Q: - How you define HTML Frames?

Using HTML Frames we can show more than one HTML document on the same browser.Where as each HTML document is called Frame.   
We use Frames using <frameset> tag.It is used to divide windows into frames.   
Using frame tag we put HTML document into the each frame.

Example:   
<frameset cols="25%,75%">   
<frame src="frame\_first.htm?phpMyAdmin=

70ac9566533a2665b6597346aab7f985&phpMyAdmin=f43d4e0b88acea2d2a393515f6bf38f2">   
<frame src="frame\_second.htm?phpMyAdmin=

70ac9566533a2665b6597346aab7f985&phpMyAdmin=f43d4e0b88acea2d2a393515f6bf38f2">   
</frameset>   
25% and 75% specifies the width of the page.Above frame told that frame\_first aquire the 25% area on the web browser and frame\_second aquire the 75% area on the same same web browser.

Q: - How you use Form's Action Attribute and Submit Button in HTML?

Using Submit Button we can sends the form to the server and Action Attribute is specified that which file we send.  
  
<form name="input" action="submit\_form.php"   
  
method="get"><input type="hidden" name="phpMyAdmin" value="f43d4e0b88acea2d2a393515f6bf38f2" /><input type="hidden" name="phpMyAdmin" value="70ac9566533a2665b6597346aab7f985" />   
  
Formname:   
  
<input type="text" name="Form">   
  
<input type="submit" value="Submit">   
  
</form>   
When we click the sumbit button web page than the submit\_form is send to the server.

Q: - Write some useful table tags that we use in HTML?

I have display the list of some tags that we used in HTML table.These tags are:   
<table>  This is started tag of table.All other   
         table tags works inside this tag.   
<th>     This is used to write header on table.   
<tr>     This is used to create row on table.   
<td>     This is used to create table cells.   
<caption>This is specifies the table caption.   
<colgroup>Specifies the group of columns in table.   
<col>    Using then we can set the attribute value   
         for one or more columns in table.   
<thead>  This is used to set table head.   
<tbody>  This is used to set table body.   
<tfoot>  This is used to set table border.

Q: - How to create tables in HTML?

We use <table> tag to create the table in HTML. After creating an table we use <tr> tag to create the rows.And use <td> to create data cell on each row.These cell can store images,text,tables,list etc.   
Basic syntax to create an Table are given below:   
<table border="1">   
<tr>   
<td>row 1, cell 1</td>   
<td>row 1, cell 2</td>   
<td>row 1, cell 3</td>   
</tr>   
<tr>   
<td>row 2, cell 1</td>   
<td>row 2, cell 2</td>   
<td>row 2, cell 3</td>   
</tr>   
</table>

Q: - What is a tag?

In HTML, a tag tells the browser what to do. When you write an HTML page, you enter tags for many reasons -- to change the appearance of text, to show a graphic, or to make a link to another page.

Q: - How comfortable are you with writing HTML entirely by hand?

Very. I don’t usually use WYSIWYG. The only occasions when I do use Dreamweaver are when I want to draw something to see what it looks like, and then I’ll usually either take that design and hand-modify it or build it all over again from scratch in code. I have actually written my own desktop HTML IDE for Windows (it’s called Less Than Slash) with the intention of deploying it for use in web development training. If has built-in reference features, and will autocomplete code by parsing the DTD you specify in the file. That is to say, the program doesn’t know anything about HTML until after it parses the HTML DTD you specified. This should give you some idea of my skill level with HTML.

Q: - What is everyone using to write HTML?

Everyone has a different preference for which tool works best for them. Keep in mind that typically the less HTML the tool requires you to know, the worse the output of the HTML. In other words, you can always do it better by hand if you take the time to learn a little HTML.

Q: - What is a DOCTYPE? Which one do I use?

According to HTML standards, each HTML document begins with a DOCTYPE declaration that specifies which version of HTML the document uses. Originally, the DOCTYPE declaration was used only by SGML-based tools like HTML validators, which needed to determine which version of HTML a document used (or claimed to use).   
Today, many browsers use the document's DOCTYPE declaration to determine whether to use a stricter, more standards-oriented layout mode, or to use a "quirks" layout mode that attempts to emulate older, buggy browsers.

Q: - How do I use forms?

The basic syntax for a form is: <FORM ACTION="[URL]">...</FORM>   
When the form is submitted, the form data is sent to the URL specified in the ACTION attribute. This URL should refer to a server-side (e.g., CGI) program that will process the form data. The form itself should contain  
  
\* at least one submit button (i.e., an <INPUT TYPE="submit" ...> element),  
\* form data elements (e.g., <INPUT>, <TEXTAREA>, and <SELECT>) as needed, and  
\* additional markup (e.g., identifying data elements, presenting instructions) as needed.

Q: - How can I check for errors?

