Scripting & Computer Environments

Web Programming: CSS

IIIT-H

Sep {21,25}, 2013

...Previously & Today...

Thus far:

- Basic HTML
- XHTML
- HTML5 (an overview)

Today:

Cascading Style Sheets (CSS)

Brainstorm

- What is a “cool” website for you (from experience)? Why?

- Psycology in web design (e.g. color psycology)?

Styling: Without CSS

There are deprecated and browser-specific styling elements for:

- Backgrounds
  - The bgcolor attribute
  - The background attribute

```
<body bgcolor="gray">

<table bgcolor="#cccccc">

<tr bgcolor="green">

<body background="image.gif">
```

Styling: Without CSS (2)

- Fonts
  - The `<font>` element: specifies the face, size and color.
    - Face → {`arial, sans-serif, verdana` ...}
    - Size → between 1 (smallest) and 7 (largest).
    - Color → either in name or hex code.
  - The `<basefont>` element: sets default face, font size and color.

Without CSS (3)

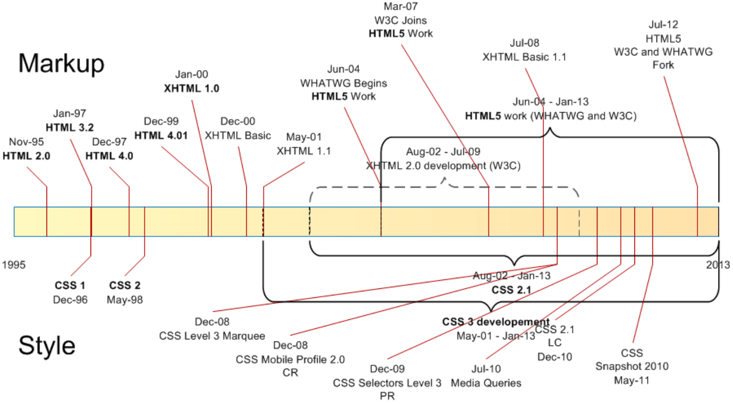
- Formatting
  - The `<s>`, `<strike>` elements for strikethrough
  - The `<u>` element for underline
  - The `<center>` element
  - The `align` attribute → {`left, right, center, justify` ...}
  - The `width` and `height` attributes

CSS: The Whats

- ≡ **C**ascading **S**tyle **S**heets.
- A styling language that defines the presentation/style of pages.
- HTML structures content; CSS *formats* the structured content.
- Defines styles for:
  - `fonts, texts, backgrounds, tables, borders, dimensions, padding ...`
- Goals:
  - Separate content/structure from style/presentation.
  - Single point of control
  - Interoperability

CSS: Evolution

- The W3C has defined various levels (“versions”) of CSS.
- Each level supersets the last and adds new features.



Markup

Nov-95
HTML 2.0

Jan-97
HTML 3.2

Dec-97
HTML 4.0

Dec-99
HTML 4.01

Jan-00
XHTML 1.0

Dec-00
XHTML Basic

May-01
XHTML 1.1

Jun-04
WHATWG Begins
HTML5 Work

Mar-07
W3C Joins
HTML5 Work

Jul-08
XHTML Basic 1.1

Jul-12
HTML5
W3C and WHATWG
Fork

Jun-04 - Jan-13
HTML5 work (WHATWG and W3C)

Aug-02 - Jul-09
XHTML 2.0 development (W3C)

1995

2013

CSS 1
Dec-96

CSS 2
May-98

Aug-02 - Jan-13
CSS 2.1

Style

Dec-08
CSS Level 3 Marquee

Dec-08
CSS Mobile Profile 2.0
CR

CSS 3 developement
May-01 - Jan-13

Dec-09
CSS Selectors Level 3
PR

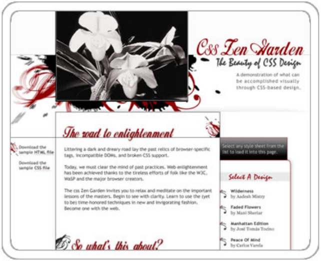
Jul-10
Media Queries

CSS 2.1
LC
Dec-10

CSS
Snapshot 2010
May-11

- An important web design consideration → browser support

Some CSS Designs



Css Zen Garden

The Beauty of CSS Design

A demonstration of what can be accomplished visually through CSS-based design.

Download the sample HTML file

Download the sample CSS file

The road to enlightenment

Littering a dark and dreary road lay the past relics of browser-specific tags, incompatible DOMs, and broken CSS support.

Today, we must clear the mind of past practices. Web enlightenment has been achieved thanks to the tireless efforts of folk like the W3C, WaSP and the major browser creators.

The css Zen Garden invites you to relax and meditate on the important lessons of the masters. Begin to see with clarity. Learn to use the (yet to be) time-honored techniques in new and invigorating fashion. Become one with the web.

So what's this about?

Select any style sheet from the list to load it into this page.

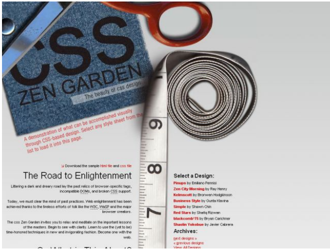
Select A Design

Wilderness by Aadesh Mistry

Faded Flowers by Mani Sheriar

Manhattan Edition by José Tomás Tocino

Peace Of Mind by Carlos Varela



CSS ZEN GARDEN

The beauty of css design

A demonstration of what can be accomplished visually through CSS-based design. Select any style sheet from the list to load it into this page.

Download the sample html file and css file

The Road to Enlightenment

Littering a dark and dreary road lay the past relics of browser-specific tags, incompatible DOMs, and broken CSS support.

Today, we must clear the mind of past practices. Web enlightenment has been achieved thanks to the tireless efforts of folk like the W3C, WaSP and the major browser creators.

The css Zen Garden invites you to relax and meditate on the important lessons of the masters. Begin to see with clarity. Learn to use the (yet to be) time-honored techniques in new and invigorating fashion. Become one with the web.

Select a Design:

Pinups by Emiliano Pennisi

Zen City Morning by Ray Henry

Kelmscott by Bronwen Hodgkinson

Business Style by Gunta Klavina

Simple by Shawn Chin

Red Stars by Shafiq Rizwan

blackcomb'75 by Bryan Carichner

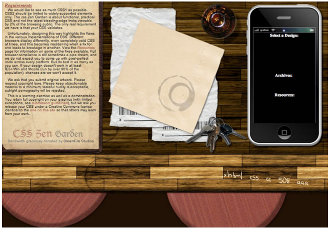
Shaolin Yokobue by Javier Cabrera

Archives:

next designs »

« previous designs

View All Designs



Requirements

We would like to see as much CSS1 as possible. CSS2 should be limited to widely-supported elements only. The css Zen Garden is about functional, practical CSS and not the latest bleeding-edge tricks viewable by 2% of the browsing public. The only real requirement we have is that your CSS validates.

Unfortunately, designing this way highlights the flaws in the various implementations of CSS. Different browsers display differently, even completely valid CSS at times, and this becomes maddening when a fix for one leads to breakage in another. View the Resources page for information on some of the fixes available. Full browser compliance is still sometimes a pipe dream, and we do not expect you to come up with pixel-perfect code across every platform. But do test in as many as you can. If your design doesn't work in at least IE5+/Win and Mozilla (run by over 90% of the population), chances are we won't accept it.

We ask that you submit original artwork. Please respect copyright laws. Please keep objectionable material to a minimum; tasteful nudity is acceptable, outright pornography will be rejected.

This is a learning exercise as well as a demonstration. You retain full copyright on your graphics (with limited exceptions, see submission guidelines), but we ask you release your CSS under a Creative Commons license identical to the one on this site so that others may learn from your work.

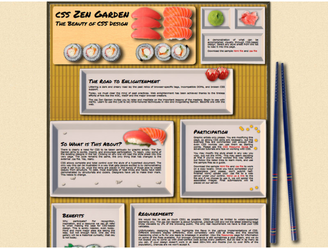
CSS Zen Garden

Bandwidth graciously donated by DreamFire Studios

Select a Design:

Archives:

Resources:



CSS Zen Garden

The Beauty of CSS Design

A demonstration of what can be accomplished visually through CSS-based design. Select any style sheet from the list to load it into this page.

Download the sample html file and css file

The Road to Enlightenment

Littering a dark and dreary road lay the past relics of browser-specific tags, incompatible DOMs, and broken CSS support.

Today, we must clear the mind of past practices. Web enlightenment has been achieved thanks to the tireless efforts of folk like the W3C, WaSP and the major browser creators.

The css Zen Garden invites you to relax and meditate on the important lessons of the masters. Begin to see with clarity. Learn to use the (yet to be) time-honored techniques in new and invigorating fashion. Become one with the web.

So What is This About?

There is clearly a need for CSS to be taken seriously by graphic artists. The Zen Garden aims to excite, inspire, and encourage participation. To begin, view some of the existing designs in the list. Clicking on any one will load the style sheet into this very page. The code remains the same, the only thing that has changed is the external .css file. Yes, really.

CSS allows complete and total control over the style of a hypertext document. The only way this can be illustrated in a way that gets people excited is by demonstrating what it can truly be, once the reins are placed in the hands of those able to create beauty from structure. To date, most examples of neat tricks and hacks have been demonstrated by structurists and coders. Designers have yet to make their mark. This needs to change.

Participation

Graphic artists only please. You are modifying this page, so strong CSS skills are necessary, but the example files are commented well enough that even CSS novices can use them as starting points. Please see the CSS Resource Guide for advanced tutorials and tips on working with CSS.

You may modify the style sheet in any way you wish, but not the HTML. This may seem daunting at first if you've never worked this way before, but follow the listed links to learn more, and use the sample files as a guide.

Download the sample html file and css file to work on a copy locally. Once you have completed your masterpiece (and please, don't submit half-finished work) upload your .css file to a web server under your control. Send us a link to the file and if we choose to use it, we will spider the associated images. Final submissions will be placed on our server.

Benefits

Why participate? For recognition, inspiration, and a resource we can all refer to when making the case for CSS-based design. This is sorely needed, even today. More and more major sites are taking the leap, but not enough have. One day this gallery will be a historical curiosity; that day is not today.

Requirements

We would like to see as much CSS1 as possible. CSS2 should be limited to widely-supported elements only. The css Zen Garden is about functional, practical CSS and not the latest bleeding-edge tricks viewable by 2% of the browsing public. The only real requirement we have is that your CSS validates.

Unfortunately, designing this way highlights the flaws in the various implementations of CSS. Different browsers display differently, even completely valid CSS at times, and this becomes maddening when a fix for one leads to breakage in another. View the Resources page for information on some of the fixes available. Full browser compliance is still sometimes a pipe dream, and we do not expect you to come up with pixel-perfect code across every platform. But do test in as many as you can. If your design doesn't work in at least IE5+/Win and Mozilla (run by over 90% of the population), chances are we won't accept it.

source: css Zen Garden

CSS: Syntax

- A CSS file is a list of rules (rule = selector + ≥ 1 declaration).
- Each rule defines a location and the style to be applied there.

General Syntax

```
selector {
  property 1: value 1;                 /* this is a comment */
  property 2: value 2;
  ...                                 /* each rule is semicolon-separated */
  property n: value n;
  }
```

- `Selector`: indicates which element(s) the rule applies to.
- `Property`: of the element(s) to be affected.
- `value`: the chosen style (specification) for the property.

Example

```
/* style.css */

p {
   font-size:10pt;
  }


h1,h3 {
      color: darkred;
      font-style: italic;
      }

body {
      color:#000000;
      font-family:sans-serif, verdana;
     }
```

CSS: Selectors

- Select an element(s) for styling.
- More specific selectors possible.
- Various ways of selection, not just using the elements' names.
- Some flavors:
  1. Simple Selectors
  2. The Universal Selector
  3. The Type Selector
  4. The Class Selector
  5. The ID Selector

The Universal Selector (*)

- Selects *all* elements in the document.

```
Usage:      * { property:value; }
```

The Type Selector

- Selects a given element(s), regardless of position.
- i.e. every instance matching an element's type name.

```
Usage:      elt1, elt 2 ...elt n { property:value; }
```

The Universal Selector (*)

- Selects *all* elements in the document.

```
Usage:      * { property:value; }
```

The Type Selector

- Selects a given element(s), regardless of position.
- i.e. every instance matching an element's type name.

```
Usage:     elt1, elt 2 ...elt n { property:value; }
```

The Class Selector

- Selects an element (often elements) with a `class` attribute.
- The value of this attribute be specified in the class selector.

Given:

```
<p class="para1"> First paragraph </P>
```

To select:

```
p.para1 { property:value; }          or

.para1 { property:value; }        (N.B. the dot)
```

The ID Selector

- Like the class selector, but acts on the id attribute.
- Instead of dot, uses the # symbol.

Given:

```
<h1 id="intro"> Heading 1 </h1>
<p id="course"> Scripting and Computer Environments </p>

#intro {property:value}                /* selects h1 */
#course {property:value}               /* selects p  */
```

Other selectors:

- The Child Selector (>)
- The Descendant selector
- Adjacent selector (+)
- Attribute selectors (based on attributes or their values)

The ID Selector

- Like the class selector, but acts on the `id` attribute.
- Instead of dot, uses the `#` symbol.

Given:

```
<h1 id="intro"> Heading 1 </h1>
<p id="course"> Scripting and Computer Environments </p>

#intro {property:value}                 /* selects h1 */
#course {property:value}                /* selects p  */
```

Other selectors:

- The Child Selector (>)
- The Descendant selector
- Adjacent selector (+)
- Attribute selectors (based on attributes or their values)

CSS: The Hows

- Applying CSS to (X)HTML document:

  1. The `inline` style sheet approach
     - By using the `style` attribute of an element.
  2. The `internal` style sheet approach
     - By using the `<style>` element inside the `<head>` tag.
  3. The `external` style sheet approach
     - By linking an external CSS file using the `<link>` element.

Selection criteria → reuse, maintenance, sharing

CSS Properties

Styles available for:

- Fonts, text formatting
- Background
- Lists
- Links

- Tables
- Borders, padding, margins
- Dimensions, positioning (Read!)
- Other fancy stuff

For a complete listing, you may visit http://www.w3schools.com/cssref/

- Font vs typeface

CSS Font Properties

Property	Value
`font-family`	{arial, verdana, sans-serif, times ...}
`font-size`	{small, smaller, medium, large, larger, px, pt, mm, cm...}
`font-weight`	{normal, bold, bolder, 100, 200... 900 }
`font-style`	{normal, italic, oblique}
`font-variant`	{normal, small-caps}
`font-stretch`	{normal, wider, narrower, condensed, expanded ...}

CSS Text Properties

Property	Value
`color`	{#FF0000, red, aqua, purple, olive, maroon, gray ...}
`text-align`	{left, right, center, justify}
`word-spacing`	specified in units of length (em, in, mm, px, pt...)
`letter-spacing`	”
`text-indent`	”
`text-transform`	{none, capitalize, uppercase, lowercase}

CSS Properties: Backgrounds

- The `background-color` property
- The `background-image` property
- The `background-repeat` - repetition of the background image.
  - Values → {`repeat` (default), `repeat-x`, `repeat-y`, `no-repeat`}

Example

```
h1 { background-color:#ccbbaa; }

body { background-image: url(‘image.gif’); }

body { background-image: url(‘image.gif’);
       background-repeat:repeat-y;
     }
```

