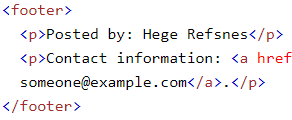
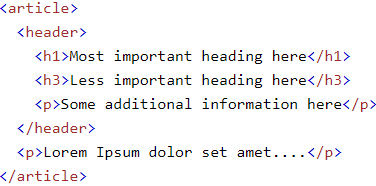
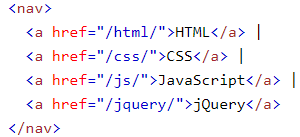
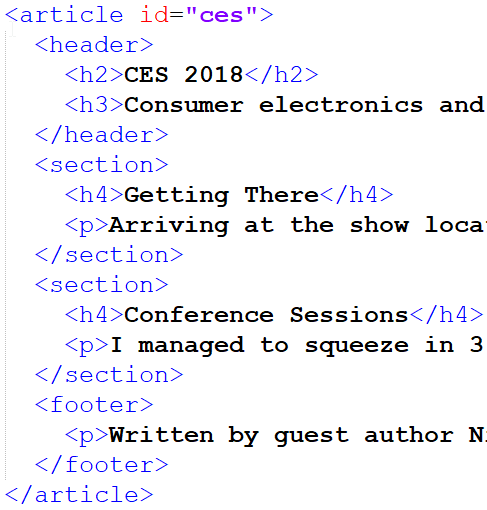
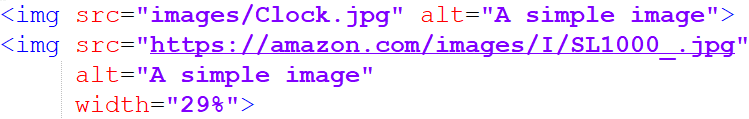
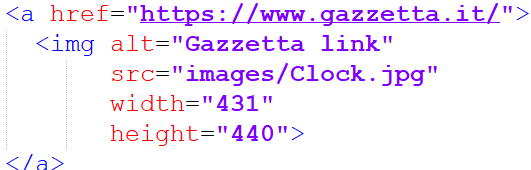
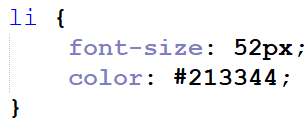
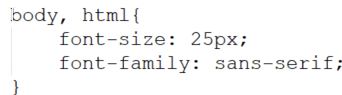
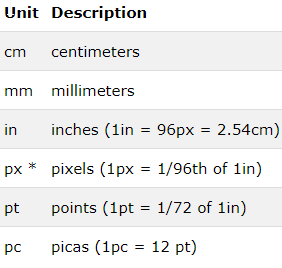
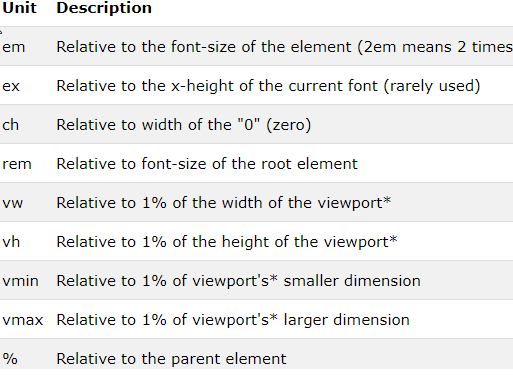
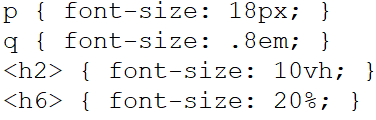
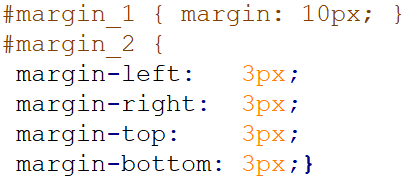
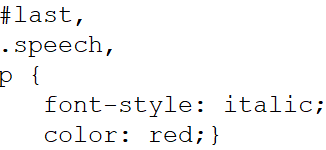
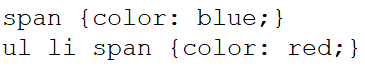
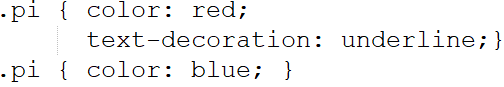
Html 5 and CSS

**W3Cx: HTML5 and CSS Fundamentals**

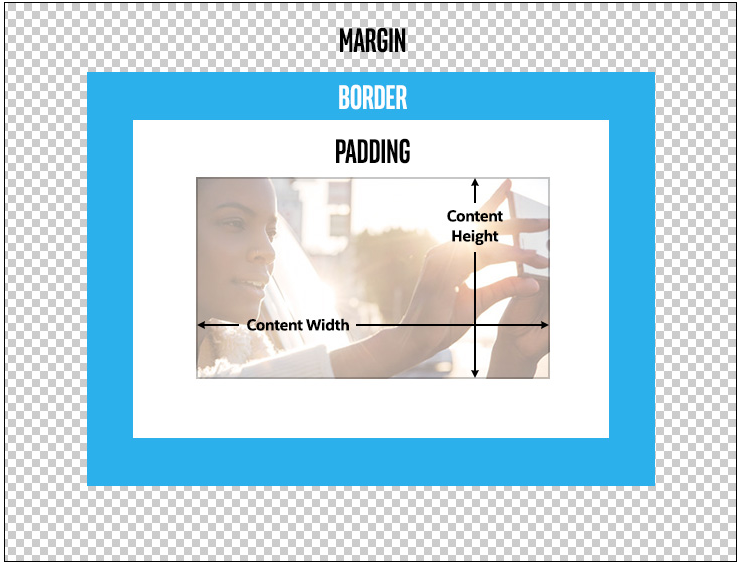
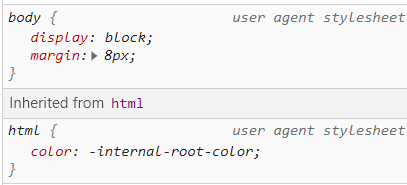
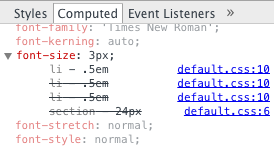
**01 Introduction to html**

* What does Markup really mean? Essentially, it means to annotate a document with extra information, things like where different sections and paragraphs begin and end, which part is the title, which things should be emphasized and so on.
* The most used tags are <p>, <h1> - <h6>, <ul> - <ol>, <hr>, <br>
  + <hr> defines a thematic break in an HTML page
  + <br> tag inserts a single line break.
* Tag names are "case insensitive", instead attributes are case sensitive.
* Double and single quotes are almost interchangeable, but they have to match.
* We have been emphasizing the general rule that HTML is for the logical structure of your content, not what it looks like. Well, nothing is perfect, including this goal. There are some HTML elements that are primarily used to satisfy certain formatting requirements: <hr> horizontal line, <br> or <br/> break line and <pre> stands for "preformatted text", meaning "text is good, don't mess with it."
