Week 3 - Material Logic

Photographed November 1979

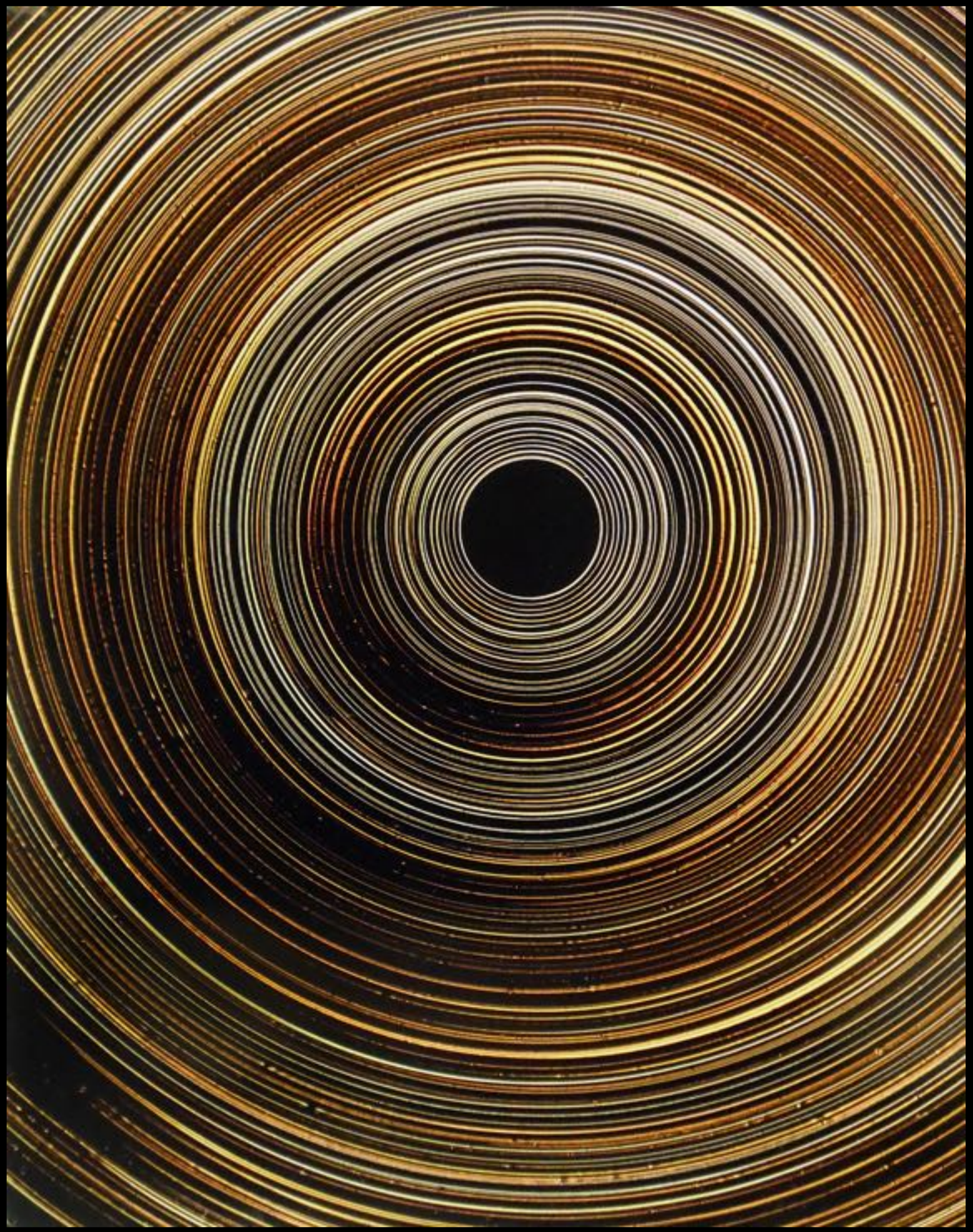
Photographed June 2004

Merril Wagner
Wall Piece, 1979
Oil Paint on Wall

Stan Brackage
Mothlight, 1963

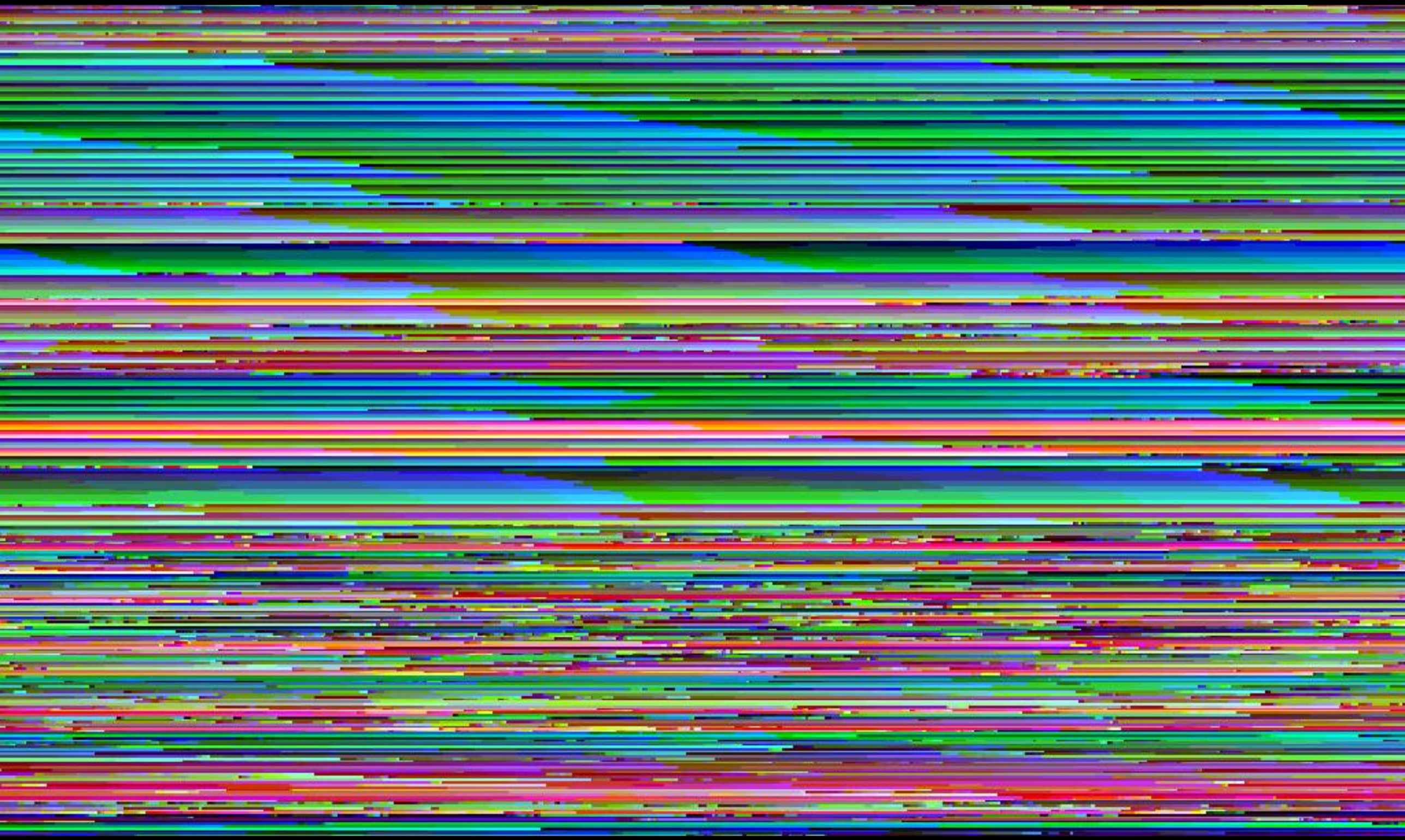
"Here is a film I made out of a deep grief…"

