

Design and CSS for Content Management Systems

Author: Giles Cambray

Wednesday, 20th December 2006



Table of Contents

1. Introduction.....	3
2. Creative Considerations	3
2.1.1. Variable height	4
2.1.2. Fonts	6
2.1.3. Navigation	7
2.1.4. Variable width.....	7
3. XHTML and CSS Considerations	8
3.1.1. Browser Compatibility.....	8
3.1.2. Modular Code	10
3.1.3. Staying true to the design	11
4. Conclusions	14
5. References	15



1. Introduction

'Do you design for CSS?' is a battle cry often rallied during the initial meeting between developers, creatives and clients; the parrying response often a reassuring 'yes, of course we do'. Then a few days later, a beautifully composed design is sent across, and the developer gives a disappointed sigh and exclaims, 'you just can't build that in XHTML and CSS'.

This seems to be a common scenario that could so often be avoided with a little more communication and tweaking to both the design and the code. Designing for the web has a whole host of different rules than that for print and they have further ramifications than just using web-safe fonts. Thinking of integrating with a CMS? That means giving the client control over the site too. Weeks of work developing an aesthetic, validated site could potentially be undone in just a couple of hours if this power is wielded improperly, and rest assured that the client will expect you to put it right again every time, and without charge. In our opinion, a reasonable request.

Communication and understanding are everything. In many cases, that funky feature could be implemented in the code, and the boxes probably could line up without using tables. But design the site in the right way and it might not matter if a search bar is not quite 57 pixels from the top, and it should certainly be possible for the client to add a 3 line news story headline instead of a 2 line one without ruining the nicely aligned layout of the site.

This paper sets out to explain just some of the techniques that could be employed on both sides to produce something pleasing for everyone. We doubt it will solve all the problems in a company culture, but if you are lucky it might get creatives and developers to enter into the same tea round.

2. Creative Considerations

A web page is probably the most fluid medium that any graphic designer will work on. Simply put, a web page will never look the same on two different computers (or two browsers) and if the content on that page changes regularly the page will never look the same on two different days.

In this instance a successful design is one that can be really flexible and be pulled and pushed around by the content and still not break. Development time will decrease if



designs allow for slight differences in width, height and alignment between browsers, instead of needing to be pixel perfect.

2.1.1. Variable height

For example, a common problem is websites that have a fixed height. It is very easy using a graphics program to design a website to put in some Greeking (say two paragraphs of *Lorem Ipsum*) and design the page to fit around it. In fact the design becomes so perfect that text almost becomes a graphical element of the page, and almost a thing of beauty in itself. Lay this on top of a complicated non-repeatable background, and you are left with a very limited layout.

