DEBUGGER!

WHAT IS IT GOOD FOR?

Stanislav Párnický @ stylers.cloud

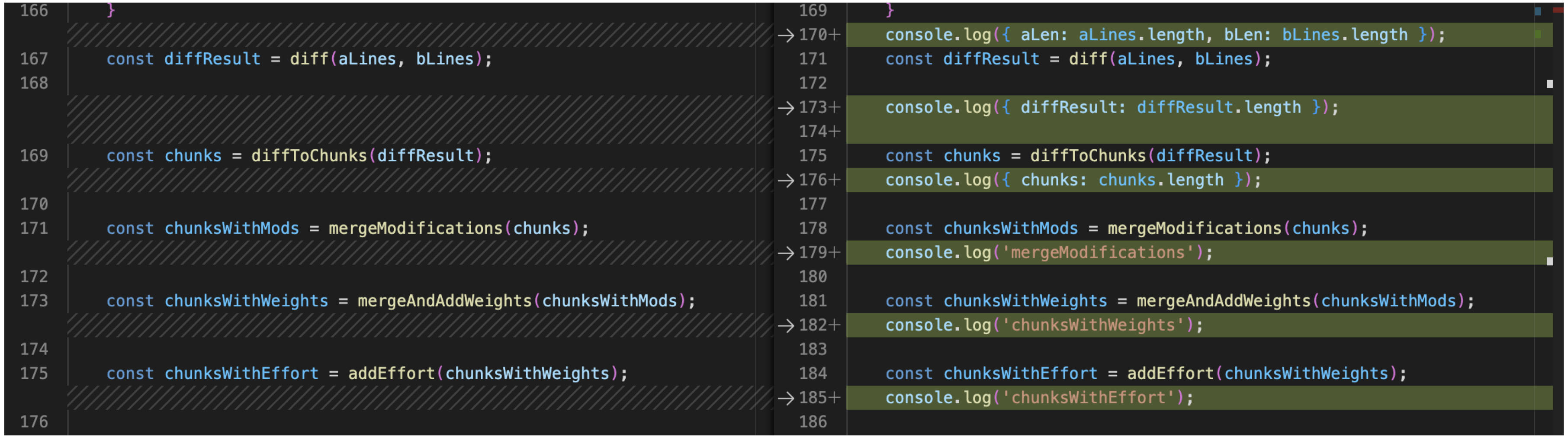
DEBUGGER

- inpect code execution
- show call stack
- read variables
- REPL

POLL: HOW DO YOU DEBUG?

1. temporary `console.log()`
2. write more tests
3. staring at code and thinking really hard
4. use debugger

