

The personal disquiet of

Mark Boulton

July 4th, 2005

Five simple steps to designing grid systems — Part 1

The first part of this Five Simple Steps series is taking some of the points discussed in the preface and putting it to practice.

Ratios are at the core of any well designed grid system. Sometimes those ratios are rational, such as 1:2 or 2:3, others are irrational such as the 1:1.414 (the proportion of A4). This first part is about how to combine those ratios to create simple, balanced grids which in turn will help you create harmonious compositions.

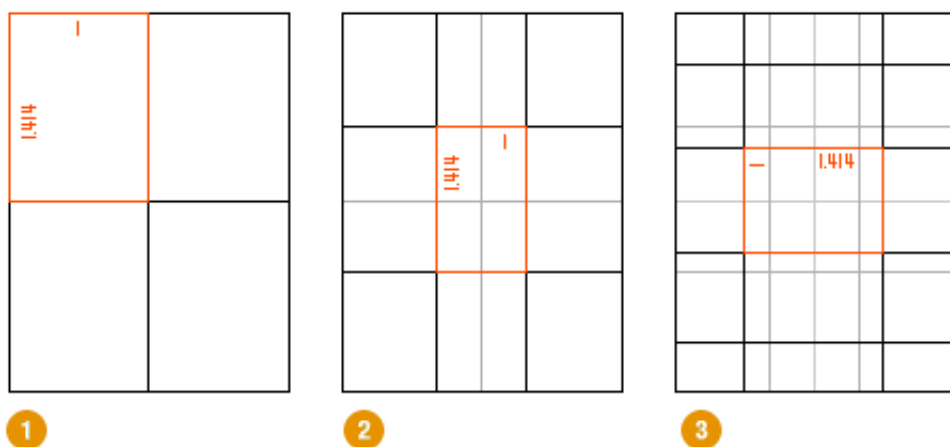
Starting with a blank canvas

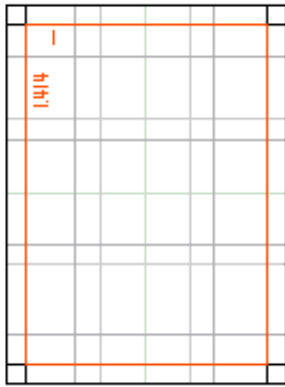
It's always easier in these kinds of tutorials to put the example in context, in some kind of real world scenario. So, this is it. You've got to design a programme for a gallery exhibition. You know you want the size to be A4. You also know that there are going to be photographs and text, and the photographs will be of varying size. There you have it — your blank canvas.

Subdividing ratios

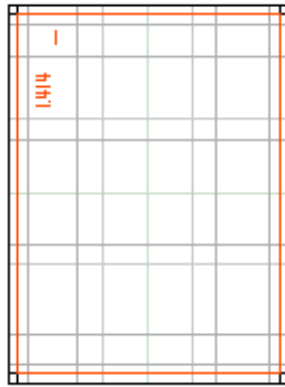
The grid system we are going to design is a simple symmetrical grid based on a continuous division of the paper size in the ratio 1:1.414. Using the paper size as a guide we can retain the proportion throughout the grid, this will give our elements within the design a **relationship to one another, the grid and the paper size**.

This is one of the easiest ways to create a balanced grid. By using the size of the paper as a guide we can divide using that ratio to begin creating the grid. You can see this through diagrams 1 — 6 that we begin by simply layering division upon division to slowly build up the grid.

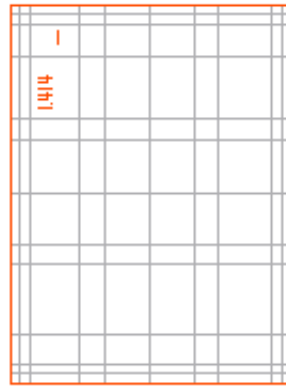




4



5



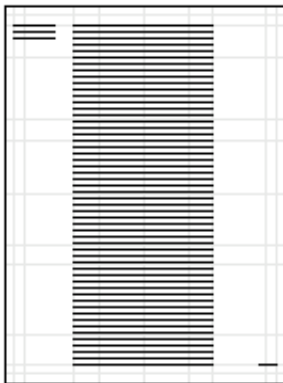
6

Diagrams kindly updated by [Michael Spence](#)

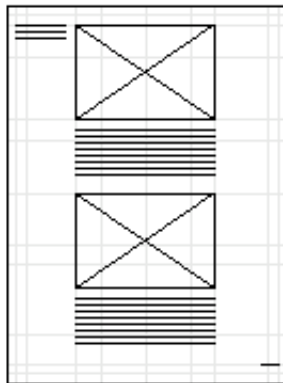
Getting creative

Many have said grid systems can stifle creativity, but I disagree. Grid systems can *facilitate* creativity by providing a framework and already answer some designers questions such as ‘where should the folios go’, ‘how wide should the measure be?’ etc. A well designed grid system will go some way to answer these questions and more.

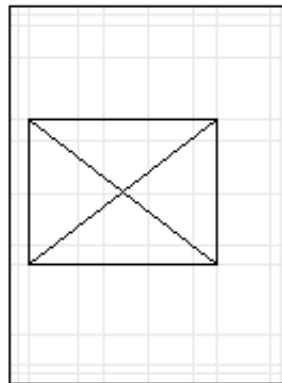
So, we have our grid. We can now begin to experiment with type areas, shapes and composition. We can explore how type and image will work together on the various types of pages our publication will have.



7



8



9

Diagram 7 shows the text area with the first elements of the access structure — running heads and folios. Diagrams 8 and 9 show how adaptable the grid is to various design options.

Short but sweet

A simple step to begin with. Next we’ll go onto to more complex ratios, such as the Golden Section, and combining multiple ratios across spreads instead of single pages.

The personal disquiet of

Mark Boulton

July 16th, 2005

Five simple steps to designing grid systems — Part 2

In [part one](#) of this Simple Steps series I talked about how to use a simple ratio, that of the paper size you are using, to create a symmetrical grid on which to create your designs. This, the second part in the series, will deal with other ratios and how they can be combined to create more complex grid systems.

Relating to grid systems

I've talked a [few times](#) about using the Golden, or 'Divine', Section in the grid systems you design. So, before you continue I suggest you read the background in [this article](#) and my article in [Design in Flight](#). For those who don't want fork out your hard earned cash on the DiF article, I'll summarise:

The Golden Section is a ratio which is evident throughout the universe as the number Phi. You can use this ratio to good effect in design by making sure that elements of your grid conform to this ratio. Using the Golden Section can ensure a natural sense of correct composition, even though it is based in mathematics it will 'feel' right.

