Computer Science Principles

Web Programming

Web-Presentation Design with CSS

CHAPTER 16: LAYOUT MANAGEMENT II (PAGE DESIGN)

DR. ERIC CHOU IEEE SENIOR MEMBER

CSS3

Chapter 11: CSS Hierarchy and Selectors

Chapter 12: Text, Image and Foreground (Contents)

Chapter 13: Color and Background (Contents)

Chapter 14: Box Model (Padding, Border, and Margin)

Chapter 15: Layout Management (Floating and Positioning: where should the Element go)

Chapter 16: Layout Management (Page Level Planning)

Chapter 17: Layout Management (Transition, Transforms, and Animation: space and time domain transformation)

Chapter 18: CSS Techniques (Put Everything Together)

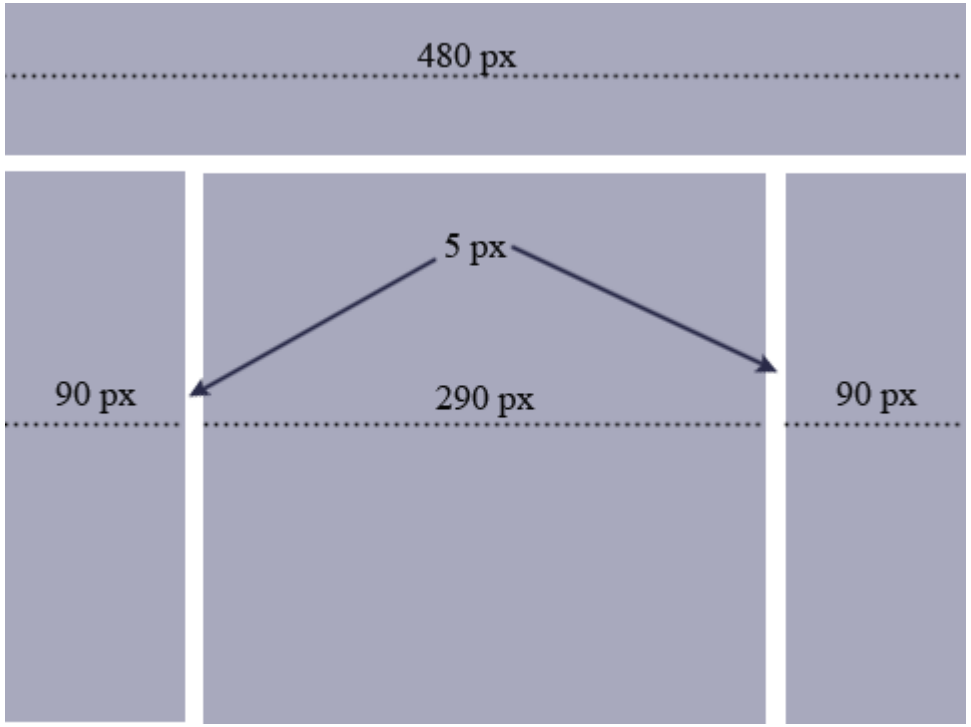
Planning

LECTURE 1

Page Layout Strategies

- **Fixed Layouts:** stay put at a specific pixel width regardless of the size of the browser window or text size.
- **Fluid (or liquid) Layouts:** resize proportionally when the browser window resizes.
- **Elastic Layouts:** resize proportionally based on the size of the text.
- **Hybrid Layouts:** combine fixed and scalable areas.

Advantages and Disadvantages of Fixed



Advantages:

- The layout is predictable and offers better control over line length.
- It is easier to design and produce.
- It behaves the way the majority of web pages behave as of this writing, but that may change as users visit the web primarily on the devices other than the desktop.

Disadvantages:

- Content on the right edge will be hidden if the browser window is smaller than the page .
- There may be an awkward amount of leftover space on large screens.
- Line lengths may grow awkwardly short at very large text sizes.
- Takes control away from the user.

An easier way to do fixed layout is to use CSS Grid Framework

Grid framework is to design web pages on grid (larger granularity).

A grid is an invisible foundation that provides the page into equal units that can be used to determine where columns, headlines, images, and so on should fall .

www.960.gs

www.blueprintcss.org

http://www.bluetrip.org

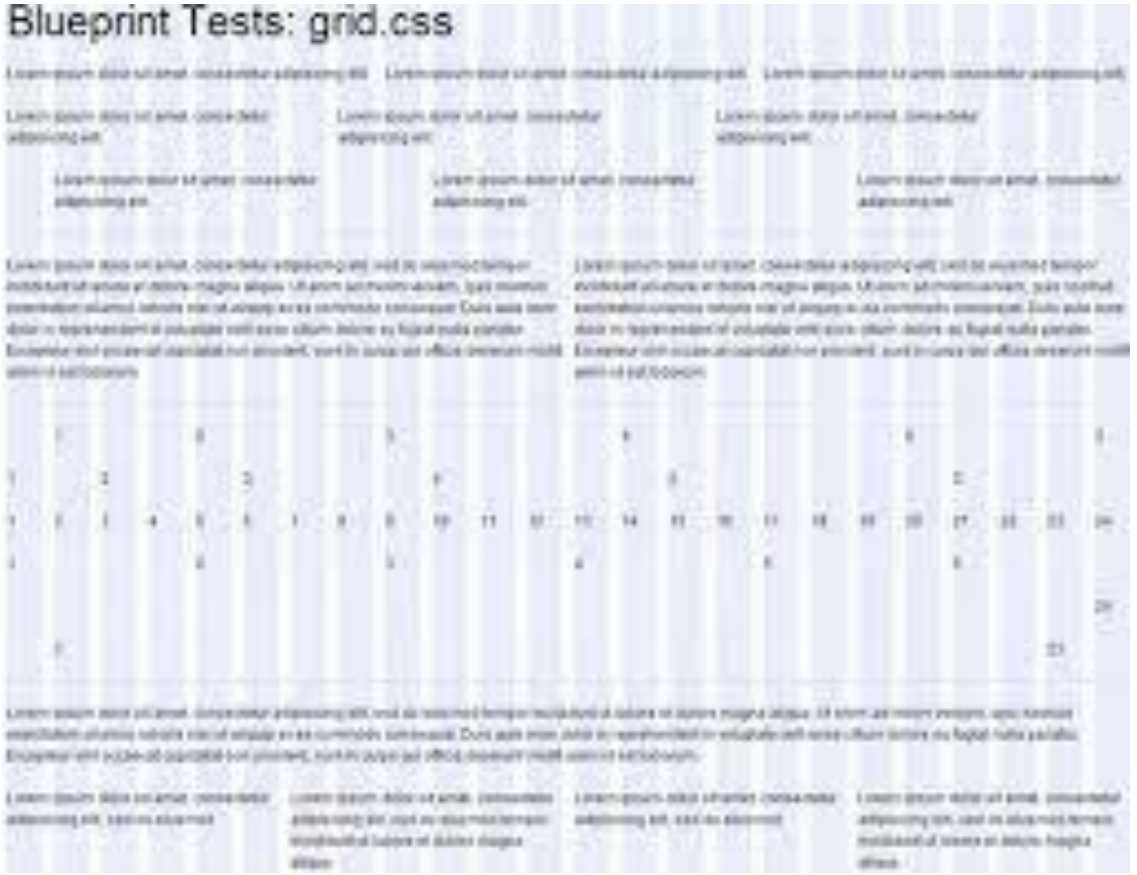
http://developer.yahoo.com/yui/grids/

http://www.cssgrid.net

http://www.getskeleton.com

http://twitter.github.com/bootstrap

http://grids.subtraction.com



Fluid Design

LECTURE 2

Fluid Page Design (Liquid Design)

- In fluid page layouts (also called liquid layouts), the page area and columns within the page get wider or narrower to fill the available space in the browser window. In other words, they follow the default behavior of the normal flow.
- There is no attempt to control the width of the content or line breaks, the text is permitted to reflow as required and as in natural to the medium.
- Fluid layouts are a cornerstone of the **responsive web design** technique.

