

DS&A: Trees

JS 401d13

Data Structures

Abstracted representation of reality

- Build a simulation of reality
- o Include just what we need
- Leave out unnecessary details
- Some general solutions are repeatedly useful:
- Strings, Arrays, key-value pair sets
- Linked lists, Stacks, Queues
- o Graphs, Trees
- Custom object constructors



```
(class-14) > tree

── LICENSE

- README.md
— dist

— admin.html

   - data
      hackerIpsum.json
       ipsumArticles.json

─ index.html

   - new.html
   - package.json
   - scripts
       i aboutController.js

→ article.js

→ articleController.js

→ articleView.js

       — githubToken.js
       ─ repo.js
       ├── repoView.js
       - routes.js
       - webdb.js

─ server.js

       ─ base.css

─ icomoon.eot

          icomoon.svg
          - icomoon.ttf
          - icomoon.woff
       ├─ icons.css

─ layout.css

─ modules.css

   - vendor

─ handlebars.js

          highlight.pack.js

→ html5sql.js

— jquery-2.1.4.js

          - marked.is
```

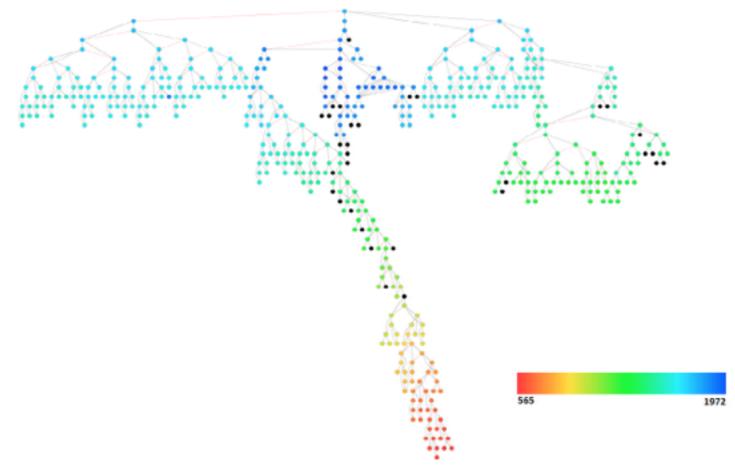
Abstraction





Reality





Abstraction





Reality



```
<100CTYPE HTML>
     civinla
       cheads
         ctitle>Chocolate Pizzac/title>
         clink rel="stylesheet" type="text/css" href="style.css" />
       cheads
       <body>
         editys
           cheaders
           cdiv class="all-things">
10
             cdiv class="image-logo">
               <ing src="images/logo.png"/>
             </div>
14
             <div id="tagline">
               <h3>cemp@elicious</em></h3>
               op class="slogan">THE BEST FOOD BLOG ON THE WEBc/p>
             c/div>
           c/div>
18
             cdiv class="line"></div>
           ediv class="all-images">
28
             cdiv class="social-images">
               cing src*"images/fb-icon.png"/>
               <img src+"images/twit-icon.png"/>
               cing src="images/gp-icon.png"/>
2.5
               <ing src="images/insta-icon.png"/>
               cing src="images/flic-icon.png"/>
               <ing src="images/pint-icon.png"/>
28
             </div>
               ediv class="rss-mail">
                 <ing src="images/rss-icon.png"/>
30
                <img src="images/mail-icon.png"/>
               c/diva
             c/div>
34
           </header>
               <section>
36
                 ch2>Chocolate Pizzac/h2>
                 odiv class="recipe-title">
                   cdiv class="posted-on">
38
```

Abstraction







Chocolate Pizza

PROTECTION TAINED DRIVEN DESIGNATION

BOTH



For the fig-swirl: Melt butter over medium heat in a saucepan. Add brown sugar and stir to dissolve. Halve all of the figs and toss in the saucepan with water and lemon juice. Cook over medium heat, stirring frequently, until you have a chunky-jammy mixture. Add salt with one or two stirs, set aside and let cool completely.

