ECS 162
Web Programming

4/5

Due Thurs 4/11

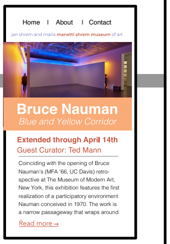
Home | About | Contact

Bruce Nauman

Blue and Yellow Corridor

Extended through April 14th

Guest Curator: Ted Mann



Home | About | Contact

jan shrem and maria **manetti shrem museum** of art

Bruce Nauman

Blue and Yellow Corridor

Extended through April 14th

Guest Curator: Ted Mann

Coinciding with the opening of Bruce Nauman's (MFA '66, UC Davis) retrospective at The Museum of Modern Art, New York, this exhibition features the first realization of a participatory environment Nauman conceived in 1970. The work is a narrow passageway that wraps around

Read more →

Exhibit assignment

- Flexbox and media queries (covered today).
- Responsive design which matches the mock-ups (Jamie's pdfs) as well as possible.
- Harder than it looks; make some progress this weekend.
- We'll talk about the button on Monday.

Responsive design

- Whatever width the window is,
  - the whole top of the page appears,
  - and it neatly fills the width of the window.
- Nothing ever gets cut off and no extra fill appears on the sides.
- Layout changes smoothly as window is resized, with maybe a jump as we switch from wide to narrower window.

We're not just showing Dana's pix

- Your program should never produce views like this

Do Not Make Your Web Pages Like This

WAYNE THIEBAUD

Exhibition at Manetti Shrem Museum

Open until May 13

free to the public

WAYNE THIEBAUD

- These are not responsive. Why?

Google fonts

- Google has a collection of fonts that a Web page can download and use. The HTML has to do the download:

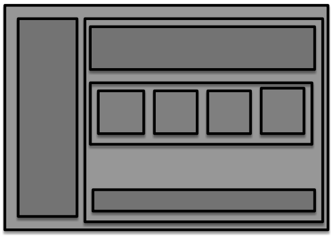
```
<link
href="https://fonts.googleapis.com/css?family=Montserrat:200,500" rel="stylesheet">

font-family: montserrat, sans-serif; font-weight: 200;
font-style: normal;
```

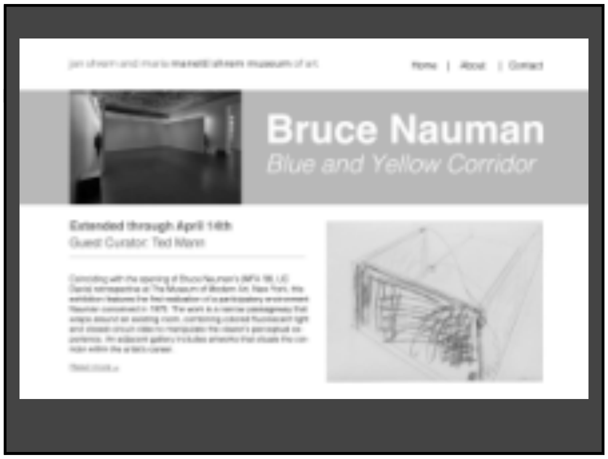
Layout and Flexbox

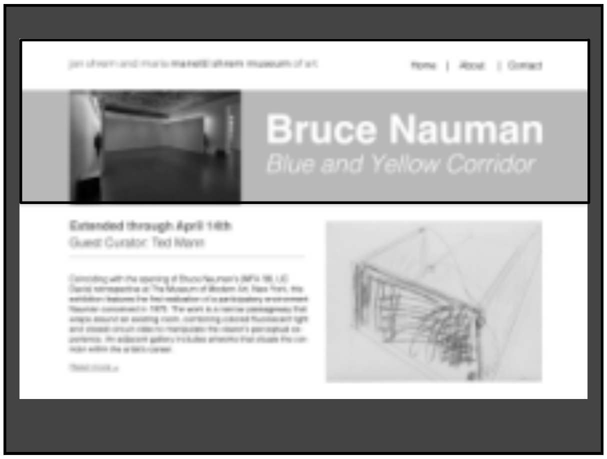
- Traditional approaches used css display modes such as inline-block and float to “trick” the default layout into doing interesting things.
- Tables and later frames were used to arrange chunks of content.
- CSS/Javascript libraries for layout (eg. Bootstrap) got popular.
- Now (eg. since 2016) CSS has two built-in layout systems, grid and flex. We’ll work with flex.