This is an important point and has been argued and debated for ages. Aesthetics can be measured and more importantly can be constructed. If you want something to be aesthetically pleasing there are steps you can take to make sure it is going in the right direction. Now I'm not saying that 'follow these rules and you will create something beautiful'. What I am saying is that by following a few of these guidelines can go some way into creating something compositionally balanced, which will inherently be more aesthetically pleasing.

Composition can make things more usable

This is a theory that exists called the 'Aesthetic Usability Effect'. I have [written about this as well in the past](#), I find it a really interesting theory. This theory suggests that things which are designed to be beautiful are inherently more usable as a result. It is an interesting theory and can certainly challenge the usability field, which is often tarnished with the 'ugly brush'.

Composing grids using theory and balanced ratios (such as the Golden Section), which in turn enable the creation of beautiful, balanced designs. These designs then have a quality which will make them more usable, according to the theory. Perhaps I'm labouring the point here, but in short:

Well designed grid systems can make your designs not only more beautiful and legible, but more *usable*.

Putting it into practice

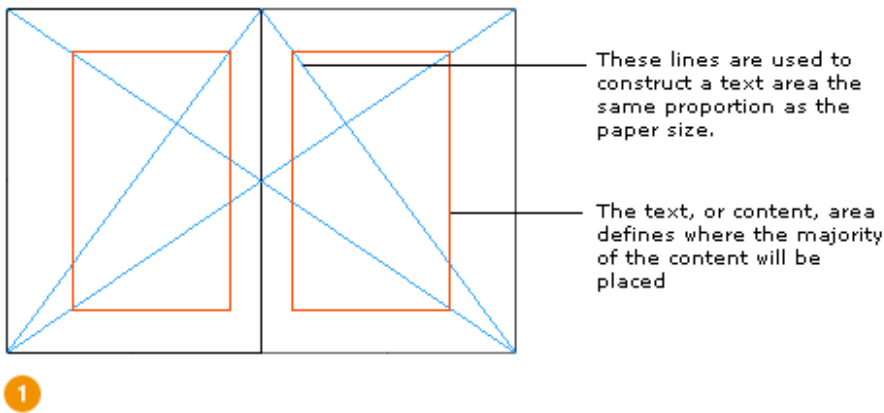
As in the first article I'm going to be designing this grid in context. For those of you who are primarily web based I'm afraid this is going to be another print example, but that doesn't mean this theory can't be applied to web. It can of course be applied to lots of different medium, from architecture and

interior design to product design and art, you just have to apply it to a different 'canvas'.

So, as in the DiF article the brief is to design a book. Unlike the first article in this series, I'm going to be applying the grid to a double page spread rather than a single page, this is called an *asymmetrical grid* as opposed to the symmetrical grid I designed in the last article.

Shaping the page

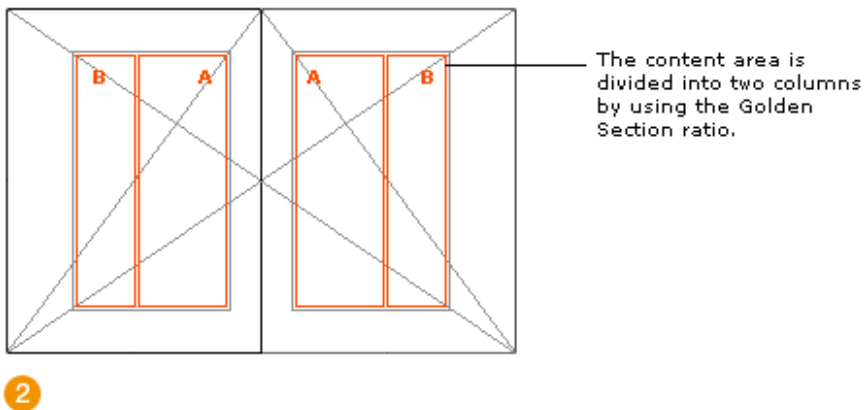
For this grid, we're going to use the ratio of the page to define the main text, or content, area of the pages. There's a very simple way of reducing this page size down to make sure the ratio is correctly placed and balanced. See diagram.



We now have an area, shown in red, in which to construct the grid.

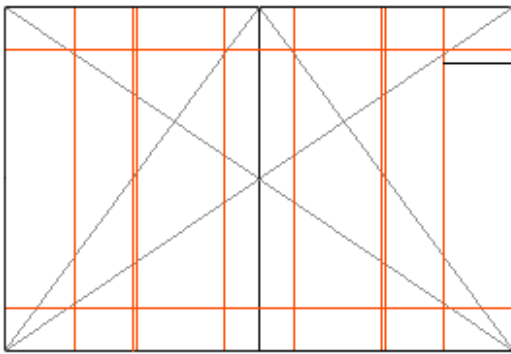
Applying the Golden Section

Now you've read the other articles you will see that applying the ratio to this area is pretty straight forward. The area is divided using Phi which gives us two columns, A and B.



Creating the system

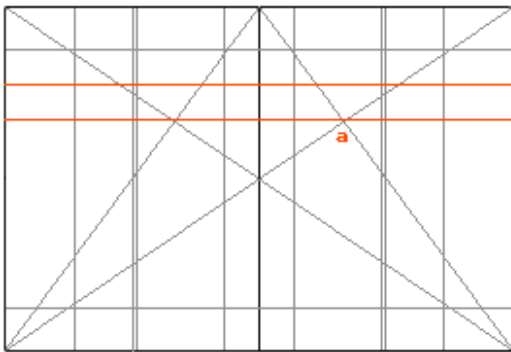
So, we've got the columns, we now need to flesh out the grid to be able to cope with the different content and page types. First off, we extend the lines of the content area and the columns.



Extending the lines out from the content area. This begins to give us more of a complete grid.

3

We then apply a horizontal rule cutting across content area creation lines. I call these 'hanging lines', not too sure what the correct terminology is. But anyway, the content 'hangs' from these lines giving us consistency throughout the book. It gives the reader a line, in the same place, to rest their eyes on page after page.

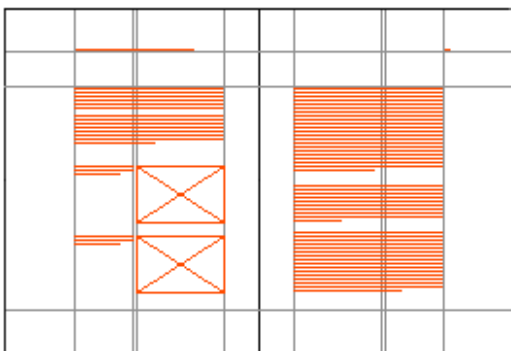


A 'hanging line' is then created by drawing a line across from the bisecting lines (a).

