

GOLDEN RULES OF CSS



12 practical attention points for designing webpages with cascade style sheets, & golden links for more information. Gathered from web, css-discuss and personal experience.

Nov. 2006,
franky

1.

Remember the "C" of CSS!

Therefore: work as much as possible in the TOP-DOWN approach of a page, and write the css in the same order.

- ❖ The main logic of css: a liquid tree upside down.
- ❖ Replacing a small stone halfway a cascade can change the bed of 20 rivers down under!
- ❖ Inside cascade branches like a `<div>`: work also a.m.a.p. in the top-down approach.

2.

Do it STEP-BY-STEP, and first the model.

If an error or something undesired is occurring: repair it first before going on.

- ❖ The main logic of css: a liquid tree upside down.
- ❖ In a heavy styled "almost finished" page it is far more difficult to discover why something doesn't work as intended. Worst case scenario: total redesign of the page from scratch ...
- ❖ Take your time for developing the model of the page. The beautiful content styles you have in mind - the real stuff things - have to wait ...
- ❖ If working on an existing page, comment out all elements on which you aren't working yet.
Maybe these elements have styles which are disturbing the rest.
Or they have styles which have to be a part of the rest, but aren't yet.
Take no risk!
- ❖ By creating **id's** or **classes** for almost every element, you're creating your own Zen model. 😊
Easy for adding special element styles in between, and for changing the design afterwards.
If the page is finished, the not needed **id's** or **classes** can be deleted in a quick move (but they don't hurt).

3.

Do it TESTING-AND-TESTING-AND-TESTING at each small step, and you will never find surprises in the end!

Test layout and code.

- ❖ Test layout: the font-scaling, the resolution. Don't forget to scroll up: "Is it also still working in what I made before?"
- ❖ Test code: the html-validator every now and then, html-Tidy for supplementing advises, the css-validator (mostly the warnings can wait).

4 .

Paste all STYLES IN THE HEAD as long as the work is not finished.

- ❖ That's is easy adapting, and quick working for:

5 .

The personal archive: SAVE EVERY STEP with an OWN VERSION NUMBER!

- ❖ Errors are cascading too... - If you notice a major or minor error after say 5 steps, and you don't have the version backups, then you have to remember all changes you made in between, starting with the last backup.
- ❖ My experience is that I always forget some: "I had such a beautiful solution, but now it doesn't work anymore. What was it again, where did I change things, what was the last step before it went wrong? My backup of 20 steps ago isn't useful - now I've to invent the beautiful solution again, while I spent hours to get it..."
- ❖ After finishing the job, be careful if you are planning to delete the developing versions. A zip file of all versions is quick made and saved: maybe you need it over 2 or 5 years...
With 30 versions of a 30kB page-model it will be less than 900kB: your hard disk can have that!

6 .

For each step: DON'T TRUST THE PREVIEW RESULTS of a WYSIWYG html-editor. See it in real browser windows!

7 .

For each step: DEVELOP FOR FIREFOX or OPERA, as (nearly) standards compliant browsers. And if it's working as you want it, then CORRECT FOR INTERNET EXPLORER afterwards.

- ❖ The other way, first developing for IE and then "correcting" for other browsers is almost impossible, due to the proprietary rules and all 1000's of known and mysterious errors and bugs of a thing called IE...

8 .

To see where you are: use TEMP BACKGROUND COLORS for the div-boxes, ul's, li's and other elements.

- ❖ Sometimes a TEMP `{border: 1px dashed red;}` can do the same. But that is a bit more risky, because adding or removing borders can influence the layout (suddenly dropping or lifting of an element can be the result, or appearing or disappearing of paddings with unintended backgrounds).

9 .**Write your css not in 1 line for each element, but ONE LINE FOR EACH PROPERTY/VALUE ITEM.**

- ❖ Then it's easier to read and to change: quick cut/copy/paste of a line, moving to another element, and so on.

10 .**Also in html: don't be soft for your hard disk, DON'T "COMPACT CODE".**

- ❖ Give indents for all `<div>`'s, ``'s, ``'s and other elements. Then it is easy to see what is what, and especially where are the ends of the elements, and where a new element is starting.
- ❖ Nobody will be pointed at, if there are comments in the source code, making clear what is done. Handy if you're coming back after some months or years, when adapting is desired.

11 .**The quick X-OUT METHOD can be used instead of commenting out some line or id with `/* ... */` (in css) or `<!-- ... -->` (in html).**

- ❖ Like this:

```
#content {
  margin: 5px;
  xpadding: 10px;
}
```

or this:

```
<div xid="content">...</div>
```

- ❖ If you forget to remove one later on (in case it was appearing that you didn't need it), no problem: the css-validator or html-validator will help to remember.

12 .**The A-B-C METHOD can be used to discover where on screen a floating `<div>` is ending, and where a `{clear: ...;}` has to be given.**

- ❖ Just type a plain letter **a** just before the first `</div>`, a letter **b** just before the second `</div>`, and so on.
- ❖ Then the position of the letters is not influenced by properties of a `<p>` or other containing element, which can be distracting or moving the position.
- ❖ For better viewing, you can use a contrasting style like:


```
<span style="background: yellow; color: red;">
  a</span>
</div>
```
- ❖ For nested floats, an extra containing div can be needed to separate from other floats.

Golden Links

retrieve: some golden links ...

- ❖ **w3c: css2.1 specifications**
<http://www.w3.org/TR/CSS21/>
- ❖ **w3c: css validator by url**
<http://jigsaw.w3.org/css-validator/validator-uri.html>
- ❖ **w3c: css validator by upload**
<http://jigsaw.w3.org/css-validator/validator-upload.html>
- ❖ **w3c: css validator by direct input**
<http://jigsaw.w3.org/css-validator/validator-text.html>
- ❖ **w3c: html home page**
<http://www.w3.org/MarkUp/>
- ❖ **w3c: html validator by url / upload / direct input**
<http://validator.w3.org/>
- ❖ **html tidy online: by url / upload / direct input**
<http://cgi.w3.org/cgi-bin/tidy>
- ❖ **html tidy extension: firefox, download page (do it!)**
<http://users.skynet.be/mgueury/mozilla/download.html>
- ❖ **webdeveloper extension: firefox etc., download page (do it!)**
<http://chrispederick.com/work/webdeveloper/>
- ❖ **checky extension: firefox etc., download page (do it!)**
<http://checky.sourceforge.net/extension.html>
- ❖ **browsershots: online screenshots different browsers**
<http://www.browsershots.org/>
- ❖ **selectoracle: css explaining**
<http://gallery.theopalgrou.com/selectoracle/index.html>
- ❖ **wasp: the web standards project**
<http://www.webstandards.org/>
- ❖ **position is everything (pie): browser bugs in detail**
<http://www.positioniseverything.net/>
- ❖ **a list apart (ala) magazine: css topics**
<http://www.alistapart.com/topics/code/css/>
- ❖ **css zen garden: experimental examples**
<http://www.csszengarden.com/>
- ❖ **cssliquid: liquid layouts in CSS based design**
<http://www.cssliquid.com/>
- ❖ **css-discuss: css wiki**
<http://css-discuss.incutio.com/>
- ❖ **css-discuss: home page**
<http://www.css-discuss.org/>

and on a more modest scale:

- ❖ this Golden Rules of CSS on internet:
<http://home.tiscali.nl/developerscorner/golden-rules-of-css.htm>
- ❖ *francky's developers corner*: home
<http://home.tiscali.nl/developerscorner/>