Fluid Page Design (Liquid Design)

Advantages:

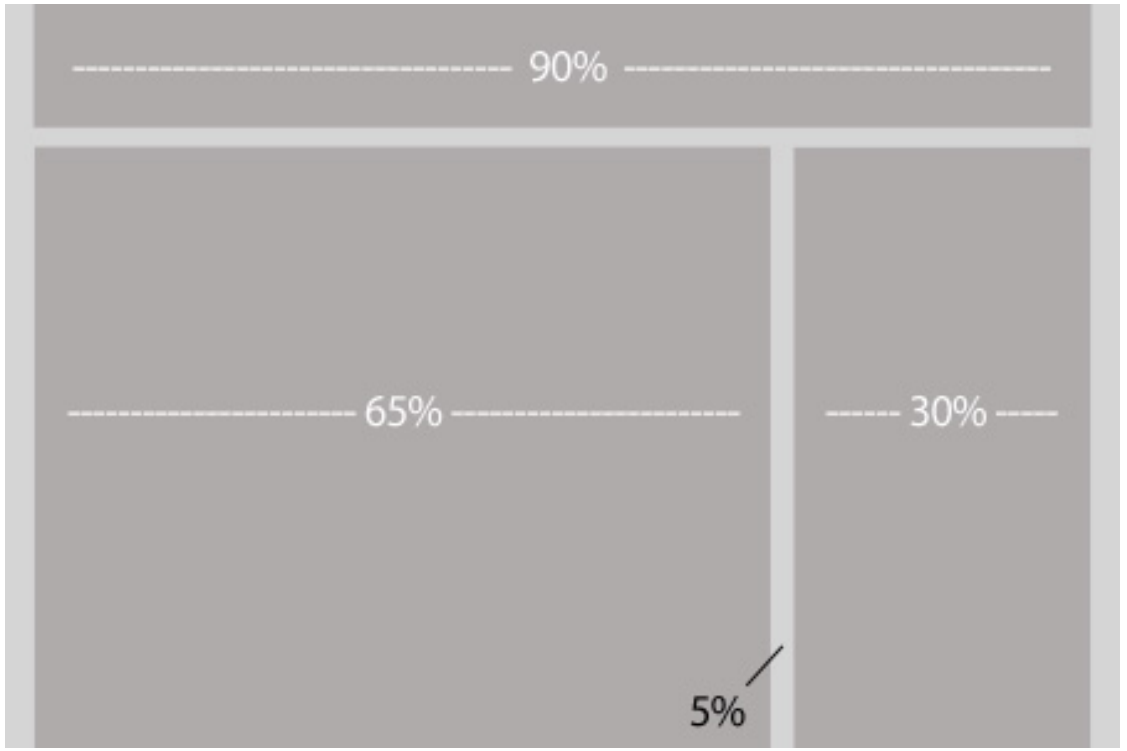
- Fluid layouts keep with the spirit and nature of the medium.
- They avoid potentially awkward empty space because the text fills the window.
- On the desktop browsers, users can control the width of the window and content.
- No horizontal scrollbars.

Disadvantages:

- On large monitors, line lengths can get very long and uncomfortable to read.
- They are less predictable. Elements may be too spread out or too cramped at extreme browser dimensions.
- It may be more difficult to achieve whitespace.
- There is more math involved in calculating measurements

Fluid Page

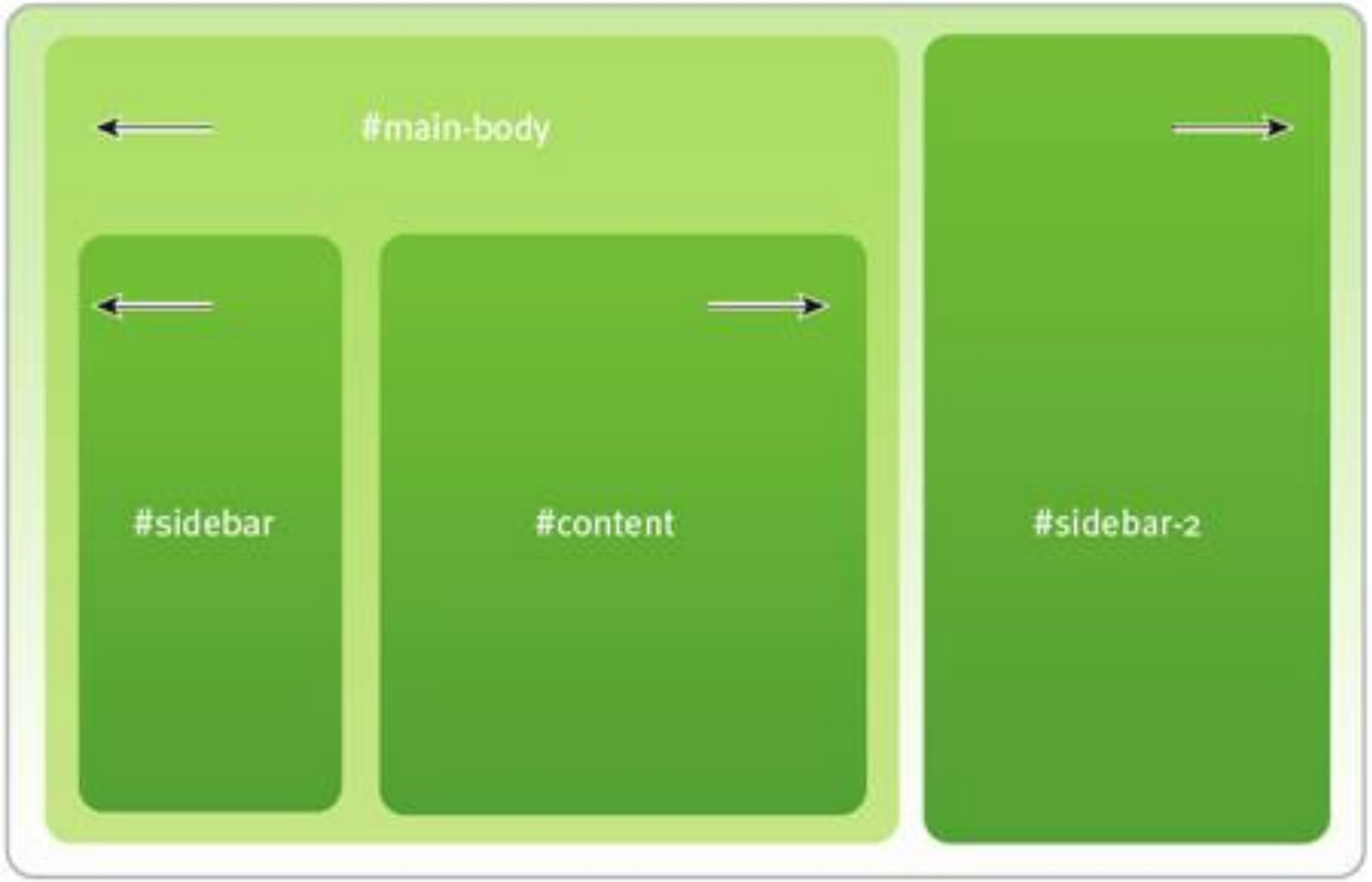
(All in Percentage. Text Font size fixed. Text will flow down.)