Dividing the space between line (b) and the top of our content area (c) gives us our hanging line.

4

Using the extended lines we can then add areas for the access structure of the book—folios etc. These typically sit outside of the content area, usually with plenty of white space around them, as to show that they are different 'types' of content.



Content is now added to the grid, including running heads and folios.

You can see from the grid construction that there are a lot of possibilities for varying the design across different spreads.

5

We can then add various designs to this grid comfortable in knowing that the individual elements of the design—text, images, access structure elements—will all have a relationship to each other and to the book size.

Relating to grid systems

Creating grid systems in this way—using ratios to create *related* lines on which to construct composition—ensures a balanced grid.

I'm afraid it isn't an exact science though. A lot of grid design is experimentation with ratios, it's experimentation with using white space and elements of content such as photographs and text. It's also about conventions. Don't reinvent the wheel needlessly, study the conventions used in magazines from all sectors—from architecture to nursing (seriously, some magazines from unexpected professional sectors have fantastic grid designs).

What I'm saying is, have a play with grid design. Just because I'm talking about ratios, subdivisions and modularisation, doesn't mean designing grids should be dull. Have a mind on the end product, but not at the expense of the *process* of designing your grid.

The series

This is the second installment of this “Simple Steps...” series.

1. [Subdividing ratios](#)
2. Ratios and complex grid systems
3. [Grid systems for web design: Part 1](#)
4. [Grid systems for web design: Part 2 Fixed](#)
5. [Grid systems for web design: Part 3 Fluid](#)

This entry was posted on Saturday, July 16th, 2005 at 5:45 pm and is filed under [Design](#), [Simple Steps](#), [Typography](#).

You can follow any responses to this entry through the [RSS 2.0](#) feed.

Responses are currently closed, but you can [trackback](#) from your own site.

Related Posts:

- [Five simple steps to designing grid systems - Part 1](#)
- [Five simple steps to designing grid systems - Preface](#)
- [Five simple steps to designing grid systems - Part 5](#)
- [Five simple steps to designing grid systems - Part 4](#)
- [Five simple steps to designing grid systems - Part 3](#)

26 Responses to “Five simple steps to designing grid systems — Part 2”



Ian Lotinsky said on: [July 16th, 2005 at 8:44 pm](#)

In diagram (2), are A and B divinely proportional because the splice line falls in the middle of each page or are the dimensions of the pages such that the split of the page just happens to fall near the middle of the page?

(Thanks for the articles!)



Mark Boulton said on: [July 16th, 2005 at 9:08 pm](#)

The personal disquiet of

Mark Boulton

August 9th, 2005

Five simple steps to designing grid systems — Part 3

The third installment to this series is going to be a little different. The previous installments have been talking through some of the basics of grid construction using ratios as the primary device. They've also dealt with grid construction for print media. Unfortunately, as designers for web media, we don't have some of the luxuries as our print designer colleagues.

Rather than go through tutorials (I'll be covering these in the last two installments), I'll be using this installment as a platform to discuss some of the challenges and rewards of designing grid systems for the web.

A whole load of considerations

Designing grid systems for print is considerably more straight forward than designing grid systems for the web. First off, in print, the designer has a *fixed* media size — the paper size (or packaging, poster, whatever). Let's say a print designer has designed a magazine. The reader of this magazine can't suddenly increase the font size if they find it difficult to read — well they just move it closer to their eyes I guess. This is just one consideration, there are more but I'm sure you get the point.

So, that's media size. On the web you have other considerations such as the browser, the OS platform, the screen size and the actual devices that web sites can be viewed on, from PDA's and Mobile's to assistive technologies such as screen readers. How do you design a grid for all of this? It's a really good question and I'm not claiming I have the answer.

In an ideal world

We all know about the problems with websites rendering differently across different browsers, platforms, devices etc. But, just for a moment, let's forget about that.

Designing a grid for the web should not be difficult, in fact, it shouldn't be any different from designing a grid for an media. As discussed in the previous parts of the series, you can construct a grid in the same manner for screen as you do for print. Base it on ratios, experiment with form and white space etc. Use pixels as your base measurement and go from there in the knowledge that your design will look exactly the same in every browser. After all, you, the designer, knows best for your reader right? You *know* they want light grey, 10px verdana with a measure of 600 pixels.

In the real world

Good designers for the web *know* that the users who use their sites may want different, and know, with the web, they have the power to change things. The designer has lost, to a degree, the ability to *control*. For a lot of designers (including me), this has been a tough transition. We're taught for

years to create the delicate balances of white space, the manicured typography and delicate colour palettes, all designed to create harmonious designs which do their job very well.

Then some short sighted user comes along and increases the text size... and... and... totally breaks *your* design.

I think you get the idea. We can't be upset when the user wants to change something like the text size. What we can do is design grid systems to *adapt* to those changes.

Not just columns

Over the past couple of years, coupled with the increase in CSS based sites, we've seen a rise in certain grid configurations which are all based on the amount of columns. 2, 3, 4 column layouts — float this, position that etc. Why, even this site falls into the '750 px, 3 column' category. These grids have quickly become a convention, and for good reason too. They are quick to create, fairly stable across many platforms and don't degrade to the same degree as table based layouts. This is all good. What isn't so good though is the general lack of understanding of grid systems when perhaps the question on most designers lips when they sit down to begin a design is, 'how many columns should I have'. This is not grid system design.

Grid systems for the web

The next two installments of this series will go through details of creating considered grid systems for the web and the implementation using CSS. I thought it would be useful to just go through some of the considerations before hand.

The series

This is the third installment of this "Simple Steps..." series.

1. [Subdividing ratios](#)
2. [Ratios and complex grid systems](#)
3. [Grid systems for web design: Part 1](#)
4. [Grid systems for web design: Part 2 Fixed](#)
5. [Grid systems for web design: Part 3 Fluid](#)

This entry was posted on Tuesday, August 9th, 2005 at 9:50 am and is filed under [Articles](#), [Design](#), [Simple Steps](#), [Typography](#).

You can follow any responses to this entry through the [RSS 2.0](#) feed.

Responses are currently closed, but you can [trackback](#) from your own site.