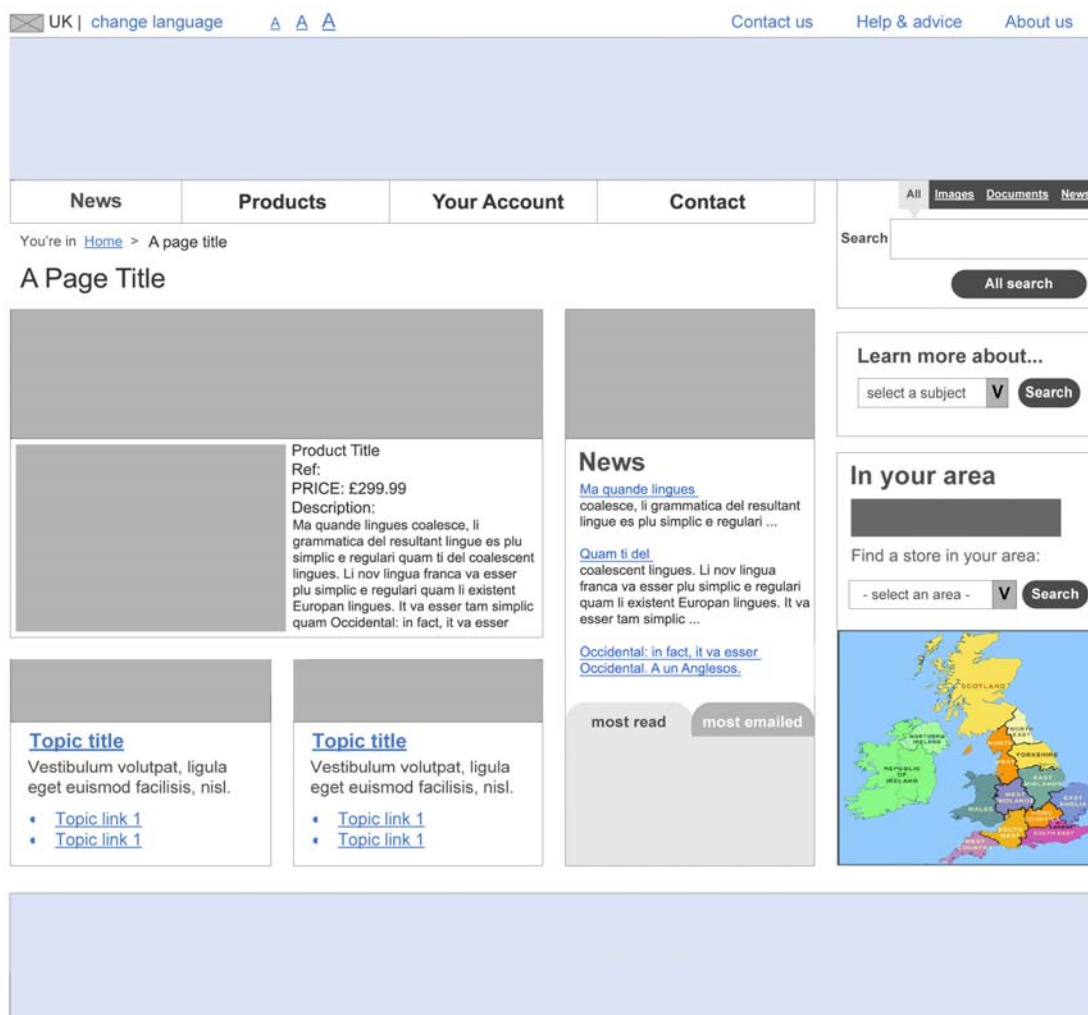


Figure 1: 'Perfect Layout' – Notice how everything lines up very nicely at the bottom, and there is enough room for only 4 menu items on the navigation bar. What would happen if there were different amounts of text on that page in each box?



This is all very well if you have a static site that won't change for months or years. But if you link the design up to a CMS, then this approach just won't cut the mustard. And it's worth noting that default word spacing will be slightly different on each browser, compared to that of the graphics program – so that perfect spacing and kerning will always be slightly different.

A common technique that is incorrectly conceived to be a solution is to make one part of the design scroll individually (whether it's part of a side bar, or the page area containing the main content). There are so many reasons why this is a bad idea.

Firstly you might be hiding a lot of important content from the user that appears below the scroll. Whilst most people are now comfortable will scrolling the whole page, not everyone will know to scroll within an individual area.

More importantly, it is a big no-no from a usability point of you. On top of hiding the information from view you are forcing the user to move the mouse to different areas of the page to scroll just an individual area. And not all browsers support the mouse wheel function for independent areas (as opposed to scrolling the whole page). In fact many mice don't even have a wheel.

And lastly – you can only perform limited amount of styling on the scrolling arrows and scroll bar and only in Internet Explorer. So scrolling often looks pretty ugly.

This is quite a consideration. Whilst it is just about acceptable for a website to be fixed width (more of this to follow) fixed height really isn't a good idea. The website content will change, and so it figures that the amount of space required will change also. For every part of the page that is content managed, it must be able to contain that content. That means the backgrounds should repeat or stretch. And there must be consideration for the fact that other content around it will be shunted around as well.

To confuse things more, it is important to appreciate how screen resolution will affect layout. On a small 800 by 600 pixel screen, taking into account the menu bar and icons at the top and bottom, you will only have around 450 pixels in height on the screen at any one time, without scrolling. Most people are comfortable with the whole page scrolling, but the trick is to give them a clue that there is more below. For example make a section of the site (like the menu bar or main info window) cross the fold on purpose. The user will see that it is cut off and will intuitively know they can scroll down.