Gordon Matta Clark
Splitting, 1974
Gelatin Silver Prints, cut and collaged



Marco Breuer

Postcards From Planet Earth, 2010
Clement Valla



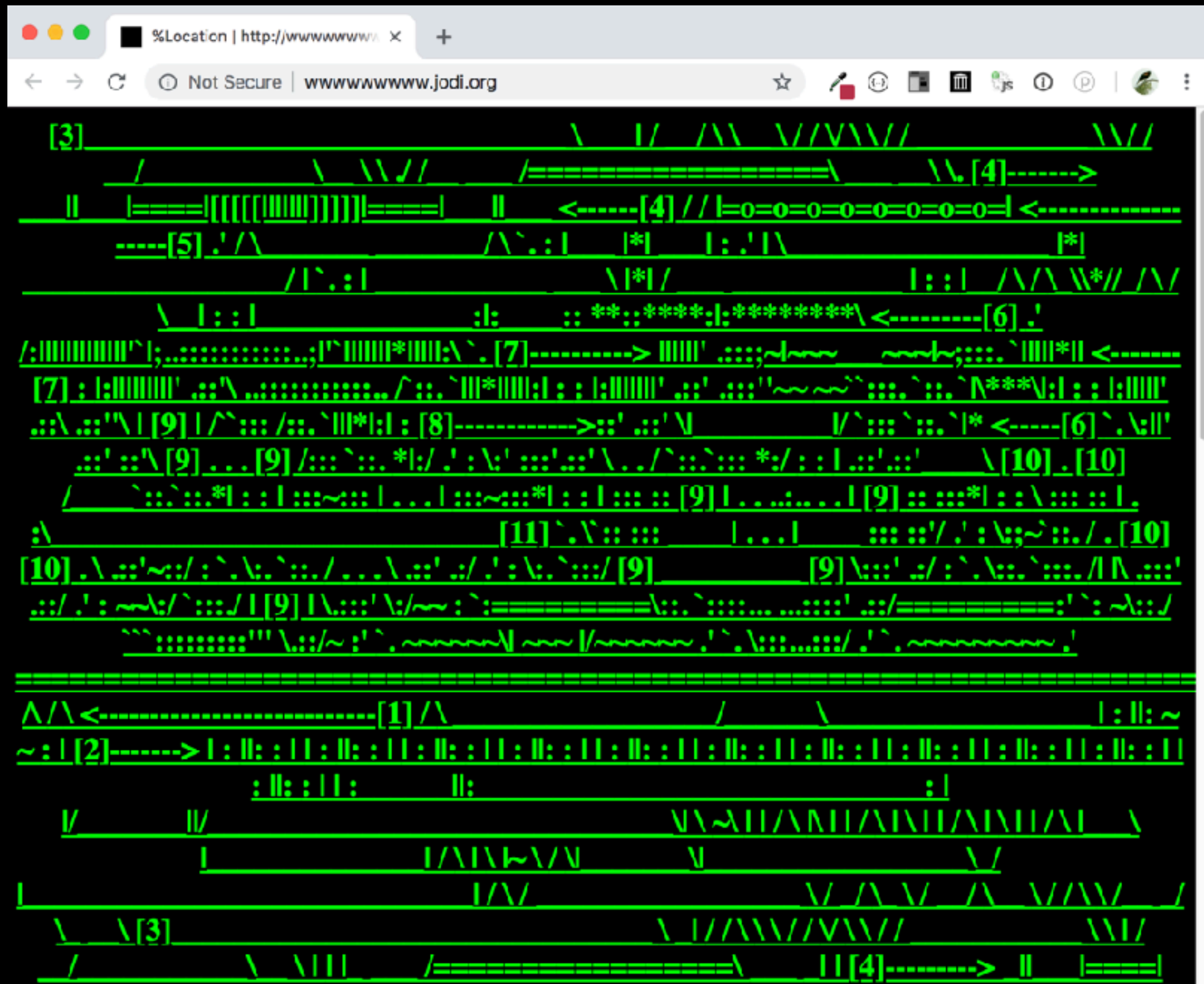
Lisa Jevbratt
1:1, 1999 - 2001

Every IP, 1999

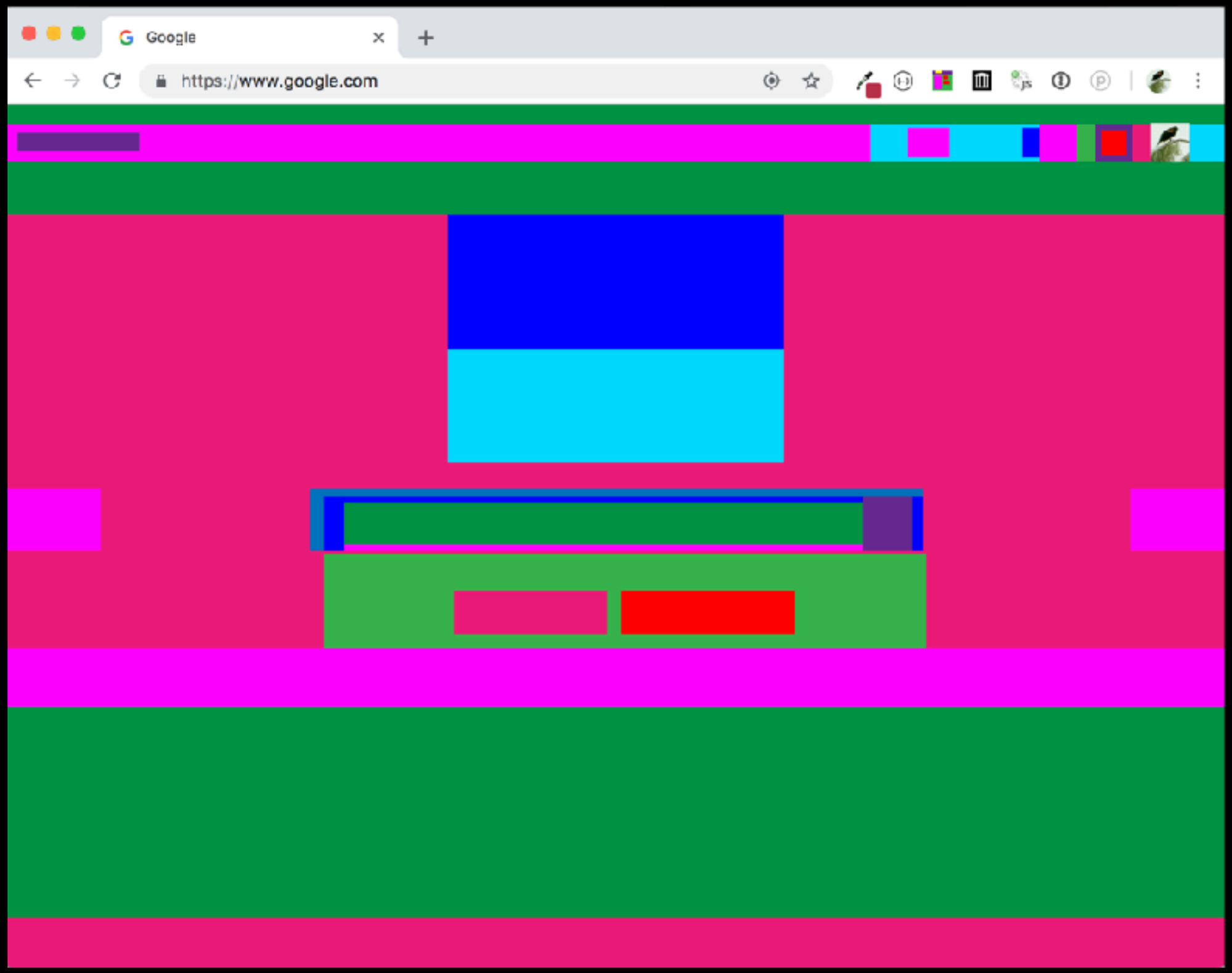
«1:1» operates with five interfaces—

Migration: reveals in one image how the Web has ›moved‹ over the last few years
Hierarchical: the Web as directory structure
Every: mapping of all web servers in the database
Random: randomly generated IP addresses from the database
Excursion: provides access to the unsearched places of the web