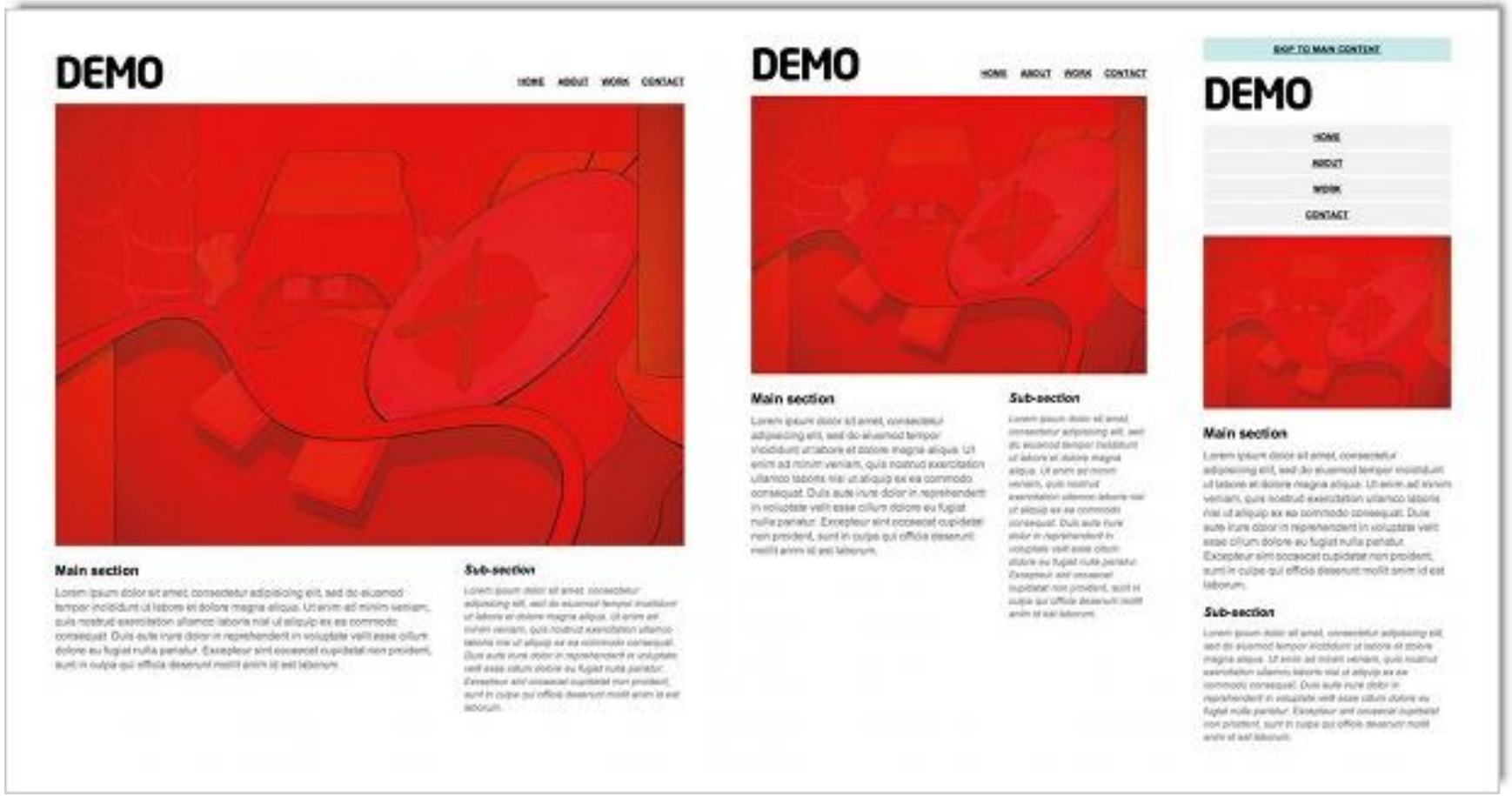
min-width and **max-width** keep the fluid page from becoming unreasonably large or small.

Fluid Page

(All in Percentage. Text Font size fixed. Text will flow down.)



Fluid Page example



Elastic Layout

LECTURE 3

Elastic Layout (width scaled by em as unit)

Elastic Page

(Width scaled elastically.)

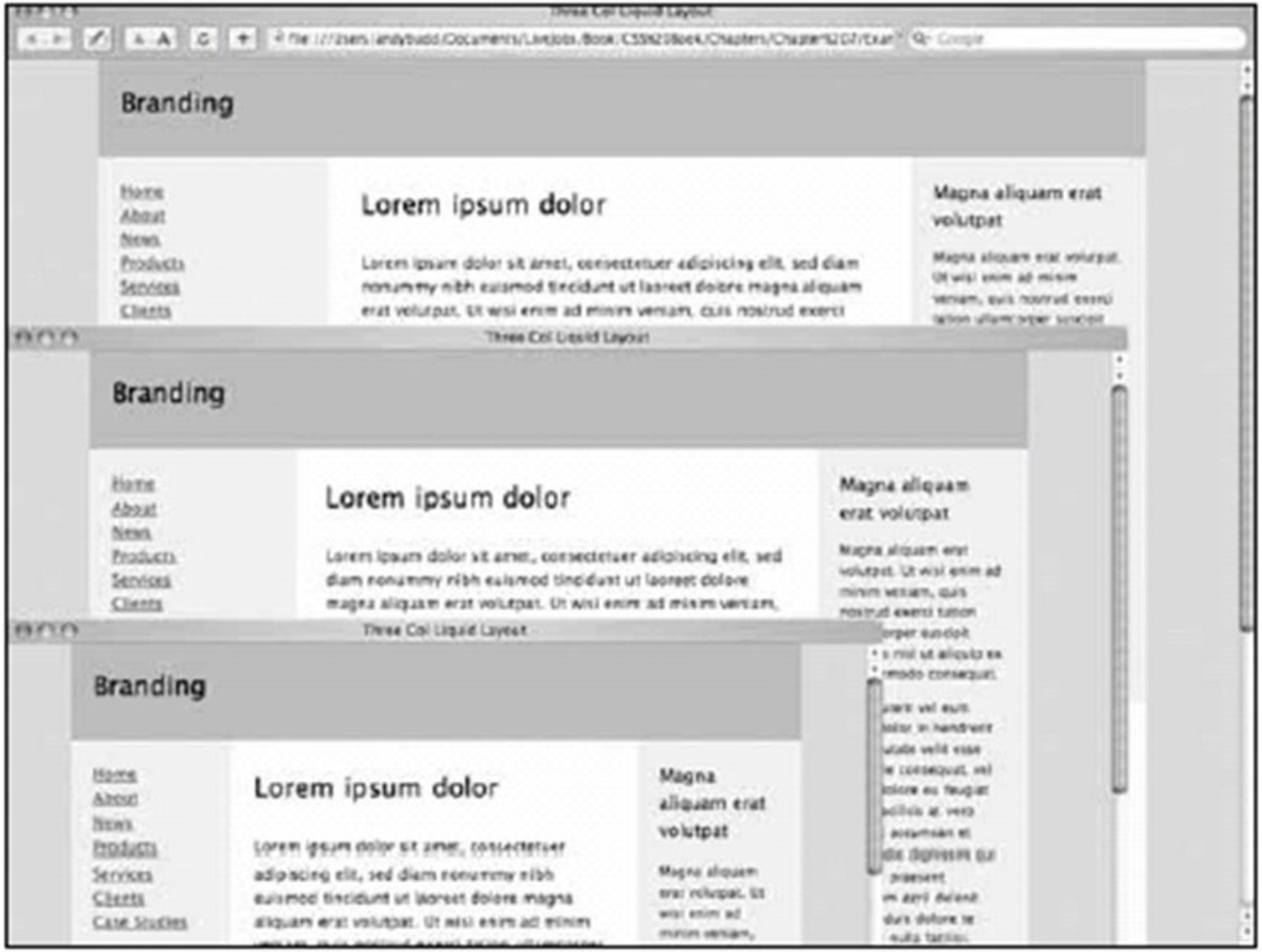
Most browser support **FULL-PAGE ZOOM**.
min-width and max-width can limit the extreme cases.

