RESPONSIVE DESIGN IN A NUTSHELL

In a Nutshell…

1. Core content: Columnar, with division of roles
2. Each screen family gets a layout
3. Layout = Core content + manipulations:
   - ✓ Extract from flow
   - ✓ Inject
   - ✓ Remove/Hide
4. Match layout to screen w/ Media Queries or Javascript

"Columnar, with a division of roles"

KAAZING

Key driver: Current smartphones & tablets

- ✓ Columns facilitate double-tap zooming & reading while zoomed

Division of roles

- ✓ Encapsulate navigation, header, footer, articles, etc. to facilitate re-layout
- ✓ Label encapsulated elements with IDs (for re-layout) and classes (for appearance)

“Each screen family gets a layout”

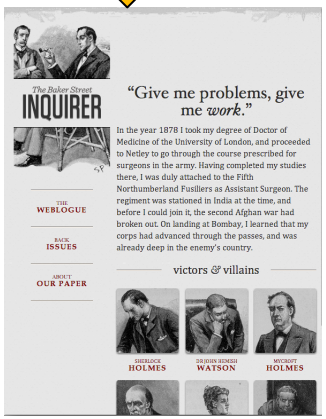
KAAZING

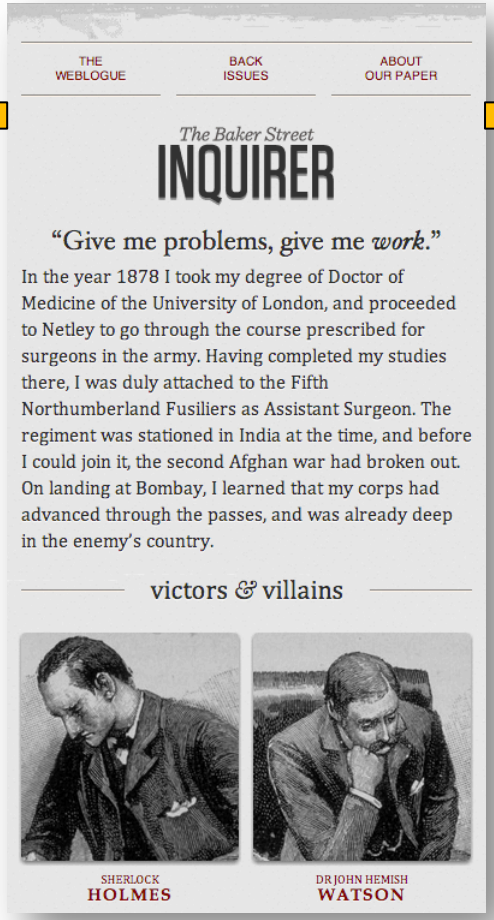
Type		Fixed width breakpoint
Desktop		≥ 1280
Tablet		
	Portrait	768
	Landscape	1024
Phone		320 x 480 (qvga)
	Portrait	320
	Landscape	480

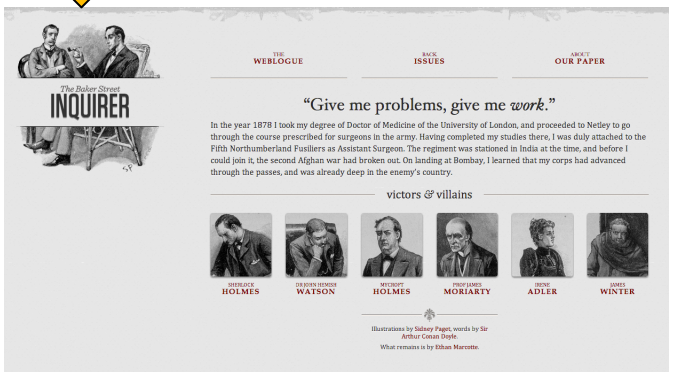
Exercise: *Should you let the content dictate breakpoints instead? Search Google for responsive design breakpoints and discuss.*

“Layout: Content + manipulations”

www.alistapart.com/d/responsive-web-design/ex/ex-site-FINAL.html







Tablet:

1. Extract nav & heading to left
2. Inject image

Desktop:

1. Extract heading to left
2. Inject image

Content manipulation options

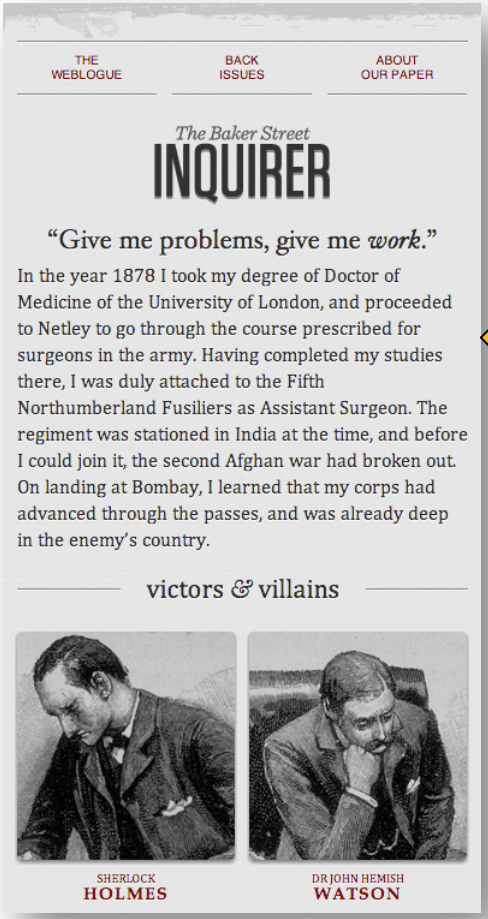
Manipulation	CSS technique	Notes
Extract from flow	float/clear, position:absolute	
Inject into page	::before and ::after	(single-colon ok for backward compatibility: IE ≤ 8, CSS2)
	background-image	
	url(…)	
Remove/hide	visibility:hidden, display:none	Understand impact on screen readers
	position:absolute, etc.	

Useful references:

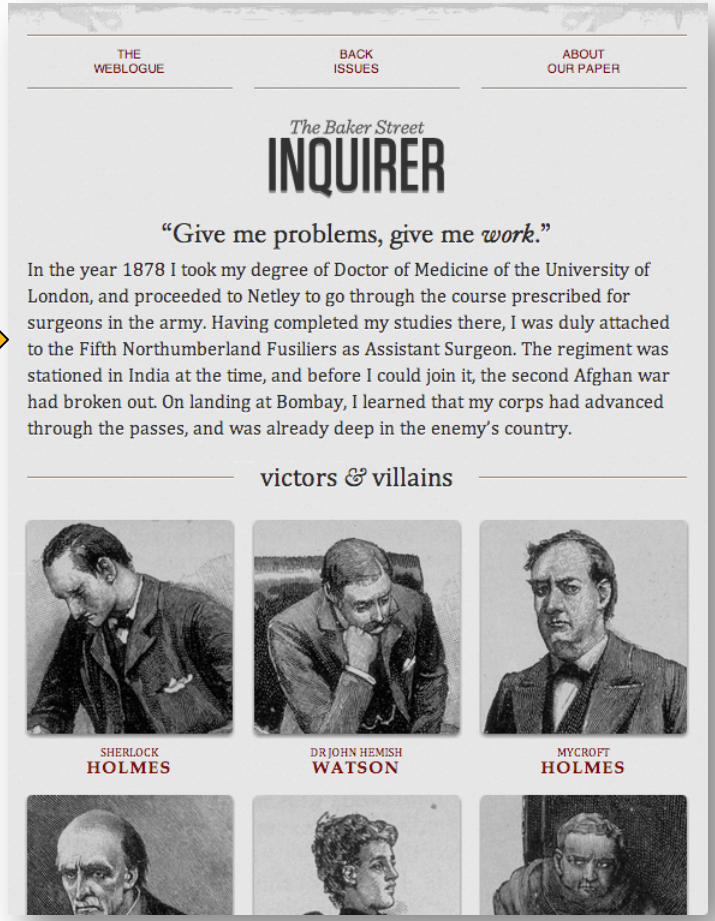
designshack.net/articles/css/the-lowdown-on-before-and-after-in-css

http://www.alistapart.com/articles/now-you-see-me/ (about hiding)

