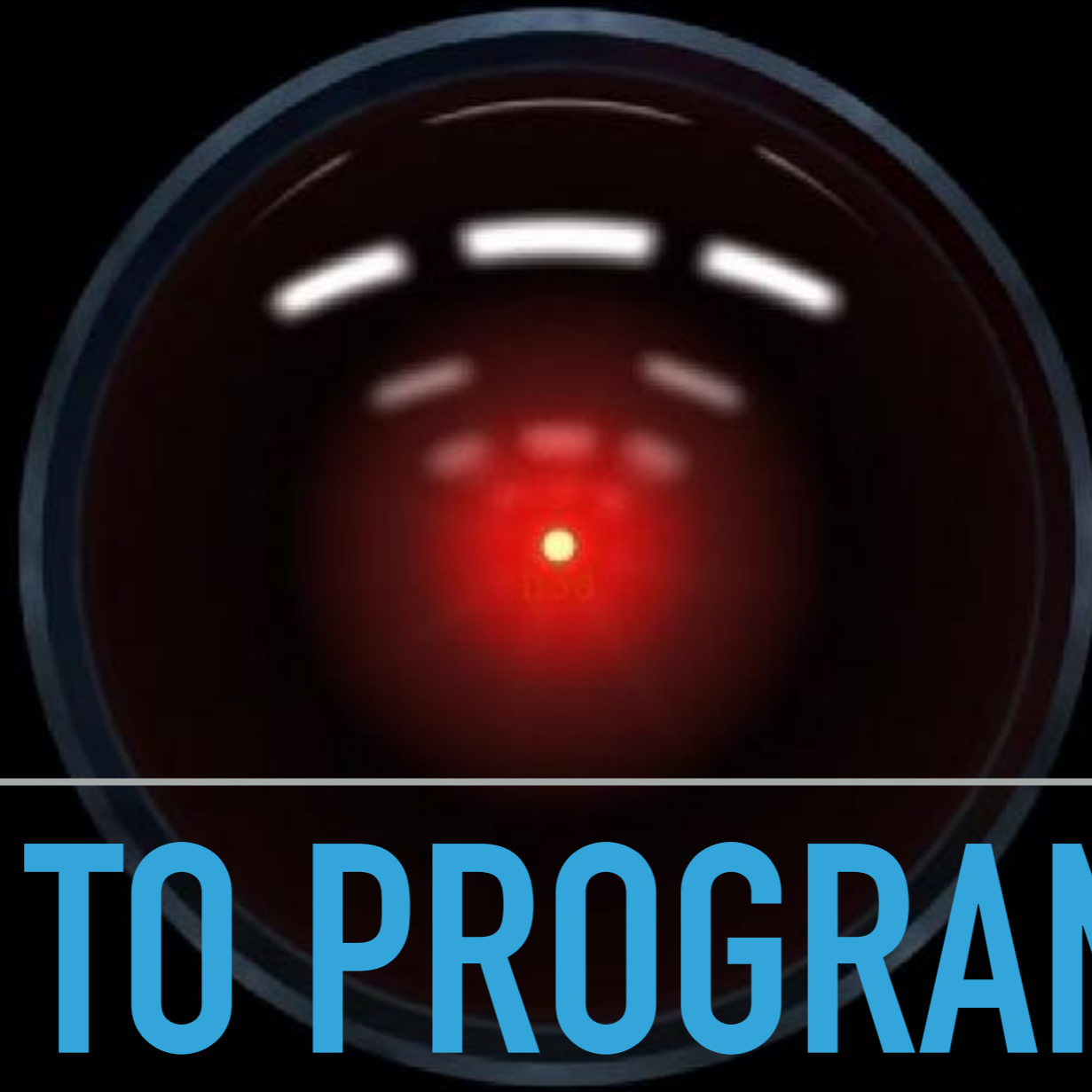


[HTTP://WWW.SIMONWELLS.ORG](http://www.simonwells.org)

[HTTP://ARG.NAPIER.AC.UK](http://arg.napier.ac.uk)



DR SIMON WELLS

INTRO TO PROGRAMMING

WHY ARE WE HERE?

This is not meant to be an existential question

WHAT IS PROGRAMMING?

- ▶ Telling a computer what to do [solving problems]
- ▶ Identifying parts of the solution [data]
- ▶ Working out how to handle each part [algorithms]

- ▶ Writing (Sorry [not sorry] ;)

WHAT IS PROGRAMMING?

**IF YOU CAN BAKE A CAKE/PREPARE A POT
NOODLE/PUT UP A PICTURE/WIRE A PLUG/
FIX A PUNCTURE/LIGHT A FIRE – THEN YOU
CAN PROBABLY WRITE A PROGRAM...**

**HOW DO I BECOME A (GREAT)
PROGRAMMER?**

- ▶ The programming genius
 - ▶ Just knows how to do it
 - ▶ Doesn't exist, probably
 - ▶ Hollywood has a lot to answer for :(
- ▶ Can read/follow a book/article/tutorial & I'll get it
 - ▶ Only part of the answer

HOW DO I BECOME A PROGRAMMER?

MYTHS

- ▶ Hard work & Effort
 - ▶ (but this can also be a lot of fun)
- ▶ Deliberate Practice (over time):

Thinking -> Doing -> Reflecting
- ▶ There is no magic.

HOW DO I BECOME A (GREAT) PROGRAMMER?

THE TRUTH?

PROGRAMMING IS A LIFESTYLE CHOICE

Write lots of programmes

BIO

- ▶ First Computer (age 7)
- ▶ Wrote some programmes (often from magazines & books)
- ▶ No real programming experience until university
- ▶ Nobody else in immediate family with a degree
- ▶ Nobody else with a higher degree at all (yet)
- ▶ Interested in *everything*...



