HTML

1. HTML Introduction

HTML is the standard markup language for creating Web pages.

- HTML is the standard markup language for Web pages.
- With HTML you can create your own Website.
- HTML is easy to learn!

1.1. What is HTML?

- HTML stands for Hyper Text Markup Language
- HTML is the standard markup language for creating Web pages
- HTML describes the structure of a Web page
- HTML consists of a series of elements
- HTML elements tell the browser how to display the content
- HTML elements label pieces of content such as “this is a heading”, “this is a paragraph”, “this is a link”, etc.

1.2. Example

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>

<h1>This is a Heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

1.3. Example Explained

- The `<!DOCTYPE html>` declaration defines that this document is an HTML5 document
- The `<html>` element is the root element of an HTML page
- The `<head>` element contains meta information about the HTML page
- The `<title>` element specifies a title for the HTML page (which is shown in the browser's title bar or in the page's tab)
- The `<body>` element defines the document's body, and is a container for all the visible contents, such as headings, paragraphs, images, hyperlinks, tables, lists, etc.
- The `<h1>` element defines a large heading
- The `<p>` element defines a paragraph

2. What is an HTML Element?

An HTML element is defined by a start tag, some content, and an end tag:

```
<tagname>Content goes here...</tagname>
```

The HTML **element** is everything from the start tag to the end tag:

```
<h1>My First Heading</h1>

<p>My first paragraph.</p>
```

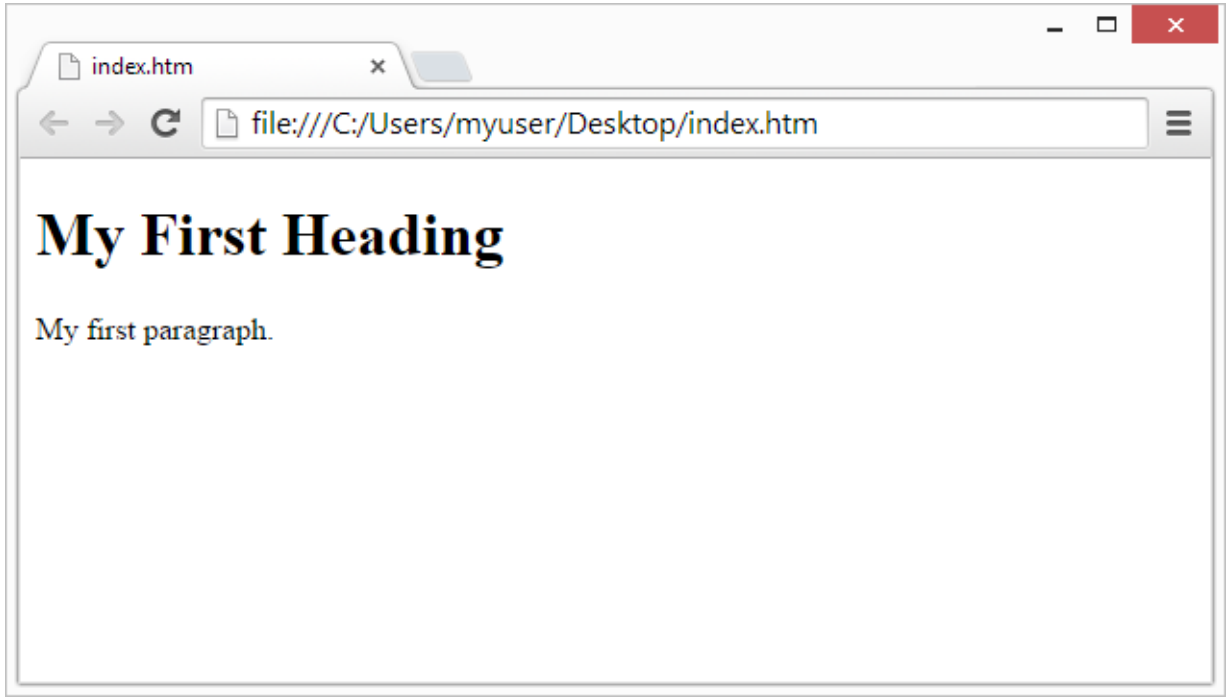
Start tag	Element content	End tag
<h1>	My First Heading	</h1>
	My first paragraph.	
 	none	*none*

> **Note:** Some HTML elements have no content (like the
> element). These elements are called empty elements. Empty elements do not have an end tag!

