# Wireframes

Building from a picture

# Some questions when you start a project (but not all of the questions)

- Are you Full stack or Front end for this ?
- Logic or Presentation or both ?
- Extending existing product or new?
- Actual designer involved?
- ...Who has non-print-based design training/experience?
- ...And UX/HCI training/experience?
- Some Client/Executive that has to be happy regardless ?

# Everyone is terrible at judging design & appearance

...so just expect that

...and don't forget that you are part of "everyone"

...Everyone can also notice a real issue or have a great thought

Because we all interact with things all the time - we all have vast amounts of very different experiences

# 4 common approaches (but there are others)

When making an idea a reality: (i.e. making a web page/app from it)

- Wireframes
- Mockups
- Redlines
- Prototypes

# Rectangles

Web UI is a bunch of boxes (this is actually true for all UI dev that I know of)

Boxes can contain boxes, one or many (and those can contain, and those…)

Boxes can be be full width, full height, or partial (or more than full!)

Boxes can be able to change height/width, or not

Boxes can be sent to the left, to the right, or to the center

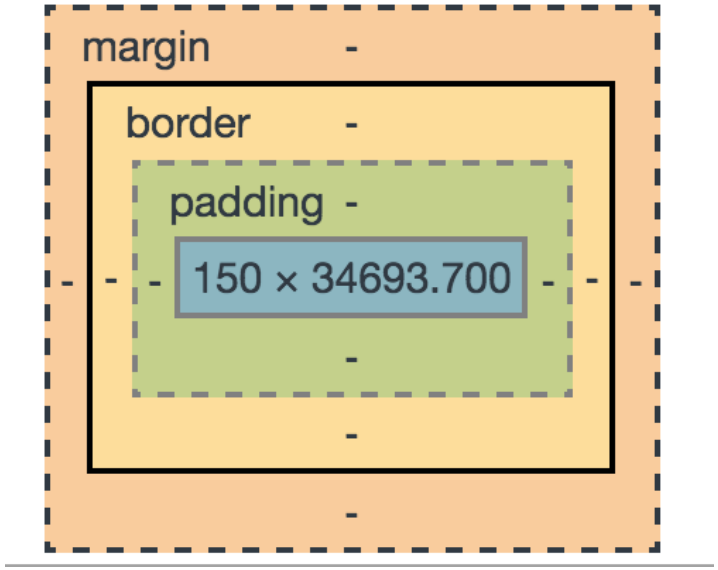
Boxes can have paddings and margins (which are like an invisible box)

# divs (and other tags) are our boxes

CSS and HTML formatting follows the "box model" (or some variant)

You can see this:  Load a page in Chrome, go to Dev Tools->Elements->Style

Scroll to bottom



# More boxes in Chrome

Go to a page (say...neu.edu)
Right click an element on the page (not just the background)

Pick Inspect.

Scroll your mouse over the elements in the Elements tab.  See how any that are visible show their dimensions

# When planning an app, think in boxes

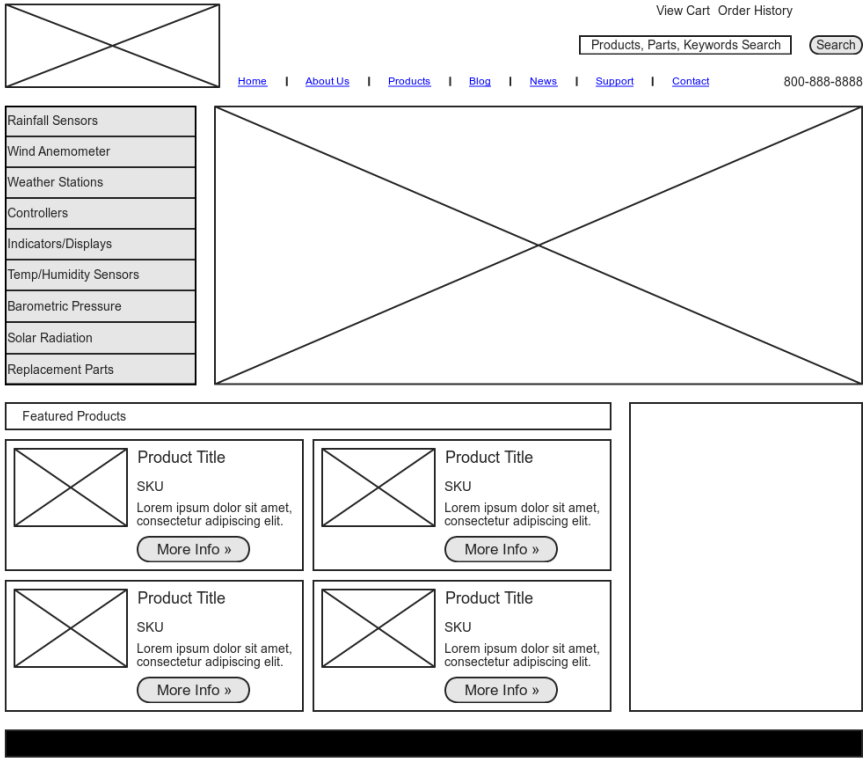
Start high level - Header, Body, Footer ( as one example)

