Lecture 17: Typesetting I

COSC 225: Algorithms and Visualization

Spring, 2023

Announcements

1. Assignment 08 not posted, now optional!
2. No Quiz Today
3. Final Projects
    - formal announcement by Wednesday
    - group assignments, tentative topic due Friday

Today

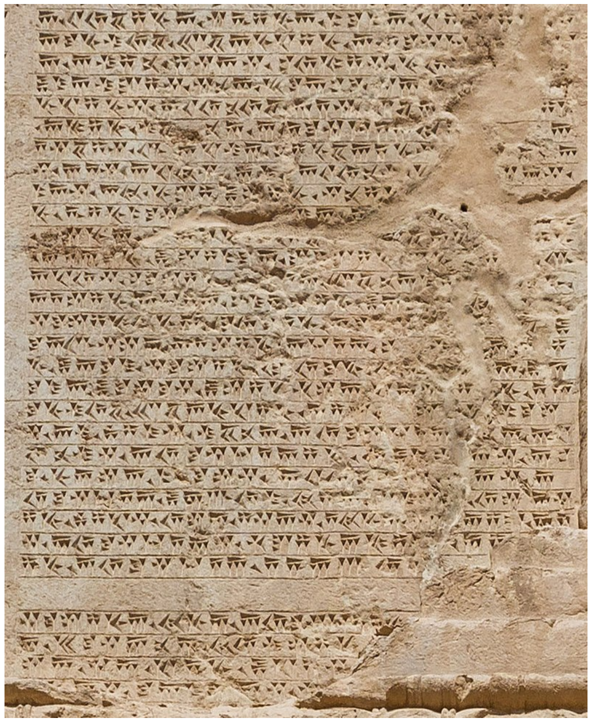
Typesetting I: Breaking paragraphs into lines

1. Intoduction to the problem
2. Some technology
3. Activity: greedy line breaking
4. Can we do better?

What Is Typesetting?

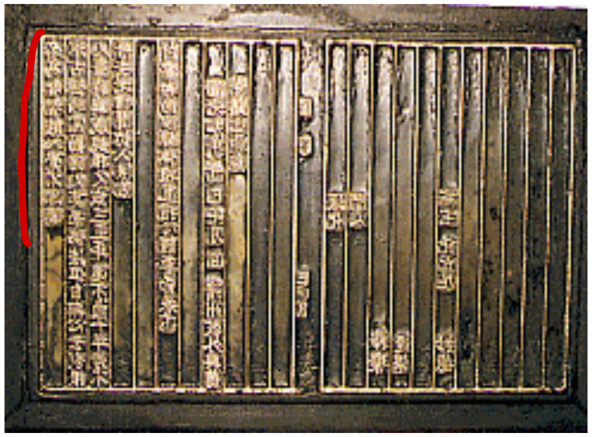
The process of arranging letters and words onto some medium.

