**CHECK MAGAZINE TUTORIAL: PART 3**

**Styling the Header & Navigation Menu**

**In Part 3, you will learn…**

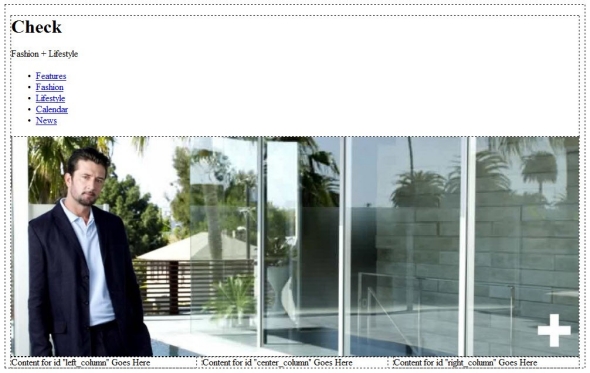
Welcome to Part 3 of this six-part tutorial series on creating your first website. This tutorial shows you how to style the main heading using Google Fonts, a library of hundreds of free fonts that you can use in your website. You’ll also learn how to eliminate the gap that often appears at the top of web pages, and how to convert an unordered list of links into a horizontal menu bar. The menu bar and main text will be maneuvered into position using a combination of relative and absolute positioning.

------------------------------------------------------------------------------------------------------------------------

**Locate your files and review your task**

In Part 1 and Part 2 of this tutorial series, you defined a site for Check Magazine, a fictional publication, and created the basic structure for the home page. If you completed the first two parts, continue working with your existing files. If you’re not sure whether your HTML and CSS is correct, you can download the check\_cs6\_pt2completed file from the Shared folder, and use its contents to replace your existing files.

At the end of Part 2, the page was still very bare (see Figure 1).



*Design view shows that apart from the main image, the page is still completely unstyled.*

In this tutorial, you'll style the header section, and add the main text area. The image below shows what the page will look like in a browser when you’re done with Part 3.



*The page is beginning to look like the original design concept.*

------------------------------------------------------------------------------------------------------------------------

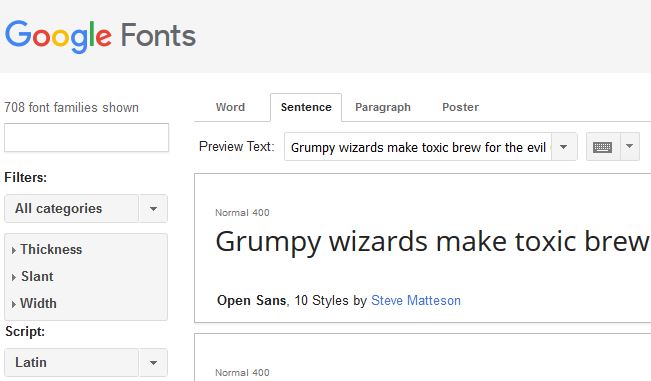
**Style the main heading and tag line**

You’ll start by using a web font for the main heading.

***Styling the heading with a Google Font***

The main heading uses a font called Sarina, which is not available on most computers, but you can add it to your web page using Adobe Edge Web Fonts, a vast library of free web fonts from Adobe, Google, and designers around the world.

1. Open your browser and go <https://www.google.com/fonts> (see Figure 3).



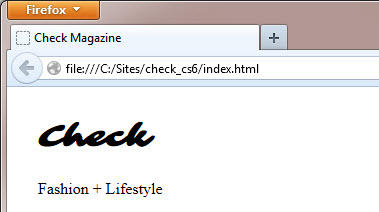
*Google Fonts give you free access to a wide range of attractive fonts*

1. In the search bar, search for the **Sarina** font. Click the “+” button to add the font to your “cart.” Open the black bar (aka your “cart”) and “check out” (aka copy the necessary links to add this font to your website). For more information on how to add Google fonts, look back in your CSS lessons to “Lesson4\_Formatting-Text.” You will have to add the style <link> to the head of your index.html page, and the style itself to the “h1” selector of your check\_cs6.css page.



*The Google Fonts site gives simple instructions for embedding the font*

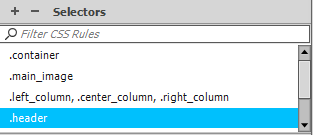
1. Save both index.html and the style sheet.
2. Unfortunately, the Dreamweaver Design view doesn’t support Google Fonts. So, either turn on the Live view option (if you have internet access), or **press F12 to launch index.html in your default browser. As** long as you’re connected to the Internet, the heading should now be styled using the web font (see Figure 5).



