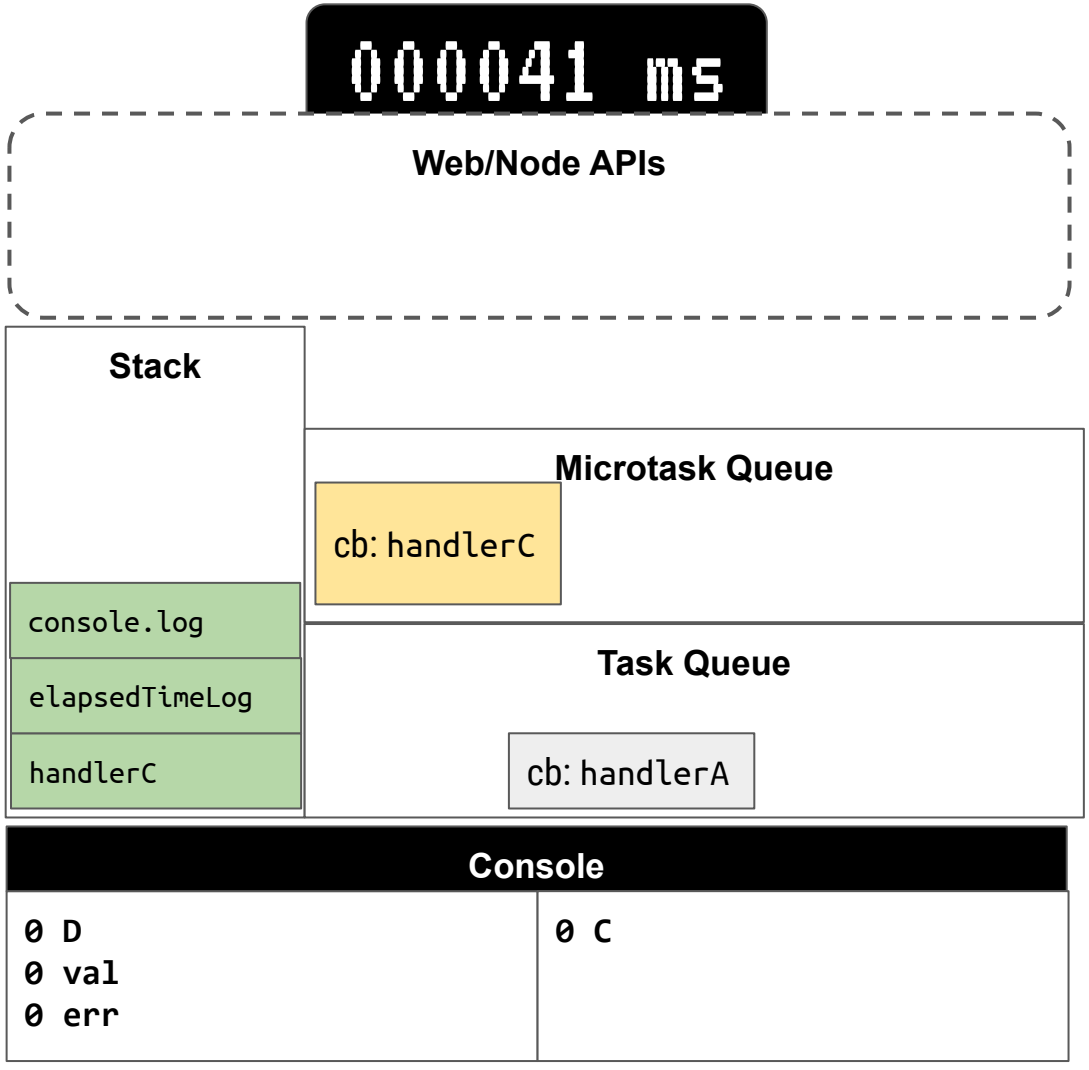
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



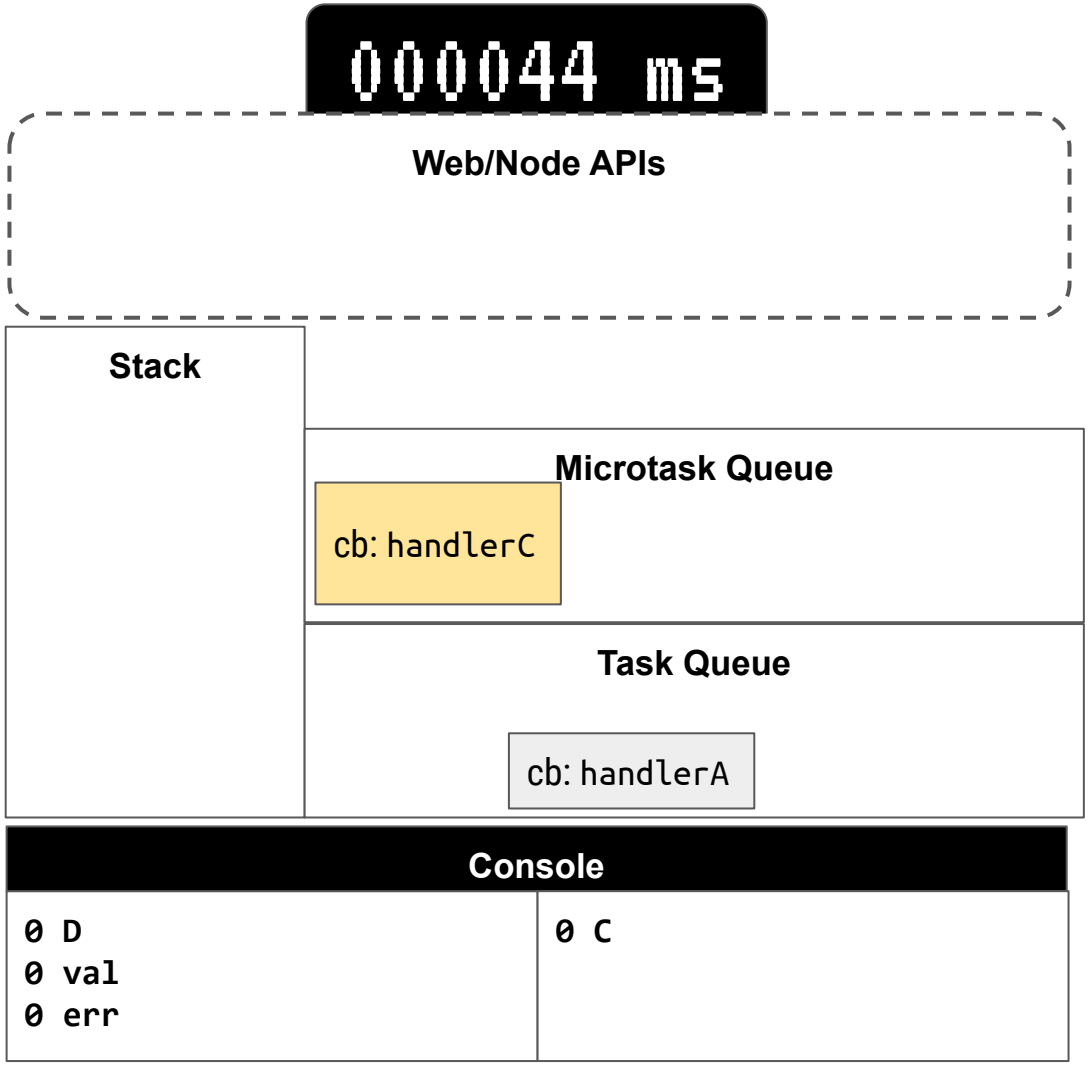
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