IN SOME WAYS MY EXPERIENCE WAS EASIER

- Immediacy
- Lower expectations
- Work with less

~~****~~ COMMODORE 64 BASIC V2 ~~****~~

64K RAM SYSTEM 38911 BASIC BYTES FREE

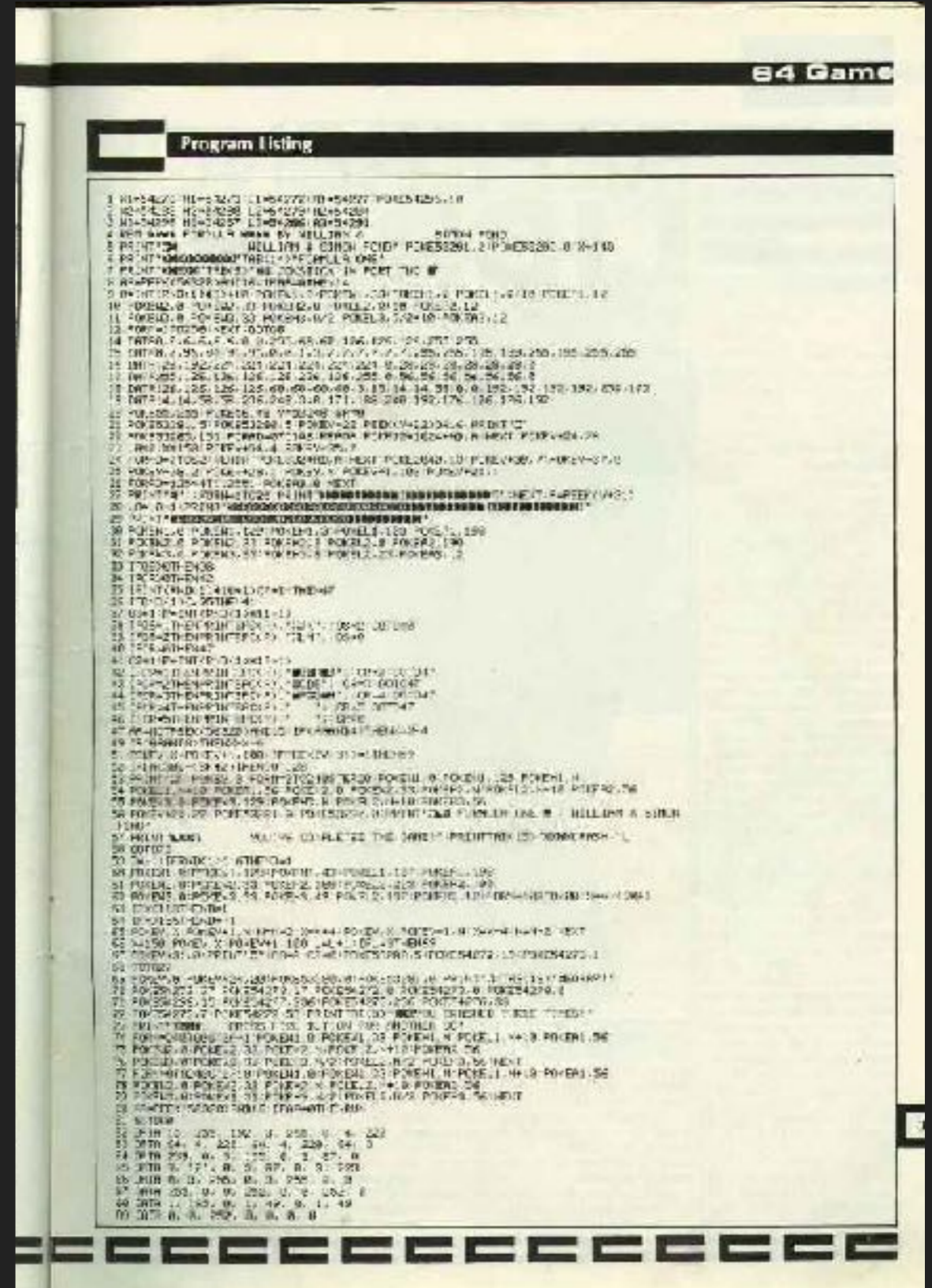
READY.
█

DOING ANYTHING WITH THIS MACHINE
INVOLVED PROGRAMMING:
WE COULD GET STRAIGHT TO THE
PROGRAMMING :D



WORK WITH LESS

- ▶ A lot fewer programmers around
- ▶ Home computers were untrusted, unreliable, and just not a mainstream consideration (for kids, for games, for the future)
- ▶ No smart phones
- ▶ No Internet/Web (we did have bulletin boards & modems though & Magazines)