What is a Tree data structure?

- Graph:
 - a set of "nodes" and connections between nodes.
- Tree:
 - A directed, acyclic, connected graph.

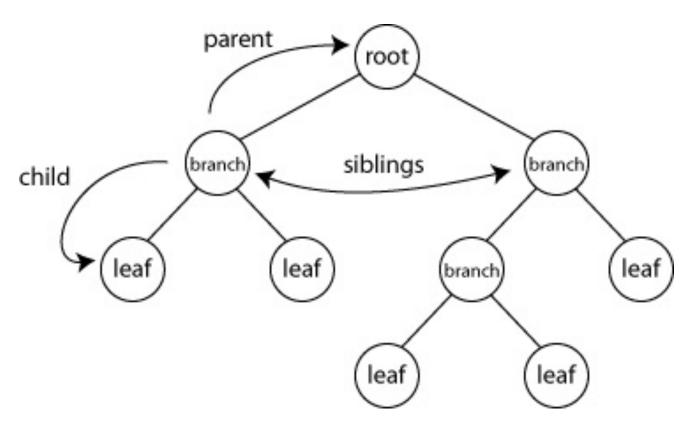
What is a Tree data structure?

- Can be defined recursively
- Tree:
 - A value, and (possibly empty) list of trees
 - The sub-trees are called "children"

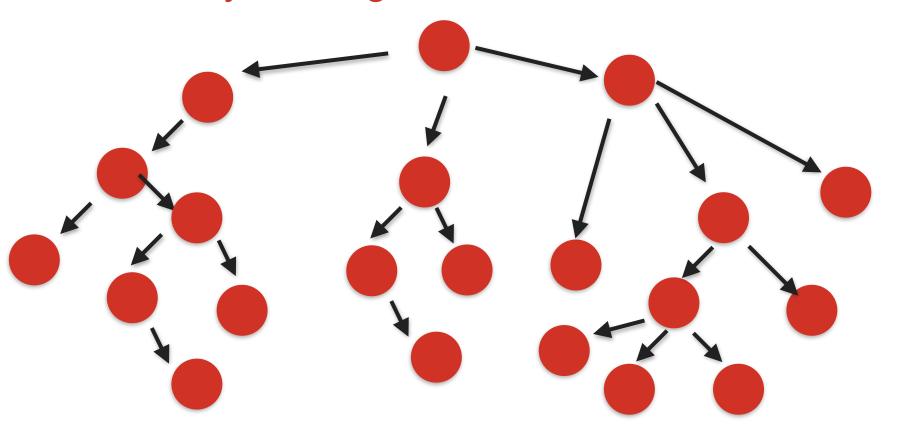
What is a Tree data structure?

- Root: The top node in a tree
- Child: A node that's a direct sub-tree of another node
- Siblings: A group of nodes with the same parent
- Leaf: A node with no children