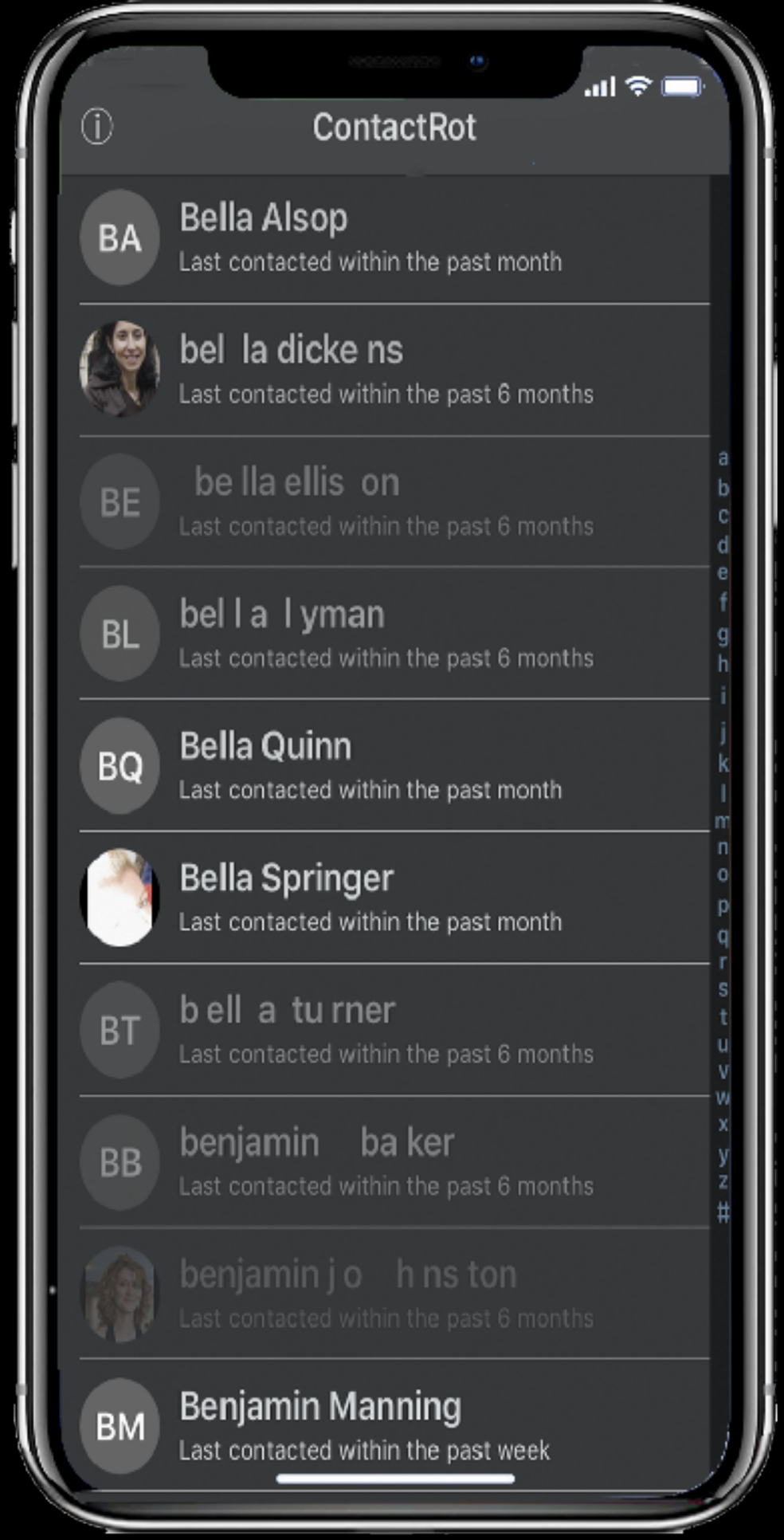
Lisa Jevbratt
1:1, 1999 - 2001



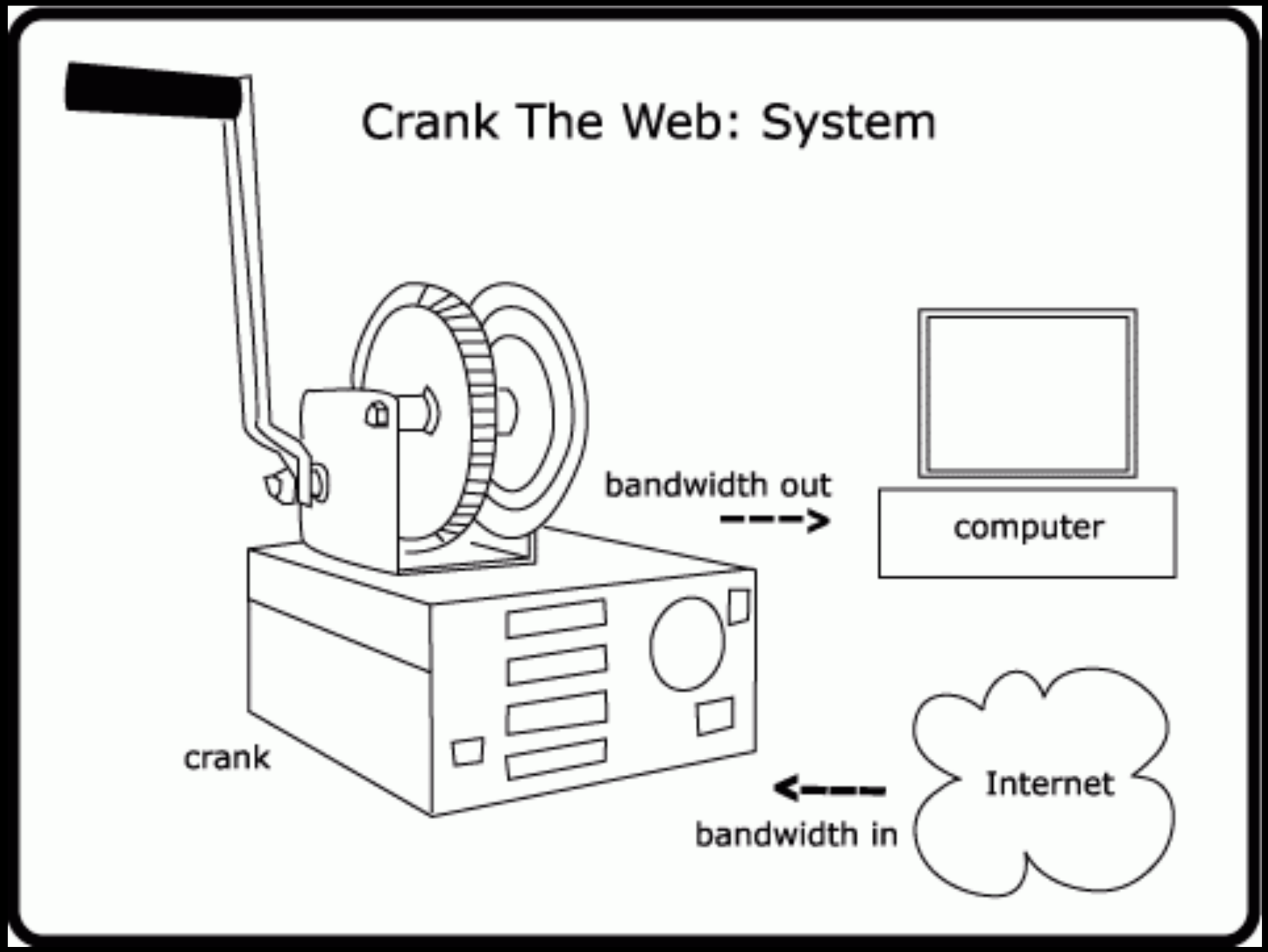
jodi.org, 1999 - 2001



Rafaël Rozendaal.
Abstract Browsing, 2014

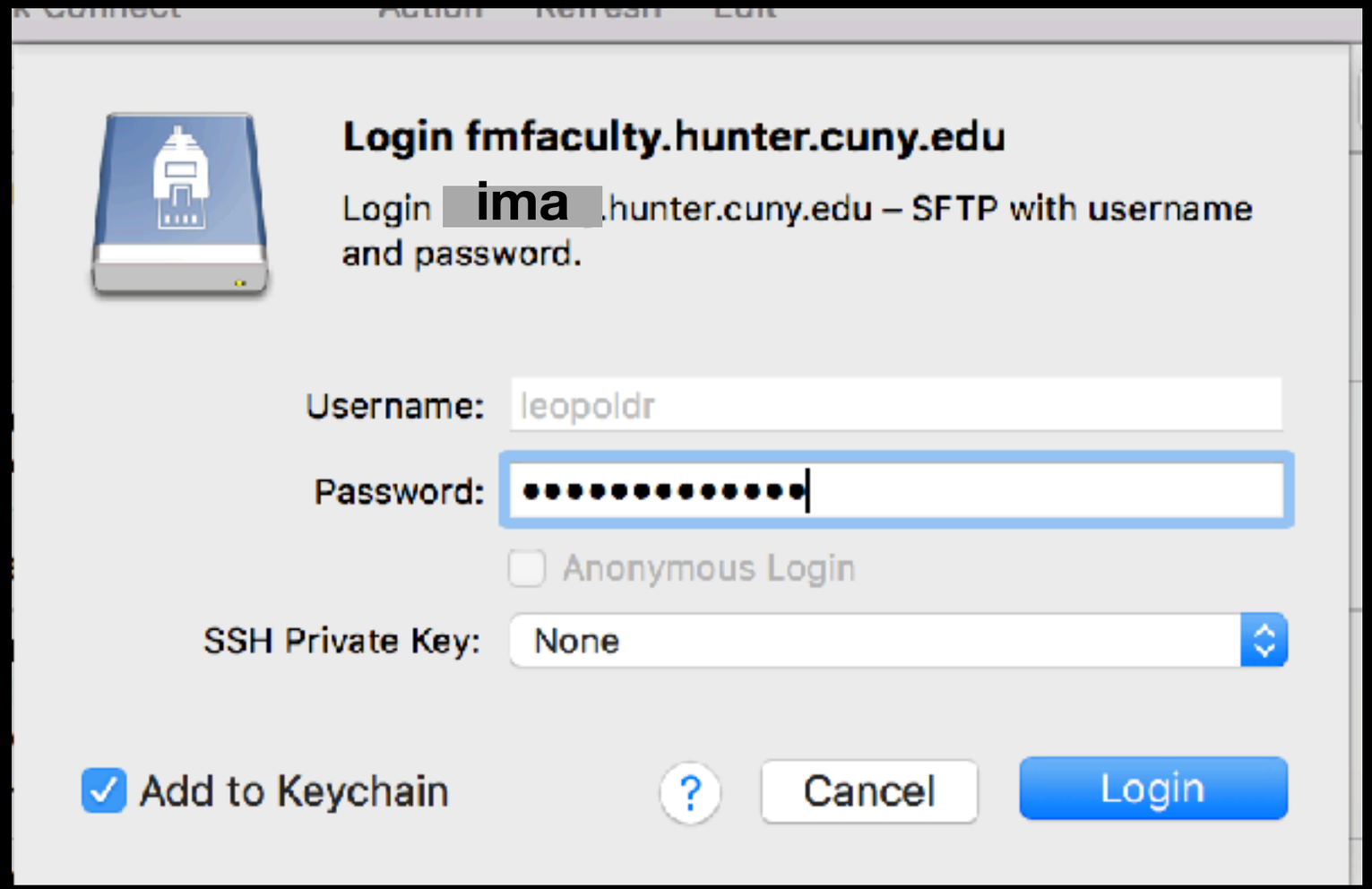


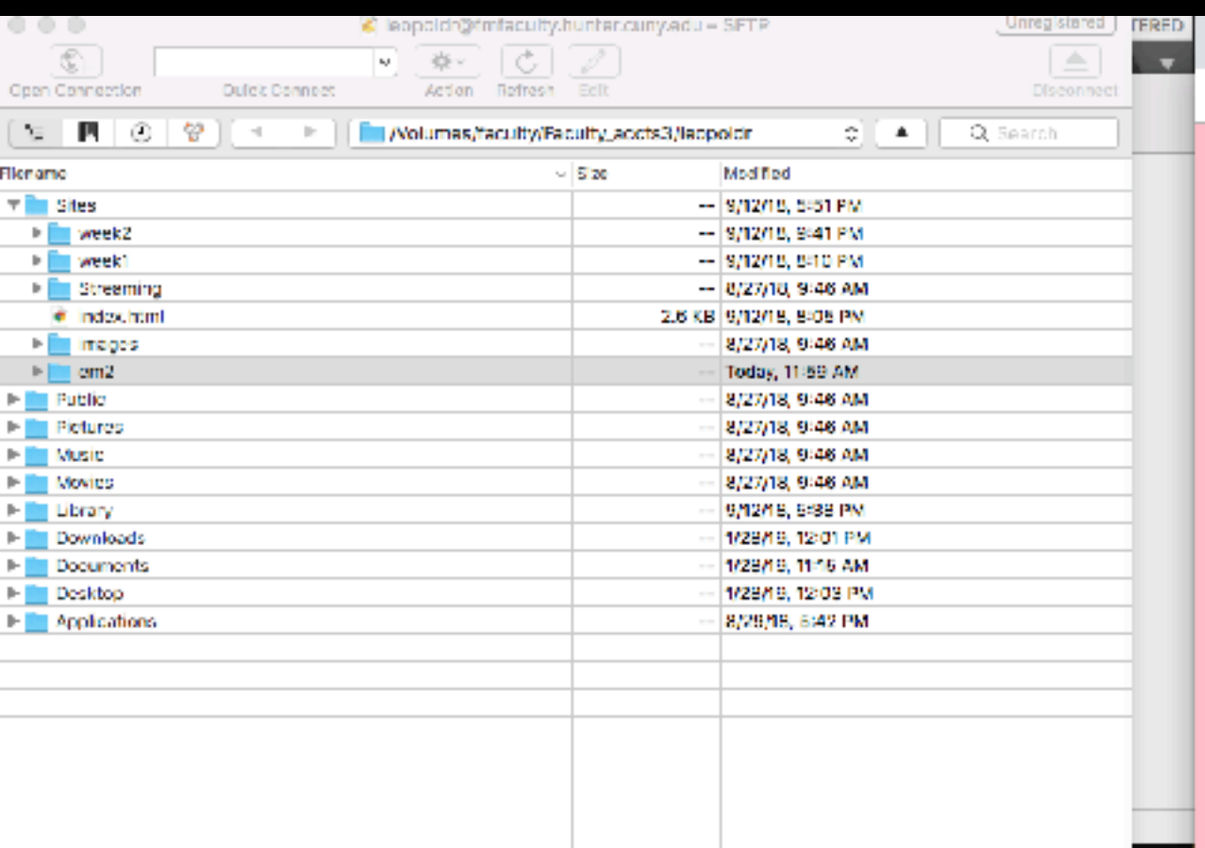
Jonah Brucker-Cohen
Contact Rot, 2018

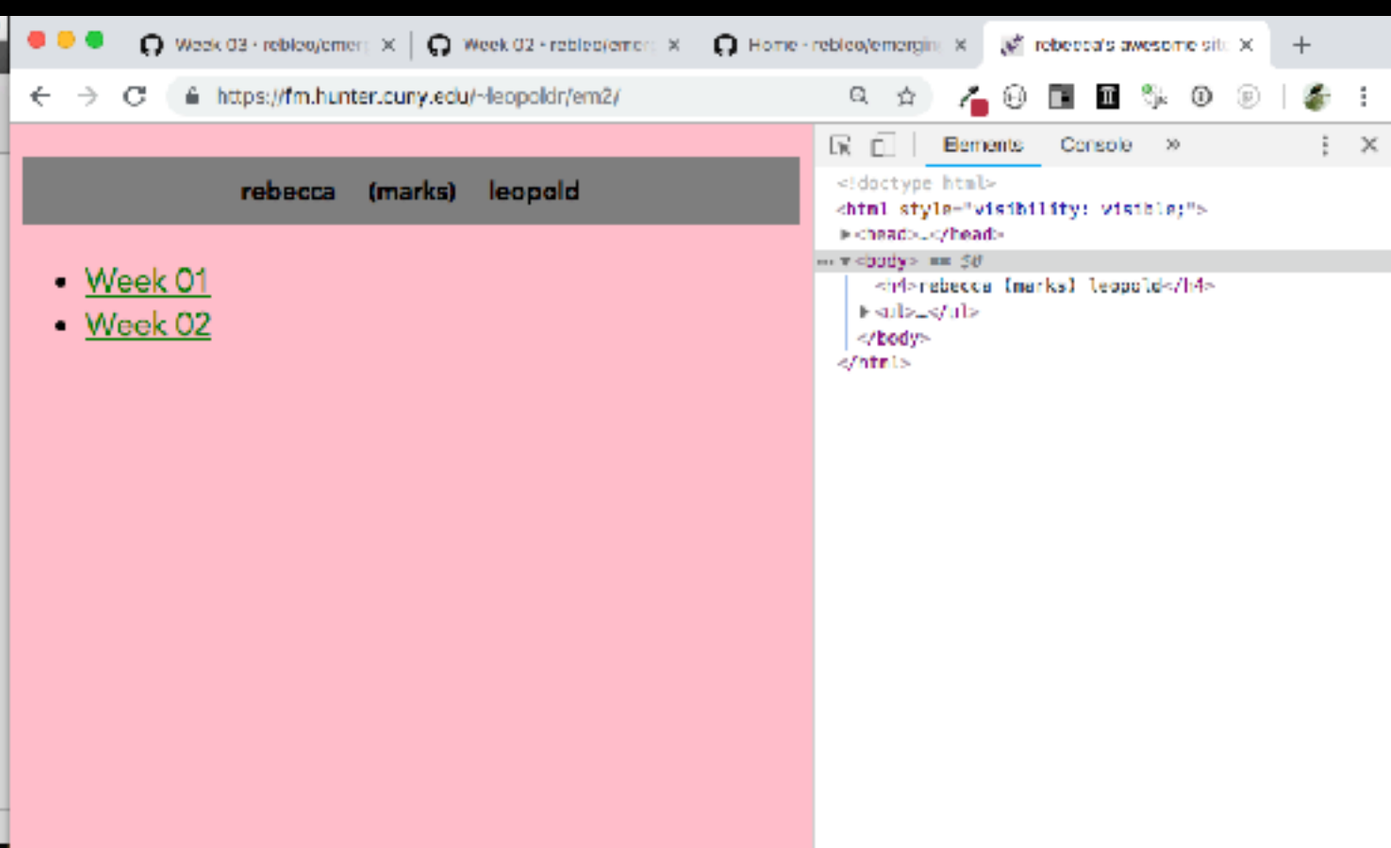


Jonah Brucker-Cohen
Crank the Web, 2001

uploading to f + m server w/ cyberduck


Login fmfaculty.hunter.cuny.edu
Login ima .hunter.cuny.edu – SFTP with username and password.
Username: leopoldr
Password:
Anonymous Login
SSH Private Key: None
Add to Keychain
Cancel
Login


Open Connection
Quick Connect
Action
Refresh
Edit
Disconnect
/Volumes/faculty/Faculty_accts3/leopoldr
Filename
Size
Modified
Sites
week2
week1
Streaming
index.html
images
em2
Public
Pictures
Music
Movies
Library
Downloads
Documents
Desktop
Applications


https://fm.hunter.cuny.edu/~leopoldr/em2/
rebecca (marks) leopold
Week 01
Week 02
Elements
Console

intro 2 CSS

* which is super awesome

**bc CSS is all about:

DESGIN

HTML - Hyper Text Mark Up

is a grammar for structuring web pages. It defines paragraphs, headings, data tables + media elements. HTML describes the content of the page - not how it looks.

CSS - Cascading Style Sheet