* Attributes are used in tags to further define the tag: name-value pairs used inside tags and should be added after a space from the tag name. The only exception to the name-value pair is if the attribute is a 'boolean attribute'. These attributes have only two types of values - true or false. But instead of writing "true" or "false" for its value the presence of a boolean attribute on an element represents the true value, and the absence of the attribute represents the false value.
  + <p id="para1" hidden> … </p>
* We should always define the language of our pages, to do this simple add the lang attribute to the html tag:
  + <html lang="en">
* Attribute id gives a unique identify to an element, it’s useful for 2 reasons: 1) Styling your element (css). 2) Specifying a link target internal at the page. You can link to a section of your HTML page using the 'id' of the section: You should reference the 'id' value with a # preceding it:
  + <P id="plutus"> ...
  + <a href="#plutus">link to plutus</a>
* There are two kind of attributes: Global and Non-global. Global attributes can be applied to all tags (id and class are examples), non-global attributes are attributes applied to a specific instance of a tag. Start is an attribute for the <ol> tag and it cannot be applied on the <p> or <h1> tags.
* Attribute src specifies the location (URL) of an external resources:
  + <img src="smiley.gif" alt="Smiley face">
  + <script src="myscripts.js"></script>
  + <audio src="horse.ogg" controls></audio>
* Semantic HTML is HTML that concentrates on the meaning of information in Web pages instead of its presentation or look. Semantic tags, like p, suggest the purpose of the content within the tags From a semantic HTML perspective, using the right tags is important: you should use blockquote to wrap a quote and not use a paragraph tag and then style it to look like a quote. For presentation purposes, you can achieve the same using CSS, but how something looks has very little to do with what it means. This is why in HTML, we separate content and style.