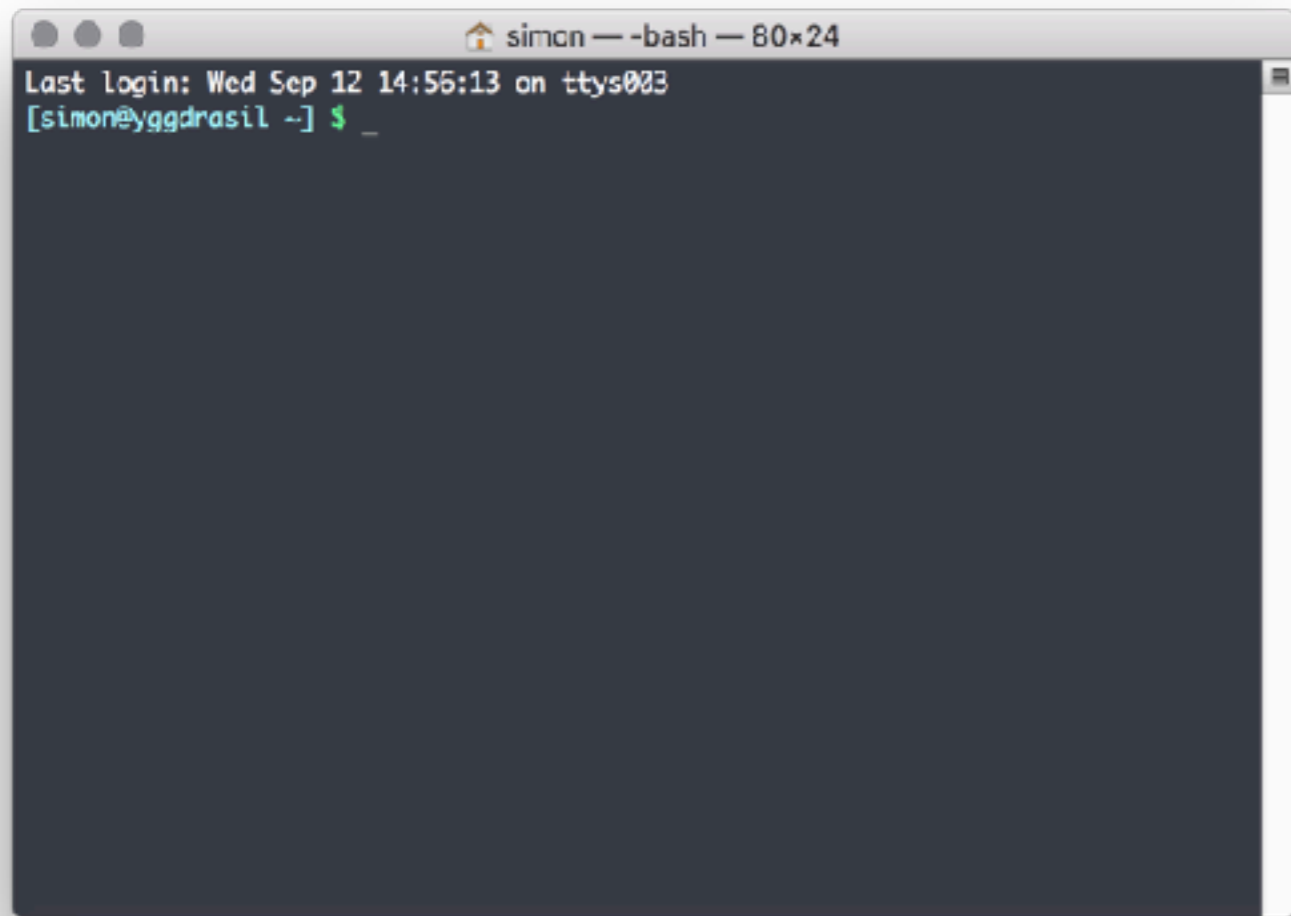


MODERN PROGRAMMING

- ▶ Getting into programming nowadays is easy:
 - ▶ Books, Web pages, Tutorials
 - ▶ Compilers, interpreters, IDEs
 - ▶ Nearly always have at least one computer on our person
- ▶ However:
 - ▶ Most computer experience is now point & click (or swipe)
- ▶ We see lots of really cool stuff but don't know how to get there from here
- ▶ **Bootstrapping is hard:**
 - ▶ there's lots of other stuff to do before you can start hacking away
- ▶ Also:
 - ▶ **What should I programme?**

- ▶ Modern computers aren't really set up to make programming accessible out-of-the-box
- ▶ Some hoop jumping: need to install programming language tools (compiler, interpreter, IDE, editor)
 - ▶ NB. Some computers already have these installed by default (Mac OS & Linux), e.g. python, ruby
- ▶ Not as straightforward as powering up the machine & getting dumped straight into a programming interface