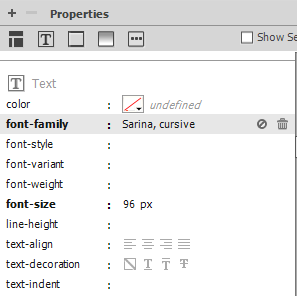
*The web font completely changes the look of the heading*

1. The heading is too small, so amend the h1 style rule by setting font-size to **96px** inside your CSS code.
2. The background color of the header needs to be black, and the color of the text should be white. In your CSS Designer, under the “Selectors” section, click on the “+” sign to add a new selector. Give the selector the name **“.header**” to correspond with the “header” class you gave one of the divs inside your html code. Click on the **“.header”** selector you just added to select it, then move onto Step 7.

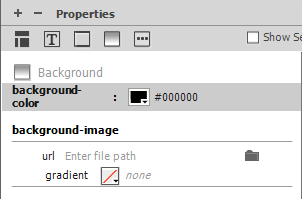
**\*NOTE:** Notice how clicking on a selector from the “Selectors” section of the CSS Designer causes that html element to be highlighted in your Design view! This is a great way to make sure that you have chosen the correct selector before you make changes to its style.



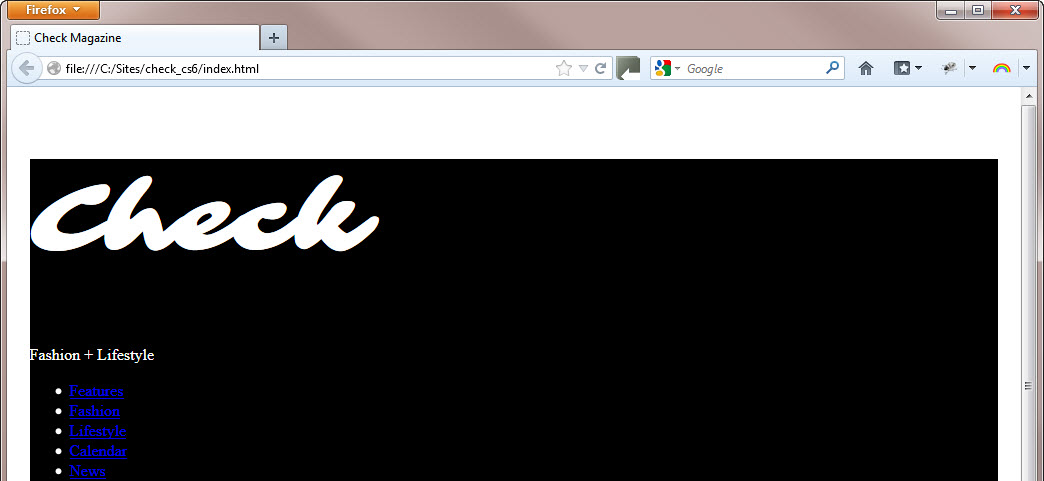
1. To set the text color: Under the “Properties” section of the CSS Designer, click on the “T” icon. This will bring up all the text-related CSS options available to you. Click on the box beside the “color” and change it to white in the color picker, or #ffffff in the hex bar.



1. To set the background color: Under the “Properties” section of the CSS Designer, click on the icon that looks like a gradient-box. This will bring up all the background-related CSS options available to you. Click on the box beside the “background-color” and change it to black in the color picker, or #000000 in the hex bar.



1. Save the style sheet, and refresh index.html in your browser (or view it in Live mode). The header should now have white text on a black background (see the image, below).



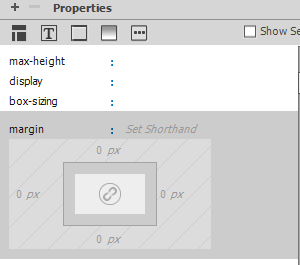
*The colors have changed, but there’s a big gap at the top of the page.*

Although there is nothing above the header, it has been pushed down the page, leaving an ugly white gap at the top. Also, the links in the unordered list haven’t changed to white like the rest of the text. Both issues need to be fixed. First, you’ll deal with the gap above the heading.