Related Posts:

- [Five simple steps to designing grid systems - Part 1](#)
- [Five simple steps to designing grid systems - Part 5](#)
- [Five simple steps to designing grid systems - Part 4](#)
- [Five simple steps to designing grid systems - Preface](#)
- [Five simple steps to designing grid systems - Part 2](#)

The personal disquiet of

Mark Boulton

August 30th, 2005

Five simple steps to designing grid systems — Part 4

Layout seems to be a [hot topic](#) at the moment, mostly prompted by the [ALA redesign](#) and the [numerous discussions](#) of the choice by [Jason](#) and the ALA team to go 1024 for a fixed width. I'm not going to go into my thoughts on ALA in too much depth here, there's [been a lot of that already](#), but it seems like the right time to get this article out.

So, fixed width grid design for the web. What is it, how do we do it and how do we implement it?

For the purposes of this article, I'm going to be focussing on the theory of creating the grid rather than the implementation. I did mention in the last series that I would cover implementation using CSS, well I'm not going to. There are just so many resources and books available telling you how to create the CSS layouts you need—I'll touch on it, but I won't be going into too much detail.

The Measurements

A fixed grid for designing for the web is as close a translation from traditional grid design as there is. The designer is using fixed measurements, pixels mostly, to construct the grid and to position elements within the grid structure and a canvas which is based on a fixed size. See, everything is fixed.

The Canvas

Now things start to get a little less concrete. The canvas size for print design is determined by the media size — paper, signage, envelope, whatever. The canvas size for fixed grid design on the web is normally determined by the browser window size, which is in turn determined by the user's screen resolution. These are not fixed. Therefore a designer should design to the minimum requirement, which is normally the average screen resolution for the majority of users.

I'm not going to quote figures here, because I'll probably be wrong, but I don't think I'm wrong in saying that 800x600 pixel screen resolution has, for quite a few years now, been the screen resolution to design to.

As I mentioned, with the relaunch of ALA, and sites like [Stylegala](#), there has been a renewed discussion about fixed width grids for 1024. So, what's my opinion on this? Well, in terms of the actual grid design it really doesn't matter what size the canvas is. What should be determining the decision to go 1024 is research into user's screen resolutions. If the user base of a certain site is shown to be using resolutions of that size and above, then a decision to use that size to design to is a valid one.

However, as some people have noted, even if you do run at a higher resolution than 800x600 does

that mean your browser window occupies the entire screen? The answer to this is, generally we don't know. I personally think that not only is it platform specific, but it's also down to the individual and their experience level. Maybe more experienced users on a PC don't use their browser's at full screen. From my experience running user tests with a wide range of people, is that more novice users, particularly on a PC, run a browser at full screen *because that is the default*, whereas on a Mac the default isn't full screen.

What I haven't touched on yet is the device you are using. This of course could be a PDA, a mobile phone or a computer. Grid designs should be looked at for each of these devices.

That is all I'm going to say on the matter for now though. Once the final part in this series is up, fluid grid design, there can be a more informed discussion I think.

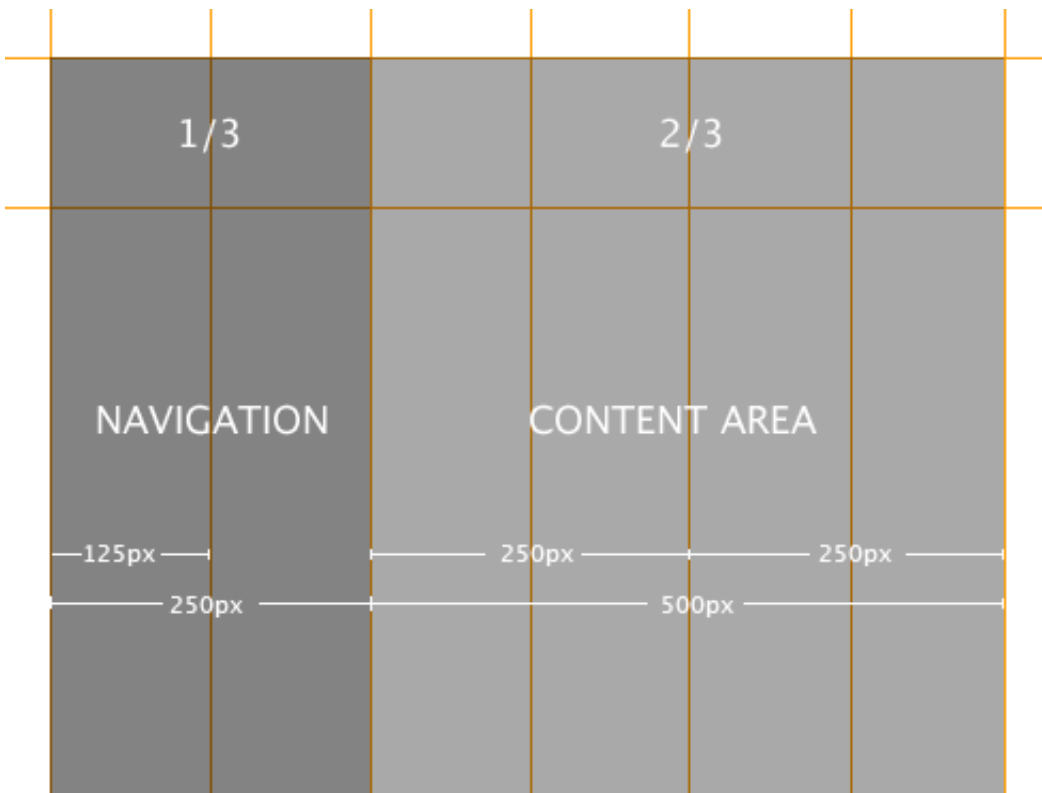
Nice, easy dimensions and thinking modular

Without further a do, let's get into this grid design.

As discussed in the rest of this series (part [1](#), [2](#) and [3](#)), we will begin our grid design by 'shaping the page'.

For the purposes of this simple (I am trying to keep it simple) article, I'll be using 800 x 600 as my default resolution to design to. For many years I've designed to a base minimum (based on 800 x 600) of 760px x 410px (410px being the fold). Don't ask me where I got these figures from, but it just stuck and seems to be ok for most platforms and browsers. Oh, you can of course go smaller than this and don't pay too much attention to the fold, in my experience most users don't have a problem with scrolling.

We begin by applying ratios to this canvas, in the same way we've done with designing grid system for print. The example I'm using for this tutorial is my own site, which uses a fixed grid and sits happily below 760px wide.



The design for my site is built around a very simple grid system. Once I had my grid, I used Photoshop to comp together the designs positioning any elements exactly on the grid lines. The grid was designed initially for a content and navigation area based on the Rule of Thirds (which is roughly approximated around the Golden Section), the dimensions of which are 250px and 500px respectively. The content area is then subdivided into two 250px columns.