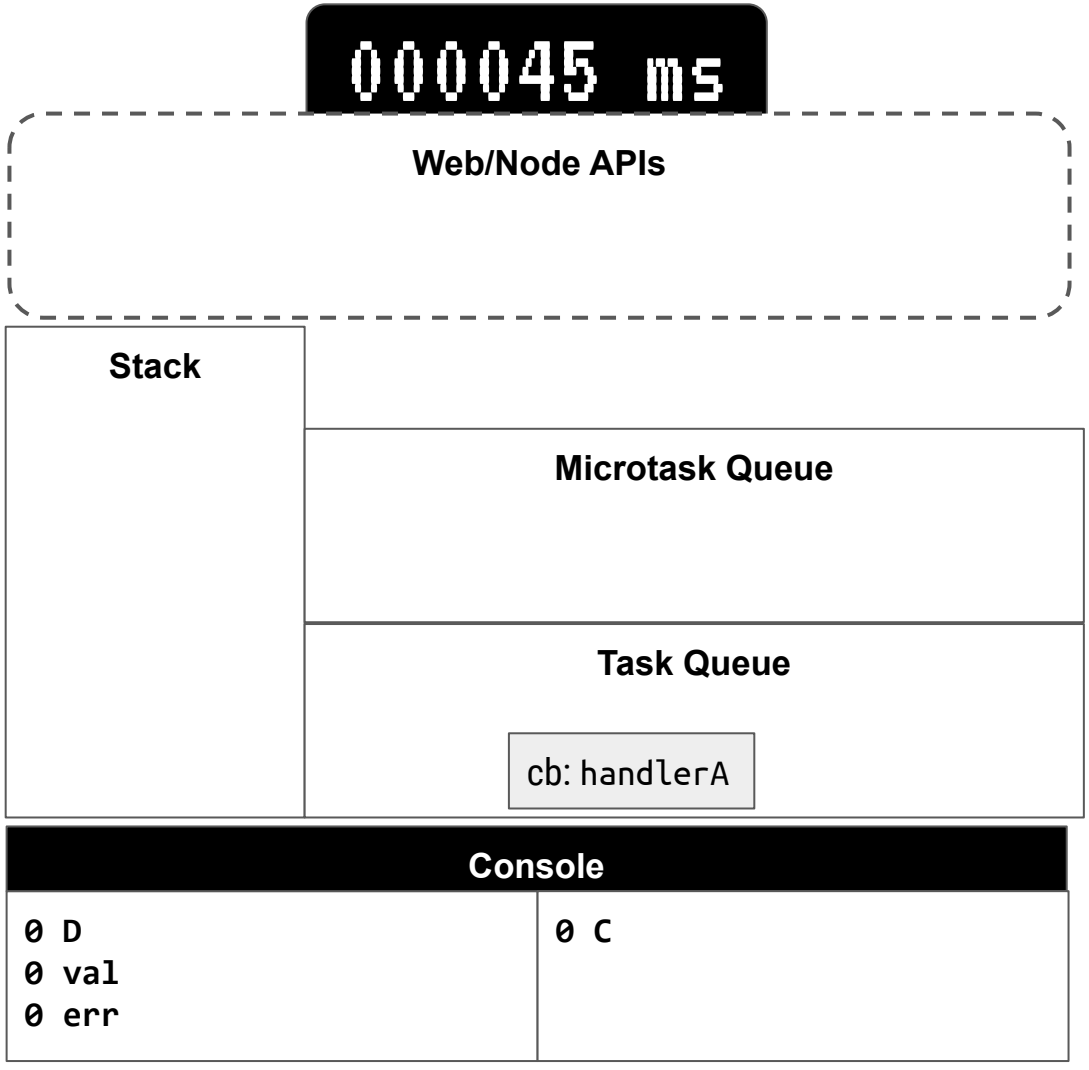
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



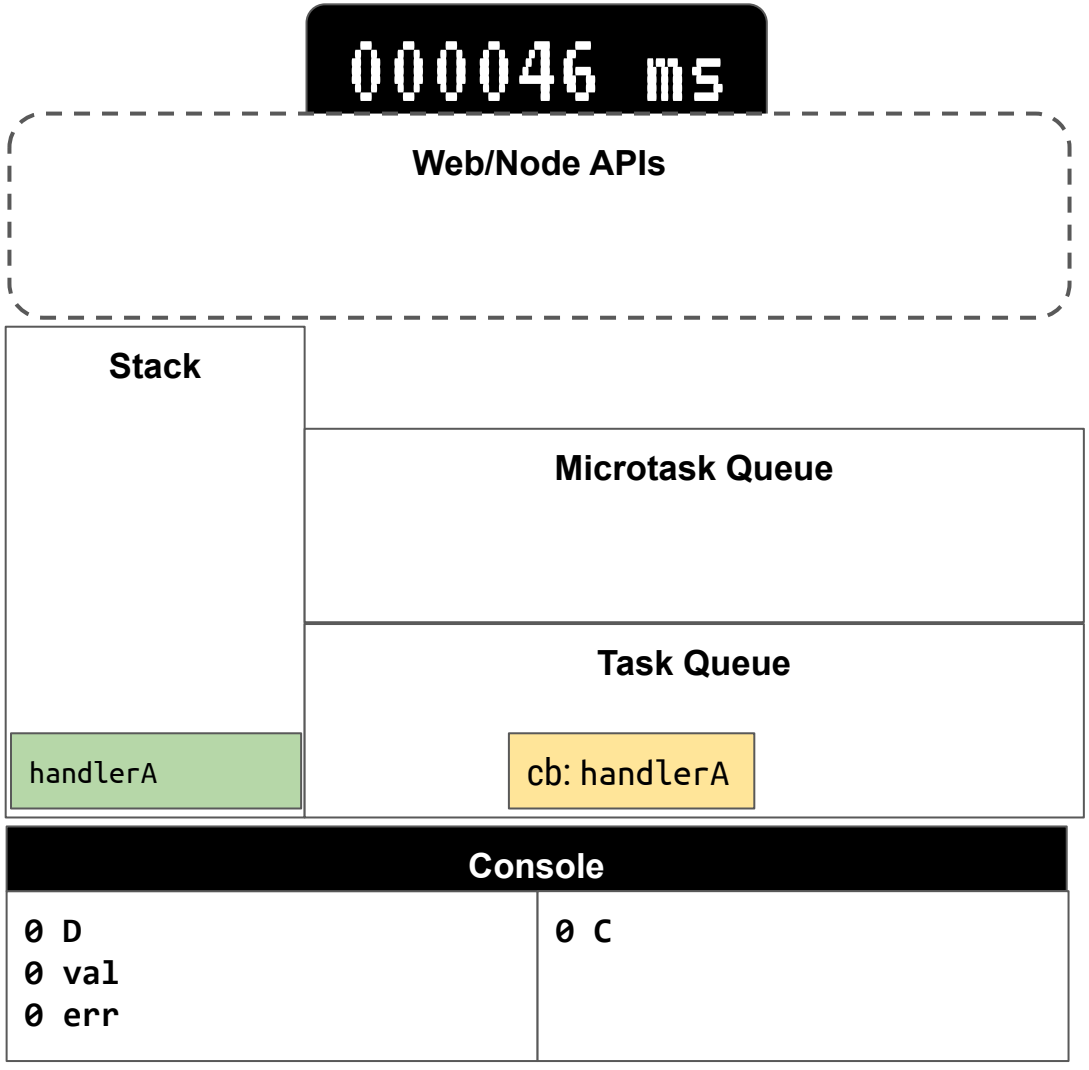
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



