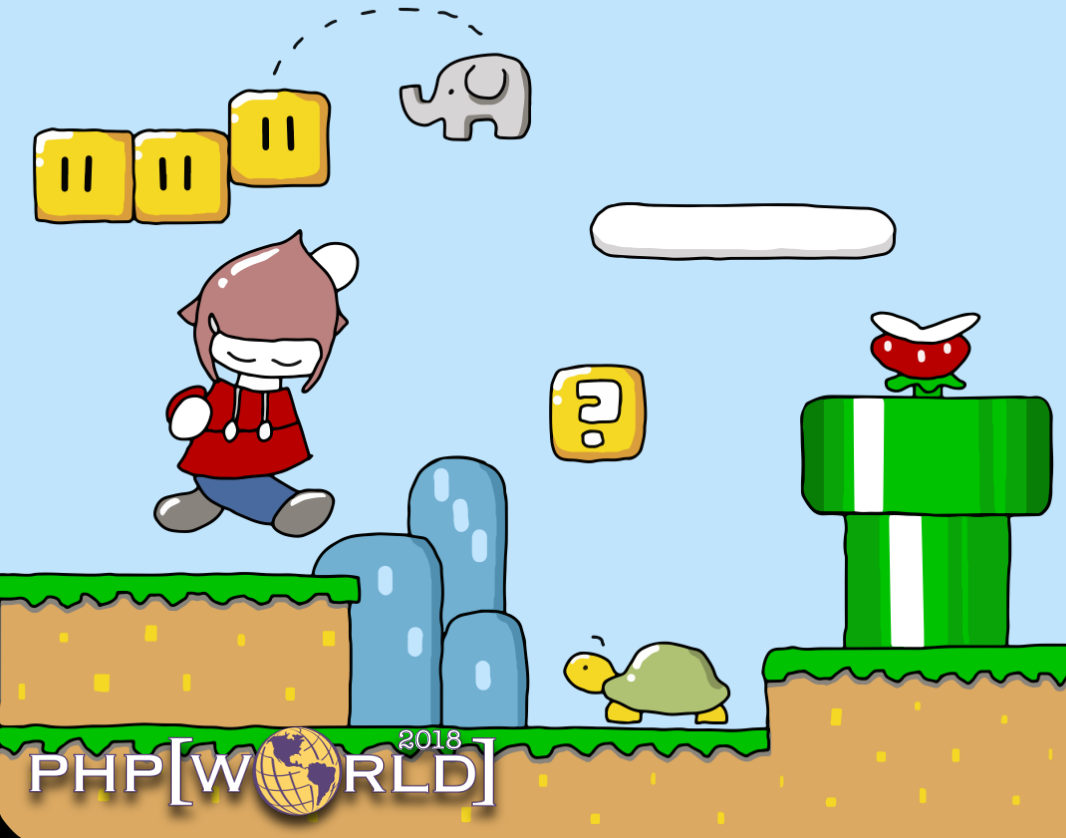


JEN KRAMER • HARVARD UNIVERSITY EXTENSION SCHOOL
@JEN4WEB

MODERN LAYOUTS WITH FLEXBOX, GRID, CALC, AND CUSTOM FIELDS

PART 1: FLEXBOX

Thanks to Our Sponsors



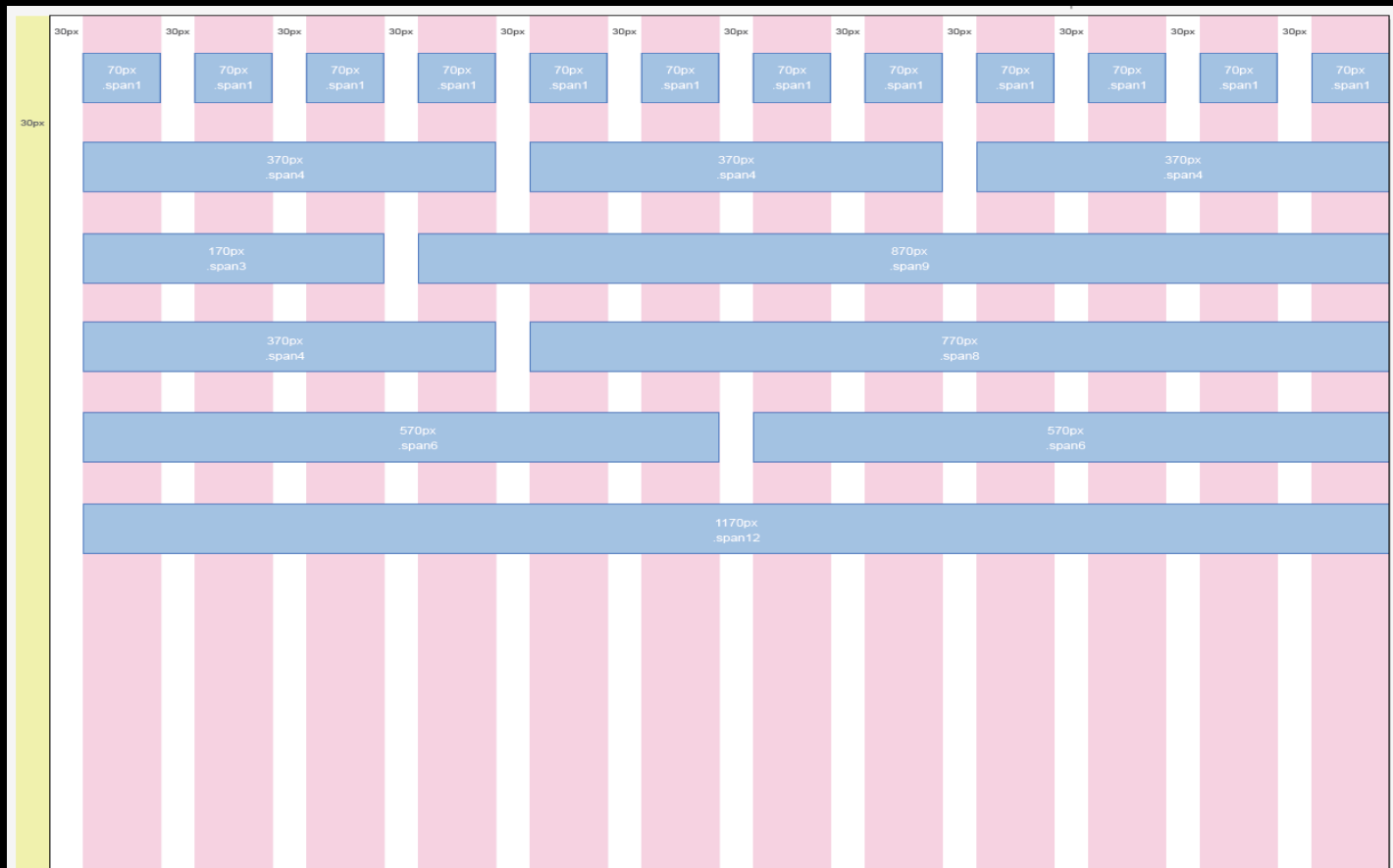
AUTOMATTIC



RESPONSIVE DESIGN

- Defined by three characteristics
 - Flexible grid-based layout
 - Media queries (CSS3)
 - Images that resize
- www.alistapart.com/articles/responsive-web-design/

GRID-BASED LAYOUT



IMAGES THAT RESIZE

- Images should change size, based on screen resolution
- Solutions available client side and server side, including new `<picture>` tag



CSS3 MEDIA QUERIES

- Browser reports screen resolution
- Based on current width, serve a stylesheet with layout for that width
- No JavaScript involved



PART 1

FLOATS

FLOATS

- A hack from the start, right after table-based layout!
- Features rows and cells.
- Rows clear the floats on the cells.
- Source ordering determines display, though some (minor) rearrangement is possible.
- Major disadvantage: equal column heights

.ROW

CELL/
.COL-1

CELL/
.COL-1

CELL/
.COL-1

CELL/
.COL-1

```
<div class="row">  
  <div class="col-1"></div>  
  <div class="col-1"></div>  
  <div class="col-1"></div>  
  <div class="col-1"></div>  
</div>
```