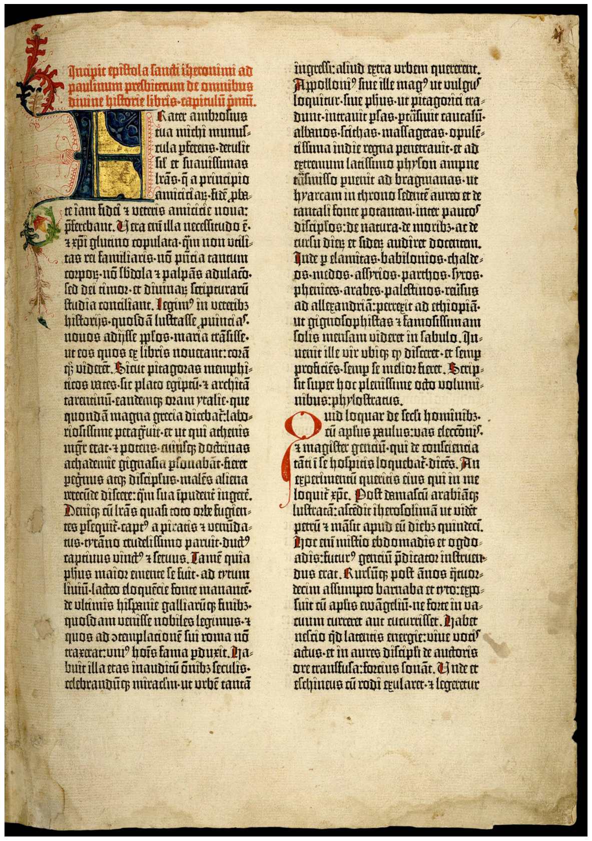
Hand Lettering



Persian cuneiform and Malnazar illuminated manuscript

Movable Printed Type

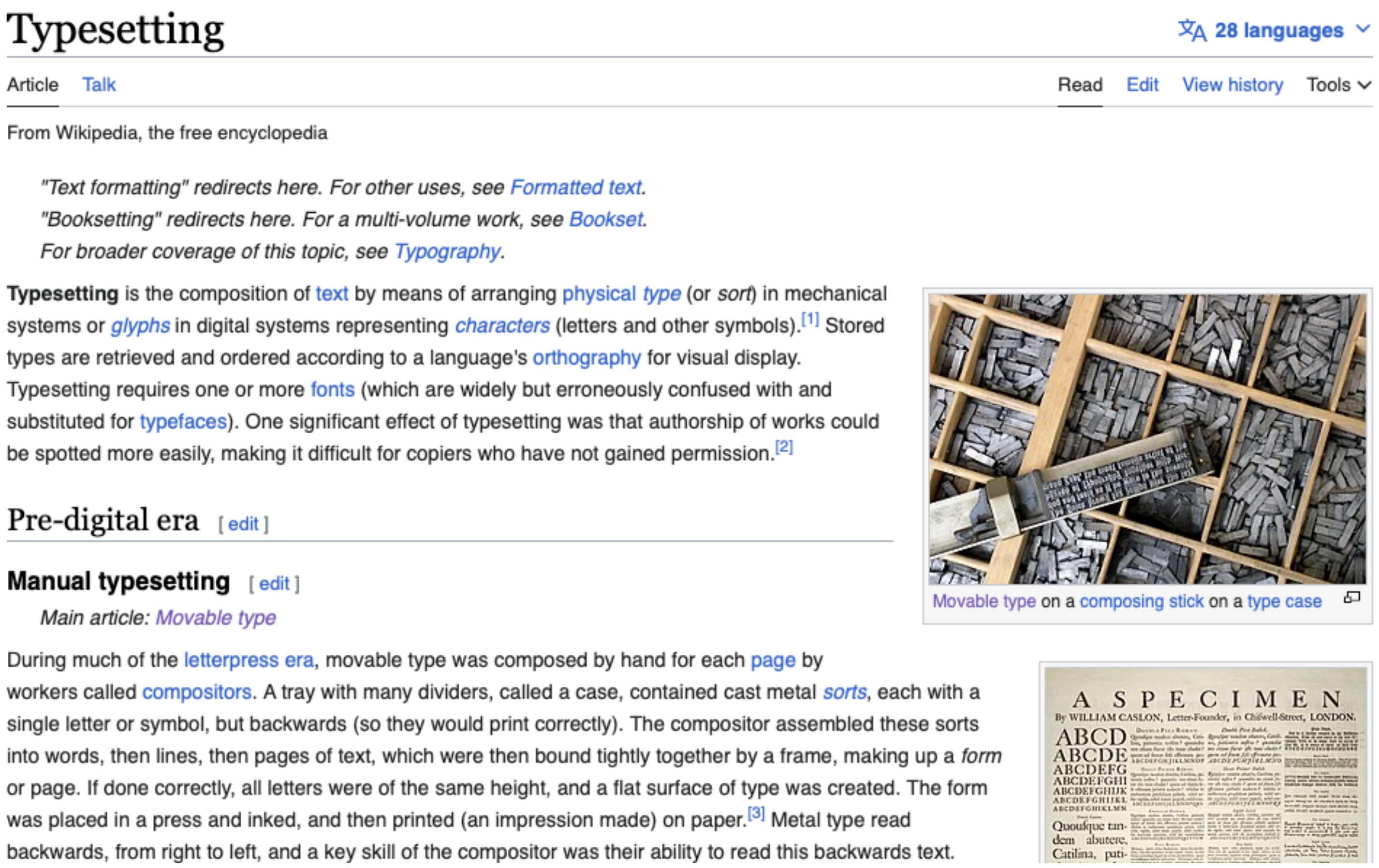




Incipit epiſtola ſancti iheronimi ad
paulinum preſbiterum de omnibus
divine hiſtorie libris·capitulū p̄mū.
Frater ambroſius
tua michi munuſ-
cula pferens·detulit
ſil' et ſuauiſſimas
lrās·q̄ a principio
amicicia℞·fidē pba-
te iam fidei ꝛ veteris amicicie noua:
pferebant. Uera eñ illa neceſſitudo ē·
ꝛ xp̄i glutino copulata·q̄m non vtili-
tas rei familiaris·nō p̄ncia tantum
corpo℞·nō ſbdola ꝛ palpās adulacō·
ſed dei timor·et diuina℞ ſcripturarū
ſtudia conciliant. Legim⁹ in veteribꝫ
hiſtorijs·quoſdā luſtraſſe puincias·
nouos adijſſe pplos·maria trāſiſſe·
ut eos quos ex libris nouerant: corā
q; viderēt. Sicut pitagoras memphi-
ticos vates·ſic plato egiptū·ꝛ architā
tarentinū·eandemq; oram ytalie·que
quondā magna grecia dicebat̄: labo-
rioſiſſime peragrauit·et ut qui athenis
mḡr erat·ꝛ potens·cuiuſq; doctrinas
achademie gignaſia pſonabāt·fieret
pegrinus atq; diſcipl'us·malēs aliena
verecūde diſcere: q̄m ſua īpudent̄ ingerē.
Deniq; cū lrās quaſi toto orbe fugien-
tes pſequit̄·capt⁹ a piratis ꝛ venūda-
tus·tyrāno crudeliſſimo paruit·duct⁹
captiuus vinct⁹ ꝛ ſeruus. Tamē quia
pħus maior emente ſe fuit·ad tytum
liuiū·lacteo eloquēcie fonte manantē·
de vltimis hiſpanie galliarūq; finibꝫ·
quoſdam veniſſe nobiles legimus·ꝛ
quos ad ꝯtemplacionē ſui roma nō
traxerat: vni⁹ hoīs fama pduxit. Ha-
buit illa etas inauditū ōnibꝫ ſeculis·