***Fixing the gap above the header***

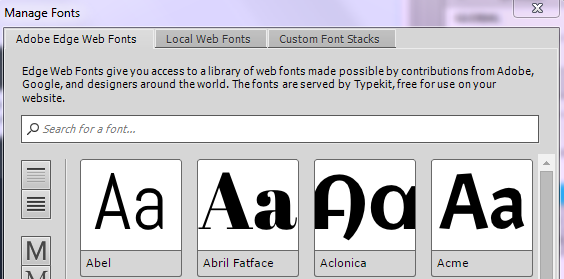
In this section, you’ll investigate and fix the gap at the top of the page. An unusual feature of CSS is that adjacent vertical margins merge to the height of the largest one. As a result, the heading’s top margin extends beyond the header <div>, pushing it away from the top of the page. Notice also that there’s a narrow white strip above the top margin. That’s caused by browsers adding a default eight-pixel margin around the <body> element. You need to fix both issues.

1. In the CSS Designer, under the “Selectors” section, select the **“h1” selector**, then the “Layout” icon (3 rectangles – see image, below).
2. Set the **“margin” shorthand to 0px.** This should clear the margins on all sides.
   * I know that, judging by the diagram below the margin property, it looks like they are already set to “0px” – but they have a default spacing value… it’s confusing. Just take my word for it – margins are never truly nonexistent, until you manually set them to 0

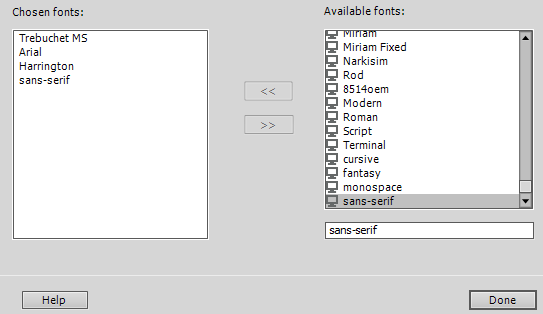


*Setting the margin property to 0px*

1. You now need to create a rule for the <body> element to remove the default margin around the page. Most rules set on the <body> element are inherited by the rest of the page. So, in addition to removing the default margin, it’s a good idea to set the default font, text color, and background color. Use what you’ve learned about the CSS Designer to add a **“body” selector**, then make the following changes to it:
   * Set the margin to 0px all around (use “set shorthand”)
   * Set the (font) color to “#3B3B3B” (readability experts recommend against using pure black text on white because the contrast is too strong. So, use a dark gray for the text)
   * Set the background-color to white
   * Set the font-family to “Trebuchet MS, Arial, Harrington, sans-serif”
     1. After you’ve clicked on the value bar for the “font-family” property, click “manage fonts…” in the dialog box that pops up.
     2. Click on the “Custom Font Stacks” tab at the top of the next dialog box that pops up. Then type each font you want into the font bar and add it to your font stack using the “<<” button!

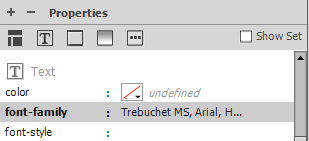


*After choosing, “manage fonts…” click on the “Custom Font Stacks” tab*



*Search for, and add, fonts to your custom font stack*

* + 1. Press the “Done” button once you’ve created your font stack.
    2. Add it to your “body” selector by, once again, clicking on the value bar beside the “font-family” property. Your font stack should appear. Click to use it.



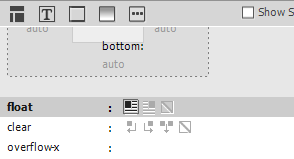
*Set your custom font-stack as your “font-family” value*

1. Notice that the font and background color has changed for all the text except the main heading. Also note that the color of the text in the header <div> is unchanged because you set a different color in the #header style rule earlier.
2. Choose File > Save All Related Files to save the changes to the style sheet.

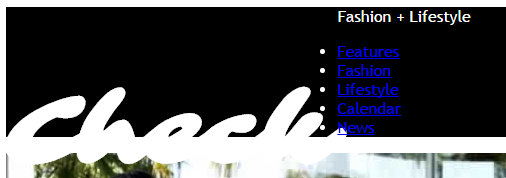
***Styling the tag line***

The Fashion + Lifestyle tag line needs to sit alongside the Check heading. The tag line is in a <p> tag, while the heading is in an <h1> tag. CSS treats both as block-level elements that occupy a new line. To move them alongside each other, you need to apply the float property to the heading. This moves it to the side, allowing other elements to move up alongside. Once the tag line moves into the space next to the heading, you can adjust its padding and margins to move it into position.