* The <header> element represents a container for introductory content or a set of navigational links. <footer> element is similar it should have information about its referred elements. Header and footer elements can also be used site-wide at the top and bottom of the body of the Web page. This type of header will typically contain logos, main heading, a search area and site-wide navigation and the footer will typically include authoring information, references and other links, copyright information etc.
* <nav> element represents a section of a page that links to other pages or to parts within the page: a section with navigation links.
  + Notice that NOT all links of a document should be inside a <nav> element. The <nav> element is intended only for major block of navigation links.
* <article> and <secion> : An article element rapresents stand-alone content. If you pick an article out of a Web page, it should make sense all by itself. Section elements are used to section the content of a page, but a single section doesn't make sense by itself.
* <div> tag is one you will likely see sprinkled all over an HTML document. It is used to define a division or a section of the document. Div is not a semantic element, however, it is commonly used when there isn't a better semantic assignment for it. It is like a generic container that can hold a variety of elements such as paragraphs, images, links, tables, etc, it’s often be used to group elements for styling purposes.
* Like div, <span> is not a semantic element, you should only use span if no other semantic element is appropriate. div and span serve the same purpose but should be applied at different levels. div is a block level element (for a block of space) while span is an inline element (for within a line or phrase).
  + To add styling to part of a sentence (inline)
  + Manipulate part of a sentence using JavaScript
  + When no other HTML element is applicable, you can use span (and div) to add attributes such as class and id.
  + <p>Hi everyone! My name is Alexa and I work for <span class="company">ABC Company</span></p>
* <img> tag defines an image in an HTML page, it has two required attribute: src and alt
  + src told you where to fetch the image from: from a local patch in your web site or from the web. Use Unix (/) path name separator instead of Windows (\) style.
  + alt attribute stands for alternate text for an image. It’s important because is the text alternative to the image for users who are unable to see the image. It is also useful to provide relevant information for search engines: they do not 'see' images but rely on the alt attribute to find out what the image is about. If you use your target keyword in alt, it will optimize the search.
  + title attribute should provide additional information about the image. The title attribute should not be relied upon for important information, and it should not be used in place of the alt attribute.
  + Attribute width and height. Use of these attributes really depends on how you are using the image. If it is part of an image grid or a list with multiple images of the same size, it is best achieved by CSS. So you don't have to bother adding the same dimensions to every image and it will be repetitive. Plus it is generally bad practice to encode dimensions directly into the HTML. However, if you are adding the image into some content and it needs to be a certain size for the visual flow of the reader, then it is best to add it to HTML using the height and width attributes.
* Hyperlink <a> is any text or image you can click and it will take you to another page. Links have a default appearance in most browsers: blue and underlined. Ensure no other text in your page is underlined to avoid confusing the user. <a< isn't the same of <link>: The anchor element is used for hyperlinks while tag is used to define a link between a document and an external resource like an external style sheet. A link can lead to three different location types:
  + <a href="https://www.gazzetta.it/"> Gazzetta </a> Another web site
  + <a href="RelativePage.html"> relative URL</a> A local link
  + <a href="#plutus">Plutus </a> another part of the same page
* Target attribute specifies the destination where the linked URL in href should be opened:
  + target="\_self" : same page
  + target="\_blank" : new window
* download attribute makes a link download a file instead of navigate to another location.

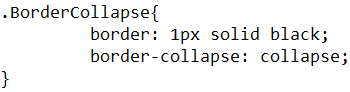
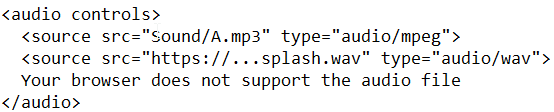
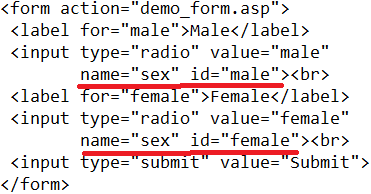
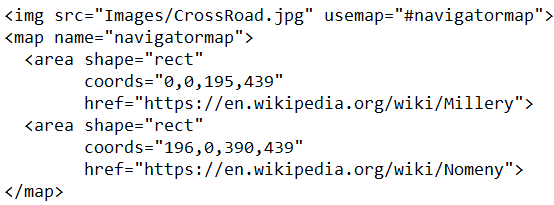
**03 Introduction to CSS**

* CSS is just a list of rules, each rule consist of a selector and a declaration. When a selector appears unprefixed by any punctuation, then it is assumed to match to an HTML tag. This li selector will apply the CSS rule to all <li> tags in the document.
