HTML Table Tutorial

Basic Table Tags

The three most important tags for tables is the opening table tag, <table> and the table row and table data tags - <tr> and <td> respectively.

The <tr> tag represents a row for the table

The <td> tag represents a cell (or column) inside the row.

Now, with that in mind, let's create a simple table:

```
<table bordercolor="#ff00ff">
	<tr>

		<td>A</td>
		<td>B</td>
		<td>C</td>
	</tr>

	<tr>
		<td>X</td>
		<td>Y</td>
		<td>Z</td>
	</tr>
</table>
```

And this is what the table would look like published:

A B C

X Y Z

Notice that by looking at the code, you can tell how many rows and columns are included just by looking at the code. The two opening <tr> tags indicate two rows and the three opening <td> tags on each line represents three data cells (or three columns).

So if you wanted to add another row, you would just start with another <tr> and so forth....

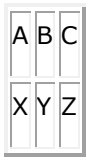
Adding Table Borders

Adding a border simply involves inserting the border attribute to the opening table tag. So in the above table the code would be adjusted accordingly:

```
<table border="2">
<tr>
<td>A</td>
<td>B</td>
<td>C</td>
</tr>

<tr>
<td>X</td>
<td>Y</td>
<td>Z</td>
</tr>
</table>
```

Notice the "2" represents the **thickness** of the border. If you had set it to "0" then there would have been no border at all. If you wanted it very thick then you could set it to 8, for example. So now your table will look like this:



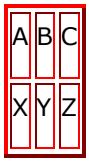
Changing a Table's Border Color

You can change the color of a table border by simply adding the **bordercolor** attribute.

```
<table border="2" bordercolor="red">
<tr>
<td>A</td>
<td>B</td>
<td>C</td>
</tr>

<tr>
<td>X</td>
<td>Y</td>
<td>Z</td>
</tr>
</table>
```

And here's what it would look like...



Adjusting Table Cell Spacing and Cell Padding

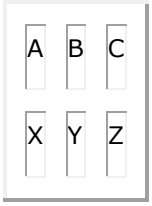
You can increase the space within the table cells and the space between the cells by using the **cellpadding**and **cellspacing** attributes.

The **cellspacing** attribute adjusts the space between the cells and **cellpadding** adjusts the space within (around) the cell.

Ex:----

```
<table border="2" cellspacing="10" cellpadding="3">
<tr><td>A</td>  <td>B</td>  <td>C</td></tr>
<tr><td>X</td>  <td>Y</td>  <td>Z</td></tr>
</table>
```

This is what the table would look like now....



See how setting the cellspacing attribute to "10" drastically increased the spacing between the cells, and the cellpadding attribute set to "3" added a little of space within each individual cell.

If you want a table to have a single border (with no border around the letters), simply set the cellspacing to "0" and your table will look like this....

A	B	C
X	Y	Z

Specifying a Table Width

You can specify the width of a table by using either a percentage or a pixel width.

```
<table width="100%" border="2">
<tr><td>A</td>  <td>B</td>  <td>C</td></tr>
<tr><td>X</td>  <td>Y</td>  <td>Z</td></tr>
</table>
```

Since the width is set to 100% that means the table will take up 100% of the screen and the columns in the table will be adjusted evenly. Here's what it would look like.

A	B	C
X	Y	Z

As we mentioned, you can also set the table width using pixels instead of percentages. So instead of setting it to 100%, you could set it to 300 pixels:

```
<table width="300" border="2">
<tr><td>A</td>  <td>B</td>  <td>C</td></tr>
<tr><td>X</td>  <td>Y</td>  <td>Z</td></tr>
</table>
```

The table would look like this:

A	B	C
X	Y	Z

Setting Column Widths

Sometimes you may not always want your columns to be the same width. If this is the case, you need to set values on your table data <td> cells. Again, you can set them by using percentages or pixel widths.

```
<table width="300" border="2">
<tr>
<td width="70%">A</td>
<td>B</td>
<td>C</td>
</tr>

<tr>
<td width="70%">X</td>
<td>Y</td>
<td>Z</td>
</tr>
</table>
```

This is what this table would look like.

A	B	C
X	Y	Z

See how the column width for the first column in both rows is set to 70%. Notice there is no value set for the other 2 columns. If you do not set a value for the remaining columns, their width will automatically be adjusted to take up the remaining space and they'll share it equally.

Since the table width is set to 300 pixels, and the first column is instructed to take up 70% of those 300 pixels (roughly 210 pixels), the other 2 columns divide the remaining 30% of the table equally (roughly 45 pixels a piece).

You could also have expressed the column widths of this table in pixels instead of percentages. The code would have looked like this:

```
<table width="300" border="2">
<tr>
<td width="210">A</td>
<td width="45">B</td>
<td width="45">C</td>
</tr>
<tr>
<td width="210">A</td>
<td width="45">B</td>
<td width="45">C</td>
</tr>
</table>
```

A	B	C
A	B	C

See how the width of the columns in each row add up to 300 (210 + 45 + 45) -- which is the width of the table.

What's the Difference Between Using Percentages and Pixel Widths?

Many people prefer to express their table width and column widths in percentages because that will ensure that the table takes up the same amount of screen no matter how big or small the screen resolution is.

If someone is using a 21 inch monitor to view your site and you have a table width set to 300 pixels, the table will show up very small on their screen. However if you set the table width to 70%, it will take up 70% of the screen no matter what size monitor the person is using.

Or, let's say you create a table that is 760 pixels wide and someone has a 15 inch monitor set to a 640x480 resolution, then that person will have to scroll left and right just to see the entire table.

On the other hand, if you set the table width to 100%, the table will fit the screen no matter what resolution the browser is set to.

So it's really up to you to decide what's the best layout for your tables. Personally I prefer to use percentages. That way the table takes up the same amount of screen no matter what size monitor/resolution people are using.

Specifying a Table's Height

You can also set the table height by adding the height tag to the table code.

```
<table height="250" width="300" border="2">
```

```
<tr>
<td width="210">A</td>
<td width="45">B</td>
<td width="45">C</td>
</tr>
<tr>
<td width="210">A</td>
<td width="45">B</td>
<td width="45">C</td>
</tr>
</table>
```

Which will produce the following table:

A	B	C
A	B	C

Horizontally Aligning the Content Inside Tables

The content inside a cell is left aligned by default, but you can also center or right align the text as well by using the align attributes.

```
<table width="300" border="2">
<tr>
<td width="210" align="center">A</td>
<td width="45">B</td>
<td width="45">C</td>
</tr>
<tr>
<td width="210" align="center">A</td>
<td width="45">B</td>
<td width="45">C</td>
</tr>
</table>
```

A	B	C
A	B	C

See how the first column is aligned to the center? You can also right align the columns by using the align="right" inside the <td> cells.

Vertically Aligning the Content Inside the Table Cells

So far we've kept the table cell alignment at the default, which is the middle. Notice in the above table that the A, B, and C are all three aligned in the middle of their cells. Well you can change their alignment to either top, bottom, or middle by using the **valign**(which stands for vertical align) tag:

```
<table height="250" width="300" border="2">
<tr>
<td valign="top" width="210">A</td>
<td width="45">B</td>
<td width="45">C</td>
</tr>
<tr>
<td valign="top" width="210">A</td>
<td width="45">B</td>
```

<td width="45">C</td>
</tr>
</table>

A	B	C
A	B	C

I've set the table height to "250" so the alignment would be more noticeable. Notice that the A in both rows are aligned to the top. You can also align to the "bottom" or the "middle".

Creating a Left Navigation Layout With Tables

As we mentioned earlier, most left and right navigations are created using tables. All you do is create a table with one row, two columns and no border. Then align both of your columns to the top (using the **valign** tag) so your text will start in the top of the columns, not the middle. Then depending on if you're going to have a right or left navigation, you'll make one column significantly smaller than the other.

Here's a simple left navigation layout:

<table width="100%" border="0">
<tr>
<td valign="top" width="**25%**">Left Nav Links Here</td>
<td valign="top" width="**75%**">Body Here</td>
</tr>
</table>

And here's what it would look like:

Left Nav Links Here	Body Here

Notice I set the border to "0" but it's still showing in the example. I just did that to show how the layout would look. If you set your border to "0" you won't see any lines around your table.

I. Introduction

You may want to consider using HTML tables in your website. In addition to creating HTML tables to present data in rows and columns, you can also create HTML tables to organize information on your web page.

The process of creating an HTML table is similar to the process that you used to create your web page and any elements that you may have already included in your page, such as links or frames. Coding HTML tables into your web page is fairly easy since you need only understand a few basic table codes.