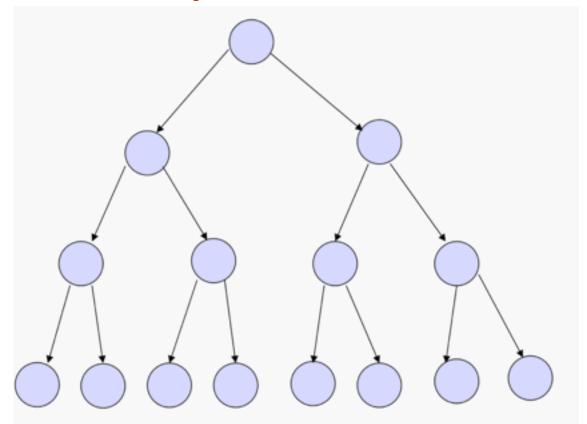




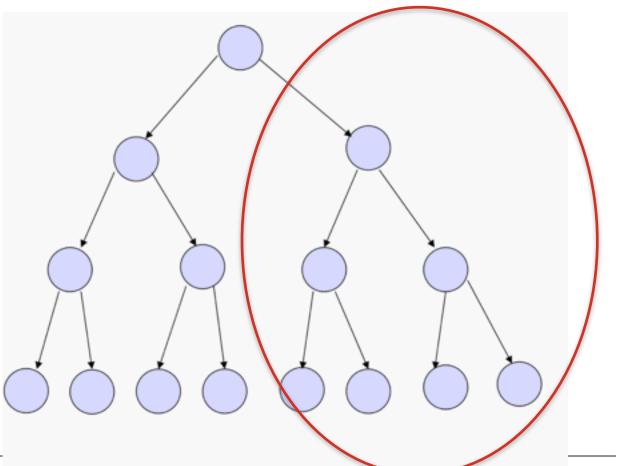
Tell your neighbor: Describe this tree



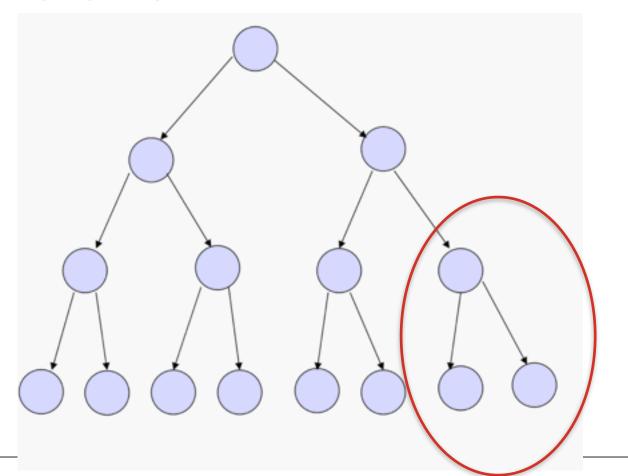
What do you see here?



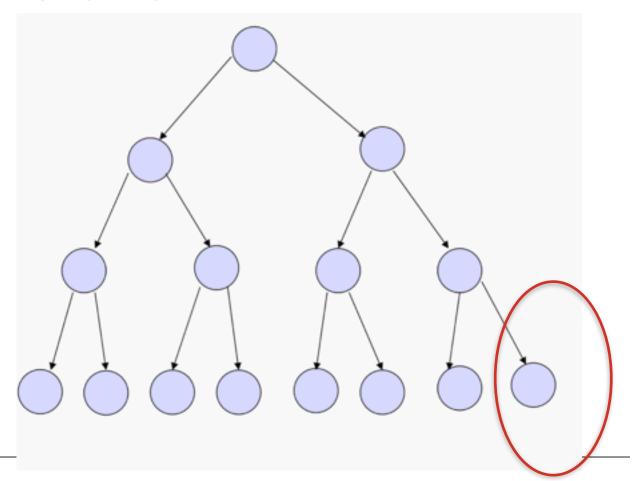
and this...?



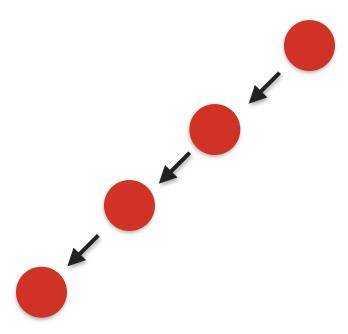
and then...?



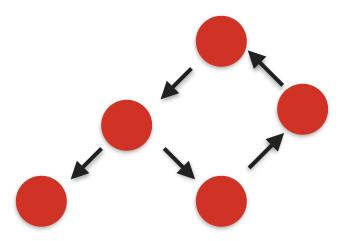
and then...?



What about this?



This?



Trees are handy!