3. Web Browsers

The purpose of a web browser (Chrome, Edge, Firefox, Safari) is to read HTML documents and display them correctly.

A browser does not display the HTML tags, but uses them to determine how to display the document:



4. HTML Page Structure

Below is a visualization of an HTML page structure:

```
<html>
  <head>
    <title>Page title</title>
  </head>
  <body>
    <h1>This is a heading</h1>
    <p>This is a paragraph.</p>
    <p>This is another paragraph.</p>
  </body>
</html>
```

Note: Only the content inside the `<body>` section (the white area above) will be displayed in a browser.

5. HTML Attributes

HTML attributes provide additional information about HTML elements.

- All HTML elements can have **attributes**
- Attributes provide **additional information** about elements
- Attributes are always specified in **the start tag**
- Attributes usually come in name/value pairs like: **name="value"**

HTML Attributes

The `<a>` tag defines a hyperlink. The `href` attribute specifies the URL of the page the link goes to:

```
<a href="https://www.demo.com">Visit W3Schools</a>
```

The `<img>` tag is used to embed an image in an HTML page. The `src` attribute specifies the path to the image to be displayed:

```
<img src="img_girl.jpg">
```

There are two ways to specify the URL in the `src` attribute:

1. Absolute URL - Links to an external image that is hosted on another website. Example: `src="https://www.demo.com/images/img_girl.jpg"`.

> **Notes:** External images might be under copyright. If you do not get permission to use it, you may be in violation of copyright laws. In addition, you cannot control external images; it can suddenly be removed or changed.

2. Relative URL - Links to an image that is hosted within the website. Here, the URL does not include the domain name. If the URL begins without a slash, it will be relative to the current page. Example: `src="img_girl.jpg"`. If the URL begins with a slash, it will be relative to the domain. Example: `src="/images/img_girl.jpg"`.

> **Tip:** It is almost always best to use relative URLs. They will not break if you change domain.

6. HTML Styles

The HTML `style` attribute is used to add styles to an element, such as color, font, size, and more.

I am Red

I am Blue

I am Big

Setting the style of an HTML element, can be done with the `style` attribute.

The HTML `style` attribute has the following syntax:

```
<tagname style="property:value;">
```

```
<body style="background-color:powderblue;">

<h1>This is a heading</h1>
<p>This is a paragraph.</p>

</body>
```

7. HTML Styles - CSS

CSS stands for Cascading Style Sheets.

CSS saves a lot of work. It can control the layout of multiple web pages all at once.

7.1. What is CSS?

Cascading Style Sheets (CSS) is used to format the layout of a webpage.

With CSS, you can control the color, font, the size of text, the spacing between elements, how elements are positioned and laid out, what background images or background colors are to be used, different displays for different devices and screen sizes, and much more!

7.2. Using CSS

CSS can be added to HTML documents in 3 ways:

- **Inline** - by using the `style` attribute inside HTML elements
- **Internal** - by using a `<style>` element in the `<head>` section
- **External** - by using a `<link>` element to link to an external CSS file

The most common way to add CSS, is to keep the styles in external CSS files. However, in this tutorial we will use inline and internal styles, because this is easier to demonstrate, and easier for you to try it yourself.

7.3. Inline CSS

An inline CSS is used to apply a unique style to a single HTML element.

An inline CSS uses the `style` attribute of an HTML element.

The following example sets the text color of the `<h1>` element to blue, and the text color of the `<p>` element to red:

```
<h1 style="color:blue;">A Blue Heading</h1>

<p style="color:red;">A red paragraph.</p>
```

7.4. Internal CSS

An internal CSS is used to define a style for a single HTML page.

An internal CSS is defined in the `<head>` section of an HTML page, within a `<style>` element.