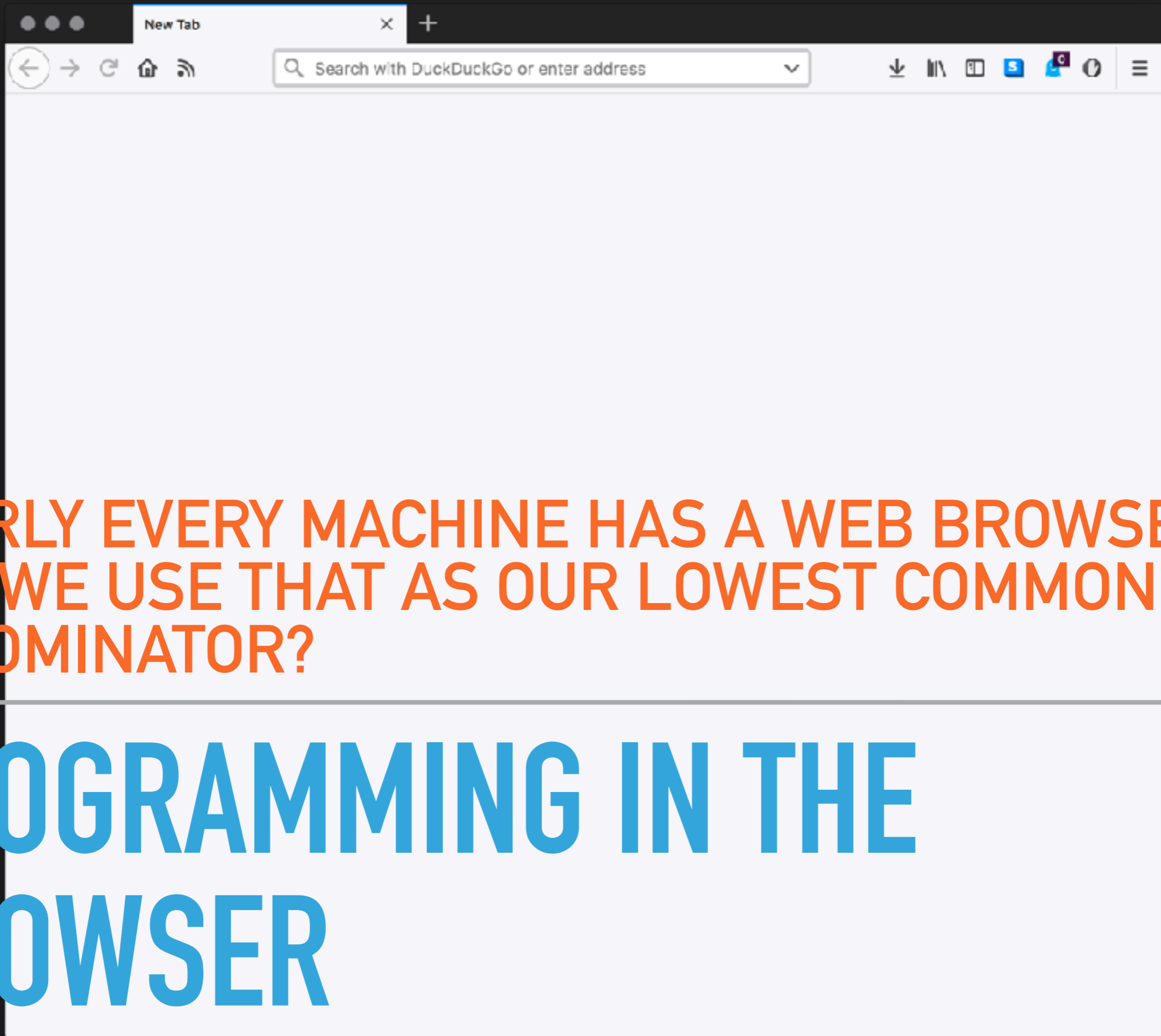
BOOTSTRAPPING IS HARD



```
simon — -bash — 80x24
Last login: Wed Sep 12 14:56:13 on ttys003
[simon@yggdrasil ~] $ _
```

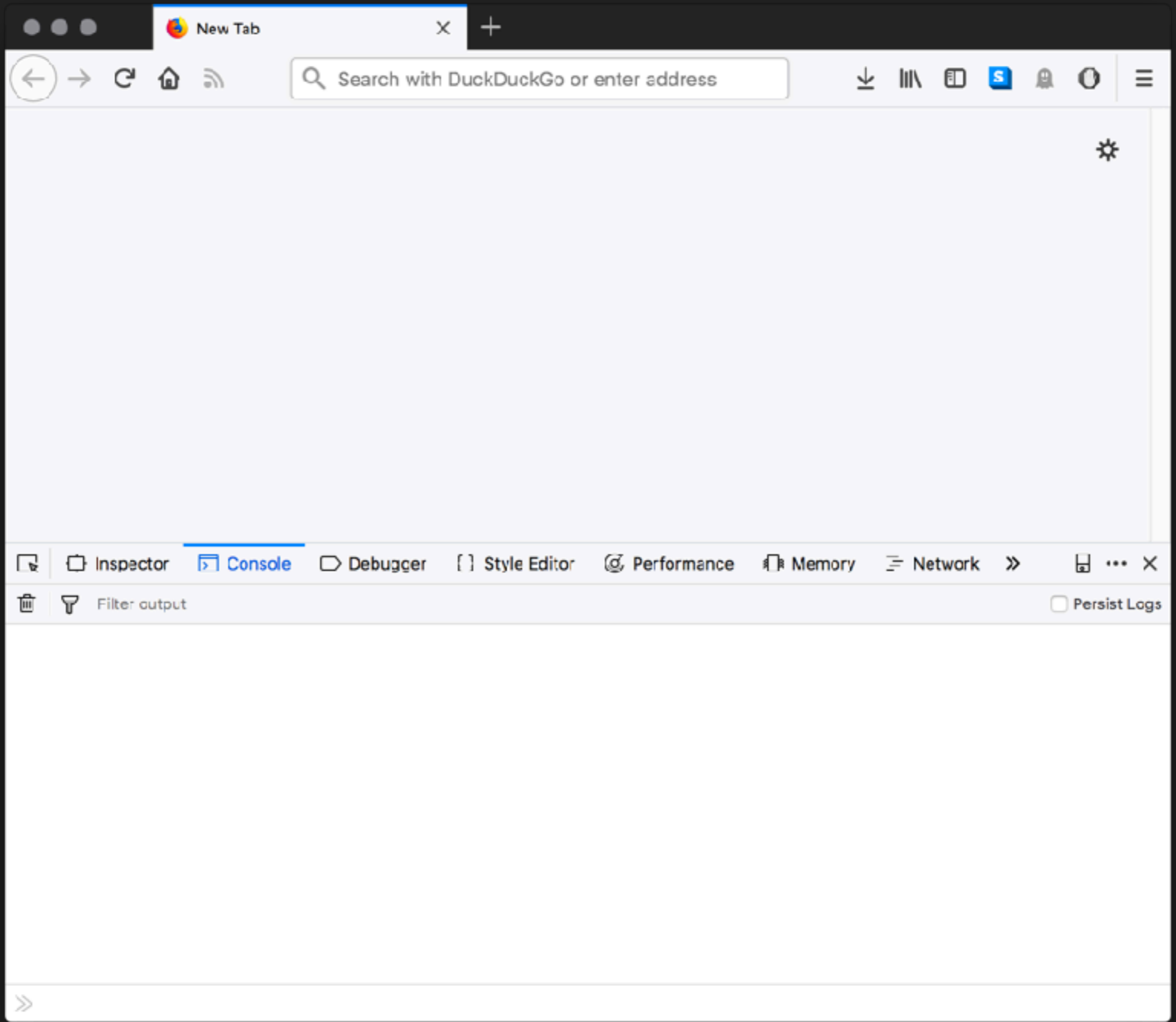
- ▶ Programming is a literate practise
 - ▶ If you only mouse around the GUI then life as a programmer is slightly more difficult
- ▶ CLI gives you the best, most fine-grained control of your computer
- ▶ Neal Stephenson "In the beginning was the command line"