1. TEMPORARY `console.log`

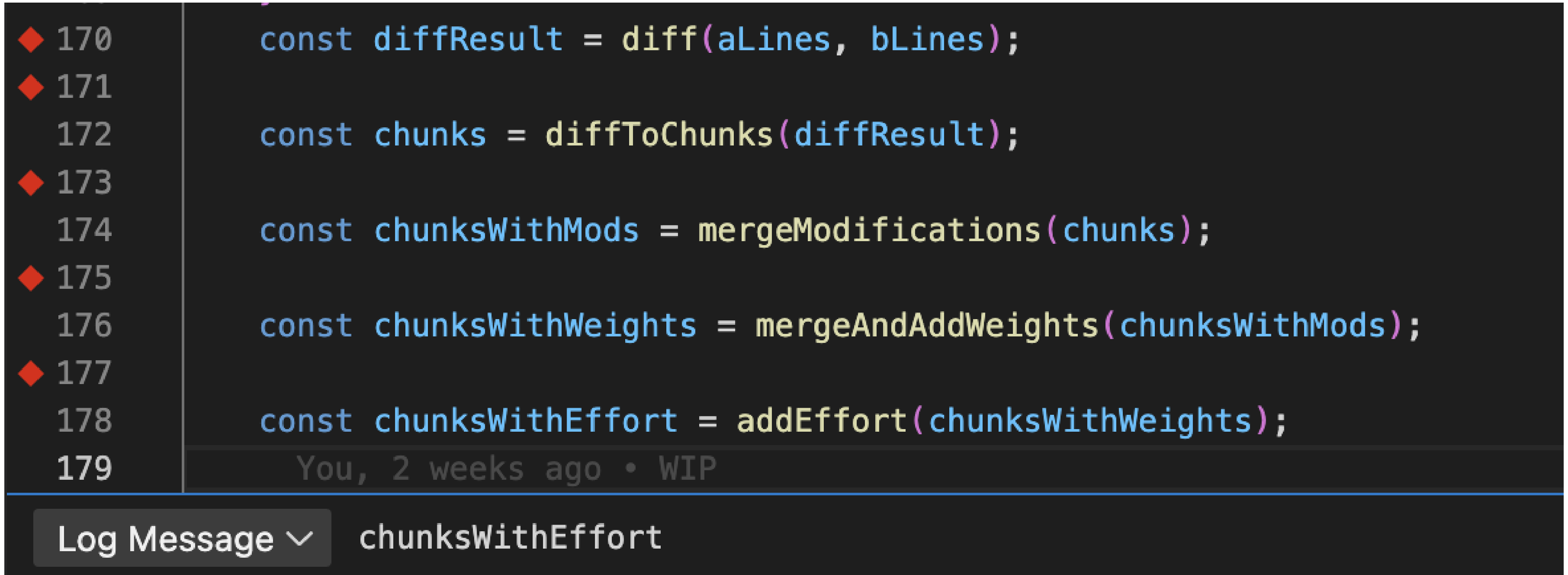


```
  }

  const diffResult = diff(aLines, bLines);


  const chunks = diffToChunks(diffResult);

  const chunksWithMods = mergeModifications(chunks);

  const chunksWithWeights = mergeAndAddWeights(chunksWithMods);

  const chunksWithEffort = addEffort(chunksWithWeights);

```

```
  }
  console.log({ aLen: aLines.length, bLen: bLines.length });
  const diffResult = diff(aLines, bLines);

  console.log({ diffResult: diffResult.length });

  const chunks = diffToChunks(diffResult);
  console.log({ chunks: chunks.length });

  const chunksWithMods = mergeModifications(chunks);