The following example sets the text color of ALL the `<h1>` elements (on that page) to blue, and the text color of ALL the `<p>` elements to red. In addition, the page will be displayed with a “powderblue” background color:

```
<!DOCTYPE html>
<html>
<head>
<style>
body {background-color: powderblue;}
h1  {color: blue;}
p  {color: red;}
</style>
</head>
<body>

<h1>This is a heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

7.5. External CSS

An external style sheet is used to define the style for many HTML pages.

To use an external style sheet, add a link to it in the `<head>` section of each HTML page:

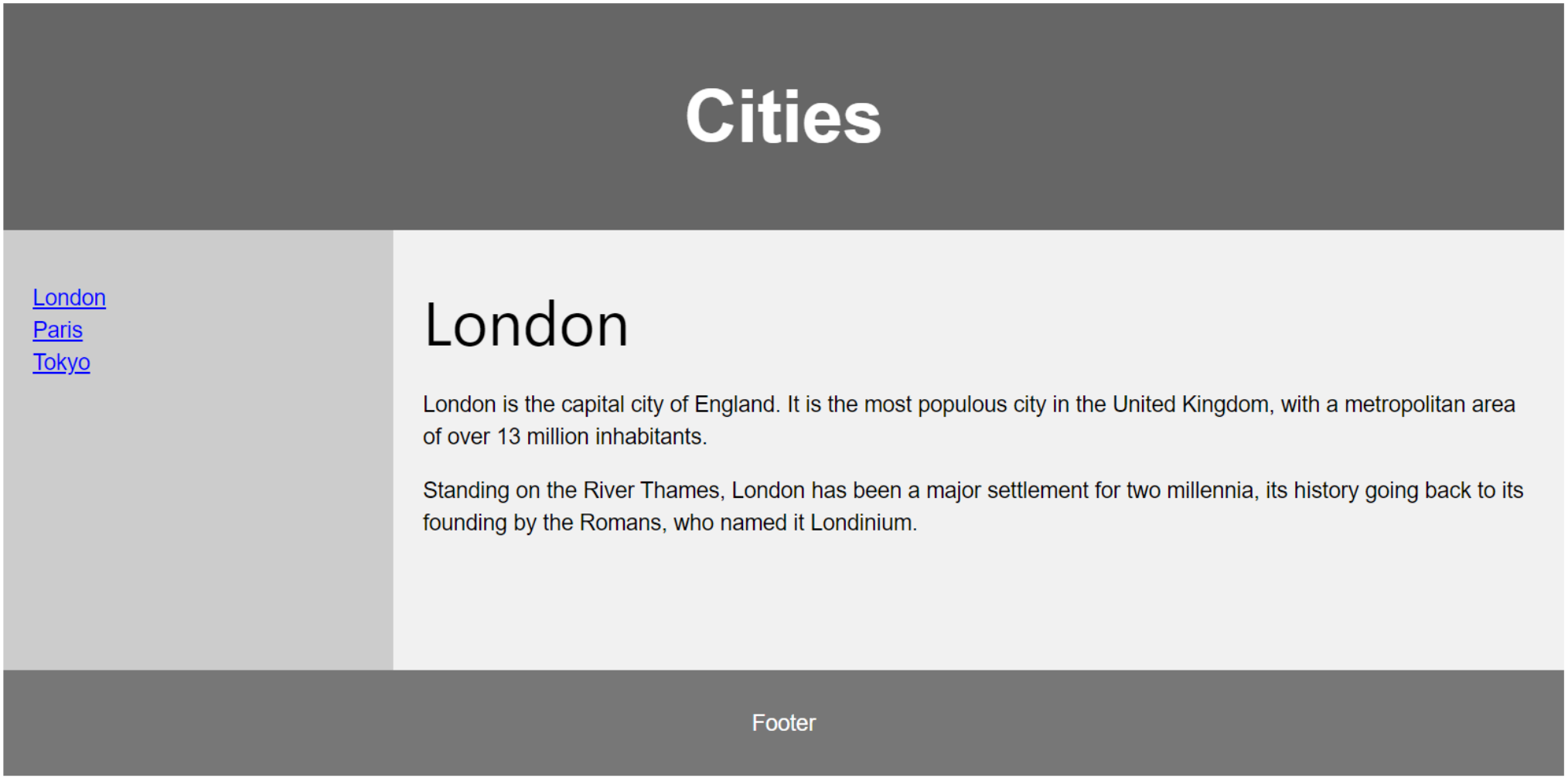
```
<!DOCTYPE html>
<html>
<head>
  <link rel="stylesheet" href="styles.css">
</head>
<body>