rules for styling a web page. Setting colors, typeface, and the layout. It can be used to consider the design of your page across different platforms and screen sizes.

Metadata: `viewport`

The user's visible area of a web page

HTML5 introduced a method to let web designers take control over the viewport, through the **<meta>** tag.

```
<!
—  - Tells the browser to match the device's width for the viewport
   - Sets an initial zoom value -->

<meta name="viewport" content="width=device-width, initial-scale=1.0”>
```

```
<meta name="viewport" content="width=device-width, initial-scale=1.0”>
```

without

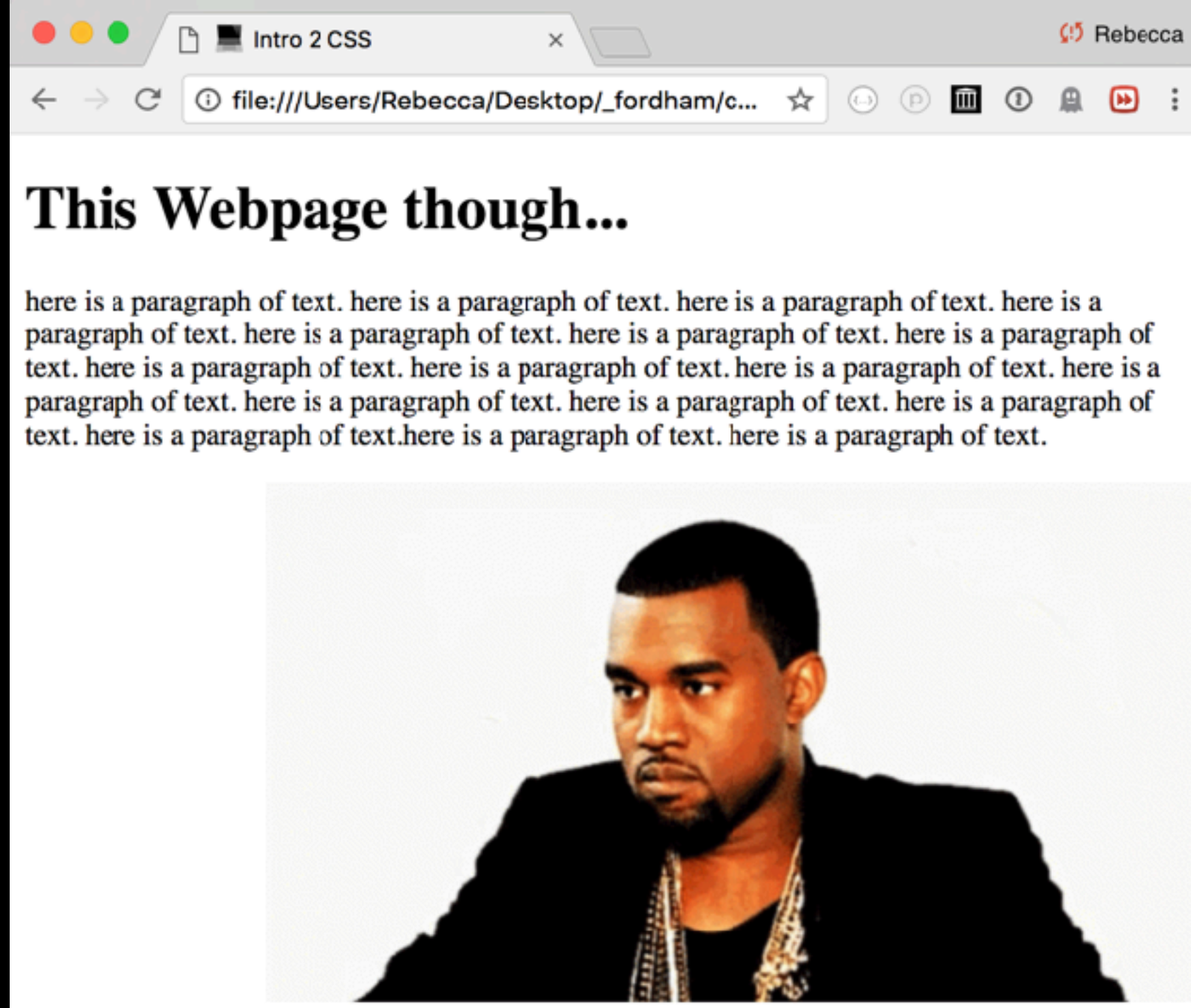
with

CSS works by associating rules with HTML elements. These rules govern how the content of specified elements should be displayed.

A CSS rule contains two parts: a selector and a declaration.

It takes 5% to learn how to write CSS rule and 95% to learn different properties that you can use.

the world w/out css



What does Cascading mean?

CSS Cascading has to do with how the styles apply when you have two or more rules that apply to the same element. Which one takes precedence?

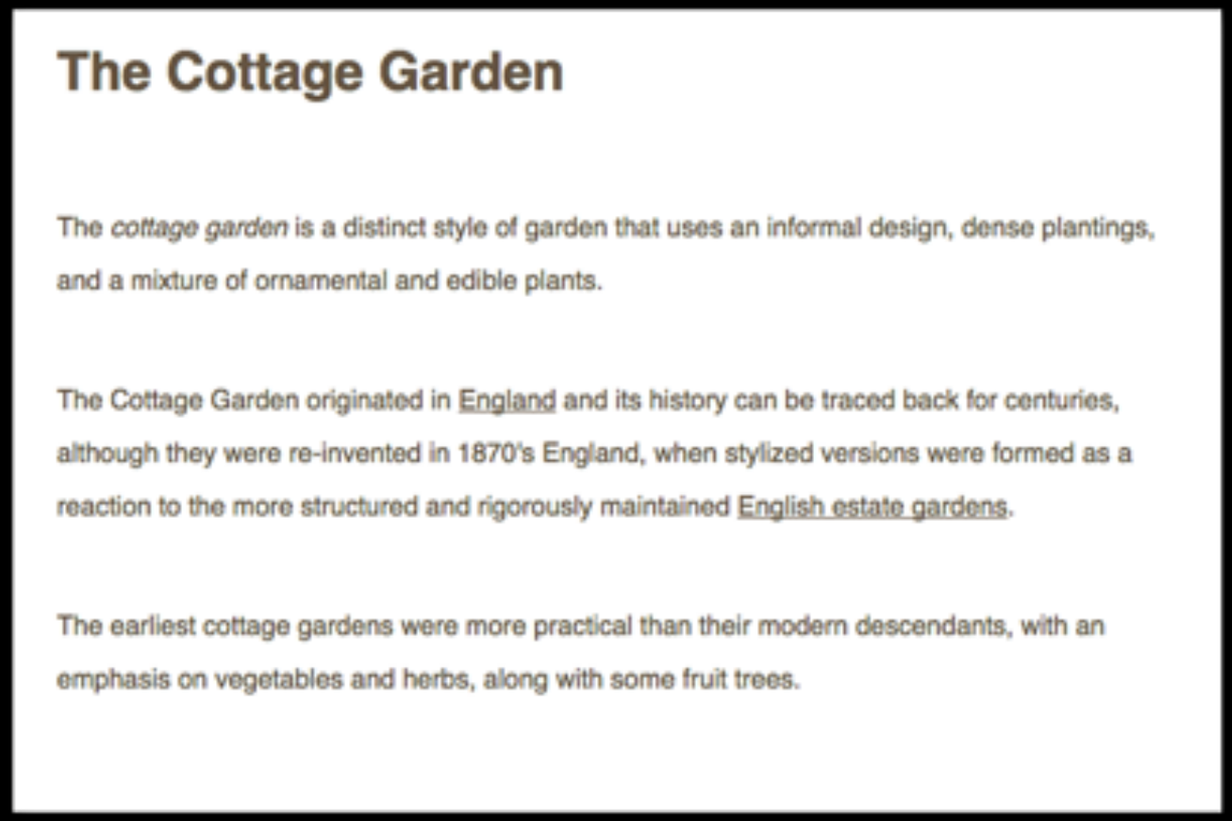
CSS Rules will cascade towards specificity and then downwards on the page. For example, if you set text to be a certain size in your **<body>**, that rule will cascade all the way down to all of the elements within body