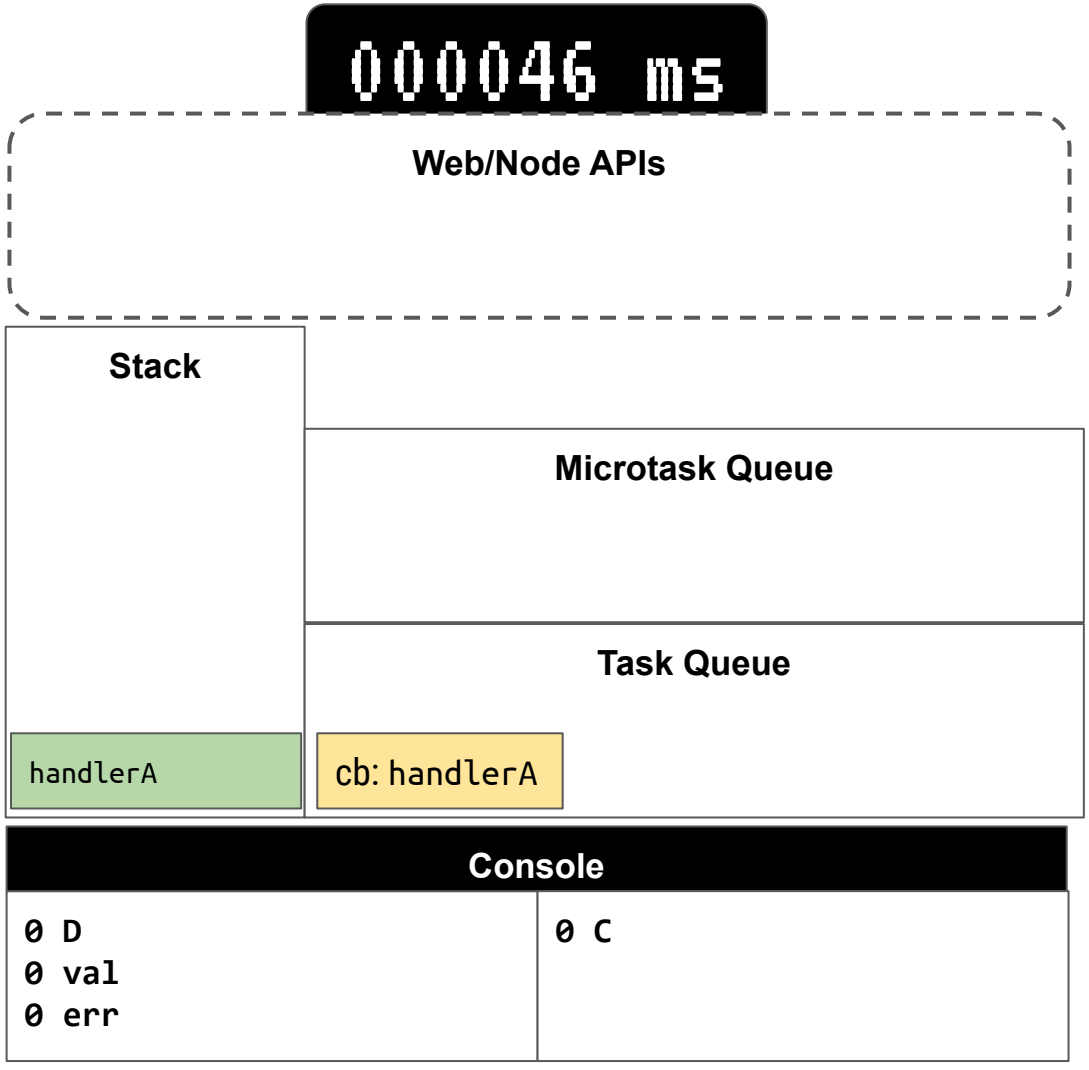
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



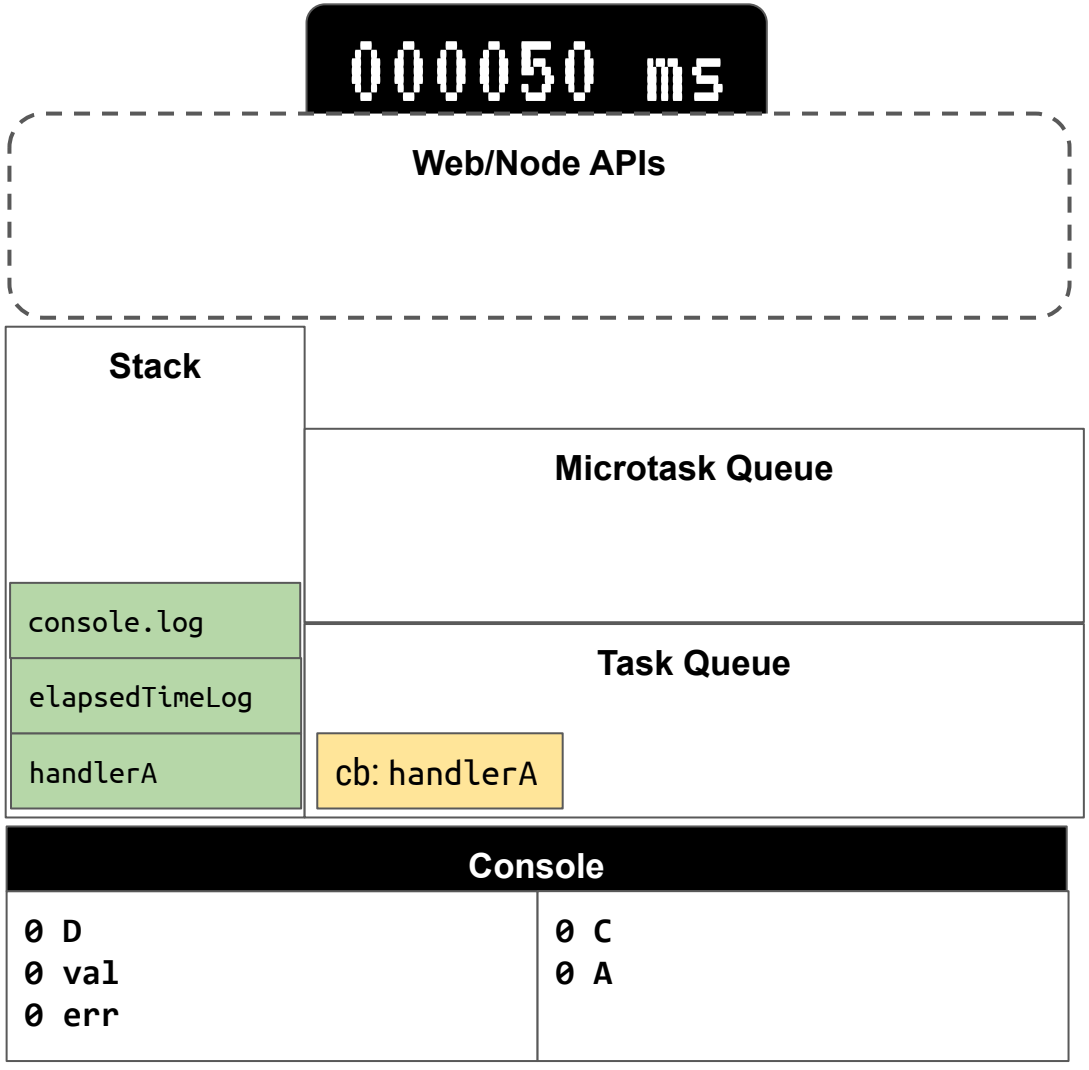