  console.log('mergeModifications');

  const chunksWithWeights = mergeAndAddWeights(chunksWithMods);
  console.log('chunksWithWeights');

  const chunksWithEffort = addEffort(chunksWithWeights);
  console.log('chunksWithEffort');

```

I will just add few console.logs...

LOGPOINTS



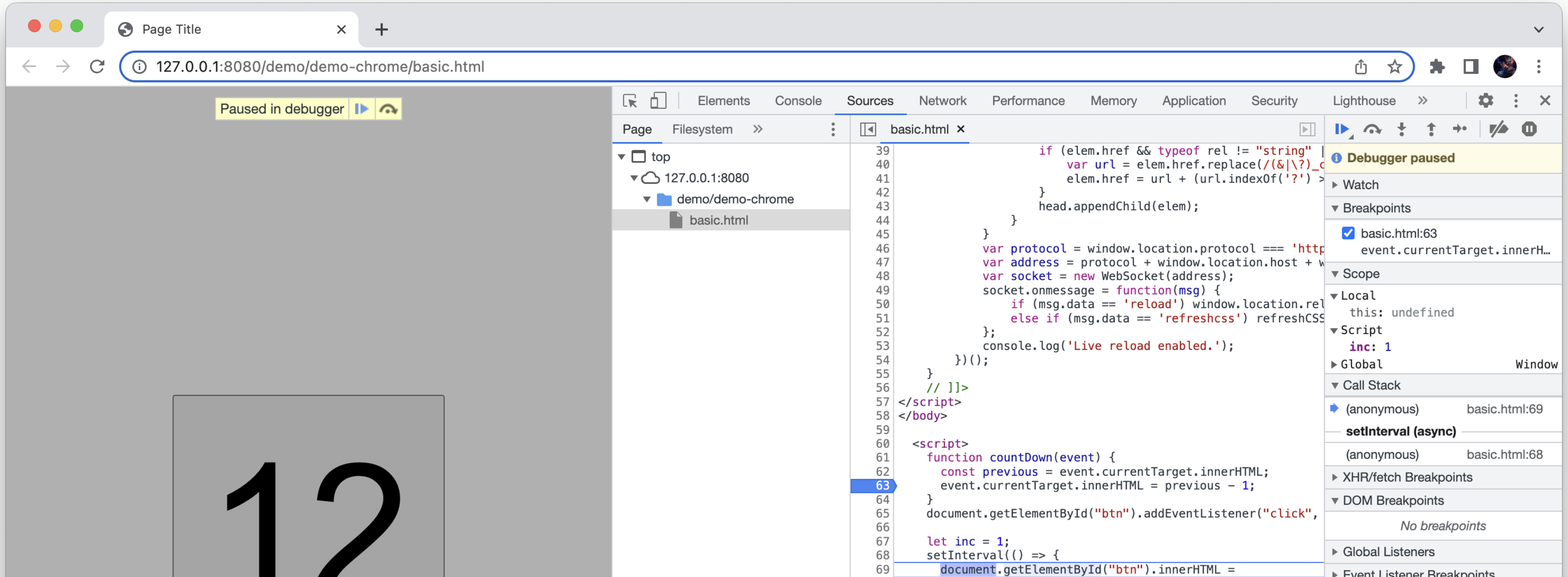
```
    const diffResult = diff(aLines, bLines);

    const chunks = diffToChunks(diffResult);

    const chunksWithMods = mergeModifications(chunks);

    const chunksWithWeights = mergeAndAddWeights(chunksWithMods);

    const chunksWithEffort = addEffort(chunksWithWeights);
      You, 2 weeks ago • WIP
```