This all comes back to the tree, parent/child structure of HTML…

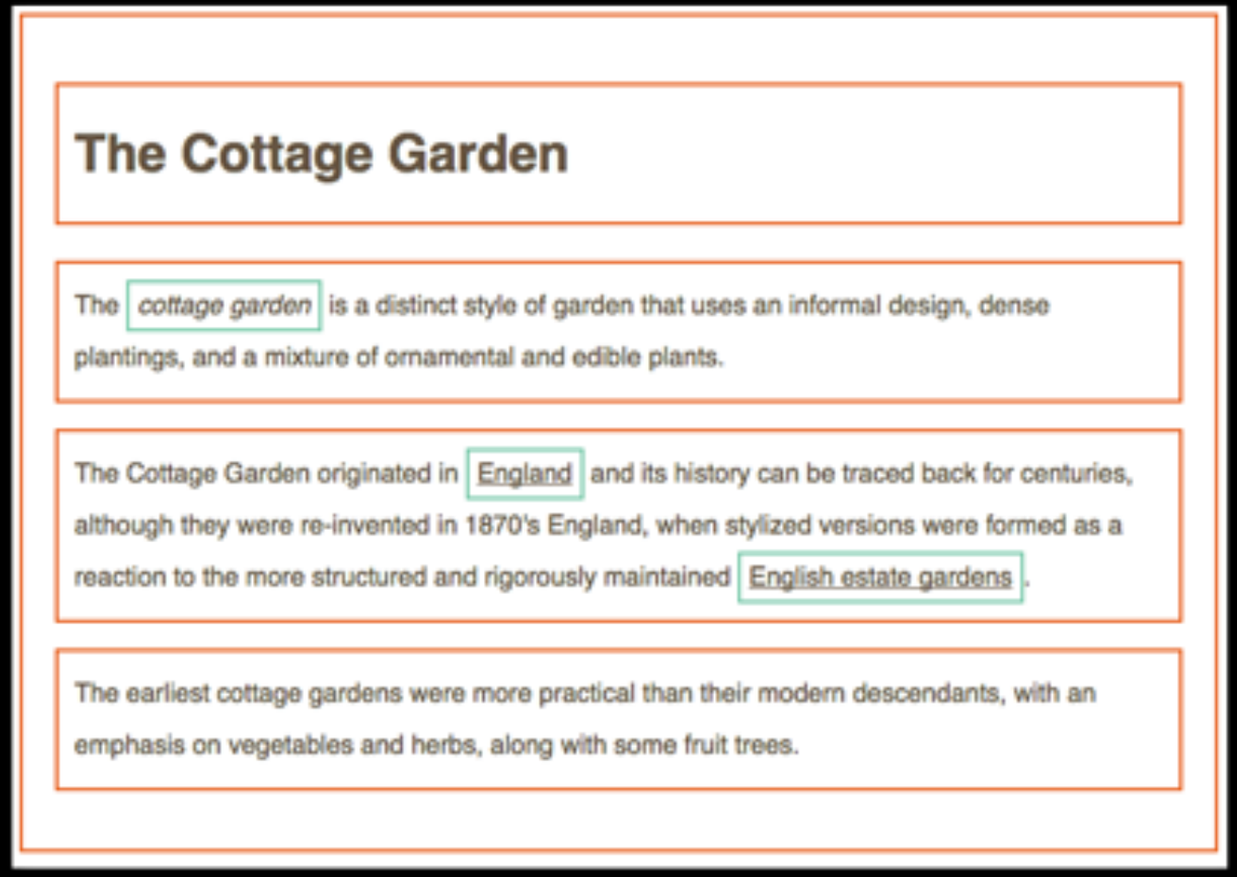
But: if you do something to a **<body>**, then do something to **<lists>** within the body, the latter will take precedence

If you do two things to those lists, the last one on your CSS page will take precedence

The key to understanding how **CSS** works is to imagine that there is an invisible box around every **HTML** element.



Block level elements are outlined w/ red + inline elements in green.



<body> creates 1st box, then **<h1>**, **<h2>**, **<p>**, **<i>** + **<a>** each create their own boxes within it.

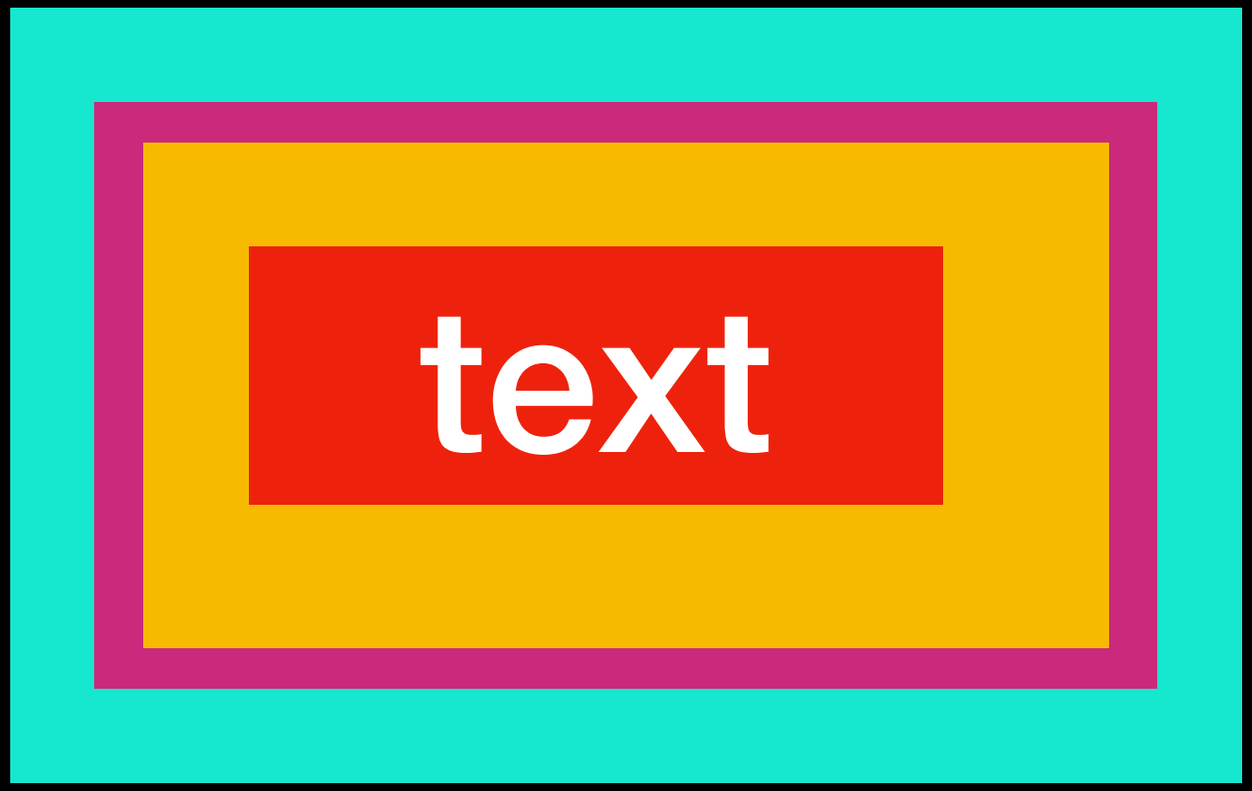
Box Model

Border

All boxes have borders even if invisible or 0px wide. It separates the edge of one box from another.

Padding

Padding is the space btw the border + any content contained within it. More padding increases the readability of its contents.

