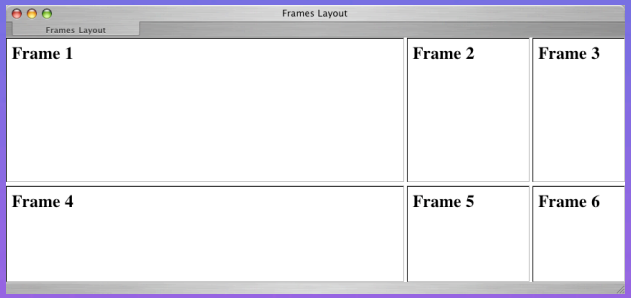
# Frames: Back to HTML

## What is a Frame?

Frames let you divide its main display window into independent window partitions or **frames**:

- Each simultaneously displaying a different document
- Something like a wall of monitors in a TV control room.




**JavaScript Loose End**: **Soon we will learn to control Frames with JavaScript**






# Frame Tags

HTML uses three new tags to comprise the frame document:

`frameset, frame, and noframes.`

- A <FRAMESET> is simply the collection of frames that make up the browser's window.
  - Column (COLS) and row (ROWS attributes for the frameset tag let you define the number and initial sizes for the columns and rows of frames.
  - The <FRAMESET> tag replaces the <BODY> tag

<FRAMESET> **and** <BODY> **tags MUST NOT BE MIXED**.






# Frame Tags (Cont.)

- The <FRAME> tag defines what document goes in the frames of the frameset
  - (Usually) HTML but could be otherwise.
  - You specify the (HTML) page to be loaded with the src attribute.
  - You may give the frame a name to use for **inter-frame document hypertext links — More Soon**.
  - You specify the link to be loaded with the name attribute.
  - Other attributes may be set, E.g. scrolling of individual frames.
- The <NOFRAMES> tag allows for browsers that can't process frames.

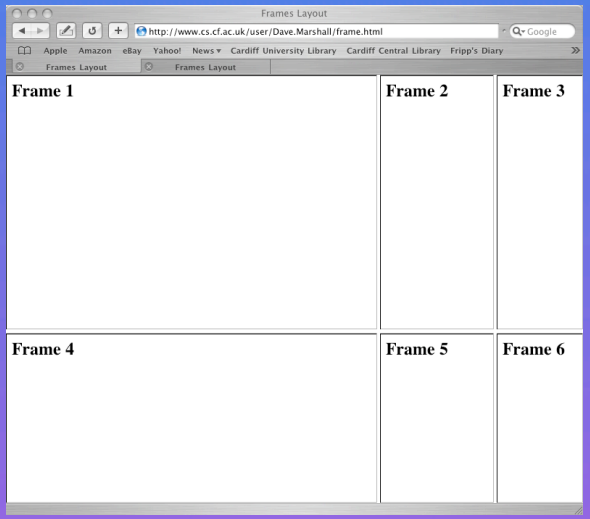





# Simple HTML Frame Example

Here is the HTML source that generated the six frame Web page, `frame.html`:

```
<html>
<head>
<title>Frames Layout</title>
</head>
<frameset rows="60%,*" cols="65%,20%,*">
  <frame src="frame1.html">
  <frame src="frame2.html">
  <frame src="frame3.html" name="fill_me">
  <frame scrolling=yes src="frame4.html">
  <frame src="frame5.html">
  <frame src="frame6.html">
  <noframes>
    Sorry, this document can be viewed only
    with browser Navigator version 2.0
    or later.
    <a href = "frame1.html">Take this link</a>
    to the first HTML document in the set.
  </noframes>
</frameset>
</html>
```









# Simple HTML Frame Example Explained

Notice a few things in the simple frame example

- The order in which browser fills the frames in a frameset is **row by row**.
- **Frame 4** sports a scrollbar because we told it to
  - Even though the contents may otherwise fit without scrolling.
  - Scrollbars **automatically appear** if the contents overflow the frame's dimensions, unless explicitly disabled with a special attribute `scrolling = no` in the frame tag.
  - The `name` attribute in **Frame 3** tags.
    - Once `name`d, you can reference a particular frame as where to display a hypertext-linked document.
    - To do that, you add a special `target` attribute to the anchor (`<a>`) tag of the source hypertext link.