Log Message chunksWithEffort

SO HOW DO I USE THIS DEBUGGER THING?

1. Open DevTools Sources tab
2. ???
3. Profit!!!

Demo in browser



```
                    if (elem.href && typeof rel != "string" |
                        var url = elem.href.replace(/(&|\?)_c
                        elem.href = url + (url.indexOf('?') >
                    }
                    head.appendChild(elem);
                }
            }
            var protocol = window.location.protocol === 'http
            var address = protocol + window.location.host + w
            var socket = new WebSocket(address);
            socket.onmessage = function(msg) {
                if (msg.data == 'reload') window.location.rel
                else if (msg.data == 'refreshcss') refreshCSS
            };
            console.log('Live reload enabled.');
        })();
    }
    // ]]>
</script>
</body>

  <script>
    function countDown(event) {
      const previous = event.currentTarget.innerHTML;
      event.currentTarget.innerHTML = previous - 1;
    }
    document.getElementById("btn").addEventListener("click",

    let inc = 1;
    setInterval(() => {
      document.getElementById("btn").innerHTML =
```

☑ basic.html:63
event.currentTarget.innerH…

```
▼ Local
  this: undefined
▼ Script
  inc: 1
▶ Global                Window
```

(anonymous)	basic.html:69
setInterval (async)	
(anonymous)	basic.html:68

2. WRITE MORE TESTS

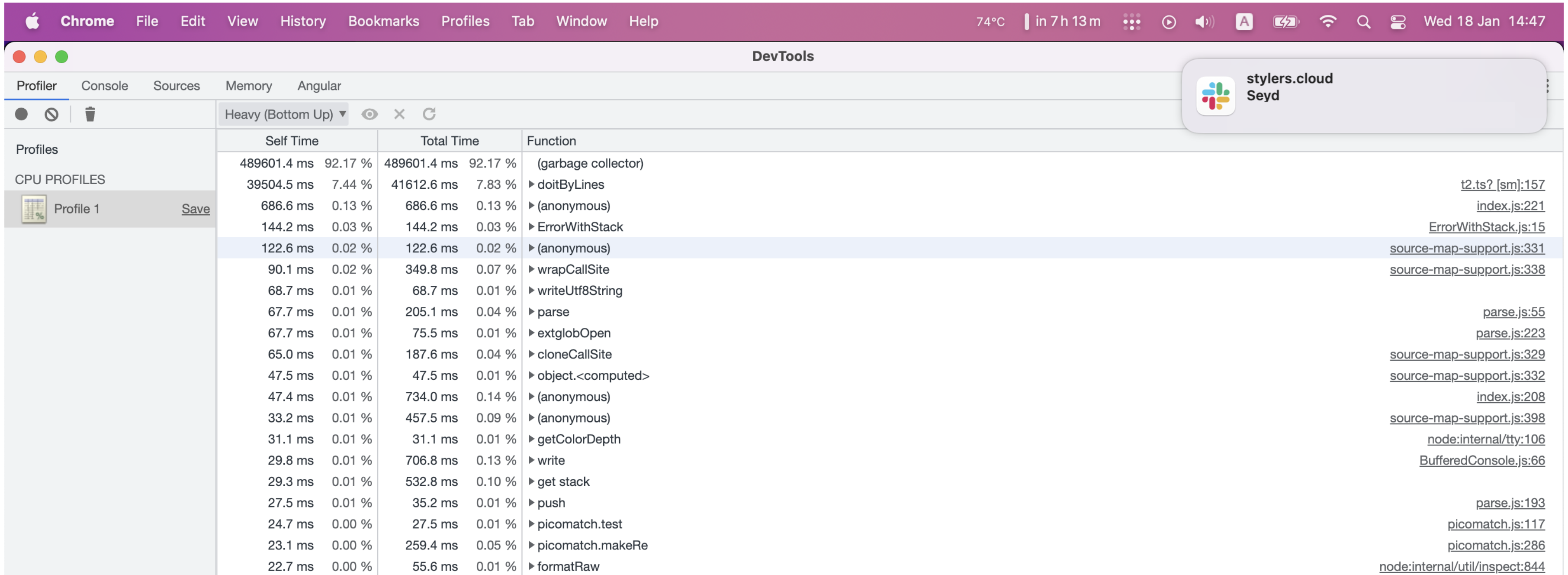
VSCODE DEBUG

demo-jest

- config launch.json
- run debug

3. STARING AT CODE AND THINKING REALLY HARD

PROFILER



Self Time		Total Time		Function	
489601.4 ms	92.17 %	489601.4 ms	92.17 %	(garbage collector)	
39504.5 ms	7.44 %	41612.6 ms	7.83 %	doitByLines	t2.ts? [sm]:157
686.6 ms	0.13 %	686.6 ms	0.13 %	(anonymous)	index.js:221
144.2 ms	0.03 %	144.2 ms	0.03 %	ErrorWithStack	ErrorWithStack.js:15
122.6 ms	0.02 %	122.6 ms	0.02 %	(anonymous)	source-map-support.js:331
90.1 ms	0.02 %	349.8 ms	0.07 %	wrapCallSite	source-map-support.js:338
68.7 ms	0.01 %	68.7 ms	0.01 %	writeUtf8String	
67.7 ms	0.01 %	205.1 ms	0.04 %	parse	parse.js:55
67.7 ms	0.01 %	75.5 ms	0.01 %	extglobOpen	parse.js:223
65.0 ms	0.01 %	187.6 ms	0.04 %	cloneCallSite	source-map-support.js:329
47.5 ms	0.01 %	47.5 ms	0.01 %	object.<computed>	source-map-support.js:332
47.4 ms	0.01 %	734.0 ms	0.14 %	(anonymous)	index.js:208
33.2 ms	0.01 %	457.5 ms	0.09 %	(anonymous)	source-map-support.js:398
31.1 ms	0.01 %	31.1 ms	0.01 %	getColorDepth	node:internal/tty:106
29.8 ms	0.01 %	706.8 ms	0.13 %	write	BufferedConsole.js:66
29.3 ms	0.01 %	532.8 ms	0.10 %	get stack	
27.5 ms	0.01 %	35.2 ms	0.01 %	push	parse.js:193
24.7 ms	0.00 %	27.5 ms	0.01 %	picomatch.test	picomatch.js:117
23.1 ms	0.00 %	259.4 ms	0.05 %	picomatch.makeRe	picomatch.js:286
22.7 ms	0.00 %	55.6 ms	0.01 %	formatRaw	node:internal/util/inspect:844

HOW IT WORKS?

- runtime
- debugger "server"
- debugger "client"

DEBUGGING MOBILE

- on mobile
    - allow USB debugging
    - open chrome
- on computer
    - `chrome://inspect`

KEY POINTS

- Debuggers are easy to set up
- Logpoints are temporary console.logs
- Breakpoints help you poke around running app
- `debugger;` works when sourcemaps fail
- Debugger has REPL
- Profilers exist

END