HTML validators check HTML documents against a formal definition of HTML syntax and then output a list of errors. Validation is important to give the best chance of correctness on unknown browsers (both existing browsers that you haven't seen and future browsers that haven't been written yet).   
HTML checkers (linters) are also useful. These programs check documents for specific problems, including some caused by invalid markup and others caused by common browser bugs. Checkers may pass some invalid documents, and they may fail some valid ones.   
All validators are functionally equivalent; while their reporting styles may vary, they will find the same errors given identical input. Different checkers are programmed to look for different problems, so their reports will vary significantly from each other. Also, some programs that are called validators (e.g. the "CSE HTML Validator") are really linters/checkers. They are still useful, but they should not be confused with real HTML validators.   
When checking a site for errors for the first time, it is often useful to identify common problems that occur repeatedly in your markup. Fix these problems everywhere they occur (with an automated process if possible), and then go back to identify and fix the remaining problems.   
Link checkers follow all the links on a site and report which ones are no longer functioning. CSS checkers report problems with CSS style sheets.

Q: - Do I have to memorize a bunch of tags?

No. Most programs that help you write HTML code already know most tags, and create them when you press a button. But you should understand what a tag is, and how it works. That way you can correct errors in your page more easily.

Q: - How do I make a form so it can be submitted by hitting ENTER?

The short answer is that the form should just have one and no TEXTAREA, though it can have other form elements like checkboxes and radio buttons.

Q: - Can I have two or more actions in the same form?

No. A form must have exactly one action. However, the server-side (e.g., CGI) program that processes your form submissions can perform any number of tasks (e.g., updating a database, sending email, logging a transaction) in response to a single form submission.

Q: - How can I use forms for pull-down navigation menus?

There is no way to do this in HTML only; something else must process the form. JavaScript processing will work only for readers with JavaScript-enabled browsers. CGI and other server-side processing is reliable for human readers, but search engines have problems following any form-based navigation.

Q: - How can I avoid using the whole URL?

The URL structure defines a hierarchy (or relationship) that is similar to the hierarchy of subdirectories (or folders) in the filesystems used by most computer operating systems. The segments of a URL are separated by slash characters ("/"). When navigating the URL hierarchy, the final segment of the URL (i.e., everything after the final slash) is similar to a file in a filesystem. The other segments of the URL are similar to the subdirectories and folders in a filesystem.   
A relative URL omits some of the information needed to locate the referenced document. The omitted information is assumed to be the same as for the base document that contains the relative URL. This reduces the length of the URLs needed to refer to related documents, and allows document trees to be accessed via multiple access schemes (e.g., "file", "http", and "ftp") or to be moved without changing any of the embedded URLs in those documents.   
Before the browser can use a relative URL, it must resolve the relative URL to produce an absolute URL. If the relative URL begins with a double slash (e.g., //www.yoursite.com/faq/html/), then it will inherit only the base URL's scheme. If the relative URL begins with a single slash (e.g., /faq/html/), then it will inherit the base URL's scheme and network location.   
If the relative URL does not begin with a slash (e.g., all.html , ./all.html or ../html/), then it has a relative path and is resolved as follows.  
  
1. The browser strips everything after the last slash in the base document's URL and appends the relative URL to the result.  
2. Each "." segment is deleted (e.g., ./all.html is the same as all.html, and ./ refers to the current "directory" level in the URL hierarchy).  
3. Each ".." segment moves up one level in the URL hierarchy; the ".." segment is removed, along with the segment that precedes it (e.g., foo/../all.html is the same as all.html, and ../ refers to the parent "directory" level in the URL hierarchy).   
  
Some examples may help make this clear. If the base document is <URL:http://www.yoursite.com/faq/html/basics.html>, then  
  
all.html and ./all.html  
refer to <URL:http://www.yoursite.com/faq/html/all.html>   
./  
refers to <URL:http://www.yoursite.com/faq/html/>   
../  
refers to <URL:http://www.yoursite.com/faq/>   
../cgifaq.html  
refers to <URL:http://www.yoursite.com/faq/cgifaq.html>   
../../reference/  
refers to <URL:http://www.yoursite.com/reference/>   
  
Please note that the browser resolves relative URLs, not the server. The server sees only the resulting absolute URL. Also, relative URLs navigate the URL hierarchy. The relationship (if any) between the URL hierarchy and the server's filesystem hierarchy is irrelevant.

Q: - Can I use percentage values for

The HTML 3.2 and HTML 4.0 specifications allow only integer values (representing a number of pixels) for the WIDTH attribute of the TD element. However, the HTML 4.0 DTD allows percentage (and other non-integer) values, so an HTML validator will not complain about <TD WIDTH="xx%">.   
It should be noted that Netscape and Microsoft's browsers interpret percentage values for <TD WIDTH=...> differently. However, their interpretations (and those of other table-aware browsers) happen to match when combined with <TABLE WIDTH="100%">. In such situations, percentage values can be used relatively safely, even though they are prohibited by the public specifications.

Q: - Why is there extra space before or after my table?

This is often caused by invalid HTML syntax. Specifically, it is often caused by loose content within the table (i.e., content that is not inside a TD or TH element). There is no standard way to handle loose content within a table. Some browsers display all loose content before or after the table. When the loose content contains only multiple line breaks or empty paragraphs, then these browsers will display all this empty space before or after the table itself.   
The solution is to fix the HTML syntax errors. All content within a table must be within a TD or TH element.

9964593682