.ROW

CELL/
.COL-1

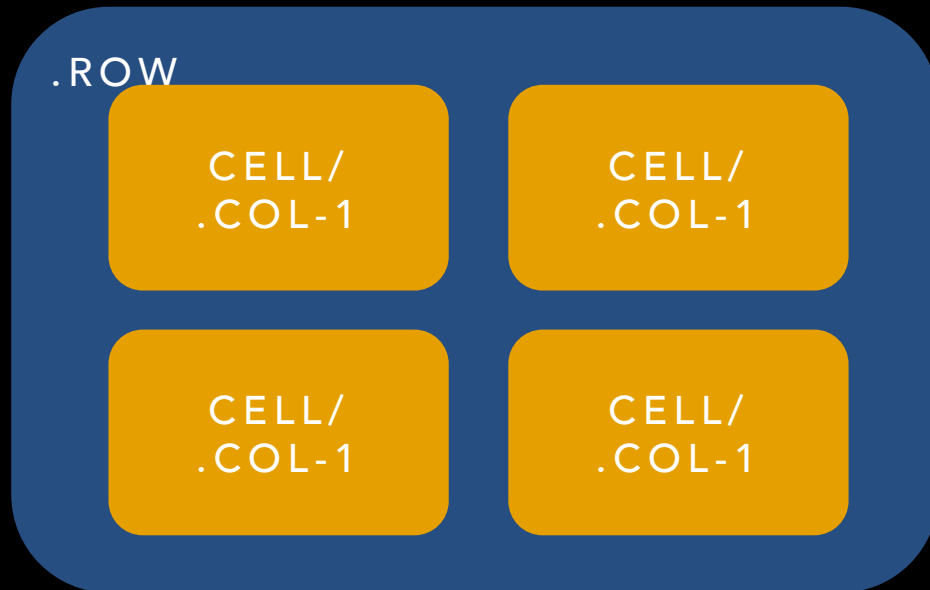
CELL/
.COL-1

CELL/
.COL-1

CELL/
.COL-1

```
.row::after {  
  content: "";  
  display: table;  
  clear: both;  
}
```

```
.col-1 {  
  float: left;  
  margin-left: 4%;  
  width: 20%;  
}
```



```
@media only screen and (min-width: 480px)
    and (max-width: 767px) {
    .col-1 {
        width: 44%;
    }
}
```

ROW

CELL/
.COL-1

CELL/
.COL-1

CELL/
.COL-1

CELL/
.COL-1

```
@media only screen and  
  (max-width: 479px) {
```

```
  .col-1 {  
    width: 98%;  
    margin: 1%;  
    float: none;
```

```
  }
```

```
}
```

.ROW

CELL/
.COL-1

1

CELL/
.COL-1
2

CELL/
.COL-1
3

CELL/
.COL-1
4

There can be layout problems with floats.

This can be resolved with JavaScript, with a column equalizer script.

```
/* rearranging the columns */
```

```
[class*="col-"] {  
    position: relative;  
}  
.col-push-1 {  
    left: 26%;  
}  
.col-pull-3 {  
    left: -74%;  
}
```