* It’s possible to have a default rules that are always valid for each elements in the document. To obtain this use ‘body, html’
  + , is used to apply the same declaration to mare than one selectors.
* CSS has several different units for expressing a length. Many CSS properties take "length" values, such as width, margin, padding, font-size, border-width, etc. Length is a number followed by a length unit, such as 10px, 2em, etc. There are two types of length units: absolute and relative
  + Absolute length units are not recommended for use on screen, because screen sizes vary so much. However, they can be used if the output medium is known, such as for print layout.
  + Relative length units specify a length relative to another length property. Relative length units scales better between different rendering mediums.
* font-size: 18px; can be used to size the text of a tag, it’s possible to use different units of measure: .8em is relative to the font-size of the element (2em means 2 times the size of the current font), 20% sets the font-size to a percent of the parent element's font size…
* line-height:3 line-height property drives the height of the space where text is drawn into. A large line-height will give the text more spacing. A small line-height will smash the text lines together. The used value (an unit less number) is multiplied by the element's font size to obtain the real height value
* text-align: justify text-align valid values are: left center right justify.
* text-decoration: underline Used to underline (and similar) the text, valid value are: underline, overline, line-through, and none;
* font-weight: bold; While common values are normal and bold, text can also be made bolder (or less bold) than its parent with the values bolder and lighter. Lastly, the font-weight can be set explicitly as a numeric value: 100, 200, 300, 400, 500, 600, 700, 800 and 900 (only this number are valid).
* font-style: italic; Valid values for this property are normal and italic
* font-family: "fantasy", "Verdana", "Arial", sans-serif; Set the font of an item, browsers only guarantee a few standard choices: serif, sans-serif, monospace, cursive, and fantasy. font-family property accepts a list of possible font choices: the browser will start with trying the first font listed, and if not available (or not having a needed glyph) it will then proceed to the next font in the list, and so on.
* margin can be a bit confusing, depending upon context, it will space an item away from its immediate neighbors (in the HTML) or from the edges of its parent. Also, there is not only one margin property, but five.
* color is used to set the color of a text
  + color: blue; this simplest form allow you to choose one of the predefined color
  + color: rgb(142,34,218); any color can be specified by mixing three components together: red, green, and blue. The amount of each component falls within a range between 0 and 255.
  + rgba(142,34,218, 0.5); similarly rgba is used to have semitransparent colors.
  + color: #3B6B0A; lastly we can provide an hexadecimal (hex) code.
* Unit of measure. Many CSS properties expect some sort of dimension value. Dimension values support a wide variety of units the most common and useful ones are: px, em, rem, %, vh and vw
  + margin: 10px; px it's short for 'pixel' which is a single dot on the screen. Text with font-size:20px is 20 pixels tall on-screen.
  + font-size: 0.9em; em maps to the height of one capital letter in the parent context, it's usually used for vertical dimensions.
  + font-size: 2rem; rem is much like em, except that it always derives its size relative to the root.
  + margin-left: 5%; % is relative to the parent dimension. If the parent element doesn't have an explicit width or height set then child percentages may be percentages of 0.
  + margin-left: 5vw; margin-right: 5vw; vh stands for viewport height, and 'vw' for viewport width. They work much like the percentage % unit but instead of percentage of the parent, it is percentage of the screen (aka viewport). Obviously, vh is for vertical dimensions, and vw for horizontal dimensions.
* We can alter the aspect of a list using attribute list-style-type, this attribute can take different value for ordered and unordered list:
  + Unordered: disc, circle, square and none.
  + Ordered: decimal, decimal-leading-zero, lower-roman, upper-alpha, simp-chinese-formal.
* There are two really used selector: id selector # and class selector .
  + #par\_1 {... with # we select a unique id present inside html <p id=” par\_1”>
  + .cl {… we select a class <div class=’cl’>
* Comma separated selectors: It’s possible to combine different selector using ‘,‘ to apply to everyone the same css rules.