celebrandūq; miraculū·ut urbē tantā
ingreſſi: aliud extra urbem quererent.
Appolloni⁹ ſiue ille mag⁹ ut vulgus
loquitur·ſiue pħus·ut pitagorici tra-
dunt·intrauit pſas·ptrāſiuit caucaſū·
albanos·ſcithas·maſſagetas·opulē-
tiſſima indie regna penetrauit·et ad
extremum latiſſimo phyſon amne
trāſmiſſo puenit ad bragmanas·ut
hyarcam in throno ſedentē aureo et de
tantali fonte potantem·inter paucos
diſcipl'os: de natura·de moribꝫ·ac de
curſu die℞ et ſyde℞ audiret docentem.
Inde p elamitas·babilonios·chalde-
os·medos·aſſyrios·parthos·ſyros·
phenices·arabes·paleſtinos·reuerſus
ad allexandriā: perrexit ad ethiopiā·
ut gignoſophiſtas ꝛ famoſiſſimam
ſolis menſam videret in ſabulo. In-
uenit ille vir vbiq; q̄d diſceret·et ſemp
proficiēs·ſemp ſe melior fieret. Scrip-
ſit ſuper hoc pleniſſime octo volumi-
nibus: phyloſtratus.
Quid loquar de ſeculi hominibꝫ·
cū apl'us paulus: vas electōnis·
ꝛ magiſter gencium·qui de conſciencia
tāti i ſe hoſpitis loquebat̄·dicēs. An
experimentū queritis eius qui in me
loquit̄ xp̄c. Poſt damaſcū arabiāq;
luſtratā: aſcēdit iheroſolimā ut vidēt
petrū ꝛ māſit apud eū diebꝫ quindecī.
Hoc eñ miſtio ebdomadis et ogdo-
adis: futur⁹ gencium p̄dicator inſtruen-
dus erat. Rurſumq; poſt ānos q̄tuor-
decim aſſumpto barnaba et tyto: expo-
ſuit cū apl'is ewāgeliū·ne forte in va-
cuum curreret aut cucurriſſet. Habet
neſcio q̄d latentis energie: viue vocis
actus·et in aures diſcipli de auctoris
ore tranſfuſa: forcius ſonāt. Unde et
eſchineus cū rodi exularet·ꝛ legeretur