% Usage of Monitor resolutions in June 2005 (Ecademy.com)

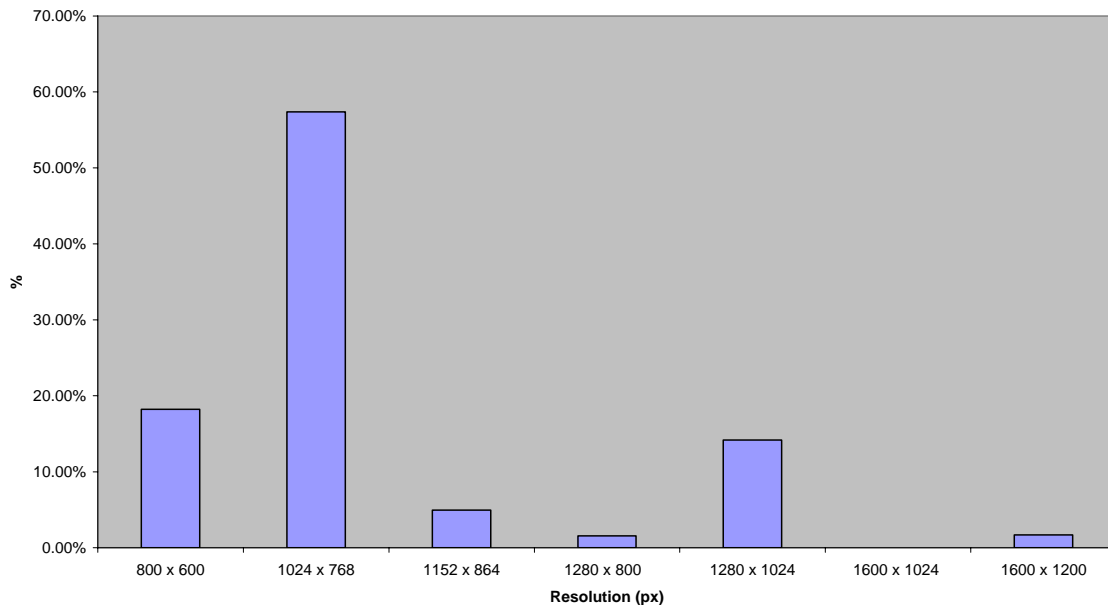


Chart 1: Popular screen resolutions in June 2005 (Ecademy.com)

2.1.2. Fonts

Whilst on the subject of text, it is often useful to realise that fonts do look different on different operating systems. Even with the Microsoft Clear Type turned on in Windows XP, the anti-alias effect is much smoother in OSX. This is not critical concern, but the client can be often quite shocked when text appears rougher on the webpage than it did in the original design. Best play safe – for content Greeking on a design, turn off the anti-alias and use a web-safe font.

Web-safe fonts are still very important and must be used, and if you are designing the site on a Mac, keep in mind that Helvetica fonts are not installed on PCs by default (and are normally replaced with Arial which doesn't look quite the same).

It is common on websites for some titles to be replaced with images that have the correctly styled font instead. This is a tried and tested method and there are several ways to make sure that search engines pick out the right information from that image using image replacement techniques in the code.

In a content management system however, you must bear in mind that titles, as with main content, will change regularly. If a text replacement technique is going to be



used, then a new image will have to be reloaded into the CMS each time, often by the client. Will the client be able to produce the correct looking graphic?

Recently a technique called *sIFR Fonts* has been employed make use of flash to implement non-web safe fonts, which when used sparingly are quite effective. More of this later.

2.1.3. Navigation

Navigation is an extremely important part of site usability. Creative companies are always coming up with novel ways in which users can click through to new pages. We would always recommend that the navigation be generated dynamically by the CMS itself. If this isn't the case, whenever the client adds a page to the site, the name of a page is changed or the page is moved from, let's say, the news page to the archive page, then the navigation has to be changed as well, rather than updating on its own. This defeats the whole point of content management.

When designing the site navigation, think about the following points:

- The navigation is generated by the back end – so that means text, not graphics
- By all means use a background image for a button – but remember that it may have to stretch to contain a webpage link with a longer name.
- Horizontal navigation often looks great, but what if a lot of pages are added at a later date? You have a finite width, but a vertical navigation could theoretically stretch all the way down the page.
- What does the sitemap look like? How deep is the site (and how many sub-menus do you need)? Drop down menus and fly out menus are a good way to display the navigation if you have small amount of space, but if there is a fly out to more than 1 – 2 levels deep you get a dip in usability.
- Content managed sites can quickly become large and complicated. How will the user know where they are on the site and not get lost?