Advantage:

- Provide a consistent layout experience while allowing flexibility in text size.
- Tighter control over line lengths than liquid and fixed layouts.

Disadvantage:

- Images and videos don't lend themselves to automatic rescaling along with the text and the rest of the layout (but there are methods to achieve this.)
- The width of the layout might exceed the width of the browser window at largest text sizes.
- Not as useful for addressing device and browser size variety.
- More complicated to create than fixed width layouts.



Difference Between Fluid Design and Elastic Design

Fluid (Flow down like liquid)

Difference Between Fluid Design and Elastic Design

Elastic (Flexible like Rubber Band)

How to create Elastic Page

- The key to elastic layouts is the **em**s, the unit of measurement that is based on the size of the text.
- It is common to specify **font-size** in ems.
- In elastic layouts, the dimensions of containing elements are specified in **em**s as well. That is how the widths can respond to the text size.

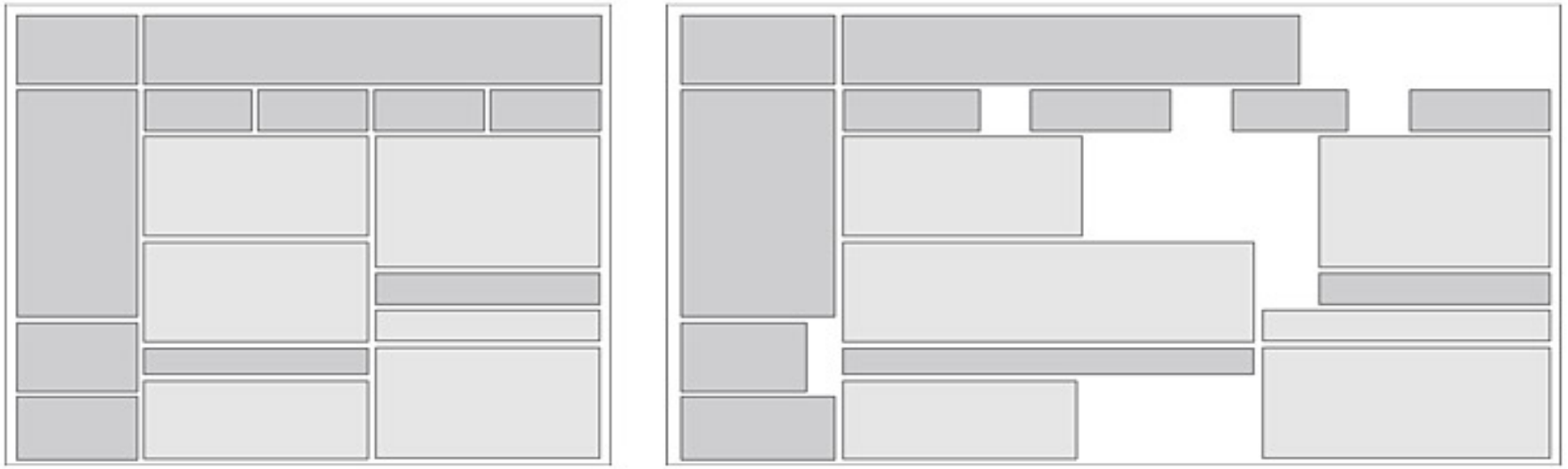
Hybrid Design

LECTURE 4

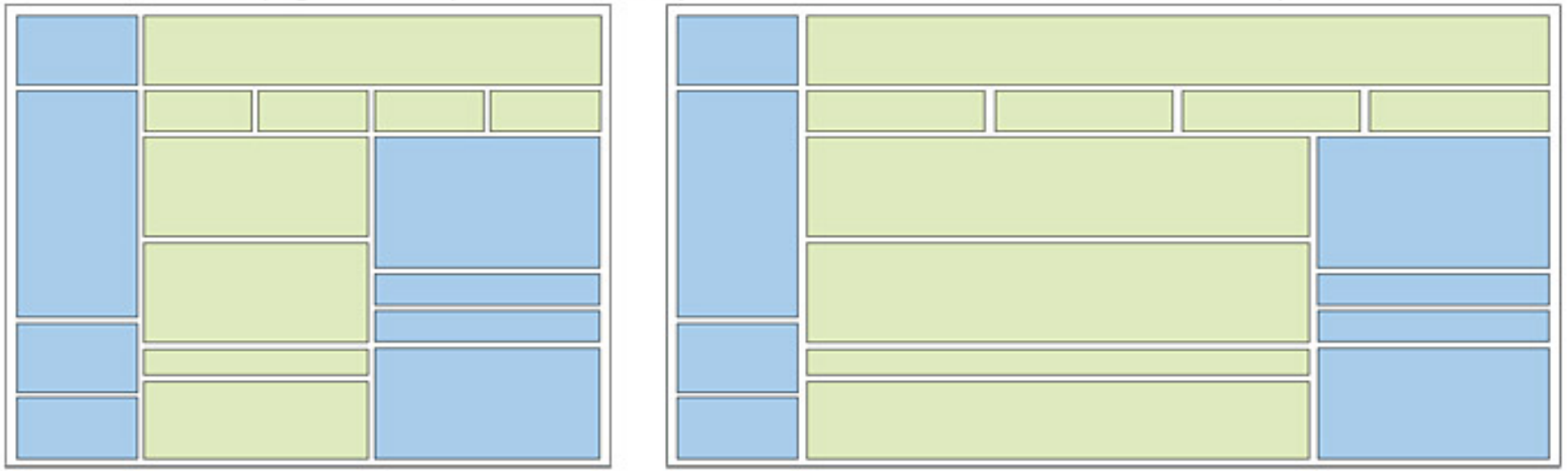
Hybrid Layouts

- Layouts that use a combination of **pixel (fixed), percentage (fluid),** and **em (elastic)** measurements are sometimes called Hybrid Layouts.
- In many scenarios, it makes sense to mix fixed and scalable content areas. For example, you might have a side-bar that contains a stack of ad banners that must stay a particular size. You could specify that sidebar at a particular pixel width and allow the column next to it to resize to fill the remaining space.