II. Creating a basic table

The basic structure of an HTML table consists of the following tags:

- Table tags: **<TABLE></TABLE>**
- Row tags: **<TR></TR>**
- Cell tags: **<TD></TD>**

Constructing an HTML table consists of describing the table between the beginning table tag, **<TABLE>**, and the ending table table tag, **</TABLE>**. Between these tags, you then construct each row and each cell in the row. To do this, you would first start the row with the beginning row tag, **<TR>**, and then build the row by creating each cell with the beginning cell tag, **<TD>**, adding the data for that cell, and then closing the cell with the ending cell tag, **</TD>**. When you finish all of the cells for a row, you would then close the row with the

ending row tag, **</TR>**.Then, for each new row, you would repeat the process of beginning the row, building each cell in the row, and closing the row.

The following table is an example of a basic table with three rows and two columns of data.

Data 1 Data 2

Data 3 Data 4

Data 5 Data 6

The codes that generated this table look like this:

```
<TABLE>
  <TR>
    <TD>Data 1</TD>
    <TD>Data 2</TD>
  </TR>
  <TR>
    <TD>Data 3</TD>
    <TD>Data 4</TD>
  </TR>
  <TR>
    <TD>Data 5</TD>
    <TD>Data 6</TD>
  </TR>
</TABLE>
```

This table contains no border, title, or headings. If you wish to add any of these elements to your table, you need to include additional HTML codes. The codes for these elements are explained in the next section.

III. Adding a border, title, and headings

In addition to the basic table tags, several options are available for adding additional elements to your table. For example, if you add a border, title, and column headings to the table in the previous section, the table would then resemble the following:

TABLE TITLE	
Column A	**Column B**
PHP	ASP.net
C# CUI	C# GUI

The following codes generated the border, TABLE TITLE, and Column A and Column B headings for this table:

```
<html>
<body>
<TABLE  BORDER="5">
<TR>
<TH COLSPAN="2">
<H3><BR>TABLE TITLE</H3>
</TH>
</TR>

<tr>
```

```
<TH>Column A</TH>
<TH>Column B</TH>
</tr>

<tr>
<Td>PHP</Td>
<Td>ASP.net</Td>
</tr>

<tr>
<Td>C# CUI</Td>
<Td>C# GUI</Td>
</tr>
</table>
</body>
</html>
```

Note: If you wish to view the codes that generated the Data 1 through Data 6 cells, refer to the previous section.

Notice that the beginning table tag, **<TABLE>**, now includes the border tag, **BORDER="5"**, which places a border around the table and frames each cell. The number that you ascribe to the border tag, **BORDER=***n*, sets the width of the table border. Depending on how you design your table, you can then determine the border size that best suits your table and the overall design of your web page.

To add a title to your table, you would place the title and the attributes of that title between the row commands, **<TR>** and **</TR>**. The heading codes, **<TH>** and **</TH>**, define a heading cell and, by default, these codes center the heading and set it in bold type. However, if you want the title to span across the columns below it, you need to include the **COLSPAN=***n* code. Since this table has two columns, the **COLSPAN="2"** code was necessary. To add emphasis to the header, you can use the header commands to make the text larger. In this table, notice that the **<H3>** and **</H3>** commands made the title larger. Finally, the **
** tag created a space above the title.

The individual column headings are also described by the heading codes, **<TH>** and **</TH>**. Since these codes, by default, center the heading and set it in bold type, no additional commands or attributes were included in the heading commands.

IV. Polishing your table

To give your table a more polished look, you can include commands that will adjust the size of your table, add space in the cell, add space between rows, and align the data in a cell. Working with these commands is basically a process of trial and error to create the most appealing presentation of your information. The type of table that you create and the overall design of your web site will help you determine what works best for your table.

Some of the commands that enable you to customize your table include:

- The **WIDTH=***n***%** command sets the width of your table as a percentage of the screen. The letter *n* designates the percentage that you assign to this command. For example, if you want the width of your table to be one half the width of the screen, you would include the **WIDTH="50%"** command in the beginning table command.

- The **CELLPADDING=***n* command adjusts the vertical dimension of the cells. The letter *n* designates the numerical value that you assign to this command.

- The **CELLSPACING=***n* command sets the space or border around the cells. The letter *n* designates the numerical value that you assign to this command.

- The **ALIGN=**(LEFT, RIGHT, or CENTER) command will horizontally align the data in a cell. For example, if you wish to place the data in the center of each cell in a row, you would include the **ALIGN=CENTER** command within the row command.

- The **VALIGN=**(TOP, MIDDLE, or BOTTOM) command will vertically align the data in a cell. For example, if you wish to place the data in the center of each cell in a row, you would include the **ALIGN=MIDDLE** command within the row command.