Then fill those in boxes with the NEXT level

And so forth.  Don't worry about appearance yet, you're just decided what is in what.  Knowing which box that left side menu is INSIDE makes a big difference long before you get it to look like much.

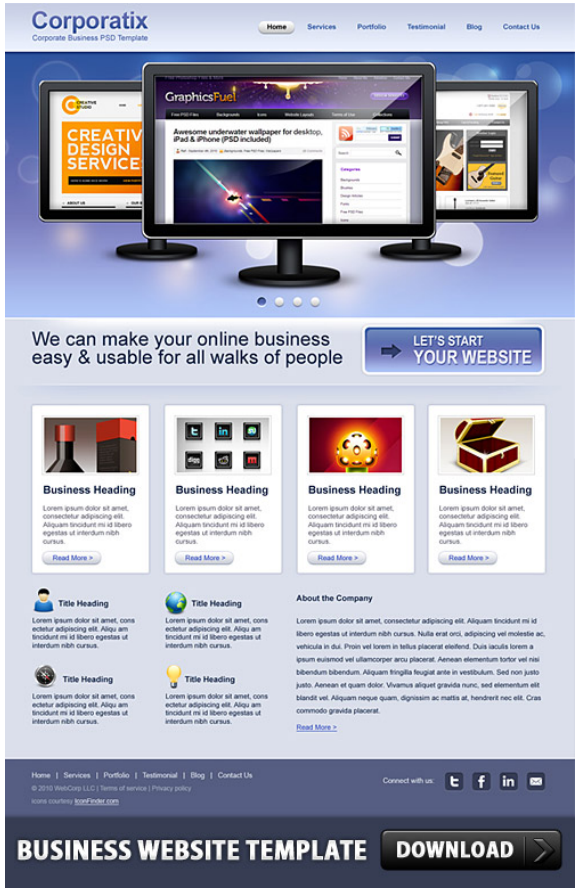
# Wireframes

An outline of the PARTS of the page without focus on the APPEARANCE of those parts



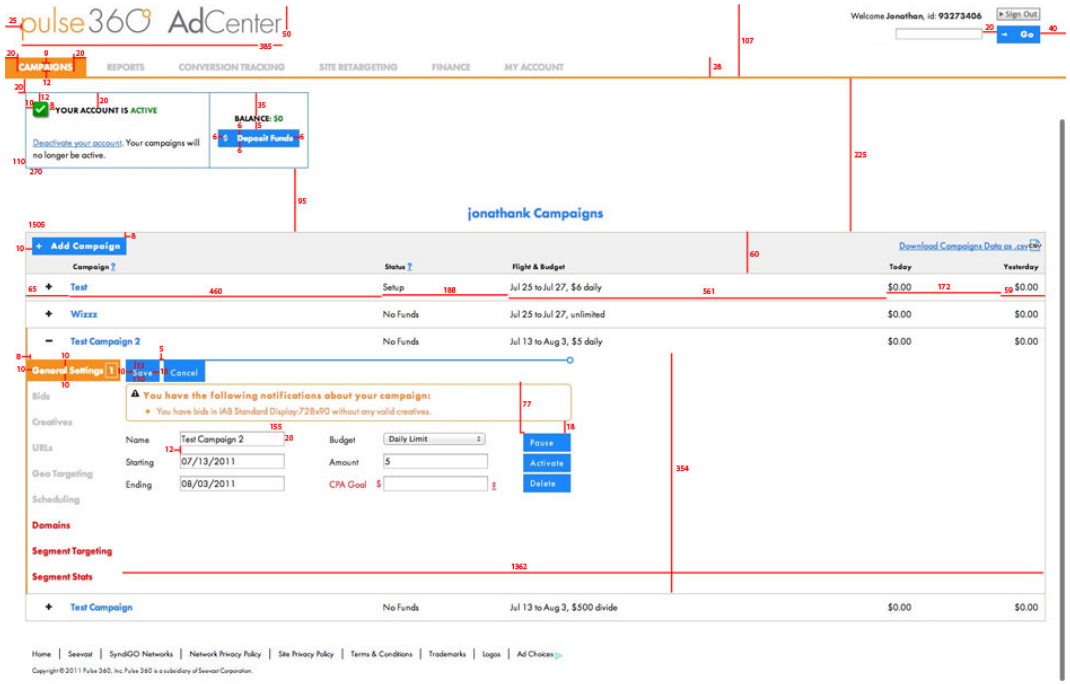
# Mockups

A sample appearance - details either come from redlines or are left to coder



# Redlines

A mockup that has formatting and spacing details



# Prototypes

A web-visible mockup that reacts to certain actions (clicks)

