**1.CSS**

Inline

Internal:

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

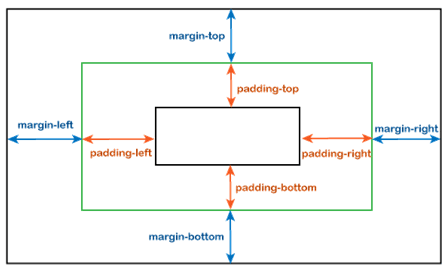
External:

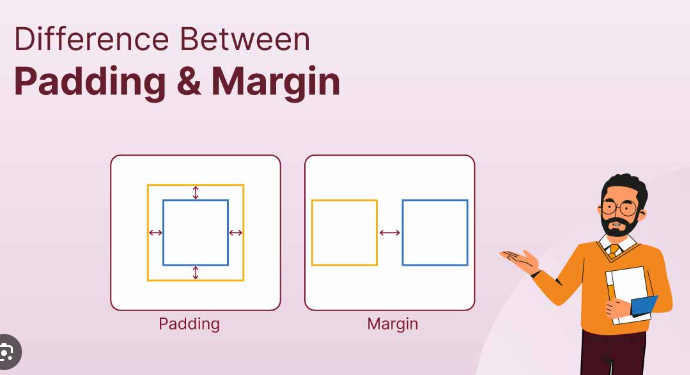
<head>  
  <link rel="stylesheet" href="styles.css">  
</head>

color: blue;  
  font-family: verdana;  
  font-size: 300%;

border: 2px solid powderblue;  
  padding: 30px;

margin: 50px;





**2.Link**

<a href=<https://www.w3schools.com/> title="Go to W3Schools HTML section">Visit W3Schools.com!</a>

->title can seen as a tooltip

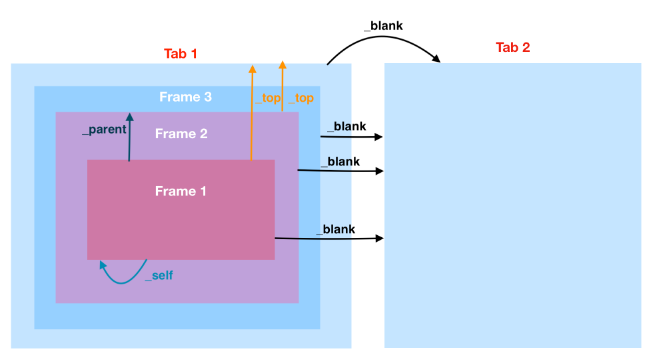
- Link to a page located in the html folder on the current web site:

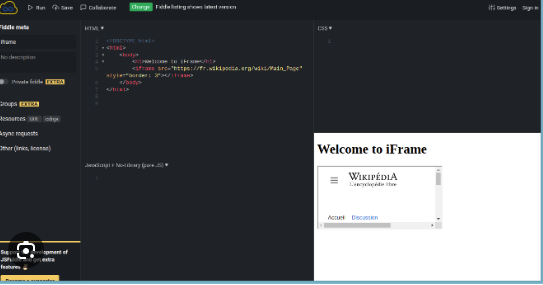
<a href="/html/default.asp">HTML tutorial</a>

- Link to a page located in the same folder as the current page:

<a href="default.asp">HTML tutorial</a>

* \_self - Default. Opens the document in the same window/tab as it was clicked
* \_blank - Opens the document in a new window or tab
* \_parent - Opens the document in the parent frame
* \_top - Opens the document in the full body of the window





 **HTTP** is suitable for non-sensitive content where security is not a major concern.

 **HTTPS** is crucial for protecting sensitive information and ensuring data security and integrity. It is now considered a best practice for all websites, not just those handling sensitive data, due to the increasing emphasis on security and user trust.

* https takes more time and produces latency, so it’s better to avoid using https when you don’t need it. Relative URLs are better in these two ways (stated below), but not in all cases.

**-scheme switching**

When you use absolute URLs like http://yourdomain.example/images/example.png and switch to HTTPS, browsers will often block these mixed-content requests. Mixed content occurs when an HTTPS page requests resources over HTTP, which can lead to security warnings or even the resources being blocked altogether.

**Example:** If your site is served over HTTPS, but your images are linked via HTTP, browsers may display warnings or fail to load those images. This is because HTTPS expects all associated resources to also be served over HTTPS to maintain security.

**Solution:** Use protocol-relative URLs (e.g., //yourdomain.example/images/example.png). This way, the resource will automatically use the same protocol as the page itself, whether HTTP or HTTPS. However, note that protocol-relative URLs are becoming less common, and some best practices recommend sticking with HTTPS URLs even for secure content.

**-domain name switching**

Using absolute URLs tied to a specific domain can cause issues when moving from a test environment (e.g., test.yourdomain.example) to a production environment (e.g., yourdomain.example). All the absolute URLs pointing to the test domain would need to be updated to reflect the new domain, which can be cumbersome and error-prone.

**Solution:** Relative URLs are more flexible and maintainable in this regard. For instance, instead of http://test.yourdomain.example/images/example.png, use /images/example.png. This way, the URLs are relative to the root of the current domain, and you only need to update the domain name in one place (your configuration) rather than in every URL.

**3.diverse Links**

<a href="default.asp"><img src="smiley.gif" alt="HTML tutorial" style="width:42px;height:42px;"></a>

* the image is a link – I love it haha—

<a href="mailto:someone@example.com">Send email</a>

* Use mailto: inside the href attribute to create a link that opens the user's email program (to let them send a new email):

Internal link example:

<style>  
a:link {  
  color: green;  
  background-color: transparent;  
  text-decoration: none;  
}  
  
a:visited {  
  color: pink;  
  background-color: transparent;  
  text-decoration: none;  
}  
  
a:hover {  
  color: red;  
  background-color: transparent;  
  text-decoration: underline;  
}  
  
a:active {  
  color: yellow;  
  background-color: transparent;  
  text-decoration: underline;  
}  
</style>

Button link:

<style>  
a:link, a:visited {  
  background-color: #f44336;  
  color: white;  
  padding: 15px 25px;  
  text-align: center;  
  text-decoration: none;  
  display: inline-block;  
}  
  
a:hover, a:active {  
  background-color: red;  
}  
</style>

**4.특수기호**