1. Use what you’ve learned about the CSS Designer to select the **“h1” selector**, then make the following change to it:
   * Set the float to “left.” *Don’t worry that the header moves away from the top of the page (see second image, below). It’s because the Fashion + Lifestyle tag line has moved up alongside the heading, and its top margin now overlaps the header <div>*

**

*Set the Float to “left”*



*Don’t worry that the header moves away from the top of the page*

1. Use what you’ve learned about the CSS Designer to add a **“.header p” selector**, then make the following changes to it:
   * Set the font-family to “Lucida Sans Unicode, Lucinda Bright, sans-serif”
   * Set the font-size to 30px
   * Set the font-variant to small-caps (cool, hey?)
   * Set the top and bottom margin (use the margin box – hint: you may have to click the chain link in the center of the box to unlink the margins) to 0px. *Note that setting the top margin to 0 removes the gap at the top of the page, but the tag line is too high. Thus, you will need to adjust the Padding, as well.*
   * Set all sides’ padding to 15px
2. At this point, in Live view, your heading should look like the image below.



*The tag line is moved into place using padding and margins.*

1. Click OK to close the CSS Rule Definition dialog box, and save the style sheet.

-------------------------------------------------------------------------------------------------------------------------

**Style the navigation bar**

The final stage of fixing the header <div> involves converting the unordered list of links into a horizontal navigation bar. You’ll start by giving the links a different color and font styles.

***Giving links a unique style***

The links in the navigation bar need to have a style that won’t be shared by other links in the page.

You do this with a selector that targets only **links** in the header <div>.

1. Use what you’ve learned about the CSS Designer to add a **“.header a” (link) selector**, then make the following changes to it:
   * Font-size: 20px
   * Font-weight: bold
   * Font-variant: small-caps
   * Color: #FFF (white)
   * Text-decoration: none
2. Select some of the placeholder text in one of the columns at the foot of the page (again, you will have to turn off Live view mode), and type **#** in the Link field of the Property panel. It’s styled differently from the links in the header because the “.header a” style rule that you have just created affects only links (a) in the header <div> (see image below)



*The links in the header have a unique style of their own, since their selectors are unique*

1. Save index.html and the style sheet.

***Converting the unordered list into a navigation bar***

To convert the links in the unordered list into a horizontal navigation bar, you need to remove the bullets, give the links a width, and turn them into block-level elements so that the full width of the link becomes clickable. Finally, you need to float them to the left.

1. Use what you’ve learned about the CSS Designer to add a **“.header ul” (unordered list) selector**, then make the following changes to it:
   * Margin, all sides: 0px
   * Padding, all sides: 0px
   * List-style-type: none *(note: though you can find this by scrolling down under the “T” icon, it may be easier to choose the “…” icon and type the property and value in manually)*
2. The bullets disappear from the unordered list, and your Live view should look like the image, below:



*The bullets disappear from the unordered list*

1. You need to add some extra properties to the style rule for the links, so use what you’ve learned about the CSS Designer to select the **“.header a” (link) selector**, then make the following changes to it:
   * Text-align (under “text”): center
   * Display (under “layout”): block
   * Width (under “layout”): 100px
2. The links should now be stacked under the main heading (see the image, below):



1. To change the layout of the nav bar so that it is a horizontal, rather than a vertical, stack, add some properties to the style rule for the actual list items within the nav bar. Use what you’ve learned about the CSS Designer to create a new selector called **“.header ul li” (link) selector**, then make the following change to it:
   * Float (under “layout”): left *(When you click OK, don’t panic if the design goes haywire. It should look similar to the image, below). The problem is caused by both the main heading and the navigation bar floating. When items float, they’re no longer constrained by the element they’re nested inside—in this case, the header <div>.)*



*Floating the menu items destroys the layout—at least temporarily*

1. To fix the issue with our layout, select the **“.header” selector**, then make the following change to it:
   * Height (under “layout”): 100px
2. The navigation bar should reappear and the main image move further left (see image, below)



*Adding a height to the header <div> brings the menu back into view*

**Note:** Don’t worry that the main image is still off to the right. We will fix that in the next section!

1. Click OK to close the dialog box, and activate Live view. The main image should be centered under the header.
2. Save all.