Lets you test actual web values and see user flow

# Your role!

First, check for common visual problems:

- length of fields/headers (particularly for i18n and buttons)
- Watch for funky fonts (slow, brittle)
- Consider browser window size (mobile?)
- Length of boxes on page - behavior on wrapping/overflow
- What if data isn't of uniform size?
- Defiance of browser conventions (that's usually bad, sometimes needed)
- Accessibility - how accessible will it be?  How accessible does it need to be?
- Timing of validations, and Space for messages ( and where do those go?)

# Then, check for data issues

- Can you get all the data on the screen? (existing services?)
- Is it too slow?  (too many services, complicated services)
- Have you considered the not-logged-in case?
- Will data always be complete?
- What is shown while data loads?
- Will you know the data necessary for any navigation/state change?
- What is the expected behavior if a service fails?
  (console.warn() only is a terrible experience)

# Finally, "box" it up:

Divide this page into "boxes" of responsibility (also visuals)

- remember to check if box dimensions are the same in a new state

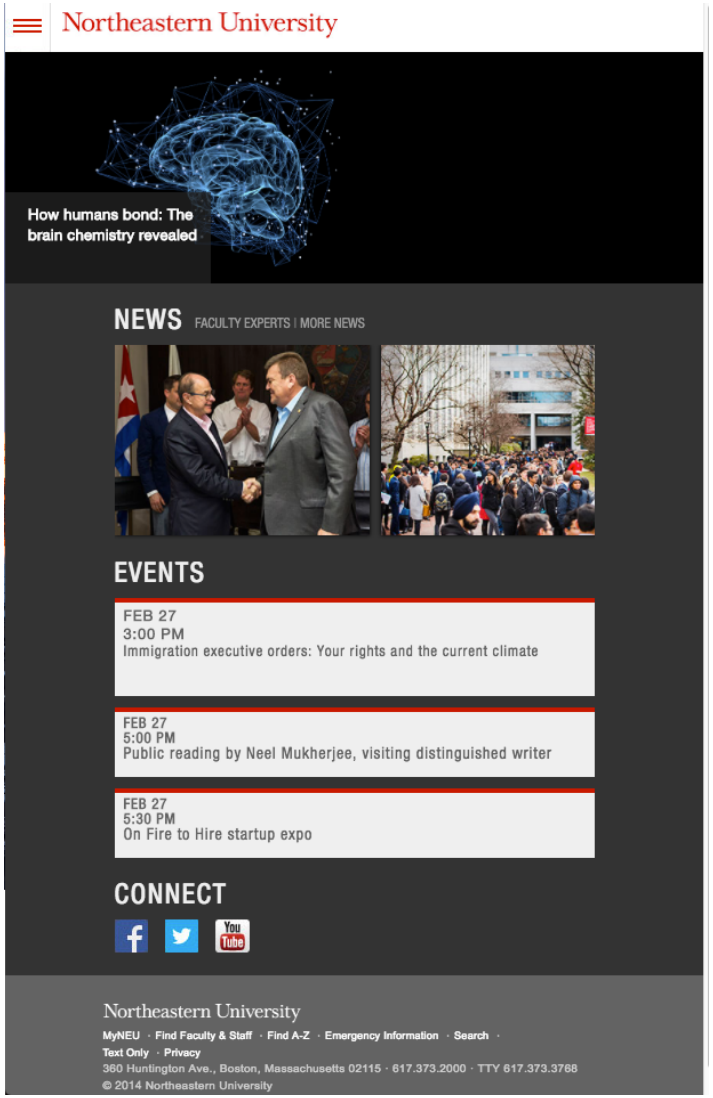
Divide those into more specific boxes, and so forth

React is easy to translate this into, because HTML elements and React components both visibly "contain" the content their rendered selves will contain.

( actual representation may not be inside )

# Break this down

Find the nested boxes



# Exercise: Now plan it in React

- No interaction, no fleshed out components
- Just do what contains what
- Don't need real text, but get the shape
- Use components that don't do anything - boxes
- Don't forget to import with ./Foo to get Foo.jsx