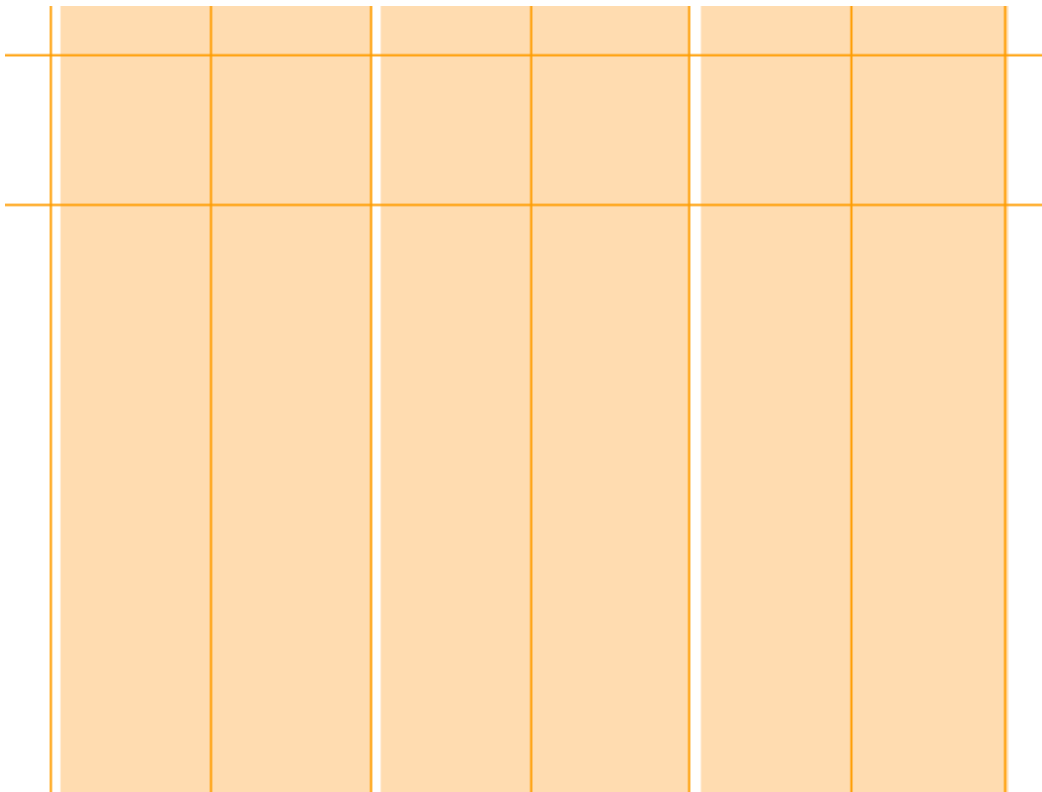
See, nice easy dimensions. However, this only deals with horizontal measurement. As discussed in the other grid articles, vertical measurement is also important, but this is where we can run into problems with designing even fixed grid systems on the web.

When users change the type size, elements move vertically (if we've fixed the horizontal widths). The vertical measurements that we've crafted suddenly disappear. Now, in the purist's eye, this is a real problem but it is something we have to design to. We really can't do anything about it when designing with fixed units such as pixels which can't be resized by the user.

Just a word about Gutters

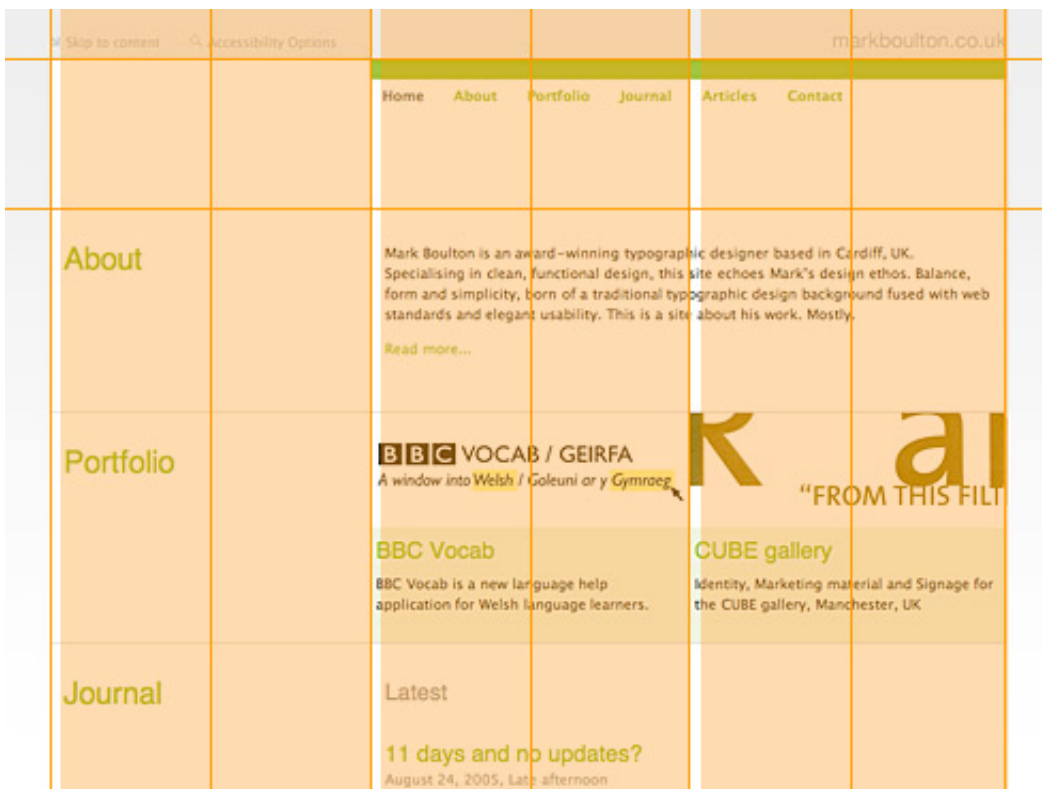
Gutters are the gaps in between columns. They are there so text, or image, from different columns don't run into each other. In grid system design sometimes, depending on what theory you read, gutters are separate to the columns. This creates practical problems for us when designing grid systems on the web because of the way we create the columns.

Generally the columns we create, using Web Standards, are 'divs' which are given widths and positioned and styled using CSS. So, ideally, if we're creating these columns, we don't want to be creating separate ones for the gutters. We therefore deal with gutters as *part* of columns and they are implemented using padding, or creating margins, on elements positioned within them, or sometimes the column divs themselves.



Creating the design

The thing about designing to grids is that in order for the grid to work *you must consistently align* items on the grid lines. I know that sounds totally obvious but designing to strong grids means you have to take a step back from what you think the design should look like (and then adding things to the grid to suit), and instead concentrate on creating a harmonious design *within* the framework you've created.

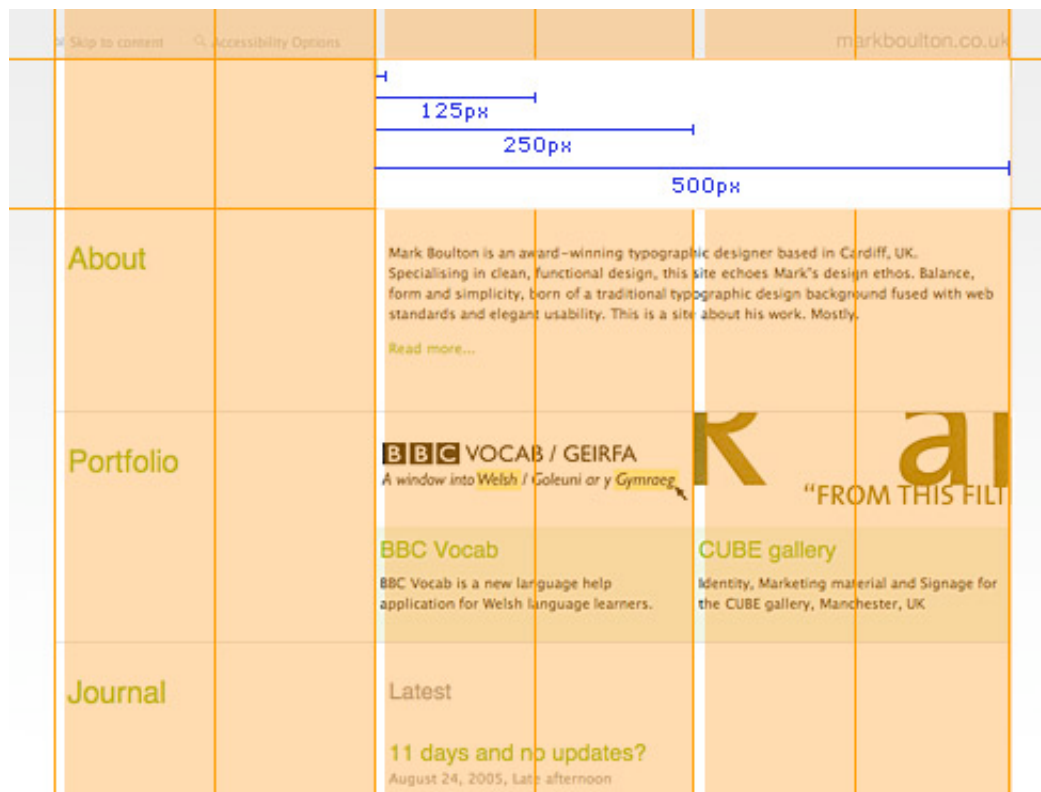


The bulk of the design work, if you exclude sketching with a pencil, is done in Photoshop. First of all I take great care in drawing the grid accurately, to the pixel, and then I overlay content elements ensuring the alignment is precise.

From Photoshop to the browser

As I stated at the beginning of this article, I'm not going to concentrate too much on how you actually build a multi-column CSS layout, there are just so many other great resources on that topic.

One of the most useful 'tools' for creating pixel-perfect grid systems for the web is [Khoi's superb idea](#) of using a grid as a background-image element on the body tag. To summarise: Using the grid I designed in photoshop, I save it out as a gif and then apply that to the background of the body tag. This provides me, throughout the build of the site, the grid so I can align all the content elements accordingly.



As you can see from the diagram, this makes production of the design incredibly easy when you have a visual reference rather than having to remember your grid or interpret a sketch.

Implementation using Web Standards

This really could be a series all on it's own. Implementation of a multi-column grid using CSS is pretty standard practice nowadays, but there are some very useful resources out there which I have used for the past 18 months or so.

Doug Bowman at [Stopdesign](#) has pioneered a technique for producing flexible column layouts using CSS and controlling them by giving a class to the body tag. This is the technique I've used throughout this site. This means if I create a new section of the site or simply decide one day that I'd rather have my navigation on the right, all I have to do is change the class on the body element

and everything switches over. Using this technique, along with Khoi's technique for sense checking the design against the grid, has been an excellent way to produce tight, grid layouts for me, give it a go and let me know what you think.

Up next: Fluid grid systems for the web.

Timely? Yes. Complicated? Yes, but they don't have to be. Fluid, or flexible, grid systems have a rightful place in grid system design for the web but they come with their own particular set of challenges. In the next installment I'll be having a look at flexible grids using relational measurements and also tying the grid design closer to the typography rather than the browser — flexible from a type perspective, rather than a browser perspective. Stay tuned.

The series

This is the fourth installment of this "Simple Steps..." series.

1. [Subdividing ratios](#)
2. [Ratios and complex grid systems](#)
3. [Grid systems for web design: Part 1](#)
4. [Grid systems for web design: Part 2 Fixed](#)
5. [Grid systems for web design: Part 3 Fluid](#)

This entry was posted on Tuesday, August 30th, 2005 at 4:00 pm and is filed under [Design](#), [Typography](#).

You can follow any responses to this entry through the [RSS 2.0](#) feed.

Responses are currently closed, but you can [trackback](#) from your own site.

Related Posts:

- [Five simple steps to designing grid systems - Part 3](#)
- [Five simple steps to designing grid systems - Part 1](#)
- [Five simple steps to designing grid systems - Part 5](#)
- [Five simple steps to designing grid systems - Part 2](#)
- [Five simple steps to designing grid systems - Preface](#)

29 Responses to "Five simple steps to designing grid systems — Part 4"



MIND Justin said on: *August 30th, 2005 at 9:07 pm*

Great article!

Articles like this show that there is much more to webdesign than frontpage and a cracked version of Photoshop



The personal disquiet of

Mark Boulton

September 19th, 2005

Five simple steps to designing grid systems — Part 5

It's been a while, but this is the final part in my series 'Five Simple Steps to designing Grid Systems'.

Flexible vs Fixed. Which one to choose? Why choose one over the other? Well you won't find the answers to those questions here. What I'm aiming to do with this article is to investigate how the theory of grid design can be applied to a flexible web page.

Lets's start by briefly examining Fixed and Flexible, or Fluid designs.

They both have their merits.

Fixed width designs are, well, just easier to produce. The designer has control over the measure, and therefore the legibility (until the user increases or decreases the font size that is).

Flexible width designs scale to the user's resolution, and therefore the browser window. There is less empty space, typically at the side of fixed width designs.

However, they both also have the down sides such as fixed layouts generally scale badly and flexible layouts tend to look very wide and short. But, this article is not about the good and the bad and it's not really the place for that type of discussion, that argument is going on elsewhere and will continue to do so for quite some time.

Flexible grids

As discussed the first few parts of this series, grid system design deals in fixed measurements — the media size, the type size and ultimately the grid size. They are all fixed. So, along comes the Web and challenges theory which has been around for generations. All of a sudden the reader can resize the media, they can change the font size, they can do all this stuff that designers didn't have to think about before. Designers have had to adapt, both to the technology and to the opportunities that media offers.

I've been giving flexible width a lot of thought over the past few weeks in preparation for writing this article. I can see the merits in both, but I've been trying to base my recent thought within the realms of purest grid theory. How does that theory translate to flexible grid design? and I think the answer is, quite well actually.

Adaptive Grid Systems

Ideally grid systems should be designed around the type size. Column widths, and therefore the Measure, should be constructed in such a way to maximise legibility based on the number of char-

acters (you can read more about that in my *Five Simple Steps to Better Typography*). This is all fine if the units of measurements are fixed, but what if they can change? But what does that mean?

1. The user can change the font size
2. The user can resize the browser window
3. The user can change their resolution

The user can of course do this with all design, fixed or flexible, but the key to flexible grid systems is the *grid must adapt* to those changes. After a bit of thought I think the key components of an adaptive grid are:

1. The grid elements adapt to the user's changes, and
2. The grid must retain its original proportions

I've never been comfortable with the descriptions of flexible grids — flexible, elastic etc. What I'm trying to convey with *Adaptive Grid Systems or Adaptive Layouts* is the thought process behind the grid design which reacts to the *user's choices*. I think it appeals to the purist in me! ;)

It's a question of ratios. Again.

In the first two parts of this series, I talked about ratios, both rational and irrational, in the construction of grid system design. But, for those who haven't read them, or have forgotten, here's a quick overview.

Grid systems can be constructed from ratios. Simple ratios such as 1:3, or 2:1 are called *rational* ratios. More complex ratios, such as those based on the Golden Section (1: 1.618) , are called *irrational* ratios.

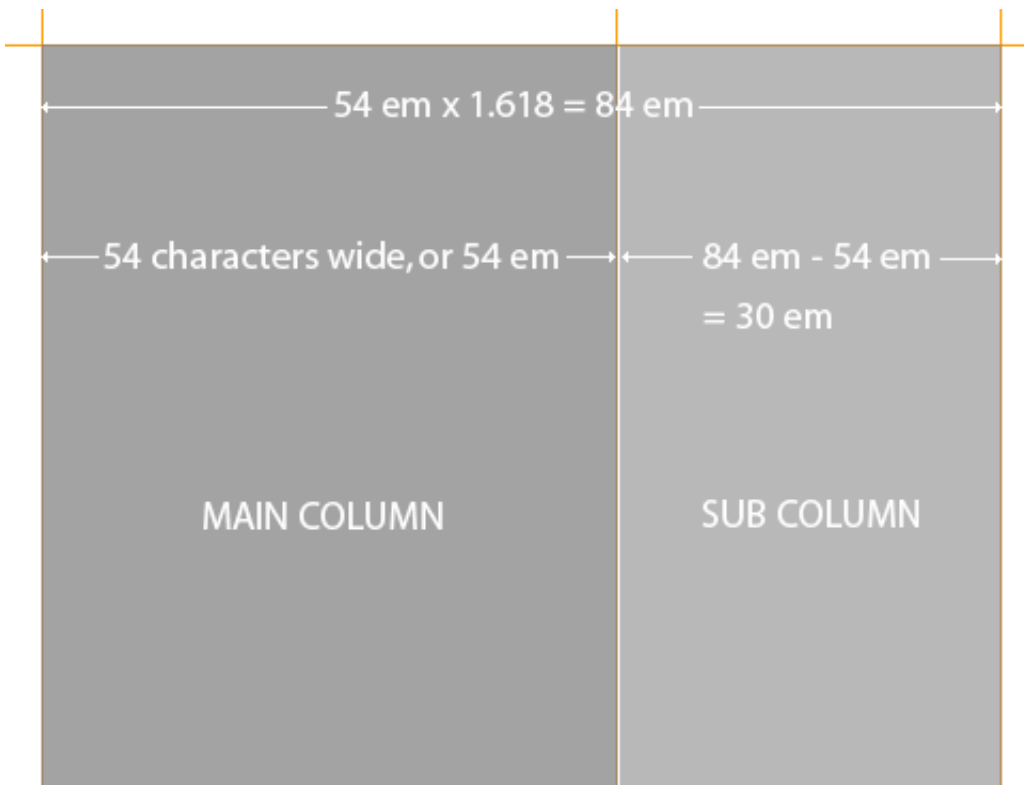
Ratios are just the job for constructing adaptive grid systems because they are independent of any unit of measurement. They are just a ratio to the whole, whatever the whole may be. This whole, be a browser window or whatever, can change and therefore so does the ratio or the grid.

Let me put this into practice now with a working grid.

Divine measurements

If you've been following these articles, you should know by now that I like to start simple. Following on from several articles I've written about the golden section, I'll continue with that and construct this adaptive grid using the Golden Section which is an irrational ratio — 1:1.618.