-------------------------------------------------------------------------------------------------------------------------

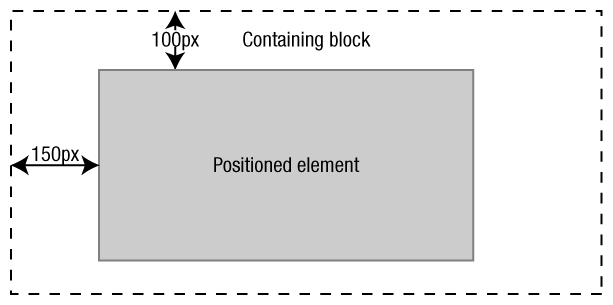
**Use absolute positioning**

The layout design calls for the menu to be at the bottom-right of the header.  To move it into position, you’re going to use absolute positioning. Before showing you how to do it, **I must sound a word of warning.** Many beginners fall into the trap of trying to lay out their entire design with absolute positioning (or AP Elements, as Dreamweaver calls them).

**Note:** Any HTML element can be absolutely positioned. It doesn’t apply only to <div> elements.

The attraction of absolute positioning is that you can place an element precisely on a page, making web design feel almost like desktop publishing. However, the web is a fluid medium. People view pages on different sized screens, and at different resolutions. Absolutely positioned elements behave in a very different way from other elements. They stay where you put them and don’t interact with the rest of the page. This often creates the illusion that the absolutely positioned element moves. In fact, the opposite happens; the rest of the page moves in relation to the absolutely positioned element. Also, an absolutely positioned element might disappear off screen on a mobile device without generating a scrollbar, making vital information inaccessible.

Although absolute positioning presents problems for laying out entire pages, it can be very effective when you want to align an element with another. You specify the position as one or two offsets from the element’s sides. For example, if you set the left offset to 150px and the top offset to 100px, the left and top sides of the element are positioned exactly that distance away from the element’s containing block (see image, below).



*The offsets of an absolutely positioned element are calculated from its containing block.*

**Note:** The offsets can be positive or negative. As the above image shows, a positive 150px left offset moves the element 150px to the right away from the left of the containing block. A negative offset moves the element in the opposite direction outside its containing block, which is simply the point from which the offsets are calculated.

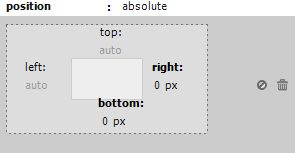
The secret to using absolute positioning successfully is understanding **the principle of the containing block.** Usually, an element’s containing block is the nearest block-level element higher up the HTML structure. In the case of the navigation bar, this is the header <div>. **However, absolute positioning is different.** The containing block is the nearest element higher up the HTML structure that has its position property set to absolute, fixed, or relative. If no such element exists, the page becomes the containing block.

To fix the navigation bar at the bottom-right of the header, the bottom and right offsets need to be set to 0. However, without first defining a containing block, the navigation bar would end up at the bottom-right of the page. To calculate the offsets from the header <div>, you need to set its position property to relative. **Relative positioning is similar to absolute positioning, except the offsets are calculated from the element’s normal position in the document.** So, if you make an element relatively positioned, but don’t set any offsets, it remains in its normal position, but with the crucial difference that it acts as the containing block for any absolute elements nested inside. It sounds complicated, but it’s very simple to implement.

***Moving the navigation bar into place***

So much for the theory. Now you can put it into practice to dispel any doubts you might have about the preceding explanation.

1. You need to add some extra properties to the style rule for the links, so use what you’ve learned about the CSS Designer to select the **“.header ul” (unordered list) selector**, then make the following changes to it:
   * Position (under “layout”): absolute
   * Placement Right: 0px
   * Placement Bottom: 0px



1. Click the Live button to activate Live view. The menu disappears. It’s actually at the bottom-right of the Document window, but you likely can’t see it because the text is white. Don’t worry – we will fix this soon!
2. To see for yourself that your nav bar has moved to the bottom of your page, in the CSS Designer, select the “.header a” selector, which will highlight your nav bar.
3. To anchor the navigation bar inside the header <div>, you need to make the <div> relatively positioned. Select the **“.header” selector**, then make the following change to it:
   * Position (under “layout”): relative
4. The navigation bar jumps into place at the bottom right of the header <div> (see image, below).