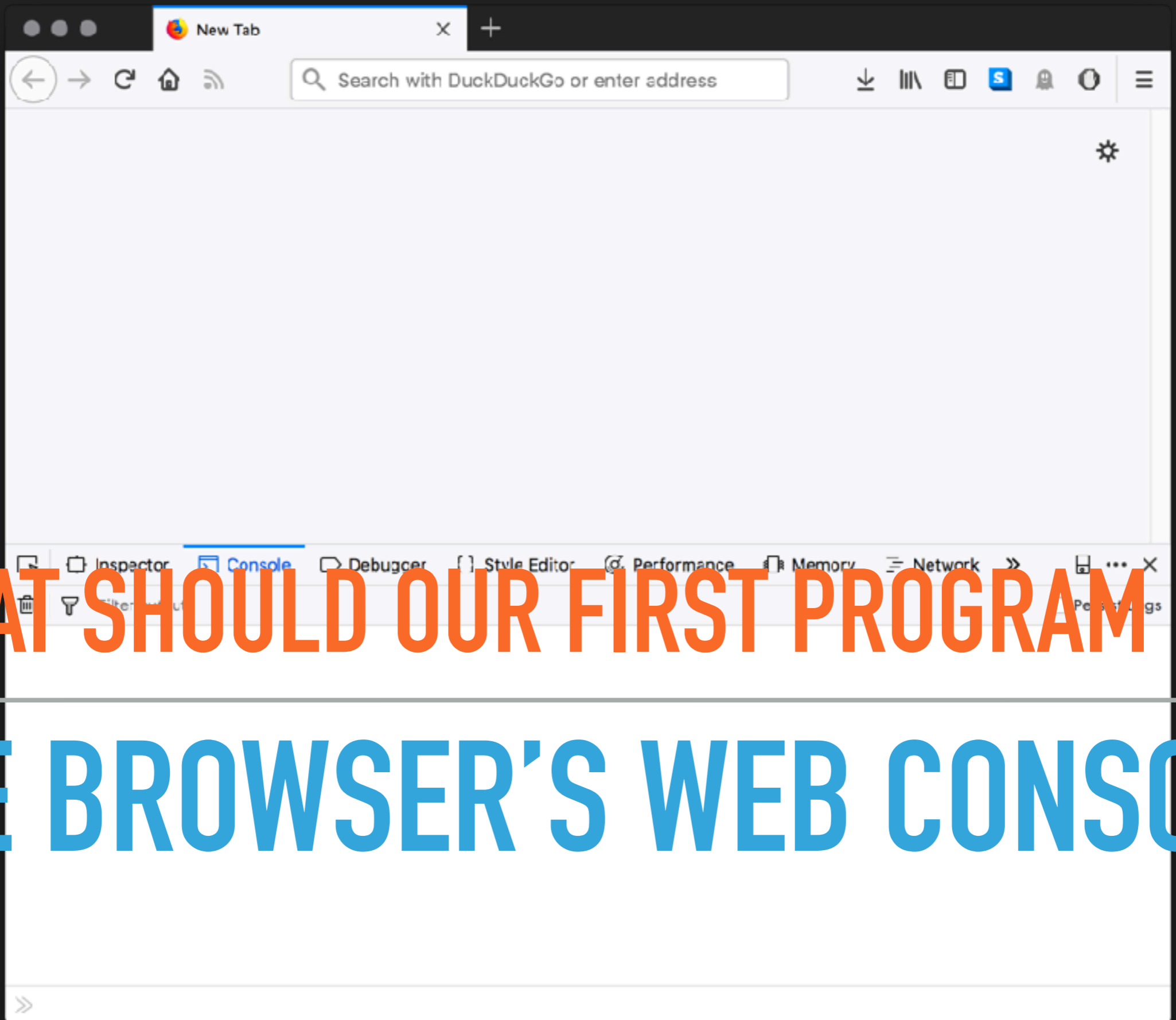
**TIP: LEARN TO LOVE THE
COMMAND LINE :)**



NEARLY EVERY MACHINE HAS A WEB BROWSER -
CAN WE USE THAT AS OUR LOWEST COMMON
DENOMINATOR?

PROGRAMMING IN THE
BROWSER





WHAT SHOULD OUR FIRST PROGRAM BE?

THE BROWSER'S WEB CONSOLE

#1

HELLO NAPIER

Browser window showing a new tab with the address bar containing "Search with DuckDuckGo or enter address". The browser interface includes navigation buttons (back, forward, refresh, home, RSS), a search bar, and various extension icons (download, print, search, social media, etc.).