* Specialized selectors: If two selectors appear next to each other with no spacing separating them, they are forming a specialized selector. To match, a candidate must match both rules. If a tag selector is used it must appear first.
  + li.talk { color: purple; }
  + p#first { color: purple; }
  + .insect.flying {list-style-type: square; }
* Descending selectors (space): if it’s necessary to enter inside the tag tree we can use tags separated by a space.
  + #intro a { color: yellow; } any <a> tag that is a descendant of #intro.
  + p #guideline { color: #00FF00; }
* Direct Descending Selector > : Sometimes you don't want to apply a style to any \_possible\_ child, but to only to the direct children. This can be done with the > symbol.
  + #direct > a { color: #00FF00; } all the <a> directly inside a tag of class direct
* Everything selector (\*) : The asterisk (\*) can be used to match any tag.
  + #oneforall \* { color: #00FF00; } any tag that is progeny of a tag of class oneforall.
* Most CSS rules once applied to an element are also applied to all the children of that element, and to their children, and so on. There are exceptions, notably the layout properties (margin, padding, position, width, etc.) and the decorative properties (border, background, etc.) do not cascade. This cascading of a CSS property is called inheritance.
  + Inheritance can be explicitly set. Many CSS properties accept the value of inherit, which means to inherit the value from the parent. …{padding: inherit;}
  + There is no reliable rule for which CSS properties are inheritable by default and which are not. Generally properties associated with positioning and layout do not inherited; likewise, the decorative properties (borders, background images, etc.) do not inherit. Most properties that begin with text- or font- inherit.
* Conflicting rules. A tag can matches different selectors and the declarations can go in contrast, the browser has some guidelines for resolving conflicts:
  + A more specific rule takes precedence over a less specific rule. A rule that more tightly matches a particular element than a general rule will be applied, color red wins over blue .
  + #id selector is always the most specific.
  + .class selector is more specific than a tag selector. Rules employing a class selector (e.g. .someclass) are considered more specific than rules without (but not as specific as an #id selector, which trumps everything)
  + Rules that come later override those that come earlier. This guideline is for two CSS rulesets with the same selector, when there are conflicts, the rules from the later one apply. In the example above, an element with the .pi class will be underlined and its color will be blue
  + You may from time to time encounter a situation where you need to apply a particular CSS property and you want it to take precedence over all others, no matter what. !important will do that. The exclamation point is required, and the whole symbol ( !important ) goes after the value and before the semi-colon ;

**04 Fixing and debugging**

* The Box Model governs 3 important spacing features of CSS: Margin, Border and Padding.
* All elements in an html document end up being treated as rectangles somewhere in the window. The content of each rectangle corresponds to the innermost rectangle in the image below. Just outside the content is the padding. This is kind of like an internal margin, meaning that it separates the contents from the border. The border essentially traces the sides of the padding rectangle.
* To manage Border we usually set three properties: border-width (the size of your imaginary pen), border-style (dashed, dotted, solid, etc.) and border-color (the color of your pen).
  + There is a shorthand syntax to set all three in one line: “border: 5px dotted red;”
* The margin specifies the position of the element relative to whatever is adjacent to it, either to the right or left, or top or bottom. The margin is always transparent, and each side can be set individually.
  + Values for any of the sides can be negative, even if that means that it overlaps with another element on the page.
* Padding controls the amount of space between the elements content and the border box. If you have no padding, then the contents of the element would be right up against the border. The background of the padded area matches the background of element, so the effective visible size of the element includes the padding.
* Inside css rules there are always two grayed-out section ‘User agent stylesheet’: these are basically the defaults values that the browser will use if nothing else is specified.
* When working with relative measures (.5em) styles panel doesn't tell us a lot about the actually font-size in absolute terms; we can use the "Computed" tab. It contains the values of all the CSS properties that apply to the current element.

**05 More HTML and CSS**

* <table> are used to arrange data in tabular format - rows and columns of cells. You can put a variety of data like text, images, forms, links and even other tables in your table.
  + In earlier days many developers resorted to tables as a means of layout, there's really no need to do that anymore. To change the layout of the page, you shouldn't be editing your content but only your CSS.
  + To declare a row use <tr> tag.
  + Inside a row we can declare cells, there are two different types: <th> declare a header cell (bold and center by default), <td> a standard one.
    - With attributes colspan and rowspan a cell can extend on more columns and rows.