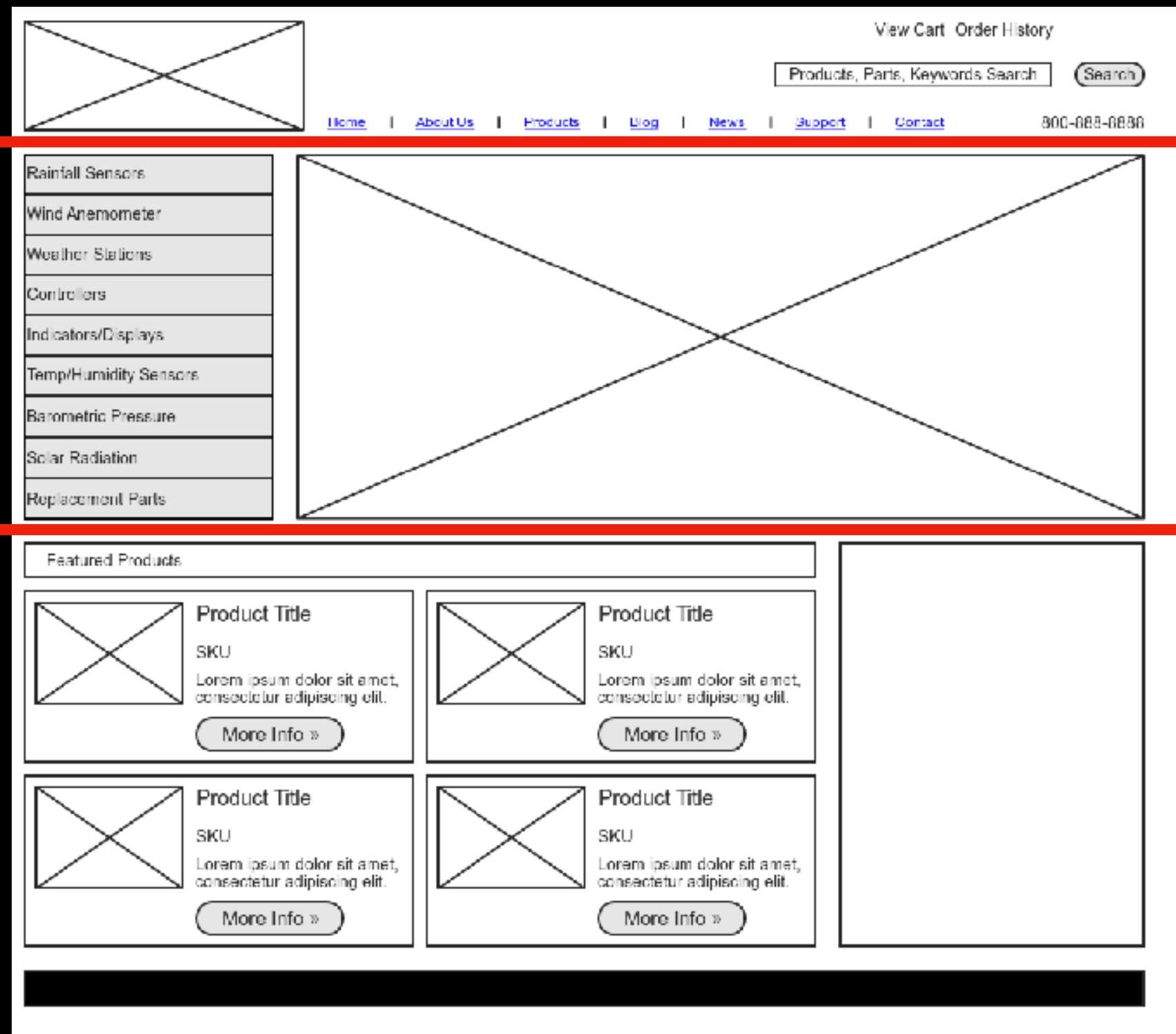


Margin

Margins sit outside the edge of the border. You can set the width to create a gap btw borders of adjacent boxes.

Content

Containing Elements



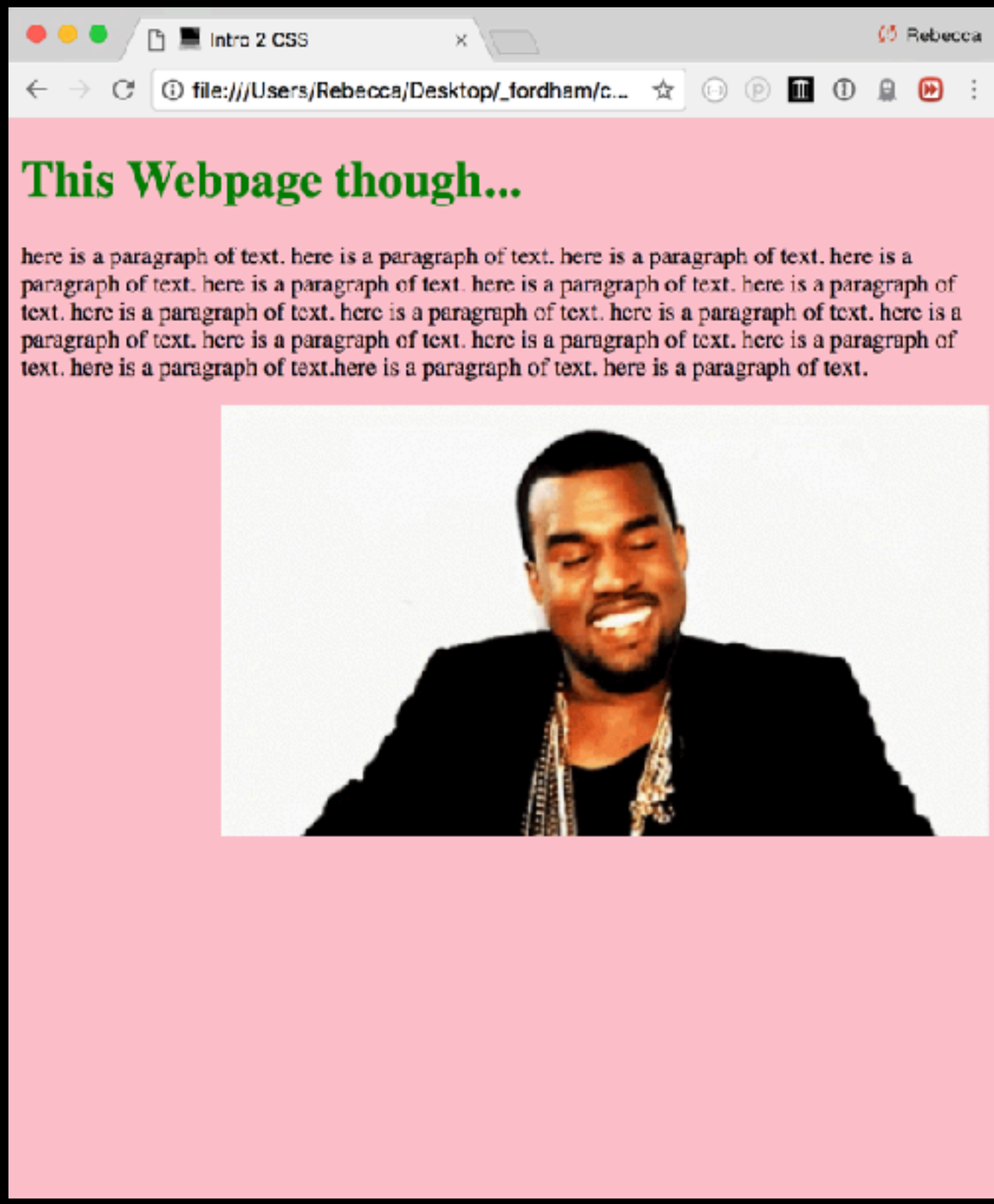
Block level elements start on a new line – if a block-level element sits inside another then the outer box is the containing or parent element.

You can write CSS 3 Different Ways:

Inline Styles

Embedded Styles

Externals Styles



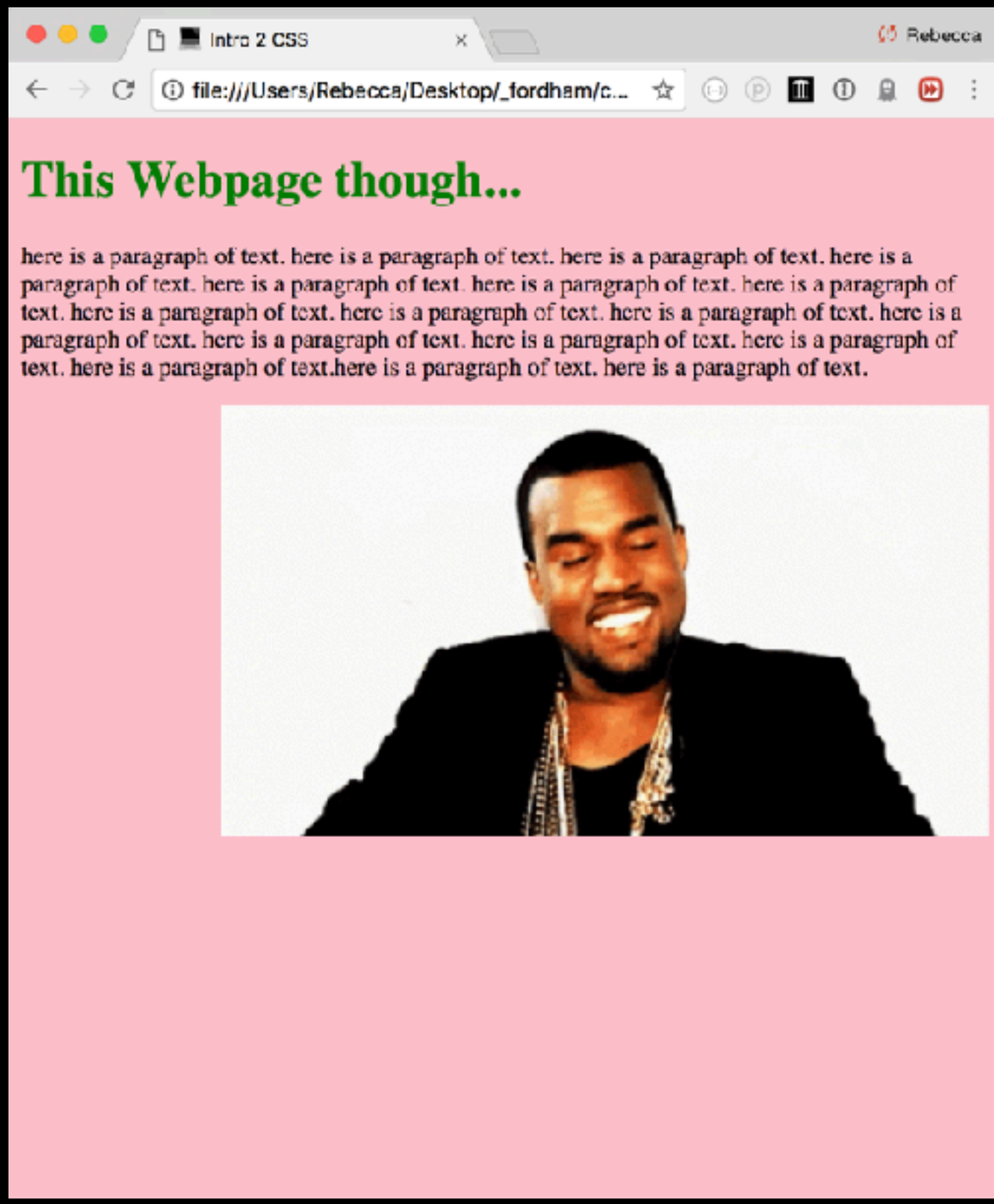
You can write CSS 3 Different Ways:

Inline Styles

```
<h1 style="color:green;">This Webpage though...</h1>
<body style="background-color: pink;">
```

Embedded Styles

Externals Styles



Inline Styles

```
<h1 style="color:white;">This Webpage though...</h1>
<body style="background-color: green;">
```