## Example: Hypertext Linking to an Individual Frame

Staying with the basis of above example, with only slight changes, `new_frame.html`:

- We may wish, to link a document called **"new.html"**
- Display **"new.html"** in (our example) window **Frame 3**, which we've named `''fill_me''`,
- The HTML anchor would be contructed as follows:

```
<a href="new.html" target="fill_me">
```

- If the browser browser user chooses the link, say in **Frame 1**, the `new.html` document will replace the original `frame3.html` contents in frame three.






# New HTML Code for Frame Linking Example

new_frame1.html
(replaces frame1.html):

```
<HEAD>
<TITLE>Frames example - Link to target
</TITLE>
</HEAD>

<BODY>
<H2>New Frame 1 target window "3"
(target="fill_me")</H2>

<a href="new.html" target="fill_me">
Click Here to Fill Frame 3</a>

</BODY>
</HTML>
```

new.html
(replaces frame3.html when linked from new_frame1.html):

```
<HEAD>
<TITLE>Frames example - Link to targe
t</TITLE>
</HEAD>

<BODY>
<H2>New Frame 1 has linked target window "3"
(target="fill_me")</H2>

Frame 3 has changed.

</BODY>
</HTML>
```






# NOFRAMES Support

For reasons of

- Legacy — not all browsers may support frames (Not perhaps so critical as 5 years ago) and
- Some special needs (special rendering of pages)

You should provide <NOFRAMES> support:

- Each of your major frame documents should provide a backdoor to your HTML document collection with the <NOFRAMES> tag.
  - Frame-compatible browsers display your frames
  - Non-compatible browsers display the alternative NOFRAMES content.






## Frame Layout

### The `rows` and `cols` attributes

- Both `rows` and `cols` attributes accept

  **a double quote-enclosed, comma-separated list of values** (`''...''`) that specify either:

  - The **absolute** width (for columns) or height (for rows) of the frames or
  - The **relative** width (for columns) or height (for rows) of the frames.

- The **number** of attribute values determines how many **rows** or **columns** of frames browser will display in the document window.






# The rows and cols attributes (Cont.)

You express **each value** in the rows or cols attribute in one of three ways:

- As an **absolute** number of pixels,
- As a **percentage** of the total width or height of the frameset, or
- As a **portion** of the space remaining after setting aside room for adjacent elements






# The rows and cols attributes (Cont.)

## Determining the Size of the Frames

- The browser will **match** the size specifications you give a frameset **as closely as possible** to the current window size.

  This is clearly **User-controlled**.

- The browser will **not** extend the boundaries of the main document window to accommodate framesets that would otherwise exceed those boundaries or fill the window with empty space if the specified frames don't fill the window.
- Rather, browser **allocates space** to a particular frame **relative** to all other frames in the row and column and resolutely fills the entire document window.
- Notice a frame document window **does not** have scroll bars.





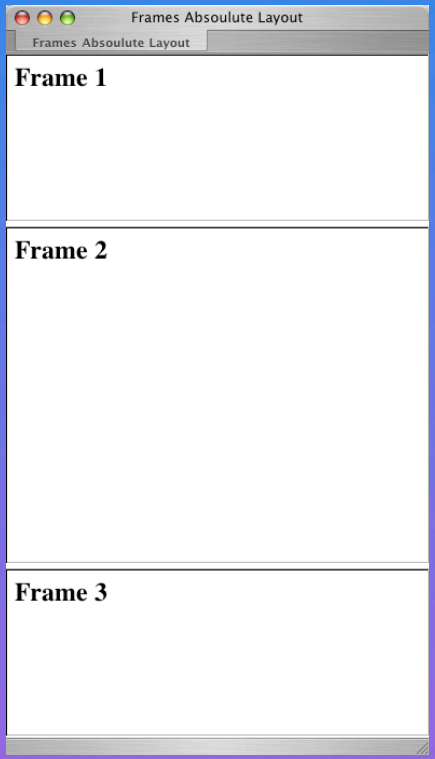

# Absolute Values Frame Size Example

For example, `frame_abs.html`:

```
<frameset rows="150,300,150">
```

- Creates three rows of frames, each extending across the entire document window.
- The first and last frames are set to 150 pixels tall, the second to 300 pixels.
- In reality, **unless** the browser window is exactly 600 pixels tall, browser automatically and proportionately stretches or compresses the first and last frames so that each occupies one quarter of the window space.

  The centre row occupies the remaining half of the window space.