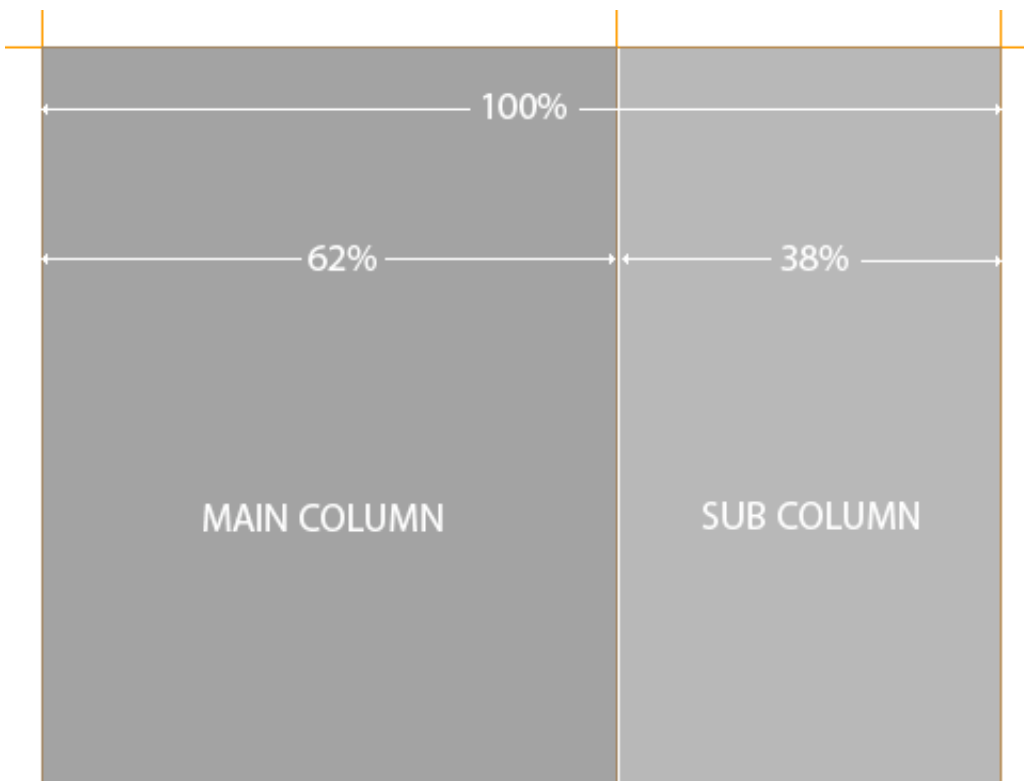
So first off we construct a simple two column grid system with the content areas being defined by the Golden Section ratio.



Getting the right units

In order for a grid to be adaptive, we have to use scalable units of measurement such as 100% or Ems. Just a reminder: An Em (pronounced 'm' NOT 'e, m') is a typographic measurement equal to the point size of the typeface you are using. We also use percentages.

To give us our column width I convert the ratio's to percentages, which gives us 61.8% for the main column and 38.2% for the right column.



That's our grid as determined by percentages. Pretty easy really. Now, on to the implementation.

Constructing the grid in CSS

Before I move headlong into implementing this using CSS, I just wanted to point out that this tutorial is about grid design and not about web standards. XHTML and CSS just happens to be the tools I use to realise this design. You can of course use tables if you want (but you'd be wrong!). Oh, and I'm just not clever enough to stuff this example full of hacks for IE this, Mac that, weird linux browser version 0.6.1 etc. etc. This example is a Best Case Scenario example. If you want to add all that stuff to it, let me know and I'll add it. Like I said, it's the grid that interests me. Disclaimer over.

Oh, and there's a small change to the percentages of the columns to get round some browser bug or another, it's basically a couple of percentage smaller so the floats work correctly.

For those who can't be bothered going through this code, here's the [example](#).

Here's the CSS, including all the global stuff such as links, typographic stuff and general body stuff which is applied to a pretty basic XHTML structure.

```
body {
margin: 0 auto;
padding: 0;
font-family: "Lucida Grande", verdana, arial,
helvetica, sans-serif;
font-size: 62.5%;
color: #333;
background-color: #f0f0f0;
text-align: center;
}

* {
padding: 0;
margin: 0;
}

/* Make sure the table cells show the right font */
td { font-family: "Lucida Grande", verdana, arial,
helvetica, sans-serif; }

/*-----
GLOBALS & GENERAL CASES
-----*/

a {text-decoration: underline; padding: 1px; }
a:link { color: #03c; }
a:visited { color: #03c; }
a:hover { color: #fff; background-color: #30c;
text-decoration: none; }

/*-----
TYPOGRAPHY
-----*/
h1, h2, h3, h4, h5, h6 {
font-family: helvetica, arial, verdana, sans-serif;
font-weight: normal;
```

```

}

/* approx 21px*/
h1 {
font-size: 2.1em;
margin-top: 2em;
}

/* approx 16px*/
h2 {
font-size: 1.6em;
margin-bottom: 1em;
}

/* approx 14px*/
h3 {
font-size: 1.4em;
}

/* approx 12px*/
h4 {
font-size: 1.2em;
}

/* approx 11/14 */
p {
font-size: 1.1em;
line-height: 1.4em;
padding: 0;
margin-bottom: 1em;
}

```

I then add the page structure css.

The columns are wrapped with a container div (called 'container'), this is defined as being 90% wide, with a minimum width of 84 em. What's that about? Well I got that number by doing this.

As mentioned earlier, our ideal minimum width for the measure is 52 em. This is the width of our main column and as determined by the Golden Section, the overall column widths combined is 1.62 multiplied by 52 em, which is 84 em. The right column is therefore 84 em minus 52 em, which is 32 em. Converting these to percentages gives us 62% and 38%, which is what you use in the CSS for each column.

There is also a minimum width of 84 em applied to the overall container, which when the user resizes the text, maintains the ratios.

Here's the CSS for the columns:

```

#container {
width:90%;
margin:0 auto;
text-align: left;
min-width: 84em;
}

#contentframe {
margin: 2em 0 0 0;
padding: 2em 0;
}

```

```
width: 100%;
text-align: left;
float: left;
border: 1px solid #ccc;
background-color: #fff;
}

/* 2 column layout c1-c1-c2 */
.c1-c2 #c2 {
float: right;
width: 36.2%;
padding: 0 0 0 1em;
margin: 0;
}

.c1-c2 #c1 {
float: left;
width: 61.8%;
padding: 0 0 0 1em;
margin: 0;
}
```

So, there we have it. [An Adaptive Grid System based on the Golden Section.](#)

Please feel free to abuse the hell out of this layout. Push it, stretch it, batter it into submission. Please let me know though what your findings are. Are Adaptive Layouts the way forward in flexible grid design for the web. Like I said, I'm interested in the grid, not necessarily in the implementation.

That wraps up another series

Well, there we are. Another series finished with. Hope you liked it, and thanks for all the comments and feedback. I've got a feeling this one's going to be interesting...

This is the fifth, and final, installment of this "Simple Steps..." series.

1. [Subdividing ratios](#)
2. [Ratios and complex grid systems](#)
3. [Grid systems for web design: Part 1](#)
4. [Grid systems for web design: Part 2 Fixed](#)
5. [Grid systems for web design: Part 3 Fluid](#)

This entry was posted on Monday, September 19th, 2005 at 9:36 am and is filed under [Design](#), [Typography](#).

You can follow any responses to this entry through the [RSS 2.0](#) feed.

Responses are currently closed, but you can [trackback](#) from your own site.

Related Posts:

- [Five simple steps to designing grid systems - Part 3](#)
- [Five simple steps to designing grid systems - Part 1](#)
- [Five simple steps to designing grid systems - Part 4](#)
- [Five simple steps to designing grid systems - Part 2](#)