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



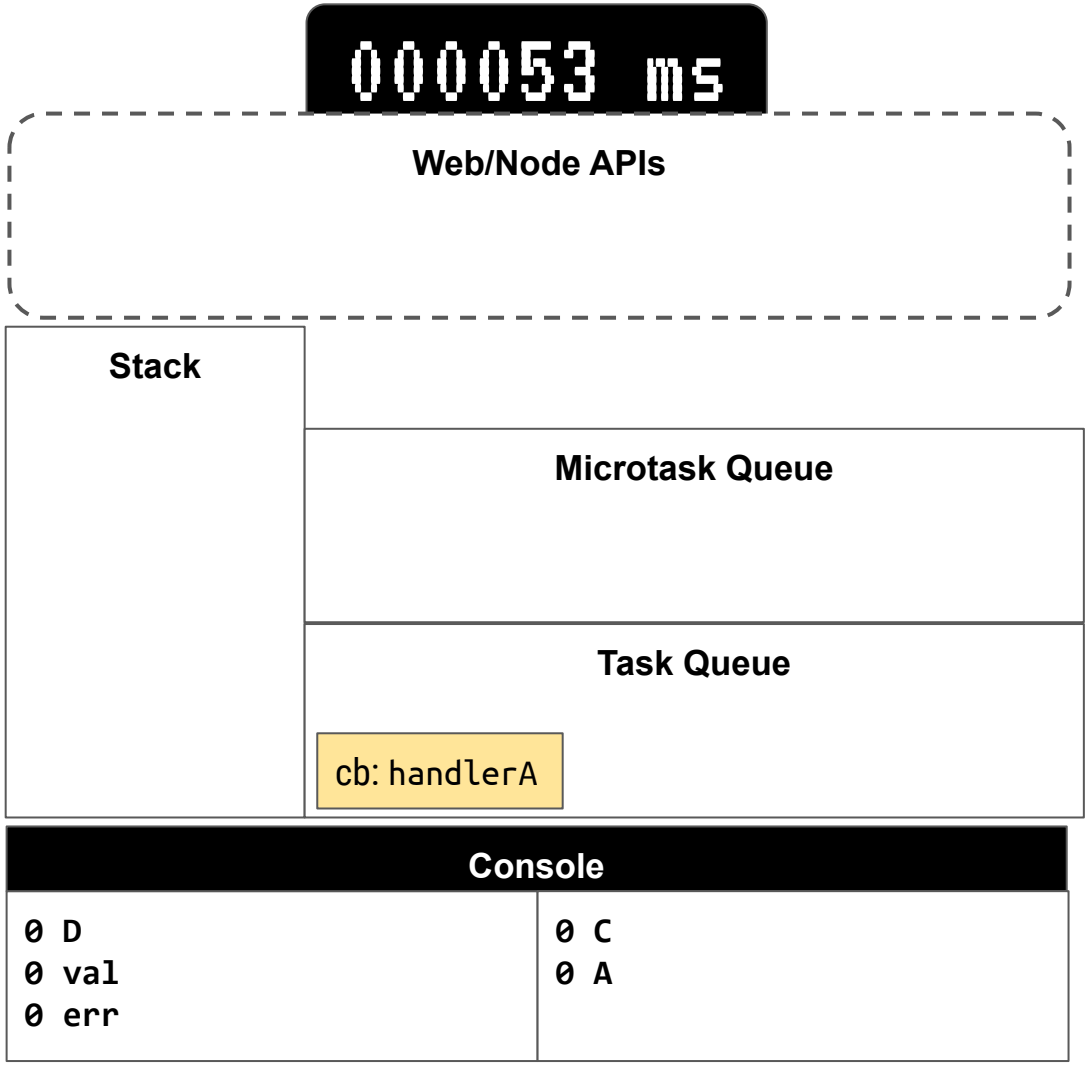
```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```



```
01: const start = Date.now();
02: function elapsedTimeLog(...args) {
03:   const elapsedMs = Date.now() - start;
04:   const sec = Math.floor(elapsedMs / 1000);
05:   console.log(sec, ...args);
06: }
07:
08: function handlerA() {
09:   elapsedTimeLog("A");
10: }
11:
12: function handlerB(...args) {
13:   elapsedTimeLog(...args);
14: }
15:
16: function handlerC() {
17:   elapsedTimeLog("C");
18: }
19:
20: setTimeout(handlerA, 0);
21: Promise.resolve("val").then(handlerB);
22: Promise.reject("err").catch(handlerB);
23: queueMicrotask(handlerC);
24: elapsedTimeLog("D");
```