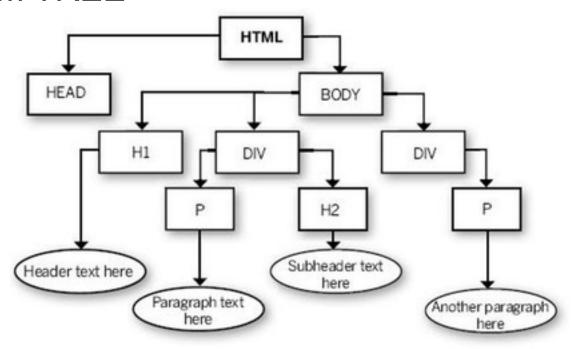
The DOM

Computer file system

Sorting and searching

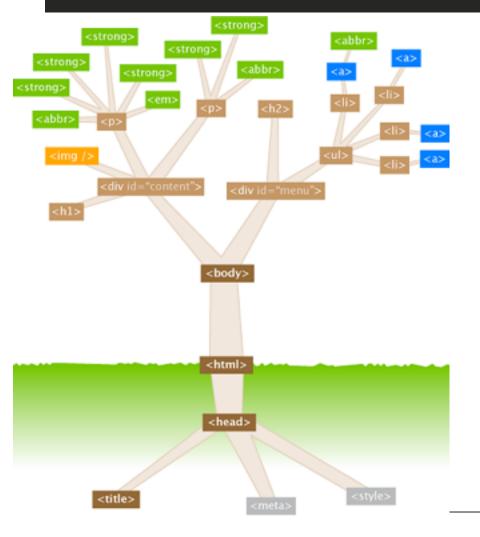


DOM TREE



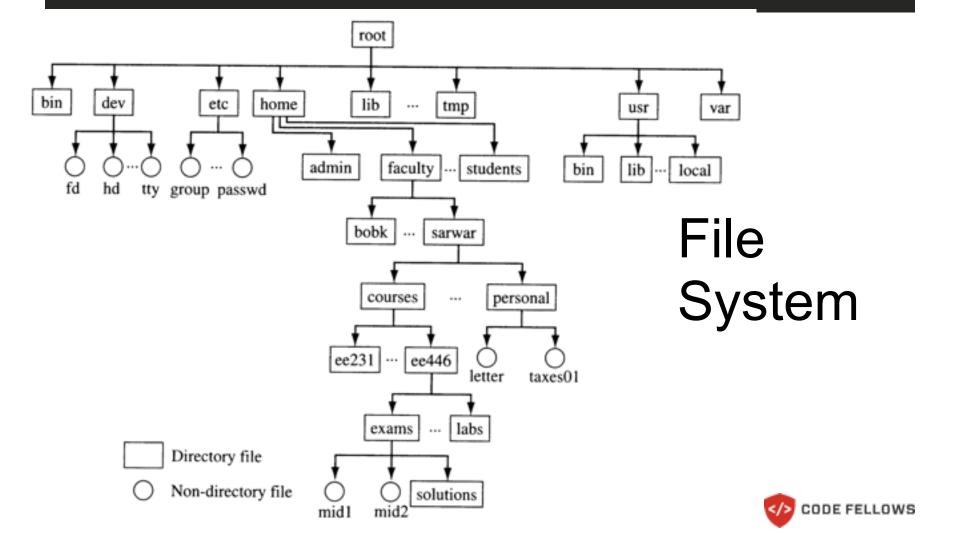
Level order: html, head, body, h1, div, div, header text, p, h2, p, para text, subheader text, another para





What is the root element of this tree?





Tree node as an object literal

object that holds a value and references to child nodes

```
{
  value: null,
  children: [ ]
}
```

Tree node as an object literal

```
value: { any: valStoredHere },

children: [ tree1138, tree42, tree9001 ]
}
```



Tree Operations

- Traversal: iterating through every node
- Insert and Delete: add/remove nodes at a position
- Search for a value, min, max
- Calculate properties: size, depth, height

```
var D = {
      data: 'D',
       children: []
    var B = {
       data: 'B',
       children: [D]
9
10
11
    var C = {
12
13
14
15
16
      data: 'C',
       children: []
    var A = {
       data: 'A',
18
       children: [B, C]
```

Consider this set of nodes

1. Which of the nodes is the root? How do you know?

2. Draw a tree diagram for these nodes similar to the tree below:

