Introduction to CSS3;

Client: Intuit;
Instructor: Denise R. Jacobs;
Company: Knowledge United;
Agenda: Day 1, Afternoon

- Review of Document Hierarchy
- Selector Review
  - Resolving Specificity
  - Units of Measure in CSS
  - CSS1
- Advanced Selectors
  - CSS2
  - CSS3
Mikey Spillane vs. Mike Hammer

Who is the toughest, smartest, and most steely? You get to be the judge by taking this survey.

Next month's survey: Sir Arthur Conan Doyle or Sherlock Holmes?
HTML Hierarchy
The Hierarchy “Family Tree”

- **Ancestor** is any element that is connected to other elements but is higher up the document tree, no matter how many levels up. For example, in the document above, both the `<html>` and `<body>` tags are ancestors of the `<p>` tag.

- **Descendant** is any element connected to an ancestor, but lower in the document tree, no matter how many levels down. In our example, the `<em>`, `<a>`, and `<img>` are descendants of the `<body>` tag.

- **Parent** is an element directly above a connected element in the document tree. A parent element is also an ancestor, but an element can have ancestors that are not its parents.

- **Child** element is directly below a connected element. A child is a descendant, but an element can have descendants that are not its children.

- **Sibling** elements share the same parent, and are on the same level as each other in the hierarchy.
Selector Review

Day 1, Afternoon
Resolving Specificity
Using specificity, you can create selectors that will zero right in on your desired element(s), but you’ve got to do it by the rules.

A little review:
1. Weight rules
2. Specificity tips and guidelines
Super-Simplified Specificity

The more specific the selector is, the higher the specificity

#id: can only be one on the page
   = high specificity (100)
.clash: can be multiple, but not everywhere
   = medium specificity (10)
element: lots on the page
   = low specificity (1)
* : everything on the page
   = no specificity (0)
Specificity Tips

- Don’t rely too heavily on specificity – leverage as many reusable selectors as possible.
- Use the cascade and source order so that you don’t have to get too specific.
- Trust specificity over source order in terms of which style will win and get applied.
Units of Measure in CSS
Units in CSS

- %: percentage
- in: inch
- cm: centimeter
- mm: millimeter
- em: 1em is equal to the current font size. 2em means 2 times the size of the current font. E.g., if an element is displayed with a font of 12 pt, then '2em' is 24 pt. The 'em' is a very useful unit in CSS, since it can adapt automatically to the font that the reader uses.
- ex: one ex is the x-height of a font (x-height is usually about half the font-size)
- pt: point (1 pt is the same as 1/72 inch)
- pc: pica (1 pc is the same as 12 points)
- px: pixels (a dot on the computer screen)
Colors in CSS1 & CSS2.1

- **color name:**
  A color name, ex. red

- **#rrggbb:**
  A hexadecimal number, ex. #ff0000

- **#rgb:**
  A hexadecimal shorthand number, ex. #fc0

- **rgb(x,x,x):**
  An RGB value,
  ex. rgb(255, 0, 0)

- **rgb(x%, x%, x%):**
  An RGB percentage value,
  ex. rgb(100%, 0%, 0%)
Colors in CSS3

- `rgba(x,x,x,y)`:
  An RGB value with alpha-opacity,
  ex. `rgba(255, 0, 0, 0.2)`

- `hsl(x%, x%, x%)`:
  A HSL value representing hue, saturation, and lightness,
  ex. `hsl(0, 100%, 50%)`

- `hsla(x%,x%,x%,y)`:
  A HSLA value representing hue, saturation, and lightness with alpha-opacity,
  ex. `hsla(0, 100%, 50%, 0.5)`
CSS1 Review
### CSS1 Selector Specification

- **Selectors**
  - Element/type
  - Class
  - ID

- **At Rules**
  - @import

- **Pseudo-elements**
  - :first-letter
  - :first-line

- **Pseudo-classes**
  - :link
  - :active
  - :visited
The element or type selector targets an HTML element, and thus uses a tag name. This enables you to select any of this kind of element in the document.

Example:

```css
p { font-size: 1em; }
```
In HTML, every single tag can have the class attribute.

A class selector targets the value of a class attribute of a tag.

A class attribute can be used multiple times in a document and applied to different elements.

Example:
\[.highlight \{color: \#ffcc00;\}\]
Selector: ID

- ID selectors target an element with a particular id attribute.

- IDs help you zero in on a particular element, because you can only use an id once in any document.

- IDs have a very high specificity weight.

Example:

```
#maincontent {background-color: #eee;}
```
Combinator Selectors: Descendent

Descendant selectors select the element that is a descendant of another element in the document tree.

Syntax:
ancestor selector (space) descendant selector

E F

Example:
div h2 {color: green;}
What is a pseudo-class?

If you wish to affect an entire element, but only in a particular state—such as the hover state—you use a pseudo-class.

Just as pseudo-elements cannot always be deduced from the document tree, pseudo-classes often cannot be so deduced.

They come in two forms: state-changing and stateless.
Pseudo-classes: Links

Link pseudo-classes target link text in its various states.

:link targets an unvisited link

:active targets an active link

:visited targets a visited link
Pseudo-elements: :first-letter

`:first-letter` targets the first letter of text within an element.

Example:

```css
h1 + p:first-letter {font-size: 110%;}
```
What is a pseudo-element?

The pseudo-element selector does not match any real HTML element in the document tree, but affects only a part of a selector.

There are two types of pseudo-elements: those that are stateless, and those that are state-changing. While this is not a complete definition, it will suffice for CSS1 pseudo-elements. We will not consider other types of pseudo-elements until later, because they are part of the CSS2.1 specification.
Pseudo-elements: :first-line

:first-line targets the first line of a line of text within an element.

Example:

p:first-line {font-weight: bold;}

Advanced CSS Selectors

Day 1, Afternoon
Advanced selectors are a good way to specifically target styles for modern browsers.

The right selector will help you achieve targeting nirvana, so it’s important to know which selectors you can use now.
Advanced Selectors: Usage Tips

• All of the CSS2 selectors are supported by the modern browsers, and almost all of the CSS3 ones are, so use them!

• It’s easy to target styles away from the IEs, but target them to the IEs with simpler combinator selectors

• There are “hacks” to target styles to specific browsers, other than the IEs
CSS2 & CSS3 Selector Support

<table>
<thead>
<tr>
<th>Selector</th>
<th>IE 5.5</th>
<th>IE 6</th>
<th>IE 7</th>
<th>IE8</th>
<th>IE9 pr3</th>
<th>FF 3.0</th>
<th>FF 3.5</th>
<th>FF 4b1</th>
<th>Saf4.0 Win</th>
<th>Saf5.0 Win</th>
<th>Chrome 4</th>
<th>Chrome 5</th>
<th>Opera 10.10</th>
<th>Opera 10.53</th>
<th>Opera 10.60</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSS 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>incorrect</td>
<td>incomplete</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>

CSS 2 has become the baseline of CSS support; without it a browser is decidedly backward.

<table>
<thead>
<tr>
<th>CSS 3</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>minimal</td>
<td>incomplete</td>
<td>almost</td>
<td>almost</td>
<td>almost</td>
<td>almost</td>
<td>almost</td>
<td>almost</td>
<td>almost</td>
<td>almost</td>
<td>almost</td>
<td>almost</td>
<td>almost</td>
<td>almost</td>
<td>almost</td>
</tr>
</tbody>
</table>

The compatibility information here is about the CSS3 modules I test. It is not necessarily valid for the browsers’ entire feature sets.

DOM Core

Node manipulation

The W3C DOM Core module defines how to access, read and manipulate an XML document. Well-formed HTML documents, so these methods and properties can be used to completely rewrite any HTML page, if you so wish.

Here you find details on how to find elements, how to create new ones, how to read out node information and how to structure the document.

DOM HTML

HTML tag manipulation

Though HTML documents are XML documents, they have a number of special features that the average XML document doesn’t, so there’s a module for those as well.

Load the update dates of the individual tables (may take a little while).
CSS2.1 Selectors
• Universal  *

• Child Combinator
  E > F

• Adjacent/Sibling Combinator
  E + F

• Attribute E[~attribute]

• At Rules
  - @font-face
  - @media
  - @page
  - @charset

• Pseudo elements
  - :before
  - :after

• State pseudo-classes
  - Dynamic
    • :hover
    • :active
    • :focus

  - Language
    • :lang

  - Structural
    • :first-child
• The universal selector is the asterisk (*).

• This selector lets you select every element on the page and apply the style rules to them.

Example:

* {font-family: Arial, sans-serif;}

Selector: Universal
A child selector targets an element that is a child of another element.

A child is a direct descendant of an element in the document tree (as opposed to a more distant descendant).

**Syntax:**

parent selector > descendant selector

E > F

**Example:**

p > strong {font-family: Tahoma, sans-serif;}


A sibling selector (also known as adjacent) selects an element that is next to another element in the document tree.

Syntax:

```
sibling selector + sibling selector
  {property: value; } 
E + F
```

Example:

```
p + ol {font-family: Georgia, serif;}
```
selector[attribute] targets a selector with a particular attribute.

Example:

- a[title] {font-variant: italic;}

Combinator Selectors: Attribute
These pseudo-classes were typically used on the link element, but you can apply them to any element on the page as well.

: hover targets any element that is being selected by a pointing device (such as a cursor)

: focus targets an element that is in focus in the browser (usually an input element)
Pseudo-classes: Others

**:first-child** targets any element that is the first child of its parent element.

Example:

```css
li:first-child { font-variant: italic; }
```

**:lang(n)** targets any element on the basis of the language that has been set for it.

Example:

```css
:lang(fr) { font-face: Gigi, sans-serif; }
```
Pseudo-elements: before & after

**:before** specifies content to be inserted before a given element.

Example:

```css
#breadcrumbnav:before {content: "Current page:";}
```

**:after** specifies content to be inserted after a given element.

Example:

```css
.time:after {content: "hours";}
```
Order of Dynamic Pseudo-classes

The order of these pseudo-classes is important: it mimics the cascade that the browser follows, and most of the states are mutually exclusive. For example, you must have a regular link state before you have a visited link state.

Many people use the mnemonic LoVe For HAte to remember the order :link, :visited, :focus, :hover, :active.

Usually, all of the link styles are written together:

```css
a {color: #3f0;}
a:link {text-decoration: none;}
a:visited {color: #0f3;}
a:focus {text-decoration: none;}
a:hover {text-decoration: underline; color: #636;}
a:active {color: #f03;}
```
CSS2.1 Selectors & IE Support

- Universal (ie7/8 – yes)
- Child (ie7/8 – yes)
- Sibling/Adjacent (ie7 no, ie8 – yes)
- Attribute (ie7/8 – yes)
- Pseudo elements (ie7/8 – no)
  - ::before
  - ::after
- State pseudo-classes, v2.1
  - :first-child (ie7/8 – yes)
  - :hover (ie7/8 – yes)
  - :active (ie7/8 – yes)
  - :focus (ie7/8 – no)
  - :lang (ie7/8 – no)
Lab Time!

Okay, just kidding 😊
Resources: CSS2 Selectors

CSS3 Selectors!
CSS3 Selector Specification

- General sibling
  \[ E \sim F \]

- Attribute presence
  - a[attribute="value"]
  - a[attribute~="value"]
  - a[attribute|="value"]

- Attribute substrings
  - a[attribute^="value"]
  - a[attribute$="value"]
  - a[attribute*="value"]

- Pseudo-elements*
  *all pseudo-elements indicated with :: in CSS3
CSS3 Selector Specification

- **Pseudo-classes**
  - Target
    - :target
  - Negation
    - :not(s)
  - State
    - :enabled
    - :disabled
    - :checked
    - :indeterminate

- **Structural**
  - :nth-child(n)
  - :nth-last-child(n)
  - :nth-of-type(n)
  - :nth-last-of-type(n)
  - :last-child
  - :first-of-type
  - :last-of-type
  - :only-child
  - :only-of-type
  - :empty
# CSS3 Selector Support

## CSS3 Selectors

<table>
<thead>
<tr>
<th>CSS3: Begins with</th>
<th>MAC</th>
<th>WIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CSS3: Ends with</th>
<th>MAC</th>
<th>WIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CSS3: Matches</th>
<th>MAC</th>
<th>WIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CSS3: Root</th>
<th>MAC</th>
<th>WIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CSS3: nth-child</th>
<th>MAC</th>
<th>WIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CSS3: nth-last-child</th>
<th>MAC</th>
<th>WIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CSS3: nth-of-type</th>
<th>MAC</th>
<th>WIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CSS3: nth-last-of-type</th>
<th>MAC</th>
<th>WIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CSS3: last-child</th>
<th>MAC</th>
<th>WIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CSS3: first-of-type</th>
<th>MAC</th>
<th>WIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>

[http://www.findmebyip.com/litmus](http://www.findmebyip.com/litmus)
Combinator Selectors: General Sibling

A general sibling selector (also known as adjacent) selects an element that shares the same parent as another element in the document tree.

Syntax:

```
sibling selector ~ sibling selector
{property: value; }
```

```
E ~ F
```

Example:

```
p ~ dd {font-family: Georgia, serif;}
```
Combinator Selectors: Attribute presence

**selector[attribute="value"]** targets a selector with a particular attribute and specific value.

Example:

```html
img[src="catchathief.jpg"] {border: 1px solid #999; }
```

**selector [attribute~="value"]** targets a selector with a particular attribute and value where the value is one of multiple values separated by a space.

Example

```html
img[alt~="CSI"] {border: 1px #ff8000 solid; }
```
selector [attribute|="value"] targets an element with an attribute that contains values separated by a hyphen in a list.

Example:

```css
img[title|="large"] { border: 1px solid #000; }
```
selector `[attribute^="value"]` targets an element with an attribute that begins with a prefix of “value”.

Example:

```css
img[title^="th_"] { border: 1px solid #000; }
```

selector `[attribute$="value"]` targets an element with an attribute which ends with a suffix of “value”.

Example:

```css
img[title$="png"] { border: 1px solid #000; }
```
Combinator Selectors: Attribute substrings

**selector [attribute*="value"]** targets an element with an attribute that contains “value” as any part of a value string.

Example:

```css
img[title|="large"] { border: 1px solid #000; }
```
Pseudo-class: Target

:target targets the target of a url on a page – most often used for fragment identifiers.

Example:

div.window:target {
  opacity: 1;
  z-index: 7;
}
Pseudo-class: Negation

:nот targets all elements that are not indicated. A very useful pseudo-class.

Example:

*:not(img) {text-align: left;}

(This would match all elements except images.)
Pseudo-classes: State

- :enabled
- :disabled
- :checked
- :indeterminate
Pseudo-elements: State

The :enabled and :disabled pseudo-classes allow developers to specify the appearance of user interface elements (form controls) that are enabled or disabled, provided that the browser allows styling of form controls.

Example:

```css
input[type="text"]:enabled {
    background: #ffe;
}
input[type="text"]:disabled {
    background: #ddd;
}
```
The :checked pseudo-class allows developers to specify the appearance of checked radio and checkbox elements. Again, this is provided that the browser allows styling of form controls.

Example:

```css
input:checked {border:1px solid #090;}
```
Pseudo-classes: Structural

- Structural
  - :root
  - :nth-child(n)
  - :nth-last-child(n)
  - :nth-of-type(n)
  - :nth-last-of-type(n)
  - :last-child
  - :first-of-type
  - :last-of-type
  - :only-child
  - :only-of-type
The :root pseudo-class

- The :root pseudo-class targets the document’s root element. In HTML, the root element is always the HTML element.
- :root actually has a higher specificity than html.

Example:
:root { background:#ff0; }
The :nth-child() pseudo-class

- The :nth-child() pseudo-class targets an element that has a certain number of siblings before it in the document tree. This argument, which is placed within the parentheses, can be a number, a keyword, or a formula.
- A **number** $x$ matches the $x$-th child.

Example:
- `p:nth-child(3) { color:#f00; }`
The :nth-child() pseudo-class

The keywords odd and even can be used to match child elements whose index is odd or even. The index of an element’s first child is 1, so this rule will match any p element that is the first, third, fifth, and so on, child of its parent element.

Example:

p:nth-child(odd) { color:#f00; }

The formula $a \cdot n + b$ can be used to create more complex repeating patterns. In the formula, $a$ represents a cycle size, $n$ is a counter starting at 0, and $b$ represents an offset value. All values are integers.

Example:

p:nth-child(3n+1) { color:#f00; }
The :nth-last-child() pseudo-class

- The :nth-last-child() pseudo-class works just like the :nth-child() pseudo-class, except that it targets an element that has a certain number of siblings after it in the document tree.

- In other words, it starts counting from the last child instead of the first, and counts backwards.

Example:

```css
tr:nth-last-child(2) {
  background:#ff0; }
```
The :nth-of-type() pseudo-class works exactly like the :nth-child() pseudo-class, but only counts those elements that are of the same type as the element the rule is applied to.

Example:

```css
p:nth-of-type(3) {
    background:#ff0;
}
```
The :nth-last-of-type() pseudo-class

- The :nth-last-of-type() pseudo-class targets an element that has a certain number of siblings of the same element type after it in the document tree.
- Just like the :nth-last-child() pseudo-class, it starts counting from the last child instead of the first, and counts backwards.

Example:
```
p: nth-last-of-type(2) {background: #ff0;}
```
The :last-child pseudo-class targets an element that is the last child of its parent element. It is the same as :nth-last-child(1).

Example:

```css
p:last-child {background:#ff00;}
```
The :first-of-type pseudo-class

- The :first-of-type pseudo-class targets an element that is the first sibling of its type. It is the same as :nth-of-type(1).
- p:first-of-type { background:#ff0; }

```css
p:first-of-type { background:#ff0; }
```
The :last-of-type pseudo-class targets an element that is the last sibling of its type. It is the same as :nth-last-of-type(1).

Example

```
p:last-of-type{background:#ff0;}
```
The :only-of-type pseudo-class

- The :only-of-type pseudo-class targets an element whose parent element has no other children of the same element type.

- It is the same (but with a lower specificity) as :first-of-type:last-of-type or :nth-of-type(1):nth-last-of-type(1).

Example:

```css
p:only-of-type { background: #ff00; }
```
The :only-child pseudo-class

The :only-child pseudo-class targets an element whose parent element has no other element children.

It is the same (but with a lower specificity) as :first-child:last-child or :nth-child(1):nth-last-child(1).

Example:

```
p:only-child {background:#ff00;}
```
Lab Time!

We are going to apply all of the selectors to a page that has already been created with descendend selectors.

Let’s see how we can make the page look exactly the same but with the new CSS3 selectors.
Nena Adornments: For Every Body

You can't resist the lure of a well-proportioned pair of earrings that combines the earthiness of semi-precious stones mixed with the clean lines of contemporary design, can you? Nor the comforting weight of a necklace of artfully combined antique African trade beads, not overly polished and semi-burnished glass. If this is true, then Nena Adornments are made for you.

For everybody who wants to adorn themselves with beautiful, thoughtful, and masterfully crafted jewelry, that delights and inspires, Nena Adornments provides exactly what you are looking for.

Delve deeper into the world of Nena Adornments to see just how this particular blend of eclectic jewelry artistry came into being. More

What's New

Recent Posts

Washed Ashore

The meeting of sand and water produces beautiful treasures of sea glass for us to appreciate. What better way to enjoy them than to incorporate them into adornments? See Nena's latest tribute to mother nature's cleverness. More

Grapes of Elegance

The new Midline evokes images of Southern France: fields of lavender, bunches of ripe juicy grapes hanging from the vine, the richness of Burgundy and Bordeaux, the allure of the Mediterranean. If only we could add scent. More

EarringLove.org Shows Some Love

Our charitable organization, EarringLove.org, was created to spread beauty and goodwill in the world by committing random acts of handmade jewelry kindness. Now you can rely upon the kindness of strangers — for earrings!

How it works: if you ever see Nena with a pair of earrings that you like, just say the magic words (and no, the magic words are not "the magic words", and she will give you the earrings — just like that!). If, however, she has given away more than 30 pairs of earrings to lucky women (and some men) all over the country, the stop the world (you think we're kidding)? We'll be in Brazil for the new year! Seriously! EarringLove and samba will make a great pair!

Want to spread some love of the earing ilk? See how you can get involved!

Video Tutorial: Effortless Elegance

Not quite sure how to work that newest fabulous piece? Can't quite figure what to pair with what? You're not alone. Sometime trying to find your own signature style takes time. The latest Nena video demonstrates how to set your jewelry off and mix and match for the best effect. Get some new ideas — and if you have some to share, add them through the comments!

Nothing but Nena

Etsy Nena

At the Nena Etsy store, you can get just about everything you want. And you can be social while shopping, right from the comfort of your own home.

Social Nena

We know it’s not possible to get enough of Nena, so here are all the other spaces you can find Nena on the internet:

Newsletter Nena

Sign up for our newsletter to get information on new releases, upcoming events, and sales.

Enter your email address below: Submit

Tagged Nena

necklaces semi-precious stones pendant rings earrings matching sets necklace ALJU! earringlove precious stones beads African trade beads matching set bound silver bracelet wax custom Dallas 2010
Nena Adornments: For Every Body

You can't resist the lure of a well-proportioned pair of earrings that combines the eye-catching allure of semi-precious stones along with the keen lines of contemporary design. It's an aesthetic that celebrates the importance of earrings in a way that complemented the clothes you wear and the women who carry it. If you aren't sure about the Nena Adornments, you've got to try them out.

For everybody who wants to adorn themselves with beautiful, thoughtful, and masterfully crafted jewelry that delights and inspires, Nena Adornments provides exactly what you are looking for.

What's New

Latest Lines

Walhed Ashore

The meeting of sand and water produces beautiful treasures of seashells for us to appreciate. What better way to enjoy them than to incorporate them into adornments? See Nena's latest tribute to mother nature's cleverness. More

Grapes of Ellis

The new midline evokes images of Southern France: fields of lavender, bunches of ripe juicy grapes hanging from the vine, the richness of Burgundy and Bordeaux, the allure of the Mediterranean sky. If only we could add scent... More

EarringLove.org Shows Some Love

Facebook

Our charitable organization, EarringLove.org, was created to spread beauty and goodwill in the world by committing random acts of handmade jewelry kindness. Now you can rely upon the kindness of strangers—for earrings.

How it works: if you ever see Nena with a pair of earrings that you like, just say magic words (and then the magic words are not "the magic words"), and she will give you the earrings — just like that! If, for her, she has given away more than 30 pairs of earrings to lucky women (and some men) all over the country. Next stop: the world! (You think we're kidding? We'll be in Brazil for the new year! Seriously! EarringLove and samba will make a great pair.)

Want to spread some love of the earring ilk? See how you can get involved!

Video Tutorial: Effortless Elegance

Not quite sure how to work that newest fabulous piece? Can't quite figure what to pair with what? You're not alone. Sometimes trying to find your own signature style takes time.

The latest Nena video demonstrates how to set your jewelry off and mix and match for the best effect. Get some new ideas — and if you have some to share, add them through the comments!
Lab Time!

Be sure to validate your code with the CSS validator:
http://jigsaw.w3.org/css-validator/
Resources: CSS3 Selectors

- http://www.delicious.com/denisejacobs/selectors+css3
- http://www.quirksmode.org/compatibility.html
- http://www.findmebyip.com/litmus
Coming Tomorrow: CSS3 Properties!