In addition to the codes that were explained in the previous sections, the table below now includes some of these commands.

TABLE TITLE	
Column A	**Column B**
Data 1	Data 2

The following codes, along with codes previously discussed, created this table:

```
<TABLE BORDER="5"   WIDTH="50%"   CELLPADDING="4" CELLSPACING="3">
  <TR>
    <TH COLSPAN="2"><BR><H3>TABLE TITLE</H3>
    </TH>
  </TR>


  <TR>
    <TH>Column A</TH>
    <TH>Column B</TH>
  </TR>


  <TR ALIGN="CENTER">
    <TD>Data 1</TD>
    <TD>Data 2</TD>
  </TR>
</TABLE>
```

Notice that the TABLE command now includes the **WIDTH="50%"** command. This command extends the table across one half of the width of the text. Also, the **CELLPADDING="4"** command increases the vertical dimension of the cells, and the **CELLSPACING="3"** command increases the border around the cells. Finally, the **ALIGN="CENTER"** command places Data 1 and Data 2 in the center of the cell.

COLSPAN and ROWSPAN

Table cells can span across more than one column or row. The attributes `COLSPAN` ("how many across") and `ROWSPAN` ("how many down") indicate how many columns or rows a cell should take up.

For example, we might want to create header cells for each department in our table of names and phone numbers. In this table, the header cells in the first and fifth rows span across two columns to indicate the department for each group of names.

```
<TABLE BORDER=2 CELLPADDING=4>
```

```
<TR><TH COLSPAN=2>Production</TH></TR>
<TR><TD>RahaMutisya</TD> <TD>1493</TD></TR>
<TR><TD>Shalom Buraka</TD><TD>3829</TD></TR>
<TR><TD>Brandy Davis</TD><TD>0283</TD></TR>

<TR><TH COLSPAN=2>Sales</TH></TR>
<TR><TD>Claire Horne</TD><TD>4827</TD></TR>
<TR><TD>Bruce Eckel</TD><TD>7246</TD></TR>
<TR><TD>Danny Zeman</TD><TD>5689</TD></TR>
</TABLE>
```

which gives us:

Production	
RahaMutisya	1493
Shalom Buraka	3829
Brandy Davis	0283
Sales	
Claire Horne	4827
Bruce Eckel	7246
Danny Zeman	5689

It often happens with multiple-column cells that a little color helps to set off the headers, giving the table a more visually organized look. Let's add some color to the headers using BGCOLOR.

```
<TABLE BORDER=2 CELLPADDING=4>
<TR><TH COLSPAN=2 BGCOLOR="#99CCFF">Production</TH></TR>
<TR><TD>RahaMutisya</TD><TD>1493</TD></TR>
<TR><TD>Shalom Buraka</TD><TD>3829</TD></TR>
<TR><TD>Brandy Davis</TD><TD>0283</TD></TR>

<TR><TH COLSPAN=2 BGCOLOR="#99CCFF">Sales</TH></TR>
<TR><TD>Claire Horne</TD><TD>4827</TD></TR>
<TR><TD>Bruce Eckel</TD><TD>7246</TD></TR>
<TR><TD>Danny Zeman</TD><TD>5689</TD></TR>
</TABLE>
```

which gives this table:

Production	
RahaMutisya	1493
Shalom Buraka	3829
Brandy Davis	0283
Sales	
Claire Horne	4827
Bruce Eckel	7246
Danny Zeman	5689