CSS Properties: Lists

- Unordered vs Ordered lists (HTML)
- The CSS list properties set different styles of numbers and markers, including images.
- The `list-style-type` property controls the style.
    - Possible values for ordered lists + unordered lists:

`decimal`	`(1,2,3 ...)`
`decimal-leading-zero`	`(01, 02, 03, ... )`
`lower-roman`	`(i, ii, iii, ...)`
`upper-roman`	`(I, II, III, ...)`
`lower-alpha`	`(a, b, c ...)`
`upper-alpha`	`(A, B, C ...)`

`{none, disk, square, circle}`

- The list-style-image property specifies an image as a marker.

Example

```
ol.a {list-style-type: lower-roman;}    /* class="a" */
ol.b {list-style-type: lower-alpha;}    /* class="b" */
ul.c {list-style-type: circle;}
ul.d {list-style-type: square;}


ul { list-style-image: url(‘image.gif’); }
li { list-style-image: url(‘image.gif’); }
```

CSS Properties: Links

- Can be styled with the `font`, `text`, `background`, etc properties.
- Also, state-based styling (a.k.a. `link pseudo-classes`):
  1. `a:link` → a normal, unvisited link
  2. `a:visited` → a visited link
  3. `a:hover`[1] → a mouse-over link
  4. `a:active`[2] → a selected link
- Common link style properties:
  - `text-decoration` {`none`, `underline`, `overline`, `line-through`}
  - `background-color`

[1] Must come after the first two

[2] Must come after `a:hover`

Next...

- The CSS Box Model
- Table Styling
- CSS3
- CSS Design Examples

Brainstorm

- Why is it called *cascading* anyway?

Brainstorm

- Why is it called *cascading* anyway?

Properties of an element cascade in some order (4 = highest priority):

1. Browser's default styles
2. External style sheets
3. Internal style sheets
4. Inline style sheets

Brainstorm

- Why is it called *cascading* anyway?

  Properties of an element cascade in some order (4 = highest priority):

  1. Browser's default styles
  2. External style sheets
  3. Internal style sheets
  4. Inline style sheets

- Conflict resolution (intra-sheet, inter-sheet, selectors, etc)?

  e.g.

```
p, h1, h6 { color:red; font-style:italic; }
p h6 { background-color:gray; color:green; }
```

CSS: Lengths

- Many CSS properties demand that length be specified.
- Can be specified in either:

1. Absolute units

`pt`	`a point`	`1/72 of an inch`
`in`	`an inch`	
`mm`	`a milimeter`	
`cm`	`a centimeter`	
`pc`	`a picameter`	`1/12 of an inch`

CSS: Lengths (2)

2. Relative units

`px`	`a pixel`	`(smallest unit of resolution)`
`em`	`1 em is equal to the current font size.`	
`ex`	`height of lower x`	`(depends on font size & type)`

3. Percentage (%)

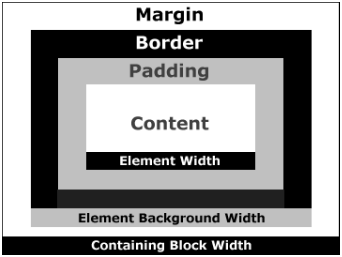
The CSS Box Model

- In CSS, each element is seen as a box.
- Each box has the following properties:

Border - around the element.

Padding - space b/n content & border

Margin - space b/n border & the next box.



```
width = content width + L/R padding + L/R margin

height = content height + T/B padding + T/B margin
```

The CSS Box Model: Borders

- The CSS border properties specify the look and feel of borders.