<h1>This is a heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

styles.css

```
body {
  background-color: powderblue;
}
h1 {
  color: blue;
}
p {
  color: red;
}
```

8. HTML Layout

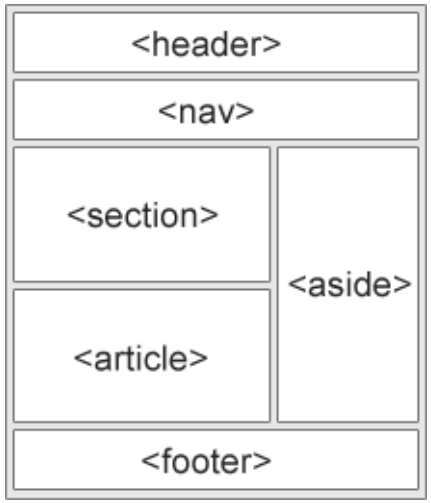
Websites often display content in multiple columns (like a magazine or a newspaper).



8.1. HTML Layout Elements

HTML has several semantic elements that define the different parts of a web page:

- <header> - Defines a header for a document or a section
- <nav> - Defines a set of navigation links
- <section> - Defines a section in a document
- <article> - Defines an independent, self-contained content
- <aside> - Defines content aside from the content (like a sidebar)
- <footer> - Defines a footer for a document or a section
- <details> - Defines additional details that the user can open and close on demand
- <summary> - Defines a heading for the <details> element



8.2. HTML Layout Techniques

There are four different techniques to create multicolumn layouts. Each technique has its pros and cons:

- CSS framework
- CSS float property
- CSS flexbox
- CSS grid

8.3. CSS Frameworks

If you want to create your layout fast, you can use a CSS framework, like **W3.CSS** or **Bootstrap**.

8.4. CSS Float Layout

It is common to do entire web layouts using the CSS `float` property. Float is easy to learn - you just need to remember how the `float` and `clear` properties work.

Disadvantages: Floating elements are tied to the document flow, which may harm the flexibility.

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>CSS Template</title>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<style>
* {
  box-sizing: border-box;
}

body {
  font-family: Arial, Helvetica, sans-serif;
}

/* Style the header */
header {
  background-color: #666;
  padding: 30px;
  text-align: center;
```

```
  font-size: 35px;
  color: white;
}

/* Create two columns/boxes that floats next to each other */
nav {
  float: left;
  width: 30%;
  height: 300px; /* only for demonstration, should be removed */
  background: #ccc;
  padding: 20px;
}

/* Style the list inside the menu */
nav ul {
  list-style-type: none;
  padding: 0;
}

article {
  float: left;
  padding: 20px;
  width: 70%;
  background-color: #f1f1f1;
  height: 300px; /* only for demonstration, should be removed */
}
```

```
/* Clear floats after the columns */
section:after {
  content: "";
  display: table;
  clear: both;
}

/* Style the footer */
footer {
  background-color: #777;
  padding: 10px;
  text-align: center;
  color: white;
}

/* Responsive layout - makes the two columns/boxes stack on top of each other instead of next to each other, on small screens */
@media (max-width: 600px) {
  nav, article {
    width: 100%;
    height: auto;
  }
}
</style>
</head>
<body>

<h2>CSS Layout Float</h2>
```

```
<h2>CSS Layout Float</h2>
<p>In this example, we have created a header, two columns/boxes and a footer. On smaller screens, the columns will stack on top of each other.</p>
<p>Resize the browser window to see the responsive effect (you will learn more about this in our next chapter - HTML Responsive.)</p>

<header>
  <h2>Cities</h2>
</header>

<section>
  <nav>
    <ul>
      <li><a href="#">London</a></li>
      <li><a href="#">Paris</a></li>
      <li><a href="#">Tokyo</a></li>
    </ul>
  </nav>

  <article>
    <h1>London</h1>
    <p>London is the capital city of England. It is the most populous city in the  United Kingdom, with a metropolitan area of over 13 million inhabitants.</p>
    <p>Standing on the River Thames, London has been a major settlement for two millennia, its history going back to its founding by the Romans, who named it Londi
  </article>
</section>