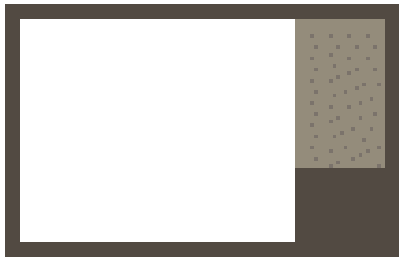
Pages that look reasonable in a fixed layout (below, left) can fall apart when converted to a stretchy "liquid" layout (right)



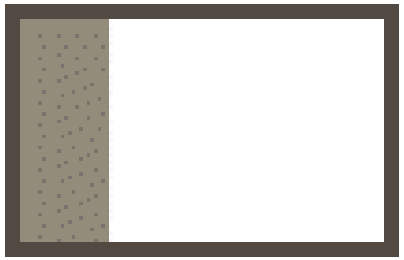
Careful planning and a hybrid approach that mixes liquid (green) and fixed-width layout elements (blue) can give you the best of both worlds: pages that adapt to a wide variety of screen sizes and media, with some areas of predictable widths



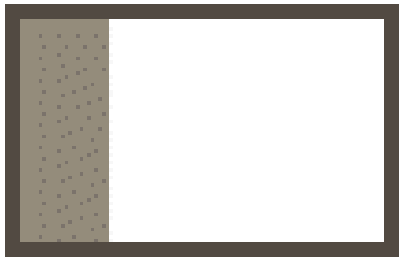
Mixed Layouts



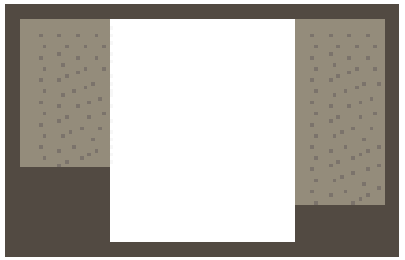
Menu and content dynamic



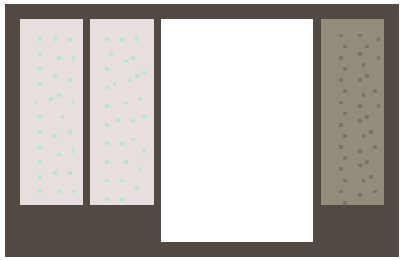
Menu fixed, Content dynamic



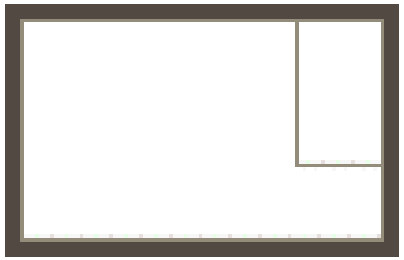
Menu and content dynamic



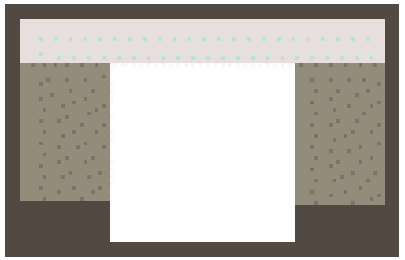
3 columns, all dynamic



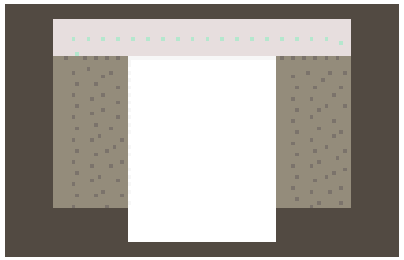
4 columns, all dynamic



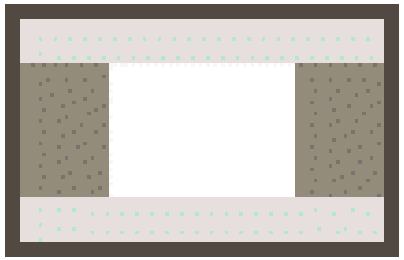
Menu floating



Menu fixed, content & header dynamic



3 columns fixed centered



dynamic with header and footer

Page Layout Techniques

- CSS Templates: two column and three column layout examples using CSS. Templates can provide general purpose layout for many applications.
- The next section provides templates and techniques for the following:
  - Two- and three-column layouts using floats
  - A source-independent layout using floats and negative margins
  - A multicolumn layout using positioning

The future of CSS Layout

- Column (www.w3.org/TR/css3-multicol): use the property **column-count** to specify a number of columns or a specific column width that will repeat until it runs out of room.
- Flexbox (http://www.the-haystack.com/2012/04/learn-you-a-flexbox): The CSS Flexible Box Layout Model provides a much simpler way to arrange element boxes in relation to one another. For example, you can line children elements up within a parent, select where extra space appears, center things horizontally or vertically, and even change the order of appearance – all without resorting to floats and margin offsets and the tricky calculation s that come with them.
- Grid Layout System (http://dev.w3.org/csswg/css3-grid-layout , http://msdn.Microsoft.com/library/ie/hh673536.aspx#-CSSGrid): Initiated by Microsoft
- Regions and Exclusions (http://dev.w3.org/csswg/css3-regions, http://dev.w3.org/csswg/css3-exclusions): Initiated by Adobe. (http://html.adobe.com)

Multiple Column Design

LECTURE 5

Multi-column Layouts Using Floats

- Floats are the primary tool for creating columns on web pages. As a tool, it is flawed, but it's the best that we've got as of the book.
- The **advantages** that floats have over absolute positioning for layout are that they prevent content from overlapping other content, and they make it easier to keep footer content at the bottom of the page.
- The **drawback** is that they are dependent on the order in which the elements appear in the source, although there is a workaround using negative margins, aw we'll see later in this section.

Two column, fluid layout

The strategy

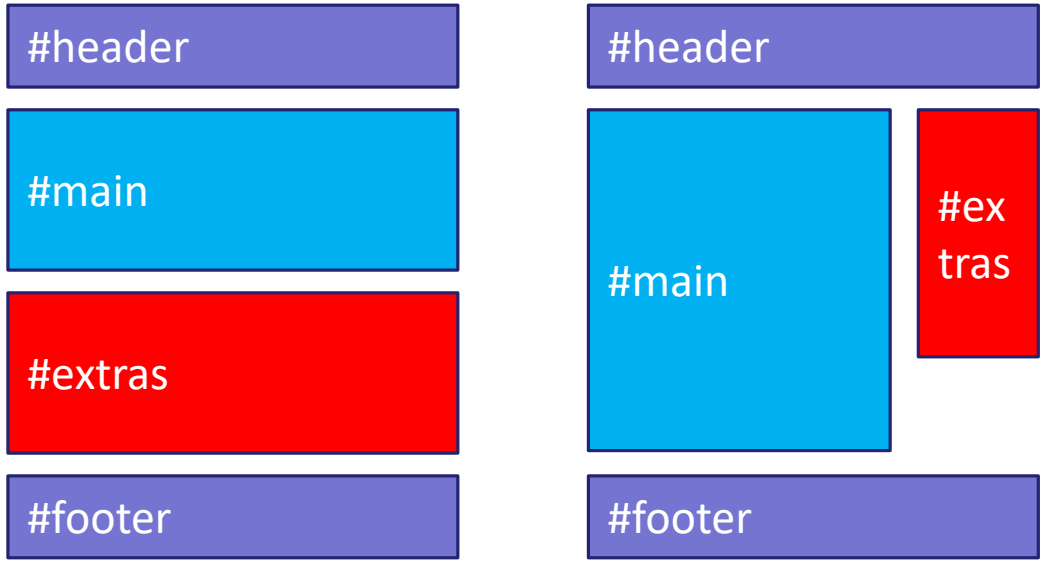
Set widths on both column elements and float them to the left. Clear the footer to keep it at the bottom of the page.

