Layout Basics Cheat Sheet

Span

- A generic wrapper for any inline content. `span`s are generally used to group small sections of content for styling purposes.

Div

- A generic wrapper for any block content. `div`s are used to group other elements together or to provide style to a specific area.

Header

- These elements kick off your page with introductory content like logos, headlines, titles, and links.

Footer

- This element wraps the content at the bottom of your web page. This element often contains copyright information, links to career pages, contact information, terms of use, etc.

Main

- These elements contain all the primary content — articles, blog posts, images, videos, etc. — between the `<header>` and the `<footer>`.

Aside

- These elements contain all the secondary content between the `<header>` and `<footer>`, including supplemental information like recommended stories or archived blog links.

Nav

- Any navigational links that help users get around your page should be wrapped in `<nav>` tags.

Section

- These elements are used to group the content of a page into related chunks.

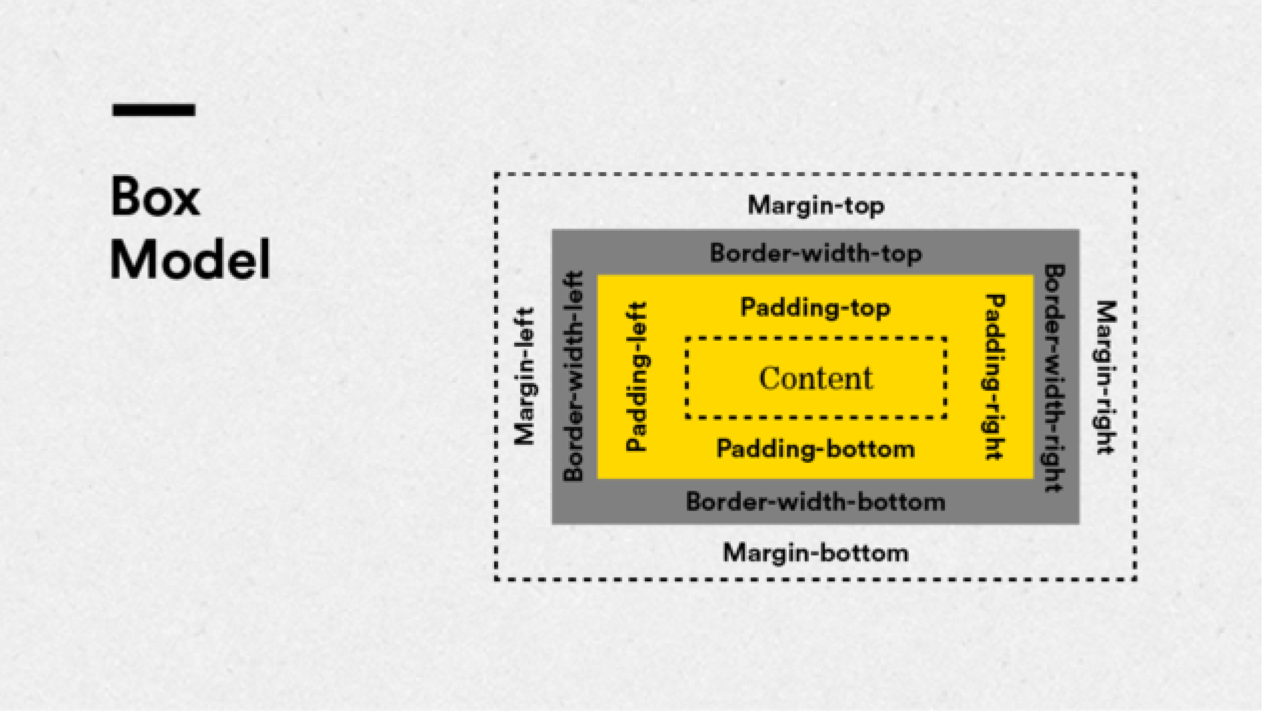
Article

- These elements are used to contain standalone blocks of text, such as news articles, blog posts, etc.

Box Model

- Every HTML element on the page is in a box, and the box model dictates what the element will look like. The model consists of the content, padding, border, and margin.

Here's a diagram to help you remember the box model.



Content

- The text and images that are included within any element's opening and closing tags.

Padding

- The area outside the content, but within the `border`. `padding` will extend the `background-color` and contribute to the overall `width`of the object.

Border

- A line of variable thickness, color, and style that goes around the `padding` and content.

Margin

- The area outside of the `border` that separates the object from other objects around it. `margin` has a transparent `background-color`.

Inline Elements

- HTML elements that take up the width of their contents and do not begin with new line. They cannot have `height` or `width`assigned.

Block Elements

- HTML elements that break to a new line before and after the element and they take up the width of their containers by default (this is often the browser window itself). They can have `padding`, `margin`, `height`, and `width` assigned.

Span

- An element used to apply CSS to `inline` HTML.

Div

- A `block` element used to apply CSS to blocks of HTML. Divs typically wrap big sections of content on a page. Use semantic elements over divs whenever possible.