Border Property	**Value**
`border-color`	colors in name, hex or RGB value
`border-style`	{`none, solid, double, dotted, dashed groove, ridge, inset, outset, hidden`}
`border-width`	in width units or {`thin, medium, thick`}

- The `border` property combines the previous 3 properties.

```
e.g.      h1,h2 { border: 5px dotted blue; }
```

- Each side of the border also can be styled individually.

```
     border-[top|bottom|left|right]-[color|style|width]
```

```
e.g.
     p { border-left-color: rgb(255, 0, 0); }

     body > div { border-bottom-style: dotted; }
```

The CSS Box Model: Padding

- The padding property specifies the amount of space b/n content and border.

```
e.g.
      body { padding:20px; border: double; }

      h2 { padding: 0px; color:#00FF00; }
```

- Possible to set each side's padding too.

```
      padding-[bottom|top|left|right]
```

```
e.g.
      table { padding-left: 100px; padding-right:100px; }
```

The CSS Box Model: Margins

- Margin = gap between boxes.
- The `margin` property specifies the gap.
- Individual margins too can be set.

```
e.g.

      <p class="one"> First paragraph goes here </p>

      <p class="two"> second paragraph goes here </p>


      .one { margin:20px; background-color:olive; }

      .two { margin-bottom:40px; background-color:aqua; }
```

CSS: Going Tabular

- CSS properties shared by `<table>`, `<th>`, `<tr>` and `<td>` too:
  - Font properties → {`font-family`, `font-size`, `font-style` ...}
  - Text properties → {`text-align`, `color`, ...}
  - Backgrounds → {`background-color`, `background-image`, ...}
  - Border, padding
  - Dimensions → {`width`, `height` ...}

CSS: Going Tabular (2)

- In addition, CSS defines various table-specific styling properties.

Property	**Purpose/value**
`border-spacing`	width b/n cells
`border-collapse`	whether table borders are collapsed into a single border or separated → {`collapse, separate`}
`caption-side`	which side of table the caption (if any) be? {`top, bottom, left, right`}

Also, checkout the `table-layout` and `empty-cells` properties.

A Bird's-eye-view

CSS3: The Whats

- The latest CSS standard (still work-in-progress).
- Adds new capabilities and features (e.g. animations)
- Split up into modules.
- Backward-compatible
- Browser support of new properties on the rise (see here).

CSS3: The Modules

- Selectors
- Box Model
- Backgrounds and Borders ✓
- Text Effects and Fonts ✓
- 2D/3D Transformations ✓
- Animations ✓
- Multiple Column Layout
- User Interface

Possiblities with CSS3:

- Rounded borders - the `border-radius` property
- Shadowed boxes - the `box-shadow` property
- Image as a border - the `border-image` property

CSS3 Modules: Fonts

- Before CSS3, "web-safe" fonts only.
- Now, you can enjoy any font you like.
- Be performance-conscious though (e.g. page load time).
- The new `@font-face` rule for user-defined fonts.

  e.g. Check out the Google Fonts API for open-source fonts.

CSS3 Modules: 2D/3D Transforms

- Change an element's size, shape and position.
- Some 2D transform methods
  - `translate()`
  - `rotate()`
  - `scale()`
  - `skew()`
  - `matrix()`
- Some 3D transform methods
  - `rotateX()`
  - `rotateY()`
  - `rotateZ()`
  - `scaleX()`
  - `scaleY()`
  - `scaleZ()`

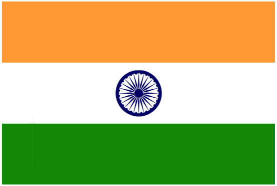
CSS3 Modules: Transitions

- An element gradually changes from one style to another.
- With CSS3, no need to use Flash animations or JS.
- Required parametres:
  - CSS property
  - Duration of the effect.

CSS3 Modules: Animations

- Pre-CSS3: animated images, Flash animations, JS, etc
- With CSS3, using the `@keyframes` rule (bound to a selector).
- Required properties:
  - Name of the animation
  - Duration of the animation.

CSS Design: Example



- Without CSS

- Using CSS (+ CSS3)

Check out!

- The W3C CSS validator
- Firebug
  - A firefox extension to inspect page elements.
  - Modifies style and layout of pages in real-time.
- W3Schools CSS Tutorial (Go for `CSS3` ✓)
- The example codes used in the class.