*Positioning the “.header ul” element relative to the “.header” element causes the navigation bar to jump back into place at the bottom right of the header*

1. Exit Live view, and save the style sheet.

**Note:** **Design view sometimes gets confused by positioned elements. If the header disappears in Design view, close and reopen index.html.**

-------------------------------------------------------------------------------------------------------------------------

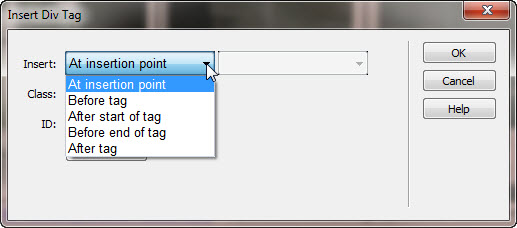
**Position text over the main image**

Although the main\_image <div> looks as though it’s filled with a large image, there’s actually nothing inside the <div>. The image is simply the background. The design calls for a block of text with a white background to be placed close to the white cross at the bottom-right of the image. You’ll use the same technique as for the navigation bar to move the text into position using a combination of relative and absolute positioning.

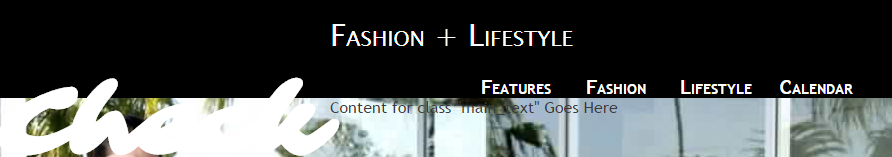
***Adding the text***

The text consists of two headings and a paragraph, so you need to create a <div> to hold them all together, and nest it inside the main\_image <div>. The most accurate way of inserting a new <div> is to use the options in the Insert Div Tag dialog box. It doesn’t matter where the Insertion point is, the options enable you to insert the <div> accurately, as long as your page elements have IDs.

1. Create the new Div tag using the same method you did in Part 2 of this tutorial: click inside the “main\_image” div (by clicking on the image in Design view, or clicking inside the “main\_image” div in Code view). In the Insert tab beside the CSS Designer tab, click Div.
2. In the Insert Div Tag dialog box, leave **At Insertion Point** selected, type **main\_text** as the class, and leave the ID blank.



1. Click OK.The new <div> appears on your page, surrounded by a dotted line in Design view, and with some placeholder text inside (see image, below).



*Some placeholder text should appear right below the Nav bar*

1. Use what you’ve learned about the CSS Designer to add the **“.main\_text” selector**, then make the following changes to it:
   * Background-color (under “background”): white
   * Width (under “layout”): 300px
2. The top-left of the main\_image <div> should now have a 300-pixel wide, white box containing placeholder text (see images, below).



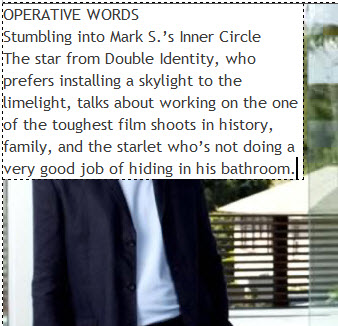
*Live view*



*Design view*

The white background of the main\_text div covers the background image, so the text can be read.

1. Now, it’s time to insert the final text. In the Files tab, next to the CSS Designer tab, expand the **assets** folder, and double-click the **main.txt** file to open it.
2. Select the entirety of the last block of text in the file, everything from "OPERATIVE WORDS" up to and including ". . . hiding in his bathroom."
3. Copy the text by choosing Edit > Copy, by right-clicking and selecting Copy from the context menu, or by pressing Ctrl+C/Cmd+C.
4. Close the main.txt file.
5. In Design view, select the placeholder text for the main\_text <div>, and paste the text from your clipboard by pressing Ctrl+V/Cmd+V or choosing Edit > Paste.The text you copied from main.txt replaces the placeholder text, and the main\_text <div> automatically expands vertically to accommodate it (see image, below).