The markup

```
<div id="header">Masthead and headline</div>
<div id="main">Main article</div>
<div id="extras">List of links and news</div>
<div id="footer">Copyright information</div>
```

The styles

```
#main {float: left; width:60% margin 0 5%}
#extras {float: left; width:25%; margin: 0 5% 0 0; }
#footer {clear: left; }
```



Two column, fixed width layout

The strategy

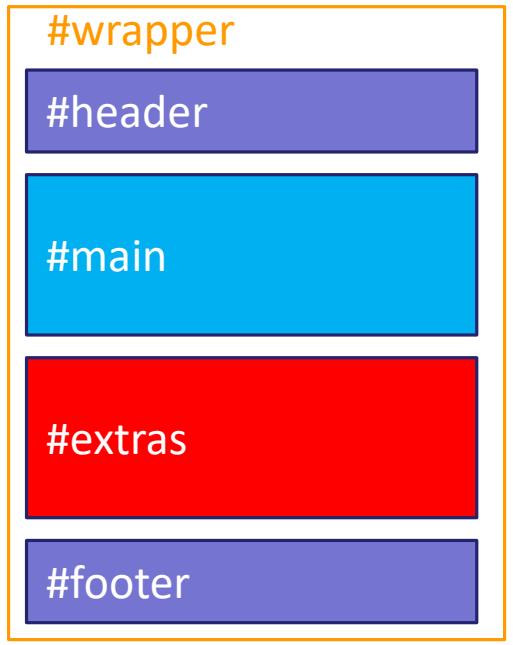
Set widths on both column elements and float them to the left. Clear the footer to keep it at the bottom of the page.

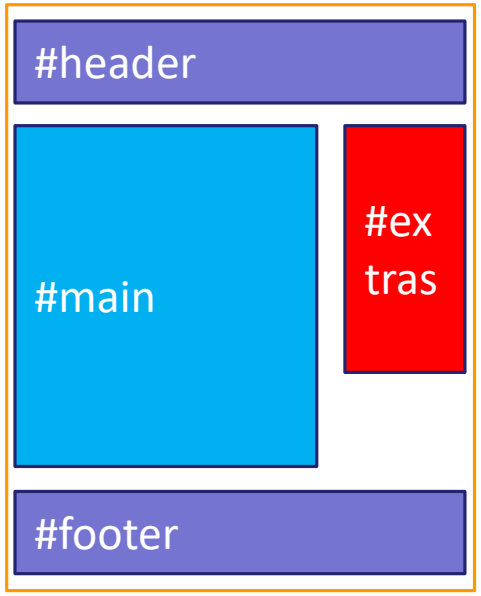
The markup

```
<div id="wrapper">
   <div id="header">Masthead and headline</div>
   <div id="main">Main article</div>
   <div id="extras">List of links and news</div>
   <div id="footer">Copyright information</div>
</div>
```

The styles

```
#wrapper {width:960px}
#main {float: left; width:60%; margin 0 5%}
#extras {float: left; width:25%; margin: 0 5% 0 0; }
#footer {clear: left; }
```





Three columns, positioned, fluid layout

The strategy

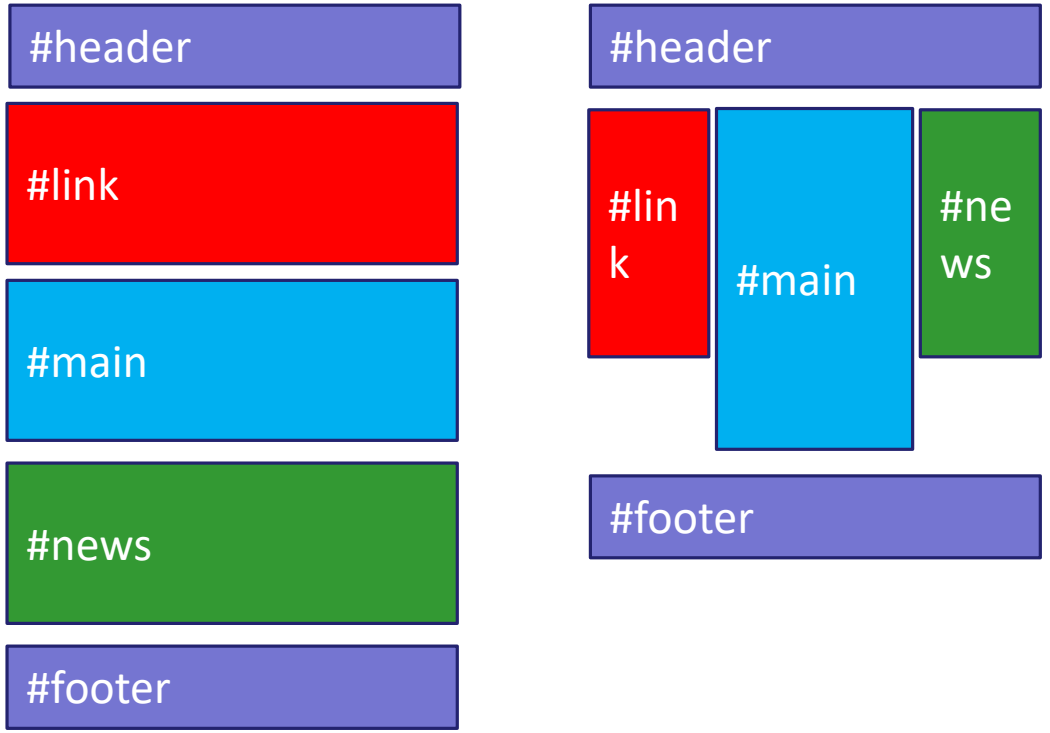
Set width on all three-column elements and float them to the left. Clear the footer to keep it at the bottom of the page.

The markup

```
<div id="header">Masthead and headline</div>
<div id="link">List of Links</div>
<div id="main">Main article</div>
<div id="news">News Item</div>
<div id="footer">Copyright information</div>
```

The styles

```
#link {float: left; width: 22.5; margin: 0 0 0 2.5%;}
#main {float: left; width:45% margin 0 2.5%}
#news {float: left; width:22.5%; margin: 0 2.5% 0 0; }
#footer {clear: left; }
```



Any order columns using negative margins

The strategy

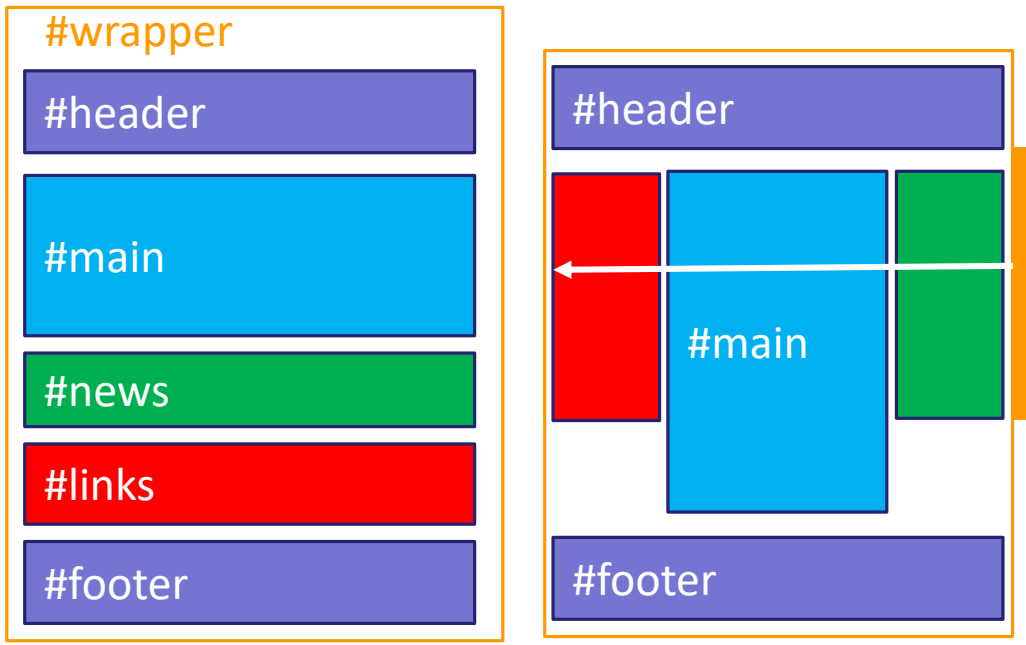
Apply widths and floats to all three column elements, and use a negative margin to “drag” the left column across the page into the left position. Notice that although #main comes first in the source, it is in the second column position. In addition , the #links div (last in the source) is in the first column position on the left. This example is fixed, but you can do the same thing with a fluid layout using percentage values.