<footer>
  <p>Footer</p>
</footer>
```

```
</body>
</html>
```

8.5. CSS Flexbox Layout

Use of flexbox ensures that elements behave predictably when the page layout must accommodate different screen sizes and different display devices.

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>CSS Template</title>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<style>
* {
  box-sizing: border-box;
}

body {
  font-family: Arial, Helvetica, sans-serif;
}

/* Style the header */
header {
  background-color: #666;
  padding: 30px;
  text-align: center;
  font-size: 35px;
  color: white;
```

```
}

/* Container for flexboxes */
section {
  display: -webkit-flex;
  display: flex;
}

/* Style the navigation menu */
nav {
  -webkit-flex: 1;
  -ms-flex: 1;
  flex: 1;
  background: #ccc;
  padding: 20px;
}

/* Style the list inside the menu */
nav ul {
  list-style-type: none;
  padding: 0;
}

/* Style the content */
article {
  -webkit-flex: 3;
  -ms-flex: 3;
```

```
  flex: 3;
  background-color: #f1f1f1;
  padding: 10px;
}

/* Style the footer */
footer {
  background-color: #777;
  padding: 10px;
  text-align: center;
  color: white;
}

/* Responsive layout - makes the menu and the content (inside the section) sit on top of each other instead of next to each other */
@media (max-width: 600px) {
  section {
    -webkit-flex-direction: column;
    flex-direction: column;
  }
}
</style>
</head>
<body>

<h2>CSS Layout Flexbox</h2>
<p>In this example, we have created a header, two columns/boxes and a footer. On smaller screens, the columns will stack on top of each other.</p>
<p>Resize the browser window to see the responsive effect.</p>
```

```
<p><strong>Note:</strong> Flexbox is not supported in Internet Explorer 10 and earlier versions.</p>

<header>
  <h2>Cities</h2>
</header>

<section>
  <nav>
    <ul>
      <li><a href="#">London</a></li>
      <li><a href="#">Paris</a></li>
      <li><a href="#">Tokyo</a></li>
    </ul>
  </nav>

  <article>
    <h1>London</h1>
    <p>London is the capital city of England. It is the most populous city in the  United Kingdom, with a metropolitan area of over 13 million inhabitants.</p>
    <p>Standing on the River Thames, London has been a major settlement for two millennia, its history going back to its founding by the Romans, who named it Londi
  </article>
</section>

<footer>
  <p>Footer</p>
</footer>

</body>
```

```
</html>
```