****

*Design view*

1. Choose File > Save All Related Files to save index.html and the style sheet.

***Moving the main text into position***

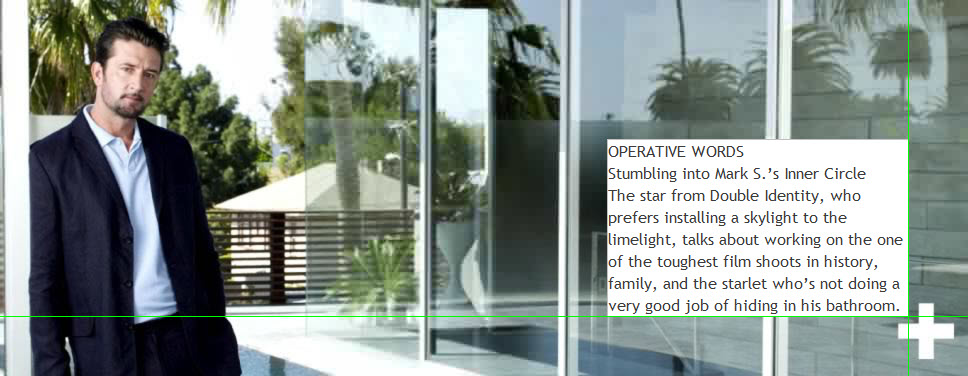
The main\_text <div> will need to be absolutely positioned with the main\_image <div> as its containing block. Although the technique is the same as used for the navigation menu, you need to work out the offsets from the bottom and right of the main\_image <div>. You could try to judge the distance by eye and keep adjusting the values until you get it right. A more efficient way is to use guides in the Dreamweaver Document window to measure the distance accurately.

1. Make sure the Document window is in Design view. Click the Design button if it isn’t.
2. (Optional) Press **F4 to hide the panels.** You need a good view of the page layout, particularly of the main image.
3. Choose View > Rulers > Show. This adds rulers measured in pixels at the top and left of the Document window.
4. Click inside the top ruler and drag down. As you drag, you’ll see a thin, green, horizontal line move down the page, with a yellow tooltip indicating how many pixels it is from the top of the page (see image, below).



*Dragging a horizontal guide from the ruler at the top of the page.*

1. Continue dragging until the green line is flush with the bottom of the main image (around 485px). You should feel the guide snap into place as you move over the image bottom.
2. Drag another guide from the top ruler until it’s in line with the position above the white cross where you would like the bottom of the main\_text <div> to be. Make the distance between the two lines 60px. To figure this out, hold down the Ctrl key while you drag. Dreamweaver displays the distance in pixels between the two guides.
3. Drag a third guide from the vertical ruler on the left of the screen.Continue dragging until the green line is flush with the right side of the main image (around 1091px). You should feel the guide snap into place as you move over the image bottom.
4. Drag fourth guide from the left ruler until the distance between this line and the one you just dragged is, again, 60px. To figure this out, hold down the Ctrl key while you drag. Dreamweaver displays the distance in pixels between the two guides.
5. You need to make the main\_image <div> the containing block for the main\_text <div>, so select the **“main\_image” selector** then make the following change to it:
   * Position (under “layout”): relative
6. You need to make the main\_text <div> the containing block for the main\_text <div>, so select the **“main\_text” selector** then make the following changes to it:
   * Position (under “layout”): absolute
   * Placement Right (under “layout”): 60px
   * Placement Bottom (under “layout”): 60px
7. Your layout should now look like the image below:



*Design view: The main\_text <div> has been accurately positioned against the guides*

**

*Live view: The main\_text <div> has been accurately positioned against the guides*

1. If your box looks like my box, you no longer need the guides, so dismiss them by choosing View > Guides > Clear Guides.

**Note:** If the main\_text area still doesn’t move. Try closing index.html and reopening it.

1. Click File > Save All Related Files to save the style sheet.

**-------------------------------------------------------------------------------------------------------------------------**

**Review Questions:**

1. When you change the color of the text within the <body> tag, why doesn’t the color of the header text (the Google Font) change?
2. What is a descendent selector and how does it work?
3. What is an example of a descendent selector within your html/css code for Check Magazine?
4. Why are descendent selectors so handy?
5. What can cause elements to go “haywire” when working with positioning?
6. Under which circumstances is it a good idea to use absolute positioning? A bad idea?

**------------------------------------------------------------------------------------------------------------**

**Where to go from here**

The Check Magazine home page is beginning to take shape, but it needs more content. You’ll add that next in [Part 4: Inserting images and styling text](http://www.adobe.com/devnet/archive/dreamweaver/articles/first_website_pt4_cs6.html).