The markup

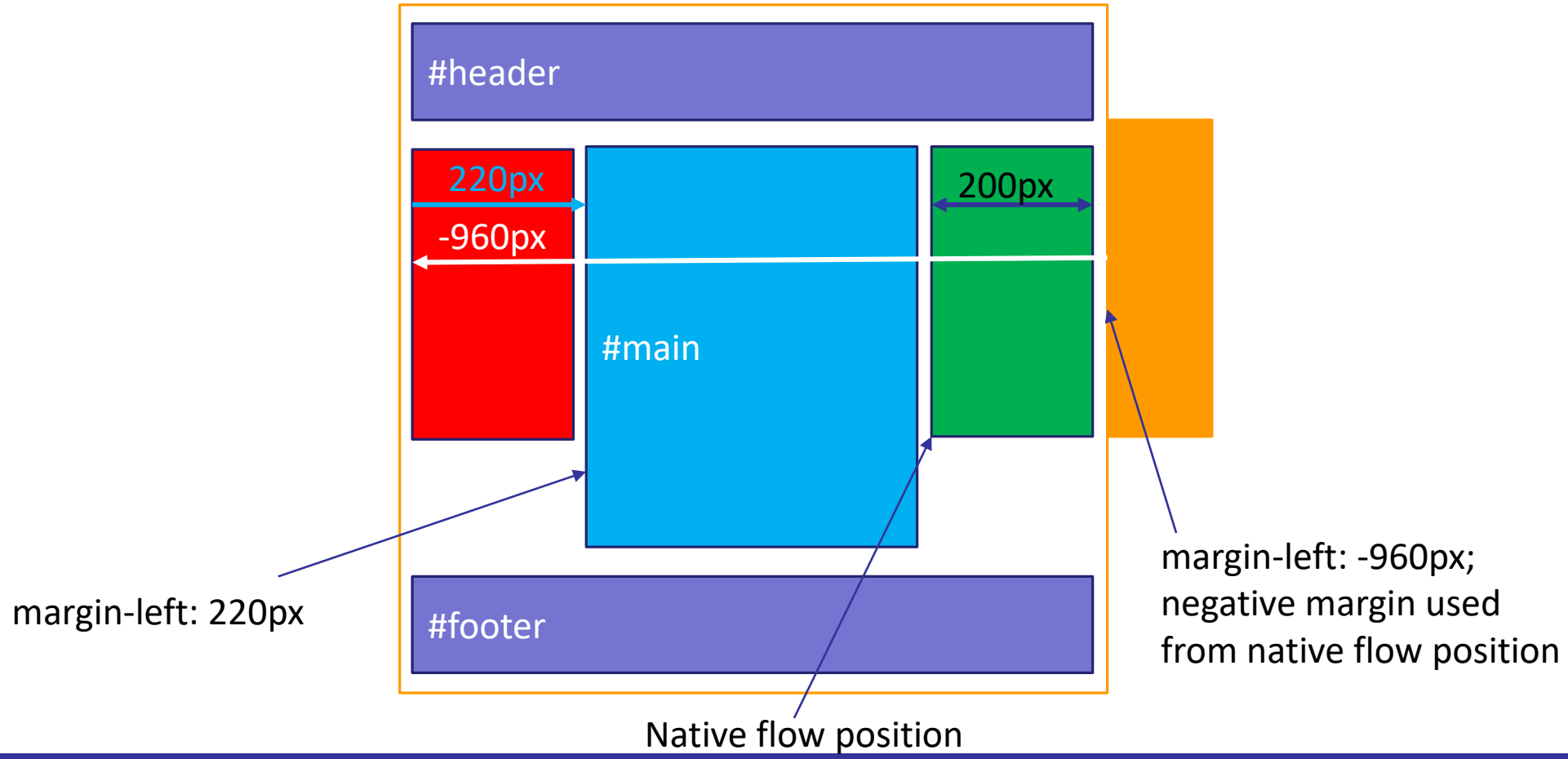
```
<div id=“wrapper”>
   <div id=“header”>Masthead and headline</div>
   <div id=“main”>Main article</div>
   <div id=“news”>News items</div>
   <div id=“links”>List of links</div>
   <div id=“footer”>Copyright information</div>
</div>
```

The styles

```
   #wrapper {width:960px; margin: 0 auto; }

   #main {float: left; width: 520px; margin-top: 0; margin-left: 200px; margin-left: 20px;}

   #news { float: left; width: 200px; margin: 0; }
   #links {float: left; width 200px; margin-top: 0; margin-left: -960px; }
   #footer {clear: left; }
```



Any order columns using negative margins



Three columns, positioned, fluid layout

The strategy

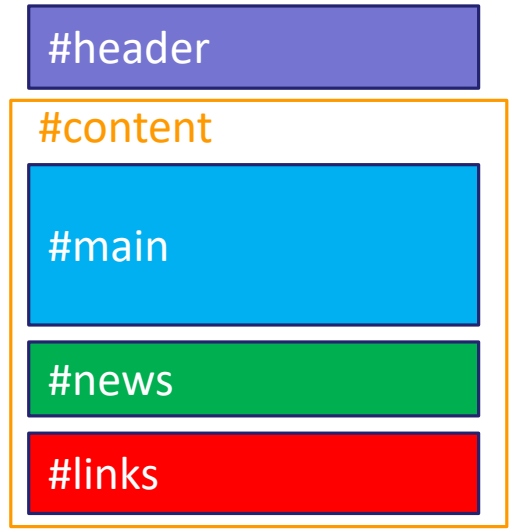
Wrap the three content divs (#main, #news, #links) in a div (#content) to serve as a containing block for the three positioned columns. Then, give the column elements widths and position them in the containing #content element.

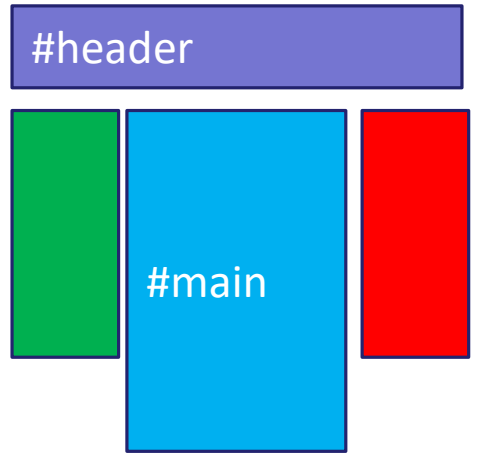
The markup

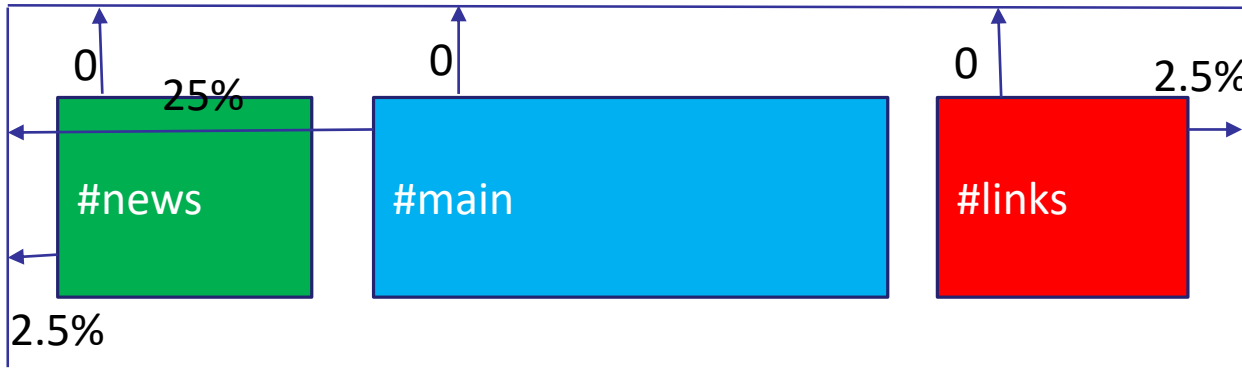
```
<div id="header">Masthead and headline</div>
<div id="content">
   <div id="main">Main article</div>
   <div id="news">News items</div>
   <div id="links">List of links</div>
</div>
```

