HTML & CSS

Martin Hutchings

Day Three

- Color codes
- Inline and Block Elements
- Float
- Flex

Repetition Questions

#1

Re-arrange these in the correct order, from least specific to most specific:

`id`, `element`, `class`, `!important`, `style=""`

Repetition Questions

#2

What are the 3 ways to use CSS in an HTML file?

Repetition Questions

#3

What is the different between a `class` and an `id`?

BONUS: Name typical use-cases for both.

Repetition Questions

#4

What is a pseudo class?

BONUS: Name 2 examples

Repetition Questions

#5

What is a pseudo element?

BONUS: Name 2 examples

Repetition Questions

#6

Name **5** different relative units of measurement in CSS.

Repetition Questions

#7

In *general* which properties are inherited *[Werden vererbt]* in CSS
BONUS: Which are not?

Repetition Questions

Answers pt.1

1. `element` < `class` < `id` < `style=""` < `!important`
2. `<style>`, `<link>` and `style=""`
3.
   - *Classes* can be used multiple times, *id's* are unique!
   - *Classes* are less specific than *id's*
   - *Classes* = `.foo{}`, id's = `#foo{}`
   - Multiple *classes* can be used for one element
   - Use *id's* for elements that appear in the page only once

Repetition Questions

Answers pt.2

4. An state *[Zustand]* of an element.
    - i.e. `:hover` or `:visited`
5. Content that is not defined in the HTML
    - i.e. `::before`, `::after`, `::first-line`
6. `px`, `em`, `rem`, `%`, `vh`, `vw`
7. Typography properties
    - Layout Properties are not

Color Codes

Hexadecimal Code

- `color: #000000;` (Black) - `color: #ffffff;` (White)
- Each digit has 16 levels:
  - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
  - (16 x 16) x (16 x 16) x (16 x 16) = 256 x 256 x 256 = 16.78 M colors
- Can be simplified to 3 digits (i.e. `#222` = `#222222`)
- Can have alpha: https://caniuse.com/css-rrggbbaa

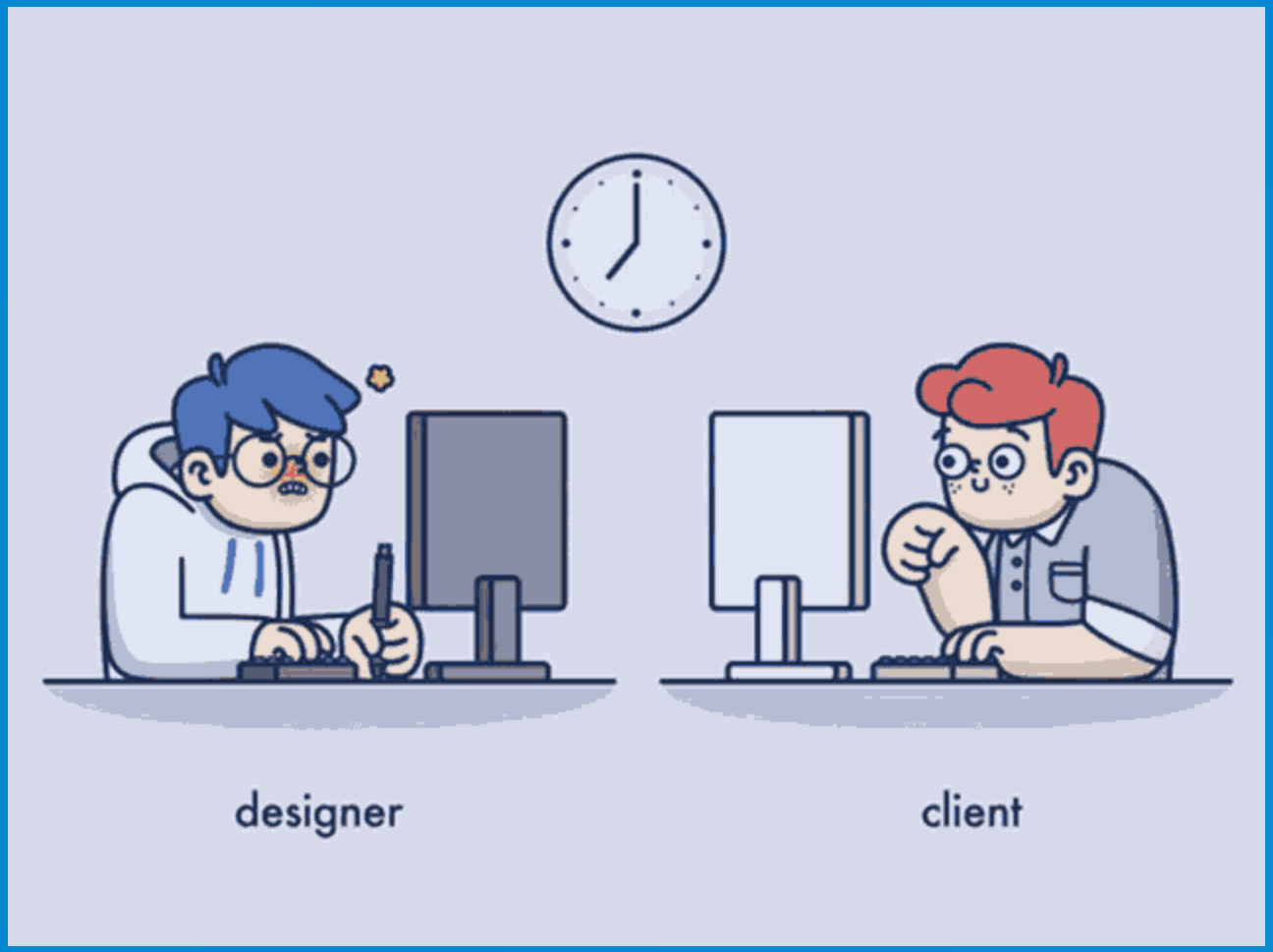
Color Codes

RGB Colors

- `color: rgb(255, 0, 0);` (Red)
- `color: rgba(255, 0, 0, 0.5);` (Red with 50% transparency)
- RGB values: `0 - 255`
- α (alpha) value: `0 - 1`
- Remember to check inspector

Inline and Block Elements

`display:`



Block Elements

- Structural elements:
  - i.e. `div`, `p`, `form`, `ul`, `li`, `h1`, Semantic Tags
- Are as **WIDE** as **possible**
  - Same width as parent elements
  - Create a new line
- Are as **HIGH** as **necessary**
- Can have `padding` and `margin`

Inline Elements

- Usually text-based elements:
  - i.e. `span`, `a`, `em`, `strong`, `label`
- Ignore **WIDTH** and **HEIGHT**
  - Don't create a new line
  - Height dependent on `line-height`
- Can be placed in the middle of texts

Inline-Block Elements

- A combination of both Block and Inline
  - i.e. `img`
- Can have **WIDTH** and **HEIGHT**
  - Don't create a new line
  - Can have `margin` and `padding`
- Can be placed in the middle of texts

The `display` Property

- Can be overwritten (you can make `div`'s inline)
- Most common uses include :
  - `block`
  - `inline`
  - `inline-block`
  - `flex`
  - `grid`

Float

Let block-elements float

- Default: `float: none`
- Start a float:
  - `float: left | right | both`
- Stop a float
  - `clear: left | right | both`

Flex

Easy way

The Ultimate Guide

Homework