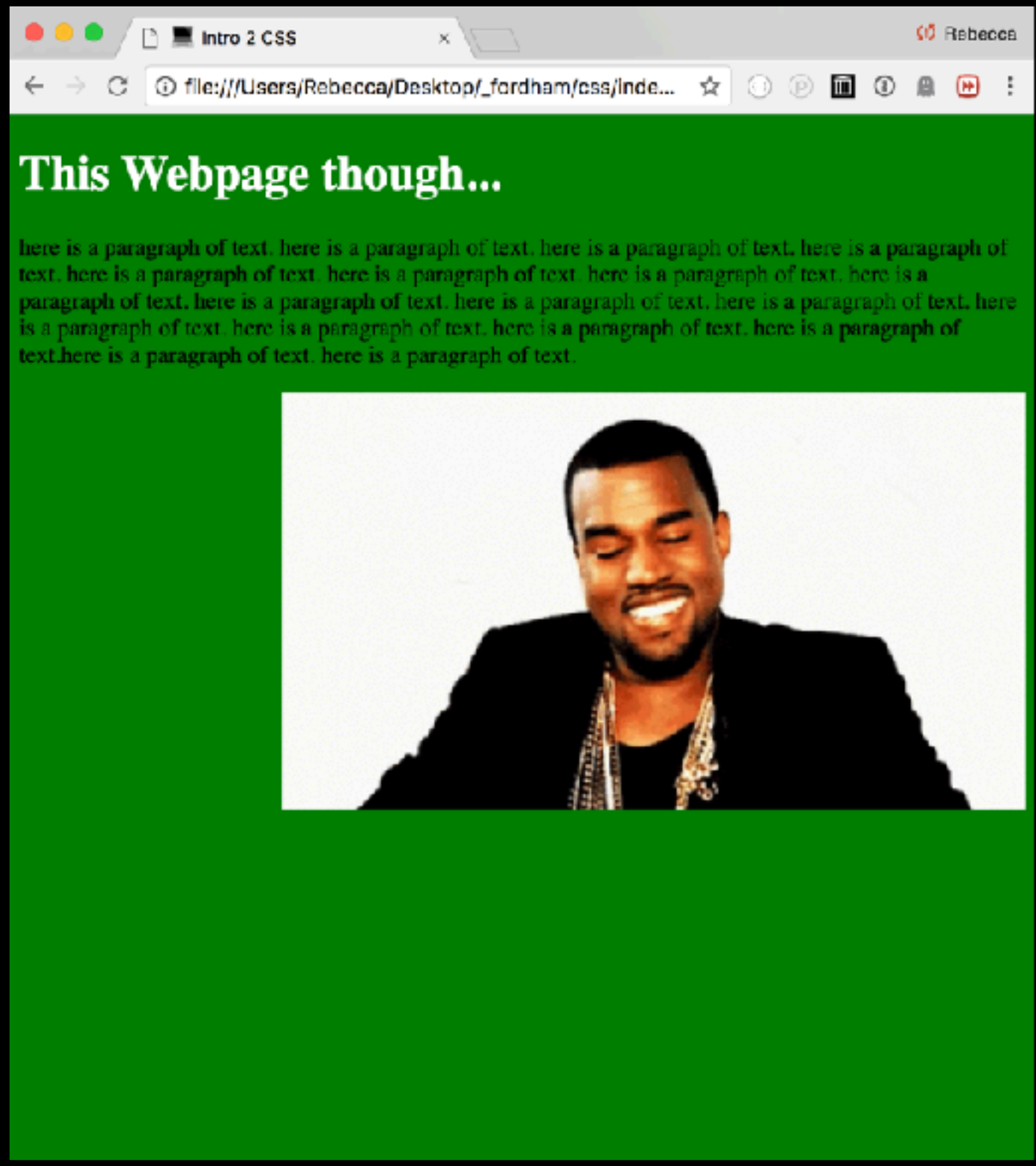
Embedded Styles

```
<html>
  <head>
        <title> 💻 Intro 2 CSS </title>
        <style type- "text/css">
            h1 {
                color: white
            }

            body {
                background: green;
            }

        </style>
    </head>
```

Externals Styles



Inline Styles

```
<h1 style="color:#FF4500;">This Webpage though...</h1>
<body style="background-color: #000080;">
```

Embedded Styles

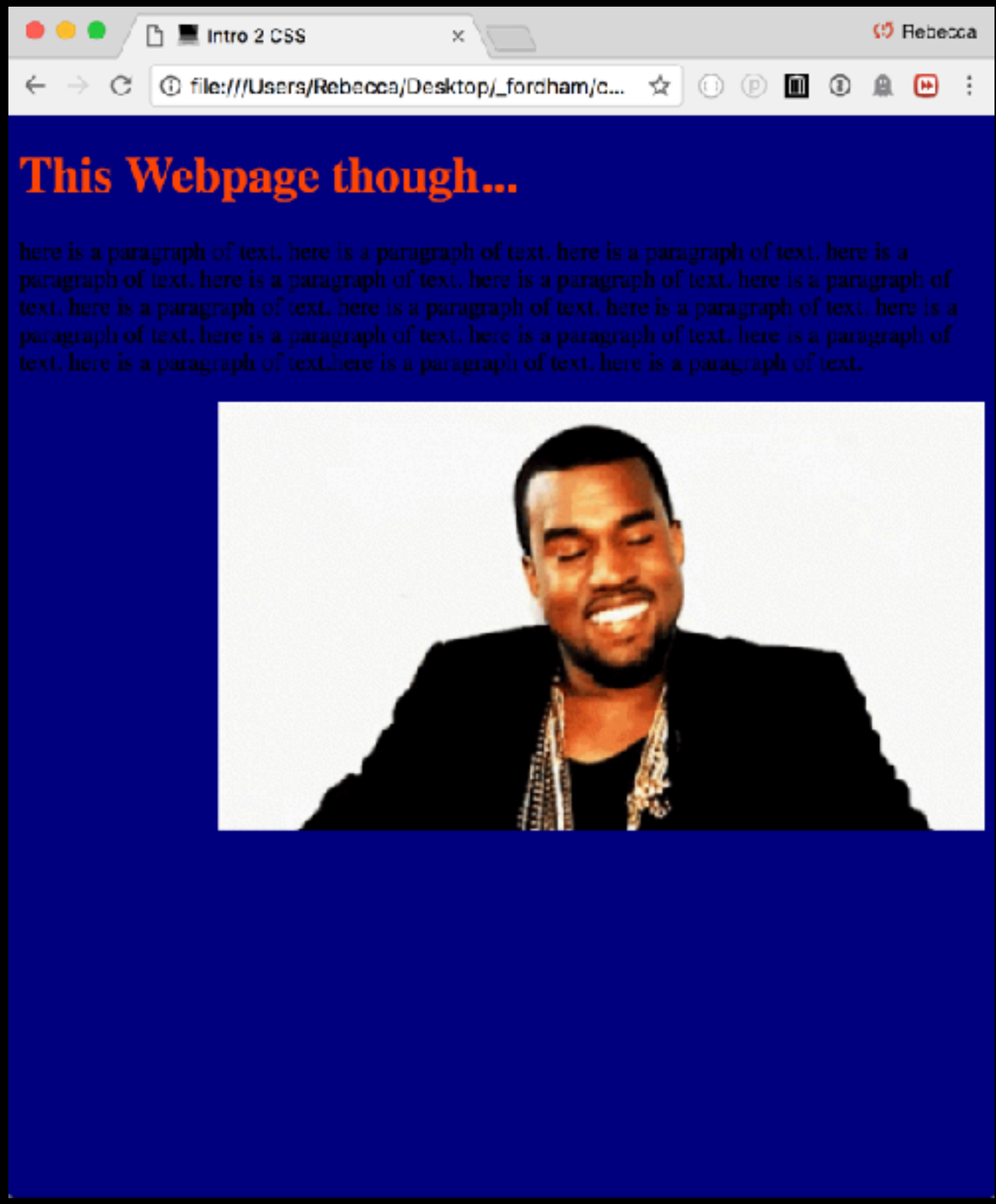
```
<html>
  <head>
        <title> 💻 Intro 2 CSS </title>
        <style type- "text/css">
            h1 {
                color: #FF4500
            }

            body {
                background: #000080;
            }

        </style>
    </head>
```

External Styles *

```
<head>
    <title> 💻 Intro 2 CSS </title>
  <link rel="stylesheet" type="text/css" href="theStyle.css">
 </head>
```



* Why Use External Style Sheet:

Same CSS for each HTML page
No need to copy or rewrite styling
Make changes to CSS automatically + can apply to the entire website
Faster download time for subsequent pages

style.css

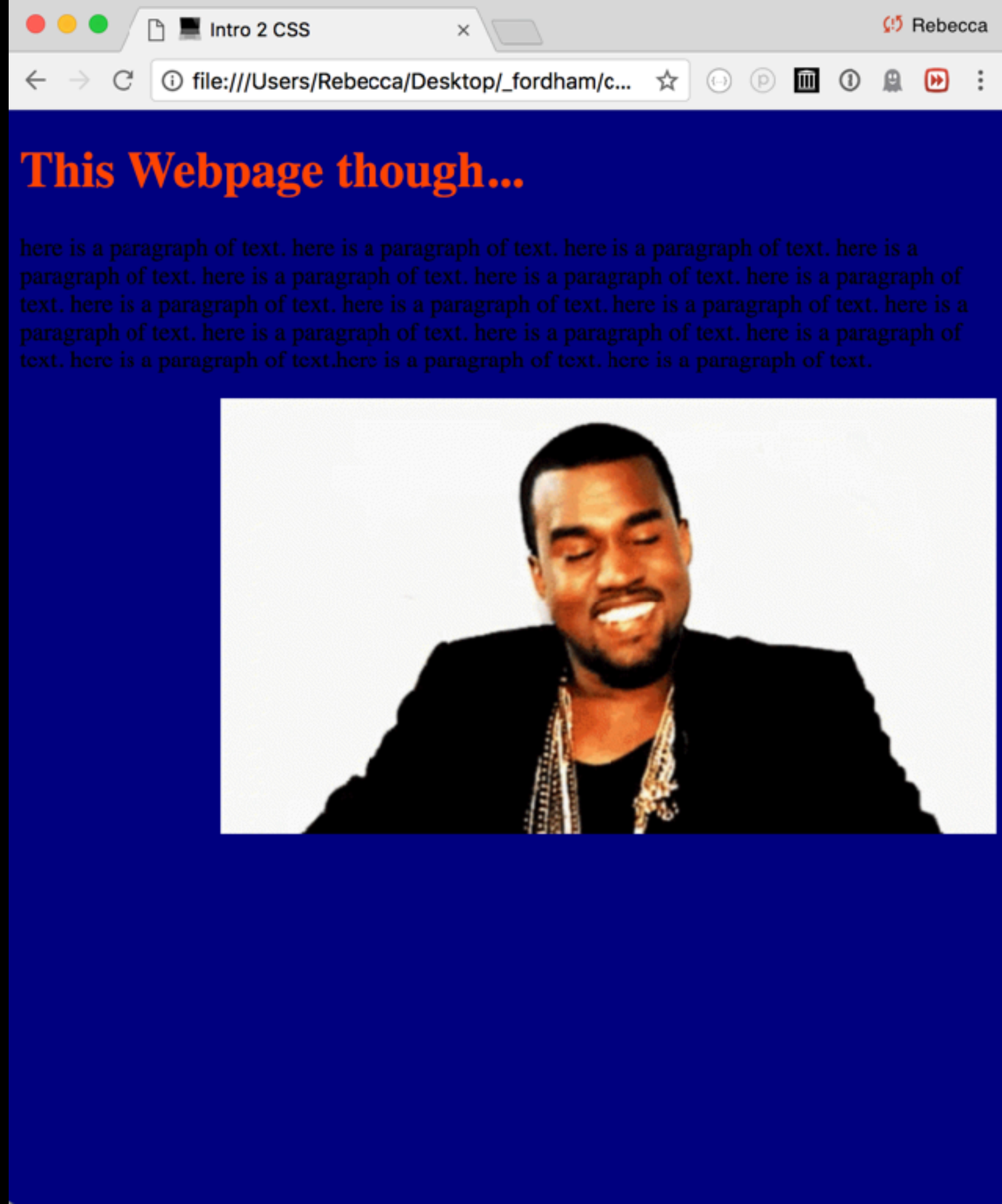
CSS SYNTAX:

selectors are used to find (select) HTML elements based on their element name, id, etc…

```
h1 {
    color: #FF4500
}

body {
    background: #000080;
}
```

selector { declaration }



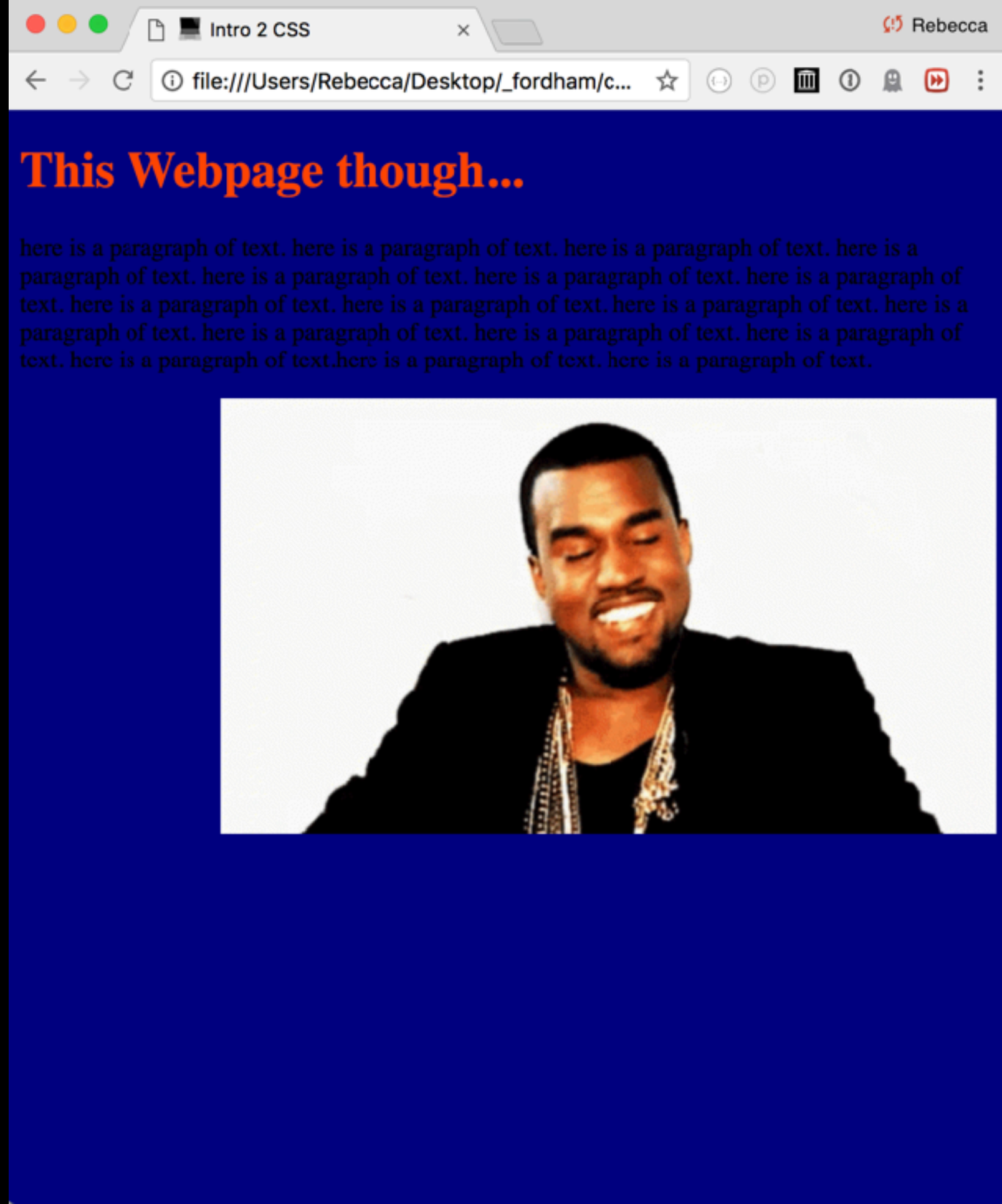
```
h1 {
    color: #FF4500
}

body {
    background: #000080;
}
```

```
selector {

  property: value ;

}
```



Selector is a term such as p, h1, div that identifies the HTML element you want to format or apply a rule to. You can add multiple selectors in a declaration.

Selector	Meaning	Example
Universal Selector	Applies to all elements in the document	* { }
Type Selector	Matches element names	h1, h2, h3 { }
Class Selector	Matches an element whose class attribute has a value that matches the one specified after the period (or full stop) symbol	.note { } targets any element whose class attribute has a value of "note} p.note { } targets only <p> elements whose class attribute has a value of "note"
ID Selector	Matches an element whose id attribute has a value that matches then specified after the # symbol	#introduction { } targets the element whose id attribute has value of "introduction"

Selector	Meaning	Example
Child Selector	**Matches an element that is a direct child of another**	li > a { } targets any <a> element that are children of an element (but not other <a> elements in the page.
Descendant Selector	**Matches an element that is a descendent of another specified element (not just a direct child of that element)**	p a { } targets any <a> elements that sit inside a <p> element, even if there are other elements nested btw them

Selector	Meaning	Example
Adjacent Sibling Selector	Matches an element that is the next sibling of another	h1+p { } targets the first <p> element after any <h1> element (but not other <p> elements)
General Sibling Selector	Matches an element that is a sibling of another, although it does not have to be the directly preceding element	h1~p { } tif you have two <p> elements that are siblings of an <h1> element, this rule would apply to both

```
/* type/element selector */
p {
    color: blue;
}

/* class attribute selector */
.blue-text {
    color: blue;
}

/* id attribute selector */
#blue-par {
    color: blue;
}

/* BONUS: grouping
selector */
p,
.blue-text,
#blue-par {
    color: blue;
}
```

selecting multiple elements:

```
h1, h2, h3 {

    color: red;
    background-color: blue;
    width: 500px;

}

p,
li {

    background-color: red;
    font-color: blue;


}
```

HTML comments are written like this

```
<!-- This is a comment -->
```

CSS comments are written like this

```
/* This is a comment */
```

```
{
text-align:

        left ;
        right ;
        center ;
        justify ;

}
```

```
{
vertical-align:

        baseline ;
        sub ;
        super ;
        top ;
        text-top ;
        middle ;
        bottom ;
        text-bottom ;


}
```

This property is NOT intended to allow you to vertically align text in the middle of a block level elements such as <p> + , although it does have this effect when used with table cells <td> + <th> elements.

It is more commonly used w/ inline elements such as <img>, or . When used with these elements, it performs a task very similar to the HTML align attribute used on the <img> element.

Interaction Design

a: link {

a: visited {

: hover {

Applied when a user hovers over an element w/ a mouse. This changes the appearance of links and buttons when a user places their cursor over them. Does not work on mobile.

: active {

Applied when an element is bingo activated by a user, like when a button is pressed or a link clicked. This added to UX.

: focus {

Applied when an element has focus. Any thing you can interact with.

Focus occurs when a browser discovers that you are ready to interact w/ an element. For example when yr cursor is in an input - that element is said mohave focus.

}

Classes and IDs

Two common attributes used to single out certain HTML elements are **class** and **id**, both are used to identify particular elements when adding CSS styling rules.**You author class + id names!!** They have no particular meaning in themselves, besides a puzzle - or code - you are creating.

Use a **class** when you have more than one element you want to share the same styling - perhaps across multiple pages.

Use an **id** when there is only one element on the page with that id, for example id="header"
With a class you can have as many elements with that styling as you like.

An element can have more than one **class**, but not more than one **id**. When there is more than one class, the class names are separated by spaces.

```html
<h1 id="myHeader">Hello World!</h1>
```

IDs

Every HTML element can carry the id attribute. It is used to uniquely identify that element from other elements on the page.

Its value should start with a letter or an underscore (not a number or any other character). It is important that no two elements on the same page have the same value for their id attributes (otherwise the value is no longer unique).

More to read on ID naming: https://mathiasbynens.be/notes/css-escapes

IDs

To select these IDs in CSS you would do so with **#myHeader** syntax

(IDs may become particularly useful when it comes to media elements - photos, videos + sound files.)

```
#myHeader{
  color: blue;
}
```

Classes

```
<div class="theAuthor">
  -- from John Duckett's <span><a
  href="https://www.amazon.com/Web-Design-HTML-JavaScript-jQuery/dp/1118907442
  /ref=sr_1_3?ie=UTF8&qid=1526310943&sr=8-3&keywords=html+and+css"
  target="_blank">HTML + CSS</span></a>
  <br>
</div>
```

To select these classes in CSS you would do so with .theAuthor syntax

```
.theAuthor{
  background: rgb(255,255,255);
    /* HSL: Hue, Saturation + Lightness
    Hue - as an angle between 0 + 360
    Saturation - as a precentage
    Lightness - as a precentage: 0% = white, %50 = normal + 100% is bl
    Alpha - expressed btw 0 _ 1.0 : 0.5 = 50% transparency, .75 is 75%
    transparency*/
  background: hsl(0,100%,100%, 0.2);
  text-align: center;
}
```