Jikji type and Gutenberg bible

Computer Typesetting



Typesetting

From Wikipedia, the free encyclopedia

"Text formatting" redirects here. For other uses, see Formatted text.
"Booksetting" redirects here. For a multi-volume work, see Bookset.
For broader coverage of this topic, see Typography.

Typesetting is the composition of text by means of arranging physical *type* (or *sort*) in mechanical systems or *glyphs* in digital systems representing *characters* (letters and other symbols).[1] Stored types are retrieved and ordered according to a language's orthography for visual display. Typesetting requires one or more fonts (which are widely but erroneously confused with and substituted for typefaces). One significant effect of typesetting was that authorship of works could be spotted more easily, making it difficult for copiers who have not gained permission.[2]

Movable type on a composing stick on a type case

Pre-digital era [edit]

Manual typesetting [edit]

Main article: Movable type

During much of the letterpress era, movable type was composed by hand for each page by workers called compositors. A tray with many dividers, called a case, contained cast metal *sorts*, each with a single letter or symbol, but backwards (so they would print correctly). The compositor assembled these sorts into words, then lines, then pages of text, which were then bound tightly together by a frame, making up a *form* or page. If done correctly, all letters were of the same height, and a flat surface of type was created. The form was placed in a press and inked, and then printed (an impression made) on paper.[3] Metal type read backwards, from right to left, and a key skill of the compositor was their ability to read this backwards text.

A SPECIMEN
By WILLIAM CASLON, Letter-Founder, in Chiswell-Street, LONDON.

Typesetting Goals?

Question. What goals might we have in typesetting? $\downarrow$

- Readability + Accessibility $\leftarrow$ size, color, shape/font/style
- Aesthetic Value
  $\hookrightarrow$ layout (esp. incorporating media)
- Responsiveness
- spacing between: letters, words, lines

Knuth on Typesetting

> *[We are] primarily concerned with high-quality technical manuscripts. [...] If you merely want to produce a passably good document—something acceptable and basically readable but not really beautiful—a simpler system will usually suffice. [...T]he goal is to produce the finest quality; this requires more attention to detail, but you will not find it much harder to go the extra distance, and you'll be able to take special pride in the finished product.*

- Knuth, *The TeXBook*

TeX

LaTeX

L. Lamport

Our Basic Task

Breaking paragraphs into lines.

Input: A plain text paragraph.

- possibly: individual typeset words

Output: locations of line-breaks

- possibly: placement of words on the page

Goals:

- readability
- beauty?
- ???

Lorem Ipsum

Placeholder text:

```
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do
eiusmod tempor incididunt ut labore et dolore magna aliqua.
Felis pellentesque suspendisse tristique in pulvinar erat integer
pellentesque nunc viverra auctor semper. Tempus eros ullamcorper
mauris turpis lacinia dictumst consequat proin facilisis et
conubia curabitur quisque egestas nullam. Nibh erat sodales
maecenas quis pulvinar auctor imperdiet platea litora id leo. Per
fusce lectus ex cursus urna fusce scelerisque. Dolor tempus augue
sit orci elit porttitor ipsum platea erat.
```

Nonsense Latin, commonly used to demonstrate graphic design

Parameters?

Question. In order to accomplish our task what parameters should we take into account?

- locations of words ← individual letters?
  - sizes (widths)
- width frame (text box)
- whitespace – for different scenarios? e.g. between words
- Fancy: floating figures/ location

How to Break Lines?

Question. Given those parameters, what procedure would you use to find line breaks?

- running total of letters' widths + blank space
- fill a line to max width w/out overflow

Greedy line breaking

→ adapt to hyphens & word breaking.

HTML/CSS/JS Stuff

Parameters:

- original text, `TEXT` (string)
- document element ← where to place words
    - `TEXT_WIDTH` maximum width of text
    - font, size, etc
    - `WORD_SEP` minimum space between words
    - ...