The Box Model

1. Set the `background-color` of `body` to `#333` and the `background-color` of `div` to `#c0dec5`.

```
body {
    background-color: #333;
}

div {
    background-color: #c0dec5;
}
```

2. Add a `margin` of `64px` on *all four sides* of the `div`s content (outside the pale blue box).

```
div {
    background: #c0dec5;
    margin: 64px;
}
```

3. Set the `div`s top `padding` to `32px`, a left and right `padding` of `48px`, and no `padding` on the bottom.

```
div {
    background: #c0dec5;
    margin: 64px;
    padding: 32px 48px 0;
}
```

4. Create a rule for `span` and set these three property-value pairs: a `font-variant` of `small-caps`, a `font-family` of `Tahoma, sans-serif`, and a `letter-spacing` of `.25em`.

```
span {
    font-variant: small-caps;
    font-family: Tahoma, sans-serif;
    letter-spacing: .25em;
}
```

5. Set the `span`s `padding` to `0 3px`. Set its `background-color` to `rgb(100,150,150)`.

```
span {
    font-variant: small-caps;
    font-family: Tahoma, sans-serif;

    letter-spacing: .25em;
   padding: 0 3px;
```

```
        background-color: rgb(100,150,150);

    }
```

6. On second thought, that color is a little dark. Let's practice with RGB and alpha and make it 50% translucent. Change the `spans` `background-color` from `rgb` to `rgba` and add a `.5` alpha.

```
span {
    font-variant: small-caps;
    font-family: Tahoma, sans-serif;
    letter-spacing: .25em;
    padding: 0 3px;
    background-color: rgba(100,150,150,.5);
}
```

7. In your HTML, the second `p` tag has a class attribute named `source`. Let's right-align that paragraph! Add the new style to your CSS.

```
.source {
    text-align: right;
}
```

Header, Footer, Nav

Section 1: Content and Page Structure

1)

```
<body>
    <header>
        <h1>ELK Web Design</h1>
        <nav>
            <a href="">Work</a>
            <a href="">About</a>
            <a href="">Contact</a>
        </nav>
        <img src="http://i.imgur.com/pdsjjxD.jpg">
    </header>
</body>
```

2)

```
<body>
    <header>
        <h1>ELK Web Design</h1>
        <nav>
            <a href="">Work</a>
            <a href="">About</a>
            <a href="">Contact</a>
        </nav>
        <img src="http://i.imgur.com/pdsjjxD.jpg">
    </header>

    <section>
        <h2>Who we are</h2>
        <p>ELK provides clean and innovative sites for small businesses, artists, and professio
    </section>

    <section>
        <h2>How we do it</h2>
        <ul>
            <li>HTML</li>
            <li>CSS</li>
            <li>Responsive</li>
            <li>Visual Design</li>
        </ul>
    </section>
</body>
```

3)

```
<body>
    <header>
        <h1>ELK Web Design</h1>
        <nav>
            <a href="">Work</a>
            <a href="">About</a>
            <a href="">Contact</a>
        </nav>
        <img src="http://i.imgur.com/pdsjjxD.jpg">
    </header>

    <section>
        <h2>Who we are</h2>
        <p>ELK provides clean and innovative sites for small businesses, artists, and professio
    </section>

    <section>
        <h2>How we do it</h2>
        <ul>
            <li>HTML</li>
            <li>CSS</li>
            <li>Responsive</li>
            <li>Visual Design</li>
            </ul>
    </section>

    <footer>&copy; 2015 ELK</footer>
</body>
```

Styling Content

1. Apply the following styles to the `body`: Set its `font-size` to `18px`and the `font-family` to `Arial`.

    ```
    body {
        font-size: 18px;
        font-family: 'Arial', sans-serif;
    }
    ```

2. Write a rule for `h1`, and set its `font-size` to `36px`.

    ```
    h1 {
        font-size: 36px;
    }
    ```

3. Create a rule for `h2`, and set its `font-size` to `27px`.

    ```
    h2 {
        font-size: 27px;
    }
    ```

4. Use only one `padding` declaration to give `h2` a top and bottom `padding` of `14px`.

    ```
    h2 {
        font-size: 27px;
        padding: 14px 0;
    }
    ```

5. Create a rule for both `h1, h2` and set its `text-transform` property to `uppercase`.

    ```
    h1, h2 {
        text-transform: uppercase;
    }
    ```

6. Let's keep styling both headings. We'll use CSS shorthand to "underline" each one with a `border-bottom`. Give `h1, h2` a `border-bottom` property that's `4px` thick, `solid`, and `black`.

    ```
    h1, h2 {
    ```

```
        text-transform: uppercase;
        border-bottom: 4px solid black;
    }
```

7. Set `h1, h2`'s `display` to `inline-block`.

```
    h1, h2 {
        text-transform: uppercase;
        border-bottom: 4px solid black;
        display: inline-block;
    }
```

8. Create a style for images, and set its `display` to `block`.

```
    img {
        display: block;
    }
```

9. Center all images on the page horizontally by setting `img`s `margin`to `0 auto`.

```
    img {
        display: block;
        margin: 0 auto;
    }
```

10. Create a `ul` style, and set its `list-style` to `none`. Set its `margin`to `0` and its `padding` to `0`.

```
    ul {
        list-style: none;
        margin: 0;
        padding: 0;
    }
```

11. Create a `footer` style in your CSS, and center align its text.

```
    footer{
        text-align: center;
    }
```

12. Set the `footer` `padding` to `18px 0`.

```
    footer{
```

```
    text-align: center;
    padding: 18px 0;
}
```