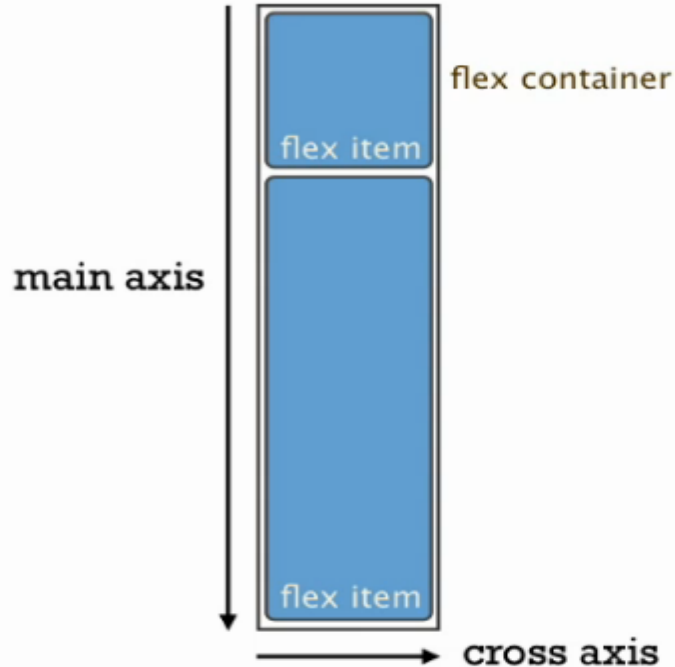
PART 2

FLEXBOX

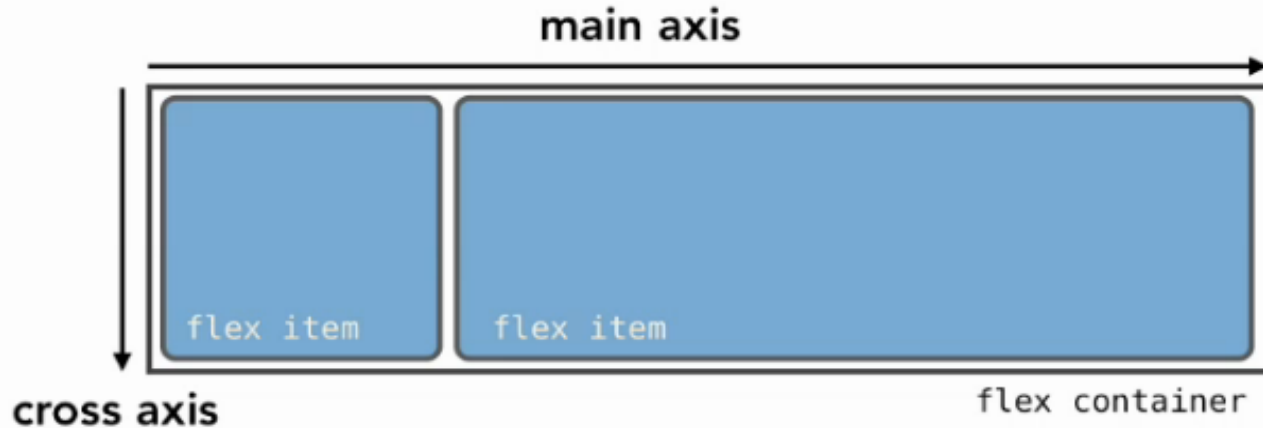
FLEXBOX

- The first layout elements – but not designed to lay out whole web pages
- Features flex-containers (row) and flex-items (cells). Both are required to work.
- Excels at vertical centering and equal heights
- Very easy to reorder boxes
- Major disadvantages:
 - Wasn't designed to be locked down for layouts! Works in 1 dimension only.
 - Browser support and syntax is challenging.

Flex container set to column



Flex container set to row



<http://www.lynda.com/CSS-tutorials/CSS-Flexbox-First-Look/116352-2.html>

THREE VERSIONS OF FLEXBOX

- 2009: `display: box;`
- 2011: `display: flexbox;` ("tweener" syntax)
- 2016: `display: flex;`
- Prefixing may still be required depending on browser support desired – but browser support is at 95% according to caniuse.com

EXAMPLE

```
ul {  
    display: -webkit-flex; /* targets Chrome, Safari */  
    display: -ms-flexbox; /* targets IE10 */  
    display: flex;  
}
```

CURRENT SUPPORT

- Internet Explorer
 - \leq IE 9: not supported
 - IE 10 supports "tweener" syntax (ms prefix)
 - IE 11, Edge: Full support (though buggy in IE 11)
- Safari 7.1/8, iOS Safari 7/8 require webkit prefix
- Others support current syntax (including Opera!)
- <http://caniuse.com/#feat=flexbox>

BASIC FLEXBOX

CODE DEMO

- Now that you have seen Flexbox in action, time to test your knowledge with either:
- Flexbox Froggy: <http://flexboxfroggy.com/>
- Flexbox Defense: <http://www.flexboxdefense.com/>