What more do we need?

<p> word </p>

Need: HTML elts for words

split TEXT into array @ whitespace

→ make elt for each word (string)

Step 1: Separate Words

Idea. Process TEXT

- break into words (use whitespace)
- create an element for each word
- put elements in an array

Questions.

- How to break the string?

"split" method for arrays

→ regular expression

- What type of element for each word?

<span> — generic inline

block

match 1 or more whitespace chars

Getting a <span> array

<span> are (default) *inline* document divisions

```
function getSpanArray (text) {
    const words = text.split(/\s+/);
    const spans = [];
    for (let word of words) {
        let elt = document.createElement("span");
        elt.innerText = word;
        spans.push(elt);
    }
    return spans;
}
```

What Next?

We have

- a `<span>` array, `spans`
- a `parent` element (say, `<p>` or `<div>`)

How to place the `<span>`s on the page?

What Next?

We have

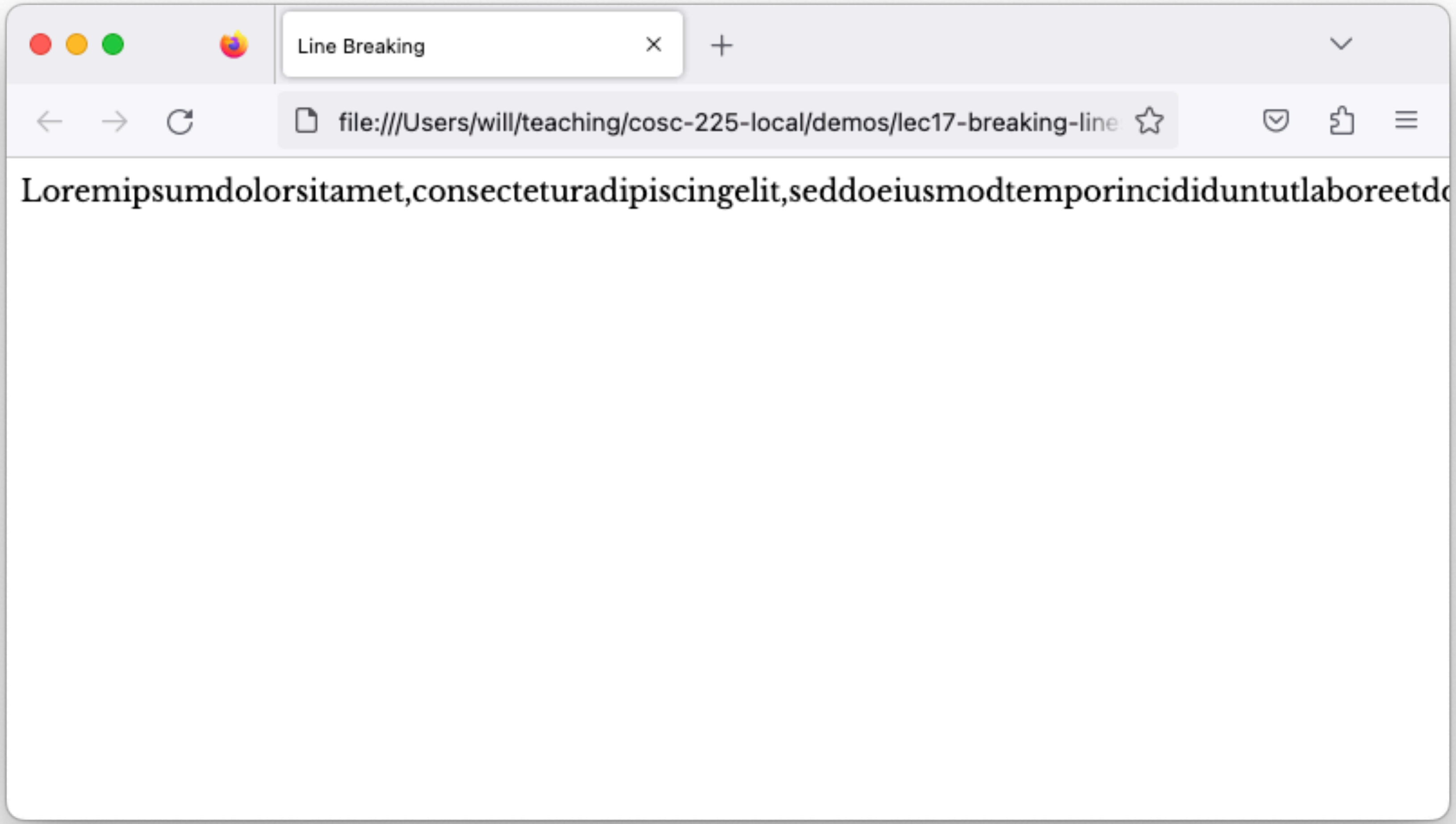
- a `<span>` array, `spans`
- a `parent` element (say, `<p>` or `<div>`)