2.1.4. Variable width

We have talked about the importance of height in websites; what about variable width websites? This is a tricky area to get right, and an understanding of the target user audience is crucial (and out of scope of this white paper as it is not directly concerned with a CMS) – but here are some pointers:

- The most common screen resolution is 1024 by 768 pixels, with just about every single monitor being at least 800 by 600 pixels. If you are going to use a fixed width it's best to use 800 pixels (or taking into account the edges of the browser and scroll bar then 760 pixels). If you have a very visibly busy



site you can increase that to 1000 pixels, but don't make it anything in between.

- Variable width is often considered to be the best solution, and will always fit to the window. Design the page to account for parts of the graphics, backgrounds and borders to stretch horizontally or repeat. For example, a drop shadow is difficult to drop onto a gradient if the gradient changes with the window width. Also bear in mind that lines longer than 10-12 words become difficult to read. Large amounts of text spanning the full width of a wide screen can be cumbersome.

3. XHTML and CSS Considerations

Converting a fully fledged validated XHTML webpage from a design is never an easy task. Whilst creating the code, it is easy to put your blinkers on and only think about validating the site at all cost. However, whilst a validating a site is extremely important, it should be weighted equally with other factors as well; SEO, accuracy to the original comp, usability, accessibility and cross browser compatibility should also be treated with equal importance.

3.1.1. Browser Compatibility

Over the past few years browsers have become much better at rendering pure CSS styled pages. Traditionally developers have been encouraged to make sure that their site works on as many browsers as possible. Whilst this is a reasonable objective, the CSS can become very difficult and costly to work with whilst the code becomes hacked to the point that it's detrimental to both the site's performance and the ability to make changes easily and without any negative effect.

Perhaps a more cost effective approach would be to apply 80-20 rule – only use the 98-2 version instead. Depending on your target audience, generally speaking, only about 2-3% of people surfing are using a browser that is older than Internet Explorer 6 (http://www.w3schools.com/browsers/browsers_stats.asp). This is all set to change once again as penetration of Internet Explorer 7 increases and in the meantime more and more people have made the switch to Mozilla Firefox, which generally renders CSS well.

2006	IE7	IE6	IE5	Firefox	Mozilla	N7/8	O7/8/9
November	7.1%	49.9%	2.9%	29.9%	2.5%	0.2%	1.5%
October	3.1%	54.5%	3.2%	28.8%	2.4%	0.3%	1.4%



September	2.5%	55.6%	4.0%	27.3%	2.3%	0.4%	1.6%
August	2.0%	56.2%	4.1%	27.1%	2.3%	0.3%	1.6%
July	1.9%	56.3%	4.2%	25.5%	2.3%	0.4%	1.4%
June	1.6%	58.2%	4.3%	24.9%	2.2%	0.3%	1.4%
May	1.1%	57.4%	4.5%	25.7%	2.3%	0.3%	1.5%
April	0.7%	58.0%	5.0%	25.2%	2.5%	0.4%	1.5%
March	0.6%	58.8%	5.3%	24.5%	2.4%	0.5%	1.5%
February	0.5%	59.5%	5.7%	25.1%	2.9%	0.4%	1.5%
January	0.2%	60.3%	5.5%	25.0%	3.1%	0.5%	1.6%

Table 1: Internet Browser usage trends in 2006.

Do be aware of older versions of Firefox, however. There are some surprising rendering differences between versions 1.0 and 2.0 that can catch you out if you don't check.

You should think about your target audience and think about what type of browsers they would use and start from there. On the whole we would recommend that you aim for Internet 6, Firefox 1.0, Safari, and Opera 8, and above.

With this in mind, there are a number of techniques that can be used to make sure that the development process is as smooth as possible.

Firstly, when choosing a doctype, use XHTML 1.1. With modern browsers there is little excuse for using anything else, and the site will be better prepared to stand the test of time. It also makes it easier to spot errors with code; a validation of a site with this doctype will return more errors, helping you debug incorrectly nested tags which will throw your layout into disarray.

It is also important to realise that all the different browsers will render content slightly differently if no stylesheet is applied. There is big gap between paragraph tags, but is it padding or a margin creating the gap, and is it placed above or below the tag, or both? Invariably with these tags, they are rendered differently in each browser, until the css file overrides it.

With this in mind a good place to start is to make sure that these basic tags are rendering in a similar manner on all browsers before you start applying the design itself.