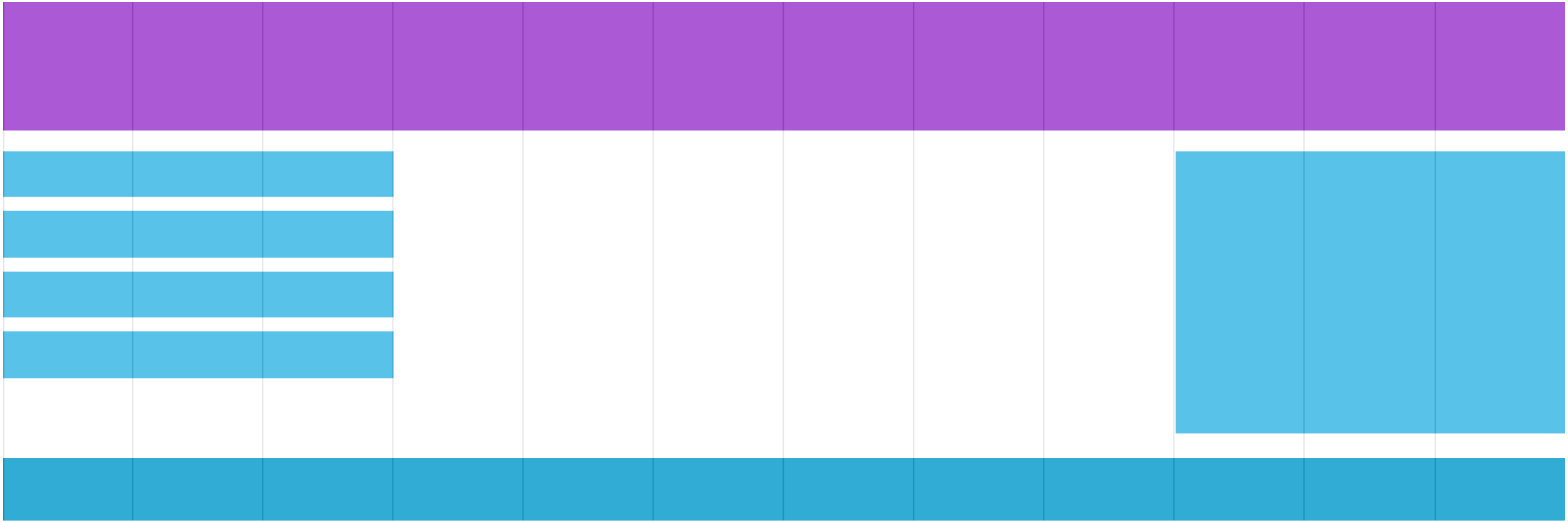
8.6. CSS Grid Layout

The CSS Grid Layout Module offers a grid-based layout system, with rows and columns, making it easier to design web pages without having to use floats and positioning.

Many web pages are based on a grid-view, which means that the page is divided into columns:

Using a grid-view is very helpful when designing web pages. It makes it easier to place elements on the page.

HTML Layout



A responsive grid-view often has 12 columns, and has a total width of 100%, and will shrink and expand as you resize the browser window.

```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<style>
* {
  box-sizing: border-box;
}

.row::after {
  content: "";
```

```
  clear: both;
  display: table;
}

[class*="col-"] {
  float: left;
  padding: 15px;
}

.col-1 {width: 8.33%;}
.col-2 {width: 16.66%;}
.col-3 {width: 25%;}
.col-4 {width: 33.33%;}
.col-5 {width: 41.66%;}
.col-6 {width: 50%;}
.col-7 {width: 58.33%;}
.col-8 {width: 66.66%;}
.col-9 {width: 75%;}
.col-10 {width: 83.33%;}
.col-11 {width: 91.66%;}
.col-12 {width: 100%;}

html {
  font-family: "Lucida Sans", sans-serif;
}

.header {
```

```
  background-color: #9933cc;
  color: #ffffff;
  padding: 15px;
}

.menu ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
}

.menu li {
  padding: 8px;
  margin-bottom: 7px;
  background-color: #33b5e5;
  color: #ffffff;
  box-shadow: 0 1px 3px rgba(0,0,0,0.12), 0 1px 2px rgba(0,0,0,0.24);
}

.menu li:hover {
  background-color: #0099cc;
}
</style>
</head>
<body>

<div class="header">
```

```
  <h1>Chania</h1>
</div>

<div class="row">
  <div class="col-3 menu">
    <ul>
      <li>The Flight</li>
      <li>The City</li>
      <li>The Island</li>
      <li>The Food</li>
    </ul>
  </div>

  <div class="col-9">
    <h1>The City</h1>
    <p>Chania is the capital of the Chania region on the island of Crete. The city can be divided in two parts, the old town and the modern city.</p>
    <p>Resize the browser window to see how the content respond to the resizing.</p>
  </div>
</div>

</body>
</html>
```

9. HTML Forms

An HTML form is used to collect user input. The user input is most often sent to a server for processing.

First name:
John
Last name:
Doe

Submit

9.1. The form Element

The HTML `<form>` element is used to create an HTML form for user input:

```
<form>
.
*form elements*
.
</form>
```

The `<form>` element is a container for different types of input elements, such as: text fields, checkboxes, radio buttons, submit buttons, etc.

9.2. The input Element

The HTML `<input>` element is the most used form element.

An `<input>` element can be displayed in many ways, depending on the `type` attribute.

Here are some examples:

Type	Description
`<input type="text">`	Displays a single-line text input field
`<input type="radio">`	Displays a radio button (for selecting one of many choices)
`<input type="checkbox">`	Displays a checkbox (for selecting zero or more of many choices)
`<input type="submit">`	Displays a submit button (for submitting the form)
`<input type="button">`	Displays a clickable button

Thank you

全国统一咨询热线：400-690-6115

北京|上海|广州|深圳|天津|成都|重庆|武汉|济南|青岛|杭州|西安