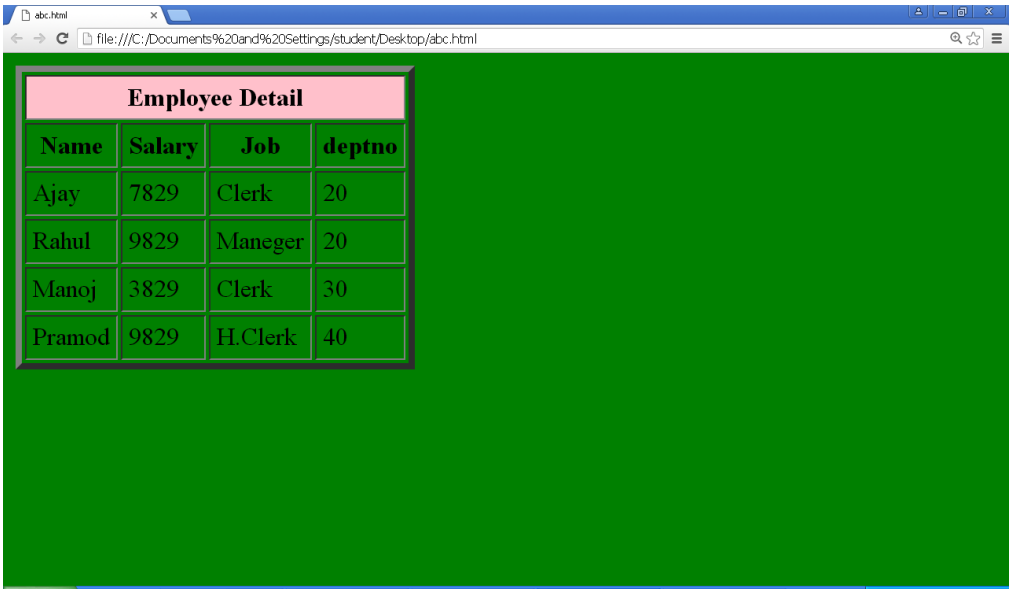
ROWSPAN sets how many rows a cell spans. ROWSPAN can get a little confusing because it requires you to think through how the cell affects the rows after the row it starts in. It's particularly useful in this situation to add borders to the table during the design process, even if the table won't ultimately use borders.

```
<html>
<body bgcolor="green">
<TABLE BORDER=4 CELLPADDING=4>
<TR><TH COLSPAN=4 bgcolor="pink">Employee Detail</TH></TR>

<TR><Th>Name</Th> <Th>Salary</Th> <th> Job</th> <th> deptno</th></TR>

<TR><TD>Ajay</TD><TD>7829</TD><td>Clerk</td> <td> 20</td></TR>
<TR><TD>Rahul</TD><TD>9829</TD><td>Maneger</td> <td> 20</td></TR>
<TR><TD>Manoj</TD><TD>3829</TD><td>Clerk</td> <td> 30</td></TR>
<TR><TD>Pramod</TD><TD>9829</TD><td>H.Clerk</td> <td> 40</td></TR>
</TABLE>
</body>
</html>
```

Output:----



This table code creates two header cells which span three rows each:

```
<TABLE BORDER=2 CELLPADDING=4>
<TR>
<TH ROWSPAN=3 BGCOLOR="#99CCFF">Production</TH>
<TD>RahaMutisya</TD><TD>1493</TD>
</TR>

<TR><TD>Shalom Buraka</TD><TD>3829</TD></TR>

<TR><TD>Brandy Davis</TD><TD>0283</TD></TR>

<TR>
<TH ROWSPAN=3 BGCOLOR="#99CCFF">Sales</TH>
<TD>Claire Horne</TD><TD>4827</TD>
```

```
</TR>

<TR><TD>Bruce Eckel</TD><TD>7246</TD></TR>

<TR><TD>Danny Zeman</TD><TD>5689</TD></TR>
</TABLE>
```

which creates

Production	Rahamat	1493
	Shalom	3829
	Brandy Davis	0283
Sales	Claire Horne	4827
	Bruce Eckel	7246
	Danny Zeman	5689

Note that in the two rows after each header, the first cell in the row ends up in the second column because the first column is taken up by the multi-column cell.

```
<TABLE BORDER=2 CELLPADDING=4>
<TR>

<TH ROWSPAN=3 BGCOLOR="#99CCFF">Production</TH>
<TD>RahaMutisya</TD><TD>1493</TD>
</TR>


<TR>
<TD>Shalom Buraka</TD><TD>3829</TD>
</TR>


<TR>
<TD>Brandy Davis</TD><TD>0283</TD>
</TR>


<TR>
<TH ROWSPAN=3 BGCOLOR="#99CCFF">Sales</TH>
<TD>Claire Horne</TD><TD>4827</TD>
</TR>


<TR>
<TD>Bruce Eckel</TD><TD>7246</TD>
</TR>


<TR>
<TD>Danny Zeman</TD><TD>5689</TD>
</TR>
```