How to place the `<span>`s on the page?

```
for (let s of spans) {
    parent.appendChild(s);
}
```

The Result

$<br/>$


Line Breaking
file:///Users/will/teaching/cosc-225-local/demos/lec17-breaking-line

Loremipsumdolorsitamet,consecteturadipiscingelit,seddoeiusmodtemporincididuntutlaboreetd

M

Oof

What do we need to fix and how?

Div for each line

- add spans to fill + (extra space)

line

- new div for each line

Getting Span Widths

We can get the dimensions of an HTML element with

```
let width = elt.getBoundingClientRect().width;
```

...but `elt` must alread be placed on the document!

Question. How to do this before we've decided where to place `elt`?

CSS Hacking:

- change display: none,
  hidden

A Trick

Create a `hidden` element

```html
<div id="hidden"></div>
```

CSS:

```css
#hidden {
    position: absolute;
    visibility: hidden;
}
```

← elt is there, but not displayed

JavaScript:

```javascript
const hidden = document.querySelector('#hidden');
for (let s of spans) {
    hidden.appendChild(s);
}
```

Widths, At Last

Finally, we can get the width of each word (in pixels).

Question. Now What?

- make divs for lines
- add spans to each line

figure out space between words?

New Units

So Far. We've mostly measured lengths in pixels (`px`)

Problem. Space between words (e.g.) should *scale* with the size of the font.

- font sizes may change

Built-in Solution. Distance unit `em`

- the width of a capital letter 'M'

Problem : convert between em and px

New Units

So Far. We've mostly measured lengths in pixels (`px`)

Problem. Space between words (e.g.) should *scale* with the size of the font.

- font sizes may change

Built-in Solution. Distance unit `em`

- the width of a capital letter 'M'

Getting the width of an `em` inside `parent` in pixels:

```
const em = parseFloat(getComputedStyle(parent).fontSize);
```

stores value of one "em" width in px.