Percentages…



Percentage-based columns to allow fluid resizing

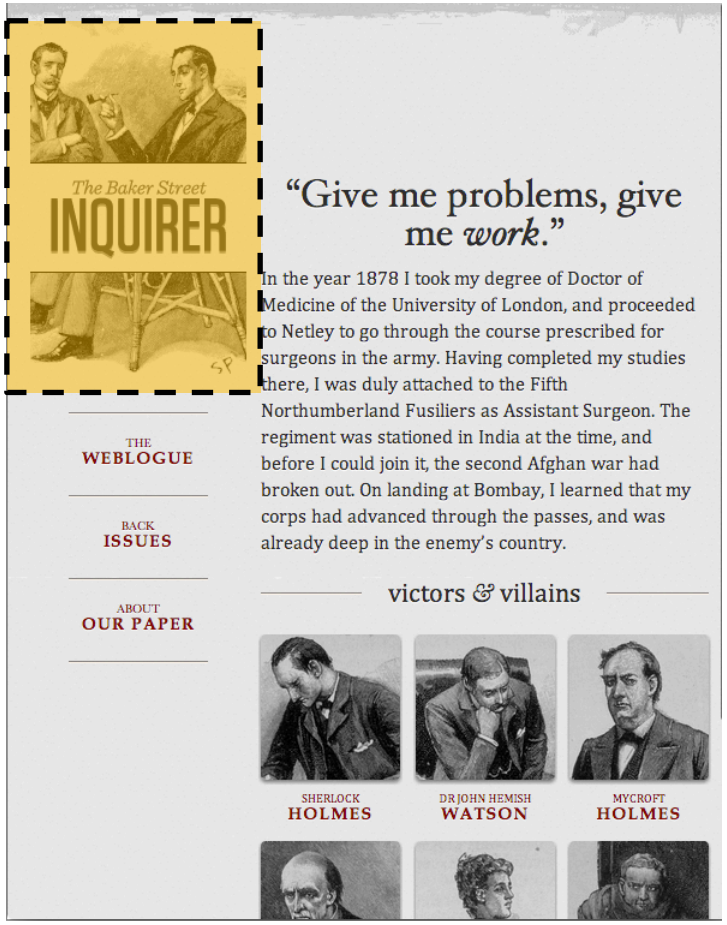


Images and scalable layouts

Set `max-width: 100%` to keep image in its bounds

For IE: use `width:100%` or hacks via `AlphaImageLoader`

See unstoppablerobotninja.com/entry/fluid-images



“Match layout to screen…”

KAAZING

Media query example (full details later)

```
nav { float: left; margin-left: -100%; width: 200px; }
.body { float: left; width: 100%; }
.body > section  { margin-left: 200px; }

@media screen and (max-width: 320px) {
nav { float: none; margin-left: 0 px; }
.body { float: none; width: 100%; }
.body > section { margin-left: 0px; }
}
```

P.S. Negative margin magic:
coding.smashingmagazine.com/2009/07/27/the-definitive-guide-to-using-negative-margins/

Other considerations

If you use fixed-width layouts:

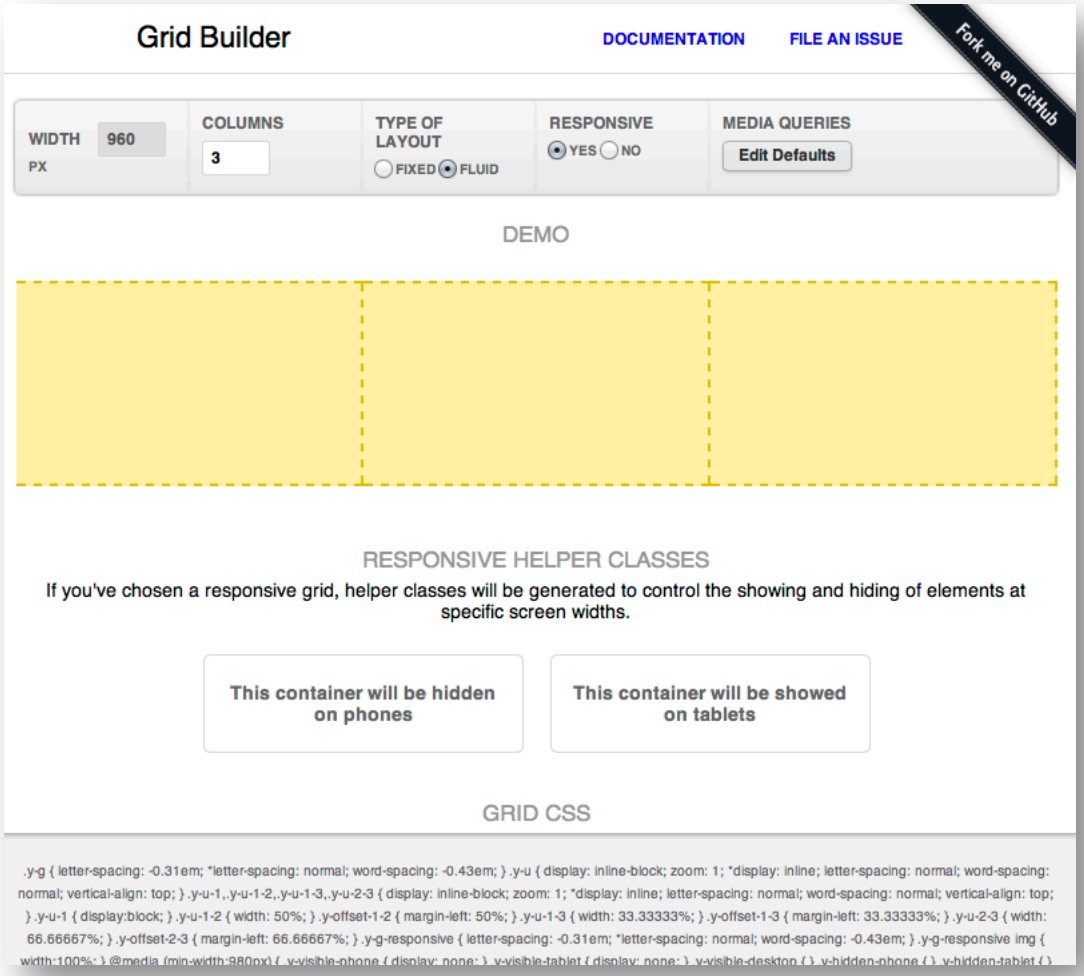
1. Still use `max-width` to constrain images
2. Percentages still useful via preprocessor or calc() element

If an element needs more than simple repositioning:

1. Use phantom elements in page
2. Hide phantoms w/ display:none (removes from flow)
3. Inject content into phantoms on larger pages

A Practical Example

yui.github.com/gridbuilder



Implementation: Screen Sizes

KAAZING

Too many to count! (In Android)

Options:

- ✓ Fixed positioning via Javascript
  - Probe screen size, perform layout
- ✓ Relative positioning vs. screen edges
  - Top, bottom, scrollable middle
  - iframe for content?
  - overflow: scroll (can set for horizontal and vertical)
- ✓ Regular flow, inline, etc. positioning
- ✓ Flexbox (future)

Pixels & Viewports

KAAZING

Device pixels vs. CSS pixels

- Device pixels: screen.width & screen.height
- Layouts in CSS Pixels
- CSS pixels == Device pixels @ 100% zoom
- 4 CSS pixels / device pixel @ 200% zoom

Details: quirksmode.org/mobile/viewports.html

Viewports

KAAZING

The Viewport

- Size of the working area (== window size on desktop)
- But on mobile…
  - Layout viewport: Virtual screen, can set width
    - Defaults: iPhone 980px, Opera 850px, Android WebKit 800px, IE 974px.*
  - Visual viewport: Window onto the layout viewport
    - Can move, zoom in/out, etc.

Details: quirksmode.org/mobile/viewports2.html

* source: quirksmode

Managing the Viewport Size

Explicit size:

```
<meta name="viewport" content="width=320">
```

Control auto-zoom on rotation:

```
<meta name="viewport"
 content="width=device-width,
          initial-scale=1.0">
```

Exercise: Thinking responsively

1. Select an site idea to make responsive (whole class)
2. Sketch layout for 320px mobile
3. Sketch tablet & desktop layouts, decide on transforms
4. Discussion: How to handle older phones & IE?