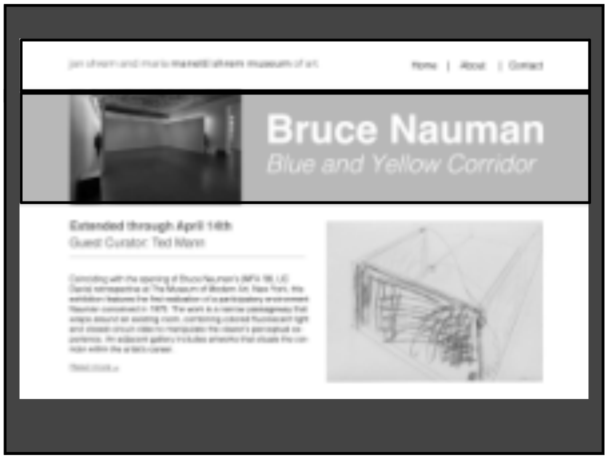
Flexbox idea:

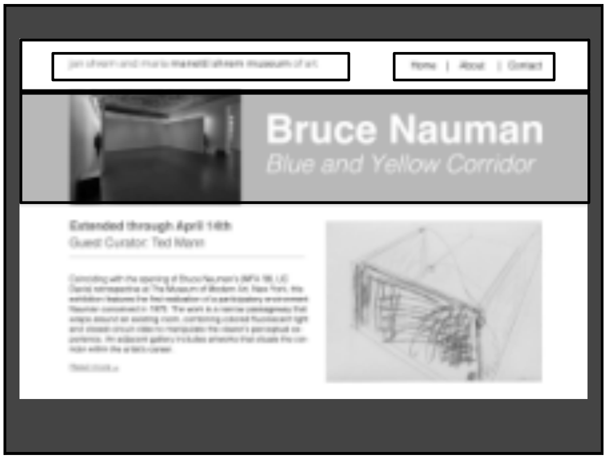


Web page is hierarchy of flexbox containers. Within a container, we lay out items (which might be other flex boxes) either vertically or horizontally.









Flexbox containers

- The css display property controls layout.
- By default display is inline or block.
- Another alternative (one of many!):

  display: flex;
- Makes the element a flexbox container. Layout of direct children of the container is handled by flexbox.

Layout of items

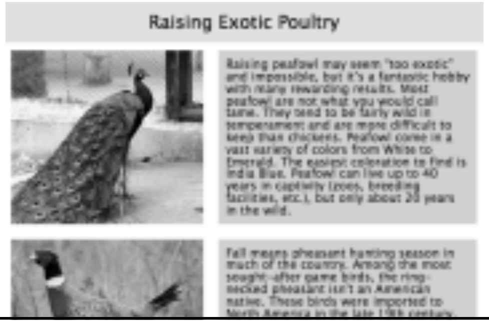
- Direct children of a flexbox container are flexbox items. Specify in container how items will be laid out:

  flex-direction: row; /* default! */

  or

  flex-direction: column;
- Direction in which items are laid out in the container is the main axis; the perpendicular direction is the cross axis.

Getting text beside pictures



Raising Exotic Poultry

Raising peafowl may seem "too exotic" and impossible, but it's a fantastic hobby with many rewarding results. Most peafowl are not what you would call tame. They tend to be fairly wild in temperament and are more difficult to keep than chickens. Peafowl come in a vast variety of colors from White to Emerald. The easiest coloration to find is India Blue. Peafowl can live up to 40 years in captivity (zoos, breeding facilities, etc.), but only about 20 years in the wild.

Fall means pheasant hunting season in much of the country. Among the most sought-after game birds, the ring-necked pheasant isn't an American native. These birds were imported to North America in the late 19th century.

Container vs item commands

- Some flexbox properties are about the html element as a flexbox container:

  flex-direction, justify-content, align-items
- Others are about the html element as a flexbox item:

  flex-basis, flex-grow, flex-shrink
- Since an html element can be ***both*** a flexbox container, and an item, keep the groups of properties separated by a space or comment in your css.

Basis, grow, shrink

- Grow is share of excess space item will take up.
- Shrink share of needed space item will contribute if it has to shrink.
- Basis is item's standard size in the main axis direction; "auto" means however big it has to be to contain it's content.
- Default:

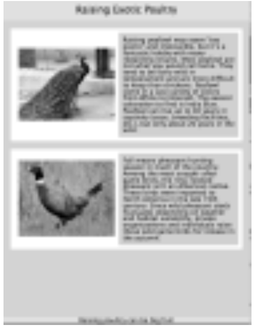
  grow=0, shrink=1, basis=auto

Shares

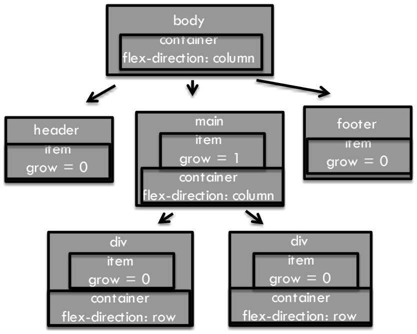
- Grow and shrink are given as shares of excess or needed space, respectively.
- If I get two shares and you get one share, then I get 2/3 and you get 1/3.
- If I get one share and you get zero shares, I get it all and you get nothing.

Example: Sticky footer

- We want the footer to sit at the bottom of the page, even if the contents don't fill up the page.


Raising Exotic Poultry

Sticky footer


body
container
flex-direction: column

header
item
grow = 0

main
item
grow = 1
container
flex-direction: column

footer
item
grow = 0

div
item
grow = 0
container
flex-direction: row

div
item
grow = 0
container
flex-direction: row