  + The <thead> element is used in conjunction with the <tbody> and <tfoot> to specify each part of a table (header, body, footer). Browsers can use these elements to enable scrolling of the table body independently of the header and footer.
  + If we gave a border to the table, table-header and table-data this creates a double line. In order to collapse them all into a single border, we use the border-collapse property.
  + Browsers automatically set the width and height for the rows and columns for your table based on the content in your cells Explicitly setting width/height of one cell will not only affect that cell but the whole column/row.
  + Using css we can select all the even and odd lines <tr> of a table:
    - .Table tr:nth-child(even){
    - . Table tr:nth-child(odd){
    - . Table tr:nth-child(3n) { with the same logic we can select all the third rows.
  + To select a row when the mouse is over it we can use: ‘.Table tr:hover{…’
* We can use <audio> tag to embed audio in our web page, in conjunction with the tag <source> it’s possible to specify multiple different source file. The advantage of providing multiple source files in different formats is that if the browser doesn't support the first format, it will automatically try the second one.
  + It’s also possible to add a test to use if the browser doesn’t support any file.
* Similarly to the previous one we can use <video> tag to embed video in our page. To select a source we can use the src attribute (single source) or the source tag (multiple source).
* Tag <iframe> allows you to put another Web page in your Web page, they are generally used in Web pages to show external content/resources. The type of content is not limited to other Web pages. You can add YouTube videos or display a PDF file. There is one significant problem with iframes: suppose you create your Web page, containing only an iframe with src="http://foo.com", with no borders, padding or margin. By all appearances, you would seem to be on the Web site foo.com. If you don't look at the URL, it might be difficult to tell, for reasons like this, some Web sites disallow their inclusion.
  + Iframes load separately from the main page, however, they do block the main page's load command until its content finishes loading. You can avoid this by applying some Javascript.
  + Useful for third party content like ads.
* Using <map> tag it’s possible to add a map to an image to enable the user to interact differently with various image’s parts.
  + The image must declare that it’s using a map with the attribute usemap.
  + A map can include any number of different <area> tags, each providing a different type of user interaction.
* What is the difference between name and id attribute? The ID of an element has nothing to do with the data contained within the element. IDs are for hooking the element with JavaScript and CSS. The name attribute, however, is used in the HTTP request sent by your browser to the server as a variable name associated with the data contained in the value attribute. With form elements the name attribute is used to determine the name-value pairs to be sent to a server-side program and should not be eliminated. Now in most elements, the name attribute has been deprecated in favor of the more ubiquitous id attribute. However, in some cases, particularly form fields (<button>, <input>, <select>, and <textarea>), the name attribute lives on because it continues to be required to set the name-value pair for form submission.
  + To ensure compatibility, having matching name and id attribute values when both are defined is a good idea. However some tags, particularly radio buttons, must have nonunique name values, but require unique id values.
  + When using radio buttons the importance is that the name remains the same throughout the options so there are no multiple selections.
* Decorative images are incorporated via CSS, not using the <img> tag. The simplest method use the background-color CSS property, it fills the rectangle of the given element with a solid background color. The background-image property is used to set an external image file as the background to a particular HTML element, the image can be local (starting point of search is the position of css file) or taken from the web. With this property there are quite a few different usage scenarios: for instance, an image can be used as repeating tile, or a background image can fit its parent element, or be a large panoramic image not fully viewed. These scenarios can be constructed with other CSS properties, like background-repeat, background-size, and background-attac.
  + By default, if the rectangular area of an element is bigger than the image itself, then the image will repeat and fill the space. The background-repeat property can be used to control this behavior, it's more commonly used values are: repeat, repeat-x, repeat-y, space and no-repeat.
  + When not repeating, it is very useful to size a background image to fit its element using the background-size property. It has two very useful values: contain and cover. The contain value will put the entire image into the space of the element, however, the space of the element may not be completely filled if the aspect ratio of the element and the image do not match (it shrinks but doesn’t enlarge the image). The cover value is the opposite. It will completely fill the element but the image may be cropped off two opposite sides (it enlarges but doesn’t shrink). Neither contain or cover will distort or squish the image, Its aspect ratio is maintained.
  + If we need a fixed dimension for the background image we can use the property background-size: it takes two values separated by a space: the first governs the width, the second the height. If none of the two values is set to ‘auto we can have image distortion.