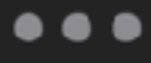
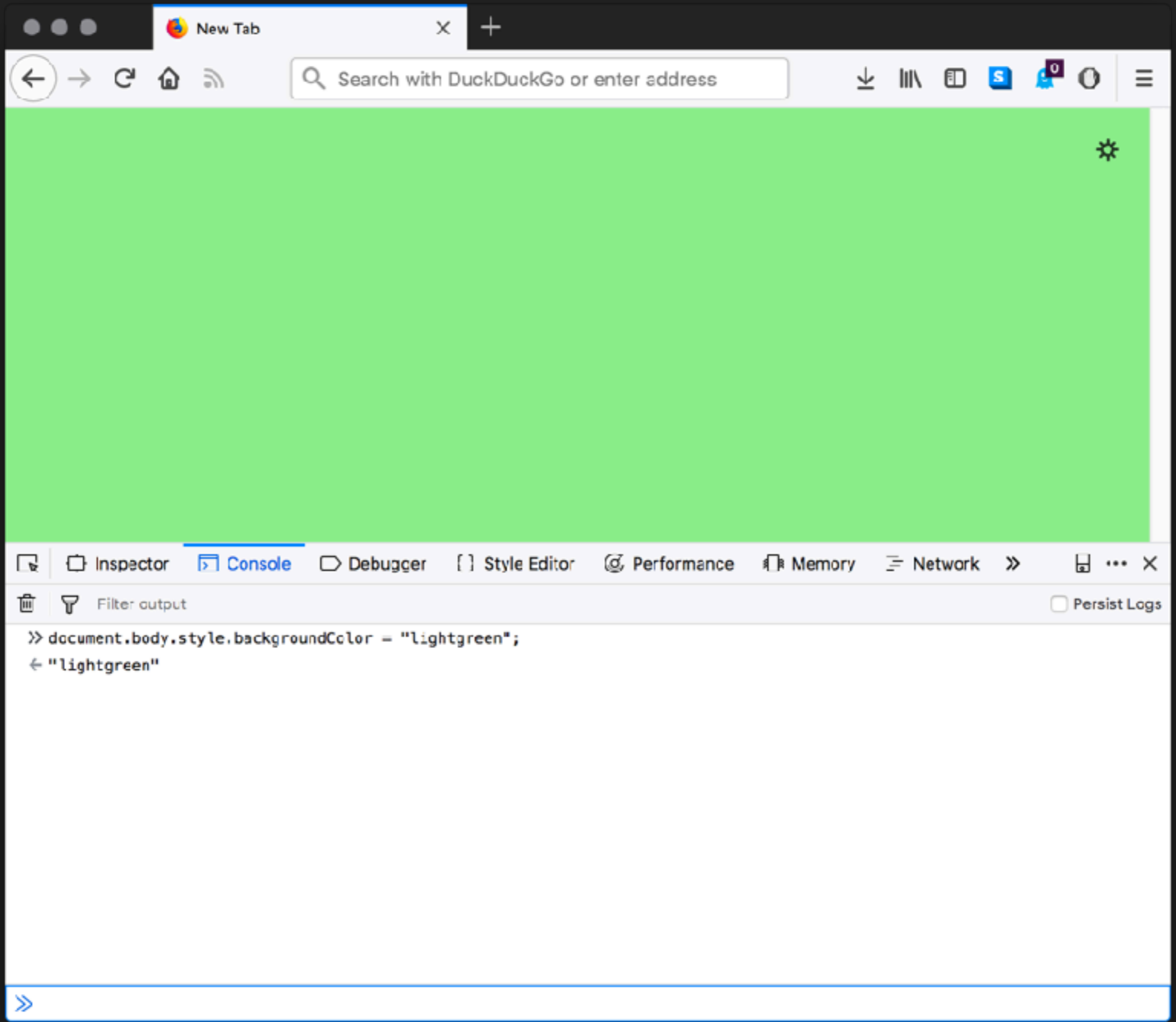
The browser's developer tools are open, showing the Console tab. The console output displays the following code and result:

```
>> console.log('hello napier!!!')
hello napier!!!
← undefined
```

The source of the code is identified as "debugger eval code:1:1". The console also includes a "Filter output" section and a "Persist Logs" checkbox.

#2

INTERACT WITH THE WEB PAGE/SCREEN



New Tab X +



Search with DuckDuckGo or enter address



Inspector Console Debugger Style Editor Performance Memory Network >> ... X

Filter output Persist Logs

```
>> document.body.style.backgroundColor = "lightgreen";
← "lightgreen"
```



#3

USE STANDARD JAVASCRIPT FUNCTIONS

The image shows a web browser window with a single tab titled "New Tab". The address bar contains the text "Search with DuckDuckGo or enter address". The main content area displays the text "Today's date is Thu Sep 13 2018 16:10:26 GMT+0100 (BST)". Below the content area is a developer console with the following code and output:

```
>> let d = new Date();  
← undefined  
>> document.body.innerHTML = "<h1>Today's date is " + d + "</h1>"  
← "<h1>Today's date is Thu Sep 13 2018 16:10:25 GMT+0100 (BST)</h1>"
```

The console also shows a "Filter output" button and a "Persist Logs" checkbox. The browser's developer tools are open, showing the "Console" tab selected. The bottom of the browser window has a blue arrow pointing right.

#4

CONSTRUCT A WEB PAGE

The image shows a web browser window with a single tab titled "New Tab". The address bar contains the text "Search with DuckDuckGo or enter address". The main content area of the browser displays "Hello Napier!!!". Below the content area, the developer tools are open, with the "Console" tab selected. The console shows the following JavaScript code being executed:

```
>> let p = document.createElement("P");  
    let t = document.createTextNode("Hello Napier!!!");  
    p.appendChild(t);  
    document.body.replaceWith(p);  
← undefined
```

The console also includes a "Filter output" section and a "Persist Logs" checkbox. The browser's navigation bar includes back, forward, refresh, and home buttons, along with various extension icons.

#5

GRAPHICS

Browser window showing a new tab with a search bar and navigation icons. The page content is a simple circle drawn on a canvas. The developer console is open, displaying the following JavaScript code:

```
>> let c = document.createElement("canvas");
var ctx = c.getContext("2d");
ctx.beginPath();
ctx.arc(95,50,40,0,2*Math.PI);
ctx.stroke();
document.body.replaceWith(c);
← undefined
```

The console also shows a filter output section and a "Persist Logs" checkbox.