</TABLE>

---

my first table	my first table

the code above gives us this

my first table	my first table

nowlets add another table row and put a single table definition in it

my first table	my first table
my first table	

the code above gives us this

my first table	my first table
my first table	

again we have a 'ghost cell' so we will add rowspan=2 to the first table definition in the first table row <TD rowspan=2>

my first table	my first table
	my first table

```
<TABLE BORDER=1>
<TR>
<TD rowspan=2>
my first table
</TD>
<TD>
my first table
</TD>
</TR>
<TR>
<TD>
my first table
</TD>
</TABLE>
```

below is the same table with rowspan=2 added to the second table definition in the first table row

my first table	my first table
my first table	

```
<TABLE BORDER=1>
<TR>
<TD>
my first table
</TD>
<TD rowspan=2>
my first table
</TD>
```

```
</TR>
<TR>
<TD>
my first table
</TD>
</TABLE>
```

this is the same table with some colors added because thats what we'll be doing soon

my first table	my first table
my first table	

as part of this rowspan tutorial you can use this example table to see if you can work out the html source code

Left	top	right
	center	
	bottom	

left top	top middle	right top
	center	right bottom
left bottom	bottom middle	
bottom table row		

- ```
  <table border="3" bordercolor="#c86260" bgcolor="#ffffcc" width="50%" cellspacing="5" cellpadding="3">

  <tr><td>MY DAILY MENU</td></tr>

  <tr><td valign="top">Breakfast</td><td>Orange juice<br>Toast<br>Black coffee</td></tr>

  <tr><td valign="top">Lunch</td><td>Tuna sandwich<br>Apple</td></tr>

  <tr><td valign="top">Dinner</td><td>Hamburger steak<br>Mashed potatoes<br>Green beans<br>Jello</td></tr>
  </table>
  ```

Example 1A - RESULT (first table row contains only one cell)

MY DAILY MENU	
Breakfast	Orange juice Toast Black coffee
  ```

Lunch	Tuna sandwich Apple
Dinner	Hamburger steak Mashed potatoes Green beans Jello

<~Missing table cell

This situation can be remedied using the `colspan` attribute whose value defines how many columns a table cell will span. Hence, a `<td>` tag using a `colspan="2"` attribute-value pair will specify that that particular cell will stretch over the equivalent of two columns worth of space:

Example 1B - SOURCE CODE (cell spans two columns)

```
<table border="3" bordercolor="#c86260" bgcolor="#ffffcc"
width="50%" cellspacing="5" cellpadding="3">
  <tr><td align="center" colspan="2">MY DAILY MENU</td></tr>

  <tr><td valign="top"><b>Breakfast</b></td><td>Orange
  juice<br>Toast<br>Black coffee</td></tr>

  <tr><td valign="top"><b>Lunch</b></td><td>Tuna
  sandwich<br>Apple</td></tr>

  <tr><td valign="top"><b>Dinner</b></td><td>Hamburger
  steak<br>Mashed potatoes<br>Green beans<br>Jello</td></tr>
</table>
```

Example 1B - RESULT (cell spans two columns)

MY DAILY MENU	
Breakfast	Orange juice Toast Black coffee
Lunch	Tuna sandwich Apple
Dinner	Hamburger steak Mashed potatoes Green beans Jello

Note that, as well as using `colspan="2"` in the 'MY DAILY MENU' table cell to make it span two columns, we also used `align="center"` to center the content.

- **`rowspan="number of rows"`** ~ The `rowspan` attribute works just like the `colspan` attribute except that you may find the situation a little more difficult to visualize when working with the source code. But once again the principle is the same. By using the `rowspan` attribute, you can force a table cell to span the number of rows specified by the respective value.

In the following example, the 'MONDAY' table cell spreads over three table rows:

```
   <table border="3" bordercolor="#c86260" bgcolor="#ffffcc"
   width="50%" cellspacing="5" cellpadding="3">
<tr><td align="center" colspan="3">MY DAILY MENU</td></tr>

<!--FIRST CELL IN THIS ROW SPANS THE NEXT 3 ROWS--><tr><td
rowspan="3">M<br>o<br>n<br>d<br>a<br>y</td><td
valign="top"><b>Breakfast</b></td><td>Orange
juice<br>Toast<br>Black coffee</td></tr>

<tr><td valign="top"><b>Lunch</b></td><td>Tuna
sandwich<br>Apple</td></tr>

<tr><td valign="top"><b>Dinner</b></td><td>Hamburger
steak<br>Mashed potatoes<br>Green beans<br>Jello</td></tr>

   </table>
```

Example 2 - RESULT (cell spans three rows)

MY DAILY MENU		
M o n d a y	**Breakfast**	Orange juice Toast Black coffee
	Lunch	Tuna sandwich Apple
	Dinner	Hamburger steak Mashed potatoes Green beans Jello