The styles

```
    #content {position: relative; margin: 0; }

    #main {width: 50%; position: absolute; top:0; left: 25%; margin: 0;}

    #news { width: 20%; position: absolute; top:0; left: 2.5%; margin: 0;
}

    #links {width:20%; position: absolute; top: 0; left: 2.5%; margin: 0;}
```







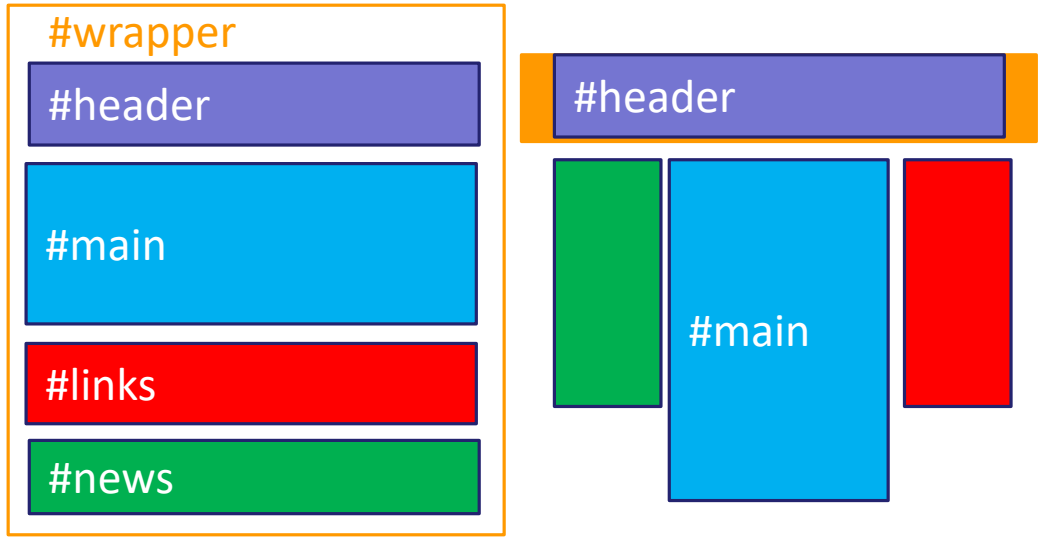
Three columns, positioned, fixed

The markup

```
<div id="wrapper">
<div id="header">Masthead and headline</div>
<div id="content">
   <div id="main">Main article</div>
  <div id="links">List of links</div>
   <div id="news">News items</div>
</div>
</div>
```

The styles

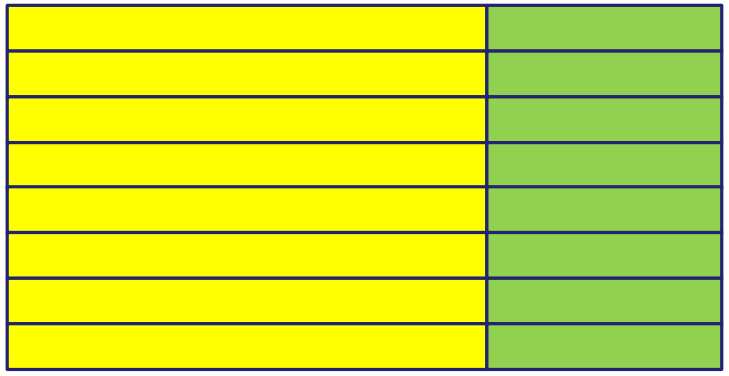
```
#wrapper {width: 960px; margin: 0 auto; }
#content {position: relative; margin: 0; }
#main {width: 520px; position: absolute; top:0; left: 220px; margin: 0;}
#news { width: 200px; position: absolute; top:0; left: 0; margin: 0; }
#links {width:200px; position: absolute; top: 0; right: 0px; margin: 0;}
```



Top-to-Bottom Backgrounds

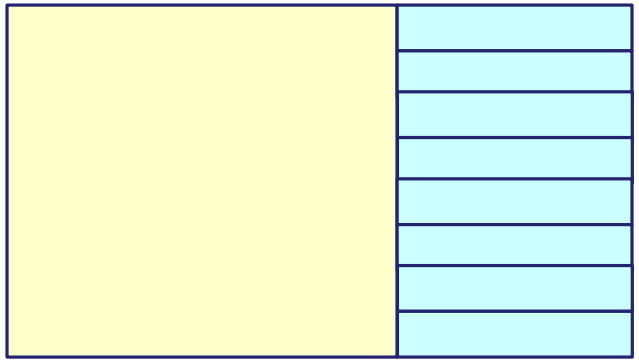
Vertical Repeated Background Image

```
#wrapper {
   width: 960px;
    margin: 0 auto;
    background-image: url(two_columns.png);
    background-repeat: repeat-y;
}
```



Faux columns for fluid layouts

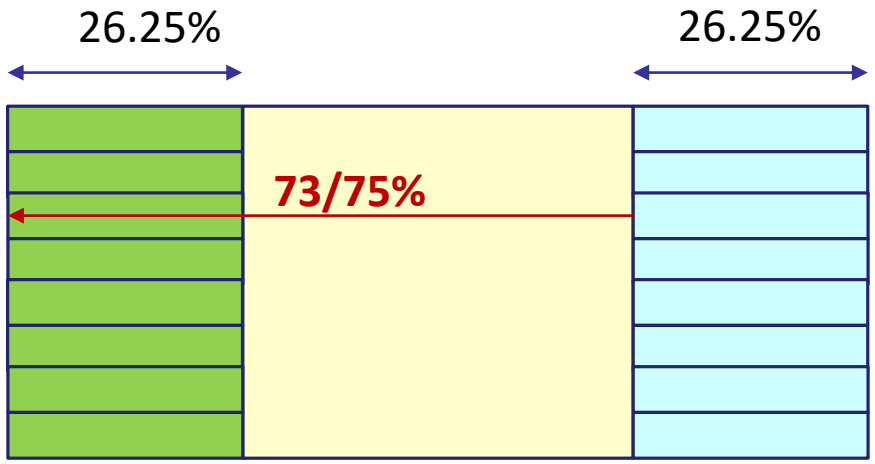
```
body {
    background-image: url(two_cols_3000px.png);
    background-repeat: repeat-y;
    background-position: 76.5%;
}
```



Top-to-Bottom Backgrounds

Three Faux Columns

```
<div id="wrapper">
 <div id="inner">
   <div id="main"></div>
   <div id="links"></div>
   <div id="news"></div>
 </div>
</div>
```




```
#links {.....;  width: 26.25%;}
#news {.....; left: 73.25; width: 26.25; }
```