#6

SOUND - BEEPS

The image shows a web browser window with a new tab. The address bar contains the text "Search with DuckDuckGo or enter address". The browser's developer tools are open, with the "Console" tab selected. The console displays the following JavaScript code:

```
>> var context = new (window.AudioContext || window.webkitAudioContext)();  
    var oscillator = context.createOscillator();  
    oscillator.type = 'sine';  
    oscillator.frequency.value = 440;  
    oscillator.connect(context.destination);  
    oscillator.start();  
← undefined
```

The console also shows a "Filter output" section with a trash icon and a "Persist Logs" checkbox. The browser's interface includes navigation buttons (back, forward, refresh, home, RSS), a search bar, and various utility icons (download, print, search, etc.).

#7

SOUND - MUSIC (AFTER A FASHION)

Browser window showing a new tab with the address bar and developer tools open.

The address bar contains the text: "Search with DuckDuckGo or enter address".

The developer tools console shows the following JavaScript code:

```
>> var context = new (window.AudioContext || window.webkitAudioContext)();  
var oscillator = context.createOscillator();  
h = window.innerHeight;  
oscillator.connect(context.destination);  
oscillator.start(0);  
document.addEventListener("mousemove", function(e) {  
    oscillator.frequency.value = e.clientY / h * 1000 + 300;  
});  
← undefined
```

The console also shows a filter output section with a trash icon and a "Persist Logs" checkbox.

- ▶ Nearly every computer has a browser so we can programme “old school” style almost anywhere at any time
- ▶ More likely to run against our own limitations right now than those of the the browser/JS
- ▶ Can build simple hackery into our daily programming habits

WHERE ARE WE?

WHAT SHOULD I PROGRAMME?

- ▶ Good Question!
- ▶ I've shown some simple things to get started
- ▶ What are you interested in?
- ▶ Key is to start small (remember the limitations & lower expectations I mentioned earlier)
- ▶ We want to make small increments without biting off more than we can chew.

WHAT SHOULD I PROGRAMME?

- ▶ Codes & Ciphers
 - ▶ This is actually an assignment in my second year web tech class (so I won't spoil it here)
- ▶ Chaos, Fractals, Artificial Life, & Cellular Automata
- ▶ Procedural Generation

WHERE DID SIMON START?

- ▶ A grid of cells that can be on or off

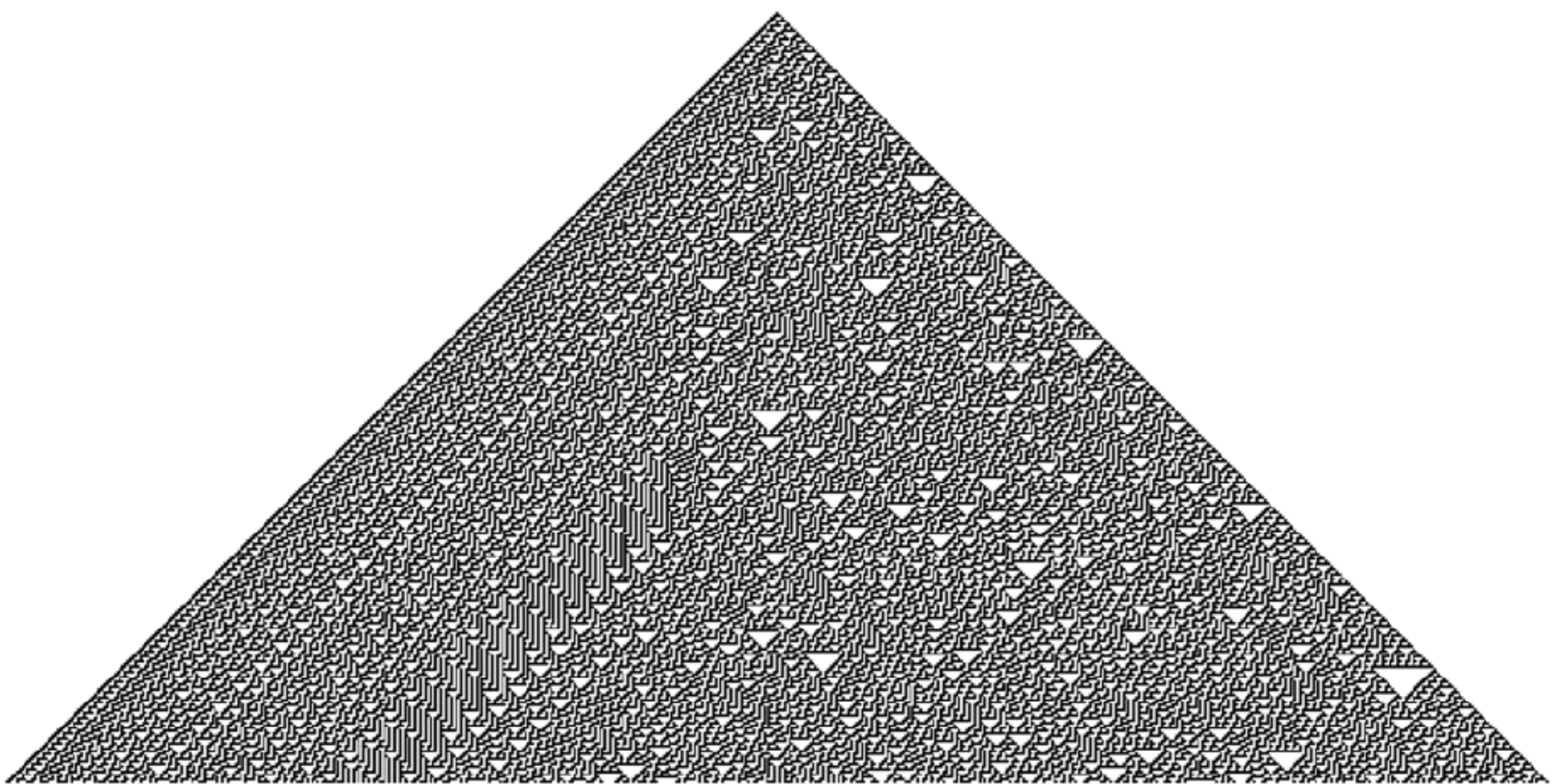
Take a starting generation

Some cells on & the rest off

Calculate the next *generation* according to some simple rules & repeat

- ▶ Can lead to very complex, sometime chaotic, behaviours
- ▶ The CompSci bit: Some CA have been proven to be able to calculate anything that a regular computer can calculate