If you need to, use browser specific css hacks (as detailed on http://www.webdevout.net/articles/css_hacks.php) to tweak your code for certain browsers. They are extremely useful and can get you out of some tricky situations. In some situations it is preferable to use these methods rather than finding a tenuous method of making the same CSS correct for all browsers.

Finally it is important to check what the site looks like without a style sheet. Create the site in a schematic way, giving parts of the page the correct tags schematically (so there is always a <h1> tag and <h2> tags and for listed data use the or <dl> or tags correctly). Not only will this aid those reading the site on a screen reader, older browser or PDA, it will also improve your search engine rankings.

3.1.2. Modular Code

A huge advantage of CMS systems is that you can easily move and change areas of the site. Some pages might have a news bar or "quicklinks" section. But they might appear in a different order, or in the absence of, for example, a login area on a particular page, they might be moved up 100 pixels.

This means that the XHTML of each of these sections must be self contained and must always validate, irrespective of where it is placed on the page. The CSS for the section should also work regardless of where it is situated on the page. For example, on the homepage, a news section might appear on the left hand side of the web page but on lower pages it might appear on the right. Defining the width of the CSS as a percentage of the parent div (ie the left or right div) allows the news section to always fit into where ever it is supposed to be. Defining the font size in 'ems' rather than pixels will allow you to make use of relative font sizes as well.

If the layout or styling needs to be radically different from one area to the next, defining the styles from a high level in the css gives you greater flexibility whilst keeping the XHTML consistent. For example, you might want to have the strapline of a news story a different colour on a different page:

The strapline style on the left might be:

```
#left .news p.strapline {  
    color: blue;  
}
```



And on the right it might be

```
#right .news p.strapline {  
    color: red;  
}
```

Whilst on both pages the XHTML code would be:

```
<p class="strapline">Lorem Ipsum</p>
```

But because this line of code lies in different divs on each page, the CSS knows that it should style it differently.

The method of defining styles is often good practice and prevents strange layout bugs occurring due to inheritance from other styles.

3.1.3. Staying true to the design

In the website development process, there is always a little bit of give and take at the point when the design turns into the code. Whilst a lot can be done to original design to make the coding process easier, there is no doubt that inevitably some advanced CSS will have to be employed. Whilst this is not a white paper on advanced CSS, there are a couple of tricks that could be used to make your life easier.

3.1.3.1. Transparent PNGs

These are a great invention, but annoyingly they are not natively supported by Internet Explorer 6 and below. There is a hefty hack that can be used to fix this which is well documented on the Web and is included below. Even though it does have its limitations, it can be used to good effect if used sparingly - for example, if you want to have a background image controlled by the client, but you want a translucent image on top of this (or a box with a drop shadow on it, for example).

- i. create the normal style for Firefox et al eg:

```
#layer {  
    background: url('images/your_transparent_pic.png');  
}
```



ii. and then underneath create this for IE 6 and below:

```
* html #layer {
background: none;
filter:progid:DXImageTransform.Microsoft.AlphaImageLoader(src='
images/your_transparent_pic.png',sizingMethod='scale');
}
```

There are 3 sizing methods to choose from:

- i. scale: This will fit to the size of the div, h1 etc
- ii. crop: This will keep the image the original size, but will allow the div to be set bigger or smaller than the image
- iii. image: This will force the div to be the same size as the image, regardless to your setting for width and height.

Beware however for clickable elements (i.e. form elements and a tags) in divs with these backgrounds. They will not be clickable in Internet Explorer. Often, however, the problem can be fixed by giving all specific clickable tags in that div relative positioning and a high z-index, thus:

```
#layer a {
    position: relative;
    z-index: 10;
}
```

For more details on this technique take a look at <http://www.alistapart.com/articles/pngopacity/>

3.1.3.2. display:table

If you have an image floated left, with a paragraph to the right of it, default CSS properties will allow the text to flow around the picture. Situations often occur where you do not want this behaviour:

```
<div>
  <img class='floatleft' src='image.jpg' alt='image' />
```



```
<div class='table'>
    <p>Some text</p>
</div>
</div>
```

By using the `display:table` on the div with the class 'table' the left hand side of that div will continue down in a straight line and not flow around the image.

