Day 3 - HTML Layout

Google Chrome Tool for Developer

1. The name is **inspect element**
2. Right click on any page area then select inspect element
3. Useful to analyze your web html & css code

Border

Every HTML element has a box around it, even if you cannot see it.

```
* {
    border: 1px solid red;
}
```

HTML Block and Inline Element/Tag

Inline Elements

Are very humble. They do not start on a new line and only takes up as much width as they need to fit their contents.

Block Elements

Are quite greedy. They always start on a new line and take up the full width of the page, forcing the next HTML element down to the next line.

http://learnlayout.com/

Inline Element/Tag

1. <span> is like <b>, <i>
2. **Everthing is inline**
3. <span> elements don't change the appearance of the page until styles are applied to them.

Block Element/Tag

1. <div> is like <h1>, <p>
2. Next element always start **on a new line**
3. <div> elements don't change the appearance of the page until styles are applied to them.

Margin & Padding

Margin
The CSS margin property defines the extra space outside the border
div { margin: 30px; }

Padding
The CSS padding property defines the extra space inside the border:
div { padding: 10px; }

Margin & Padding Multiple Value

```
div { margin: top right bottom left; }
```

Position

static
HTML elements are positioned static by default. It’s not affected by the top, bottom, left, and right properties.
div.static { position: static; border: 3px solid green }

relative
Setting the top, right, bottom, and left properties of a relatively-positioned element will cause it to be adjusted away from its normal position.
div.relative { position: relative; border: 3px solid blue; left: 30px; }

Position

fixed
A fixed element is positioned relative to the viewport, which means it always stays in the same place even if the page is scrolled. As with relative, the top, right, bottom, and left properties are used

```
div.fixed {
    position: fixed;
    bottom: 0;
    right: 0;
    width: 300px;
    border: 3px solid #73AD21;
}
```

Position

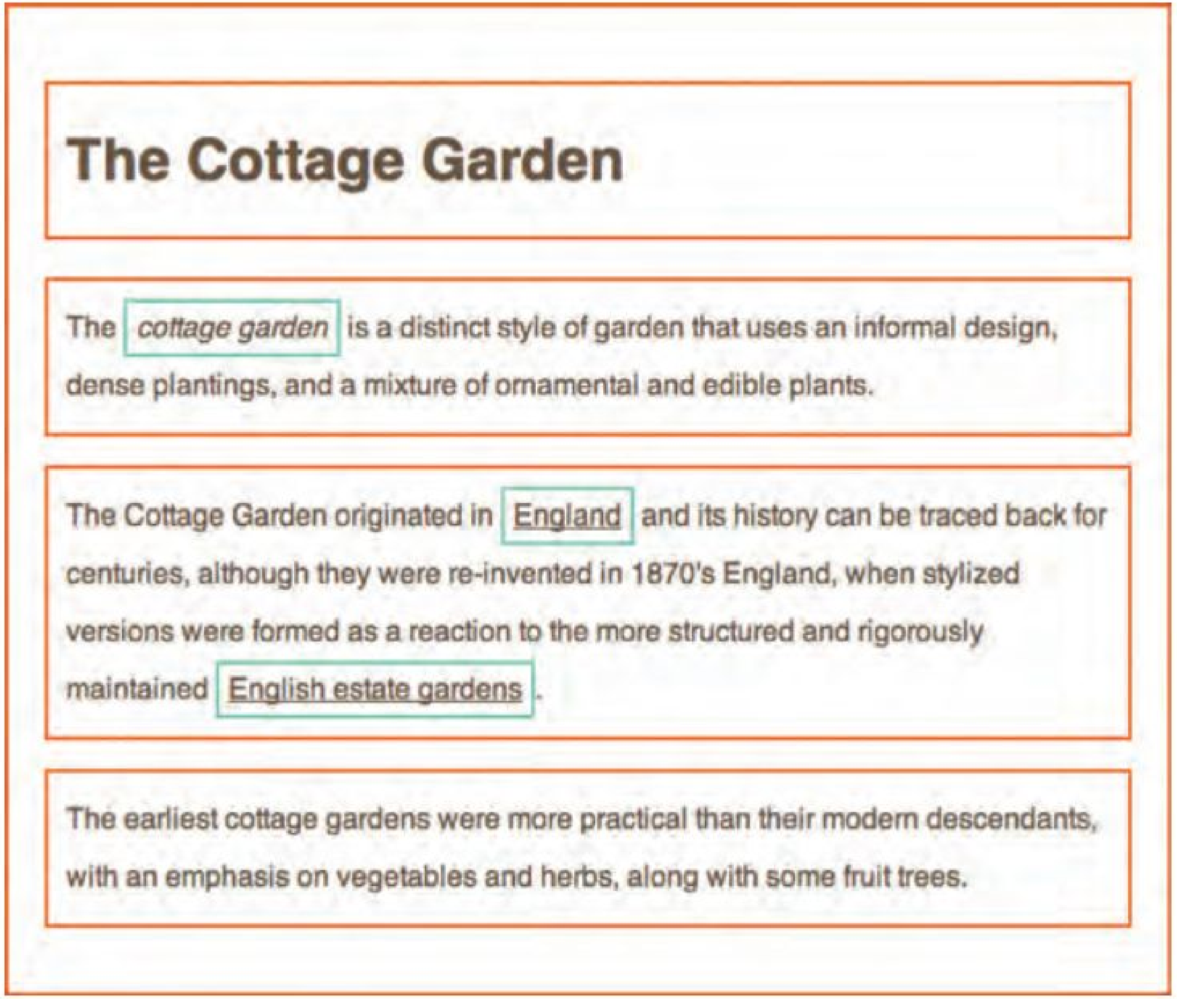
absolute
absolute is the trickiest position value. absolute behaves like fixed except relative to the nearest positioned ancestor instead of relative to the viewport. If an absolutely-positioned element has no positioned ancestors, it uses the document body, and still moves along with page scrolling.

Remember, a "positioned" element is one whose position is anything except static.

Position

```
div.relative {
      position: relative;
      width: 400px;
      height: 200px;
      border: 3px solid #73AD21;
}

div.absolute {
      position: absolute;
      top: 80px;
      right: 0;
      width: 200px;
      height: 100px;
      border: 3px solid #73AD21;
}
```


The Cottage Garden

The cottage garden is a distinct style of garden that uses an informal design, dense plantings, and a mixture of ornamental and edible plants.

The Cottage Garden originated in England and its history can be traced back for centuries, although they were re-invented in 1870's England, when stylized versions were formed as a reaction to the more structured and rigorously maintained English estate gardens.

The earliest cottage gardens were more practical than their modern descendants, with an emphasis on vegetables and herbs, along with some fruit trees.

In this example, block level elements are shown with red borders, and inline elements have green borders.

The `<body>` element creates the first box, then the `<h1>`, `<h2>`, `<p>`, `<i>`, and `<a>` elements each create their own boxes within it.

Using CSS, you could add a border around any of the boxes, specify its width and height, or add a background color. You could also control text inside a box — for example, its color, size, and the typeface used.