CELLULAR AUTOMATA



RULE 30

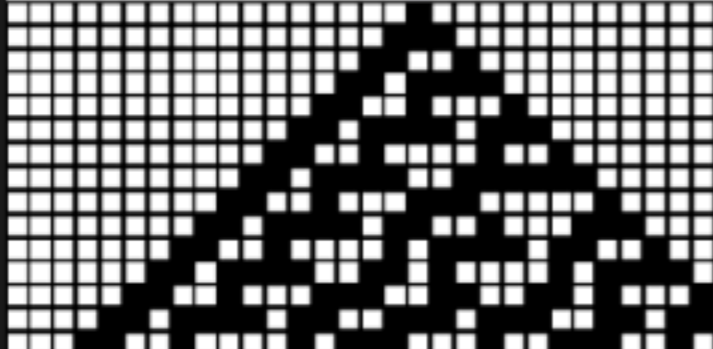
current pattern	111	110	101	100	011	010	001	000
new state for center cell	0	0	0	1	1	1	1	0

#8

1D CELLULAR AUTOMATA

New Tab

Search with DuckDuckGo or enter address



Inspector Console Debugger Style Editor Performance Memory Network

Filter output Persist Logs

```
>> function draw(generation, population){
    for (var i=0; i<population.length; i++) {
        ctx.rect(i*dimension, generation*dimension, dimension, dimension)
        if(population[i] === 1)
        { ctx.fillRect(i*dimension, generation*dimension, dimension, dimension) }
        ctx.stroke();
    }
}

function next_gen(old) {
    var old = [0].concat(old, [0]);
    var state = []; // The new state.

    for (var i=1; i<old.length-1; i++) {
        if (old[i-1] === 1 && old[i] === 1 && old[i+1] === 1) { state[i-1] = 0; }
        else if (old[i-1] === 1 && old[i] === 1 && old[i+1] === 0) { state[i-1] = 0; }
        else if (old[i-1] === 1 && old[i] === 0 && old[i+1] === 1) { state[i-1] = 0; }
        else if (old[i-1] === 1 && old[i] === 0 && old[i+1] === 0) { state[i-1] = 1; }
        else if (old[i-1] === 0 && old[i] === 1 && old[i+1] === 1) { state[i-1] = 1; }
        else if (old[i-1] === 0 && old[i] === 1 && old[i+1] === 0) { state[i-1] = 1; }
        else if (old[i-1] === 0 && old[i] === 0 && old[i+1] === 1) { state[i-1] = 1; }
        else if (old[i-1] === 0 && old[i] === 0 && old[i+1] === 0) { state[i-1] = 0; }
    }

    return state;
}

var c = document.createElement("canvas");
var ctx = c.getContext("2d");
var dimension = 10
var current = [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0];
var next = [];
var j = 60;
for (var i = 0; i < j; i++) {
    draw(i, current);
    current = next_gen(current);
}
document.body.replaceWith(c);
```

← undefined



- ▶ There are some places that collect programming problems & issue challenges:

- ▶ [Project Euler](#)

- ▶ [Stack Exchange Code Golf](#)

- ▶ [Code kata](#)

- ▶ [Reddit Daily Programmer](#)

- ▶ [Programming Praxis](#)

- ▶ [Rosetta Code](#)

- ▶ [International Collegiate Programming Contest Problems Index](#)

- ▶ [Algorithmist](#)

**I DON'T LIKE ANY OF THAT
CRAP, WHAT SHOULD I DO?**

IN SUMMARY

- ▶ Think small (until it's time to think big)
- ▶ Follow your interests
- ▶ If you don't have any interests then:
 - ▶ look around you | read more | steal from others
- ▶ Become a daily programmer
- ▶ Write LOTS of code
- ▶ Have fun

PROGRAMMING SURGERIES

- ▶ School of Computing (Merchiston Campus)
- ▶ Schedule (Weeks 02-15)
 - ▶ Monday, Lab MER_C06, 2pm-4pm
 - ▶ Wednesday, Lab MER_C06, 11am-1pm
 - ▶ Friday, Lab MER_C06, 12pm-2pm

HACKATHON

- ▶ For non-freshers, a hackathon runs during fresher's week
- ▶ Come along to D2 between 2 & 3PM this afternoon to see what they've been doing (& get an idea of what you might want to be involved in next year)

**WE ARE ALL SMART HERE.
DISTINGUISH YOURSELF BY
BEING KIND.**

RESOURCES

- ▶ Code for all of the examples (& more) is available here:

<https://github.com/siwells/oldschoolcoding/tree/master>

- ▶ If you want to find out more, these books are a good starting place for learning JavaScript:

- ▶ "JavaScript: The Good Parts" by Douglas Crockford

- ▶ "Eloquent JavaScript" by Marijn Haverbeke

- ▶ "The "You don't know JS" series by Kyle Simpson

- ▶ The MDN web docs site:

<https://developer.mozilla.org/en-US/>