This CSS property is reasonably well supported except in Internet Explorer – in order for it to work, the div needs to have a height defined. Since Internet Explorer treats a height like a minimum height (when the content gets bigger the div will just expand in height to accommodate the text) this doesn't cause too much of a problem (just define the `height:1%;` in the 'table' style). However this will cause a strange bug in Firefox where the page has to be refreshed for the float to work. Thus the definition of the height must be hidden from Firefox (using the * html CSS hack). If there is need for a minimum height then the `min-height` property can be used (IE does not recognise this).

3.1.3.3. Clearing floats

Quite often a taller floated image will stretch lower than the text to the side of it. On wider screens the chances of this happening increase with a variable width web page (the text has more width to flow). If another floated image with text were to appear beneath, it would start to the side of the original float, rather underneath it – thus would appear to be floated within the float.

This undesired effect can be removed by using the `clear` property on a `
` tag. `clear:left`, `clear:right` and `clear:both` will end the nearest float to the left, or right, or both respectively, ensuring that whatever the amount of content, width of window or size or original float, the succeeding float will always start from the beginning of that parent div.

3.1.3.4. Absolute positioning

Generally speaking the wide use of absolute positioning will not really work for a content managed site. Whilst initially it gives you a lot of control over layout, it is not flexible in terms of changing content.

One way in which it can be used to good effect however is by cropping content managed images so they always appear in the same aspect. Consider the graphic designer wants an image gallery, with image thumbnails all the same width and height.



In reality it is unlikely that all the images uploaded to the CMS will be the correct width and height (when considering portrait and landscape aspects). By placing the thumbnail images into an unordered list with a style like this:

```
.gallery li {  
    list-style: none;  
    float: left;  
    width: 120px;  
    height: 80px;  
    position: relative;  
    overflow: hidden;  
}
```

And styling the image thus:

```
.gallery li img {  
    position: absolute;  
    left: 0;  
    top: 0;  
}
```

Will make the image cropped to 120 by 80 pixels. It is important to have position: relative; on the list tag so that all absolutely positioned elements inside it are relative to its boundaries. If that style was missing, the thumbnails would all be positioned in the top left corner of the page.

This technique allows CSS developers to make schematic image thumbnail blocks without the use of tables.

4. Conclusions

Hopefully this document has highlighted how a Content Managed CSS-based website requires thought right from the beginning. How the site will work, render and be used must be considered as soon as the graphic designer fires up Photoshop for the first time. He or she must realise how the site will change and how they want the layout to adapt to these changes. Most importantly, consensus must be reached at an early stage between the creative team, the development team and the client.



5. References

Monitor Sizes

Internet Stats Compendium December 2006 – Ecademy (www.ecademy.com)

Internet Browser usage trends in 2006

http://www.w3schools.com/browsers/browsers_stats.asp

Browser Specific Hacks

http://www.webdevout.net/articles/css_hacks.php

Transparent .png

<http://www.alistapart.com/articles/pngopacity/>



About Solid State Group

Solid State Group is a content management, web applications and services consultancy, who focus on making your online presence dynamic and easy to manage, at a reasonable price. Our primary goal is to complete innovative and robust websites for our clients whilst maintaining a service second to none.

WebDeck content management system

Solid State Group's products allow you to completely control your company's brand on the internet. WebDeck is a complete accessible content management system but it's easier to think of it as the remote control for your website.

It is accessible - WebDeck produces WACG level A, Double-A, and Triple-A websites.

It is flexible - WebDeck can work with any kind of website design.

It is multi-user - WebDeck enables teams to work securely and seamlessly on a site.

It is secure - WebDeck has been tested by Deloitte consulting for FSA accreditation.

It is sticky - WebDeck comes with interactive tools such as forums, polls, quizzes, etc.

It has management tools - WebDeck has real-time integrated web statistics.

It has marketing tools - OutReach is an email marketing plug-in for WebDeck.

It uses open standards - WebDeck uses open standards like XML, RSS, CSV, SOAP.

It is platform independent - WebDeck runs on Java with a web front end.

Bespoke build services

Solid State Group also offer bespoke systems design and development. Some websites need a little extra functionality that simply won't be available out of the box from any product. We specialise in capturing the exact requirements and translating them into a working site with stylish design.

Our special offer

We offer a free consultancy meeting to capture requirements and give a no strings attached estimate for systems design and construction. If nothing else, you will at least gain a well documented specification of your requirements, so if you are interested in our services, please do get in touch.

Contact us

Call us on +44 (0)845 838 2163

Email us at info@solidstategroup.com

Visit our website at <http://www.solidstategroup.com>