FLEXBOX PROPERTIES

Parent (Flex Container)

display: flex | inline-flex;

flex-direction: row | row-reverse | column | column-reverse;

flex-wrap: wrap | nowrap | wrap-reverse;

flex-flow (shorthand for flex-direction and flex-wrap)

justify-content: flex-start | flex-end | center | space-between
| space-around | space-evenly;

align-items: flex-start | flex-end | center | baseline | stretch;

align-content (cross axis - adjust to largest item): flex-start |
flex-end | center | stretch | space-between | space-around;

Children (Flex Items)

order: <integer>;

flex-grow: <number>;

flex-shrink: <number>;

flex-basis: <length> | auto;

flex: shorthand for grow, shrink, and basis
(default: **0 1 auto**)

align-self: overrides alignment set on parent

FRAMEWORKS

FLEXBOX GRID

Flexbox Grid

A grid system based on the **flex** display property.

[Download](#)[Github](#)

Responsive

Responsive modifiers enable specifying different column sizes, offsets, alignment and distribution at xs, sm, md & lg viewport widths.



```
<div class="row">
  <div class="col-xs-12
    col-sm-8
    col-md-6
    col-lg-4">
```

flexboxgrid.com



Bootstrap

Build responsive, mobile-first projects on the web with the world's most popular front-end component library.

Bootstrap is an open source toolkit for developing with HTML, CSS, and JS. Quickly prototype your ideas or build your entire app with our Sass variables and mixins, responsive grid system, extensive prebuilt components, and powerful plugins built on jQuery.

[Get started](#)[Download](#)

Currently v4.0.0-beta



getbootstrap.com

.ROW/CONTAINER

ITEM/
.COL-1

ITEM/
.COL-1

ITEM/
.COL-1

ITEM/
.COL-1

```
<div class="row">  
  <div class="col-1"></div>  
  <div class="col-1"></div>  
  <div class="col-1"></div>  
  <div class="col-1"></div>  
</div>
```

```
.row {  
  display: flex;  
  flex-flow: row wrap;  
  justify-content: center;  
  margin: 1%;  
}
```

Change flex-flow to other values to change direction of rows – row reverse, column reverse, no wrap

.col-1 includes:

```
flex: 0 0 24%; /* desktop */
```

```
flex: 0 0 48%; /* tablet */
```

```
flex: 0 0 98%; /* phone */
```

To change widths on .col-1, change the flex-basis property. This is more flexible than width.

```
/* rearranging the columns */
```

```
.col-push-1 {  
    order: 2;  
}  
.col-pull-3 {  
    order: 1;  
}
```

- Code a Flexbox-based grid based on what you've seen here, using the folder [3-flexbox-grid](#) as your starting point
- 2 media queries/3 dimensions as before
- IF YOU HAVE EXTRA TIME:
 - Consider reordering and reversing rows
 - Consider alignment issues, nesting, offsets, etc.

- In folder [4-pie-flexbox](#), in begin folder, this is a web page laid out using a float-based grid system.
- Apply the flexbox-based grid you just generated to this web page.
 - Modify the HTML as little as possible.
 - You may need to adjust some of the code you've written to match this specific case.
 - You will likely need to add some styling beyond what you've already written.

PROPER USE OF FLEXBOX

- Flexbox grid is a hack!
- Flexbox = "flexible boxes" – grid reduces their flexibility
- How can we use Flexbox properly?

- Create an image gallery with an "unknown" number of images, from folder [5-image-gallery](#)
- Use the 17 pie images given to you, then change the order and/or number of them displayed on the page with HTML
- Find two solutions to this image gallery problem:
 - Allow images to display on the page in any number per row with any alignment you choose
 - Allow images to display in a fixed number per row for a given screen dimension

QUESTIONS?

Jen Kramer

Watertown, MA, USA

Phone: 802-257-2657

jen@jenkramer.org

www.jenkramer.org

Twitter: @jen4web

Facebook: facebook.com/webdesignjen

Code available at
www.github.com/jen4web/phpworld

Slides available at
www.slideshare.net/jen4web

CSS
IS
AWESOME