Activity: Greedy Linebreaking

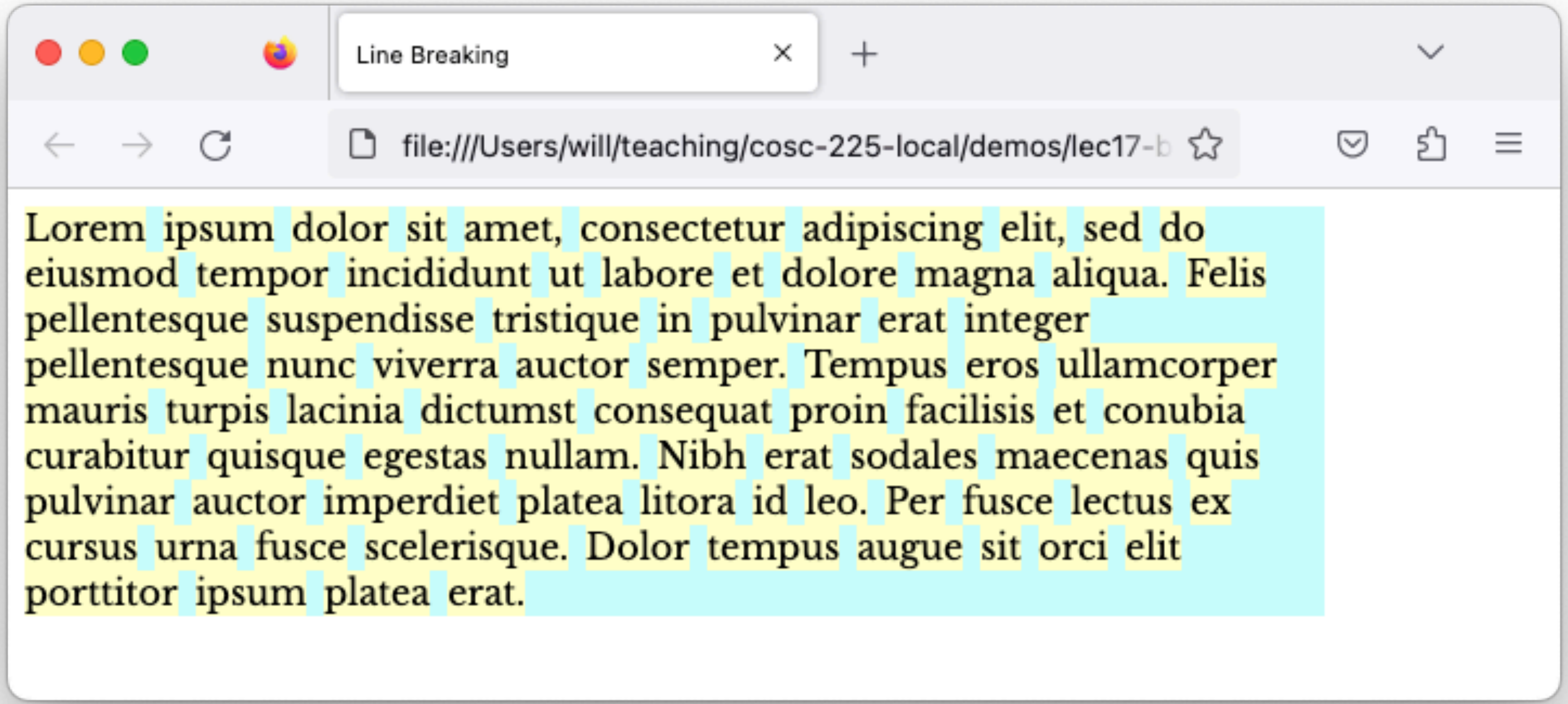
Your Task

- download `lec17-breaking-lines.zip`
- write a method `greedyLines(spans, parent)`
  - `spans` an array of `<span>` elements, one per word
  - `parent` an element (`<div>`) that will contain words
- words are displayed in lines
- each line is at most `TEXT_WIDTH` pixels wide
- words are separated by `0.5 em` space

More:

- indent first line of the paragraph
- print “justified” text instead of “ragged right”

Example Output



Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do
eiusmod tempor incididunt ut labore et dolore magna aliqua. Felis
pellentesque suspendisse tristique in pulvinar erat integer
pellentesque nunc viverra auctor semper. Tempus eros ullamcorper
mauris turpis lacinia dictumst consequat proin facilisis et conubia
curabitur quisque egestas nullam. Nibh erat sodales maecenas quis
pulvinar auctor imperdiet platea litora id leo. Per fusce lectus ex
cursus urna fusce scelerisque. Dolor tempus augue sit orci elit
porttitor ipsum platea erat.

Suggestions

1. Use JavaScript to decide where to break lines and create auxiliary elements
2. Use CSS to actually place the elements
    - `display`
    - `margin`
    - `justify-content`

How’d It Go?

Tricks: Word Spacing in CSS

```css
span {
    margin: 0em 0em 0em 0.5em;
}

span:first-child {
    margin: 0em;
}

.first-line > span:first-child {
    margin: 0em 0em 0em 2em;
}
```

Fonts

- Google Fonts demo

Global Parameters/Variables in CSS

```css
:root {
    --light-blue: rgb(200,255,255);
    --mild-yellow: rgb(255, 255, 200);
    /* font-family: 'EB Garamond', serif; */
    font-family: 'Libre Baskerville', serif;
    font-size: 16px;


}
```

```css
span {
    margin: 0em 0em 0em 0.5em;
    background-color: var(--mild-yellow);
}
```

For Your Consideration

Can we do a better job of breaking paragraphs into lines?

- What room is there for practical or aesthetic improvement?