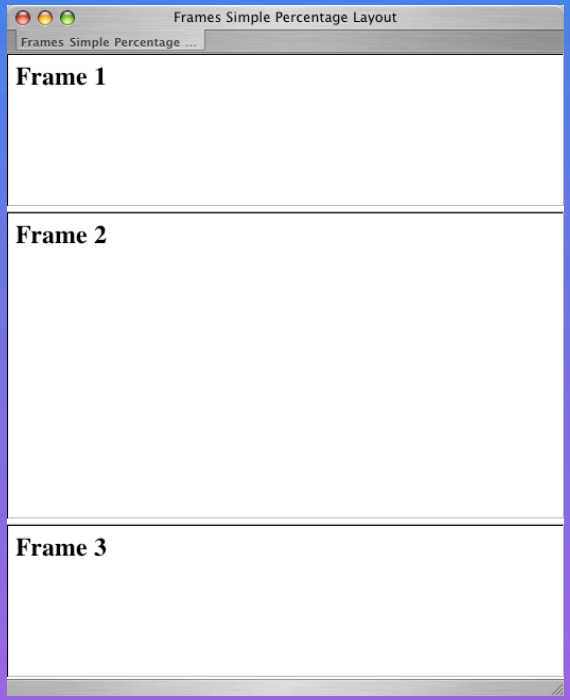
# Relative Values Frame Size Example

For example, frame_per.html:

- Frame-row and column-size values expressed as a percentage (%) of the window dimensions.
- For example, the following example is effectively identical to the previous absolute size:

```
<frameset rows="25%,50%,25%">
```

- If the percentages don't add up to 100 percent, the browser automatically and proportionally resizes each row to make up the difference.









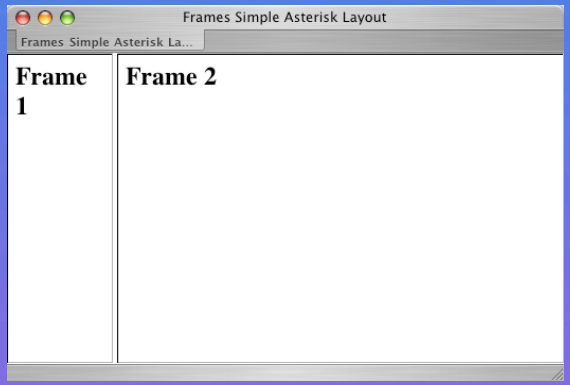
# The Wildcard Asterisk (*) Size Option

The asterisk (*) is a useful option for frameset `rows` and `cols` values:

- The browser sizes the respective column or row to whatever space is left over after putting adjacent frames into the frameset.
- For example, when browser encounters the frame tag, `frame_ast1.html`

```
<frameset cols="100,*">
```

This makes a fixed-sized column 100 pixels wide, and then creates another frame column that occupies all of the remaining space in the frameset.







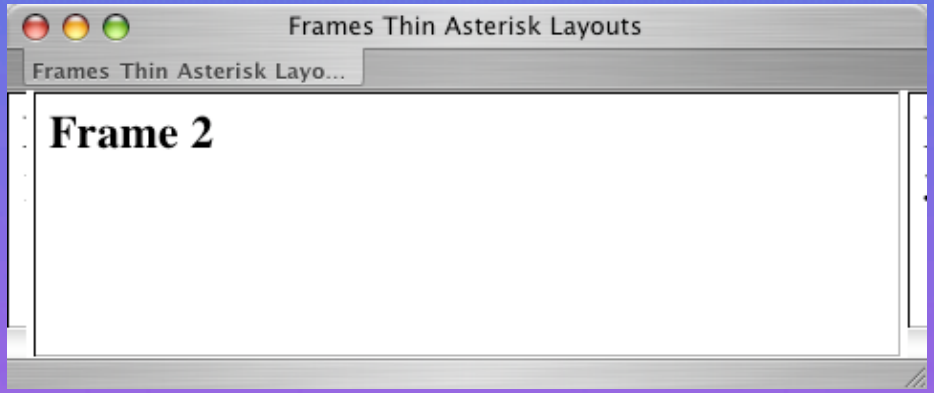


# Another Layout Example

Here's a another layout example, frame_ast2.html:

```
<frameset cols="10,*,10">
```

- This one creates two very thin columns down the edges of the frameset and gives the remaining centre portion to the middle column.



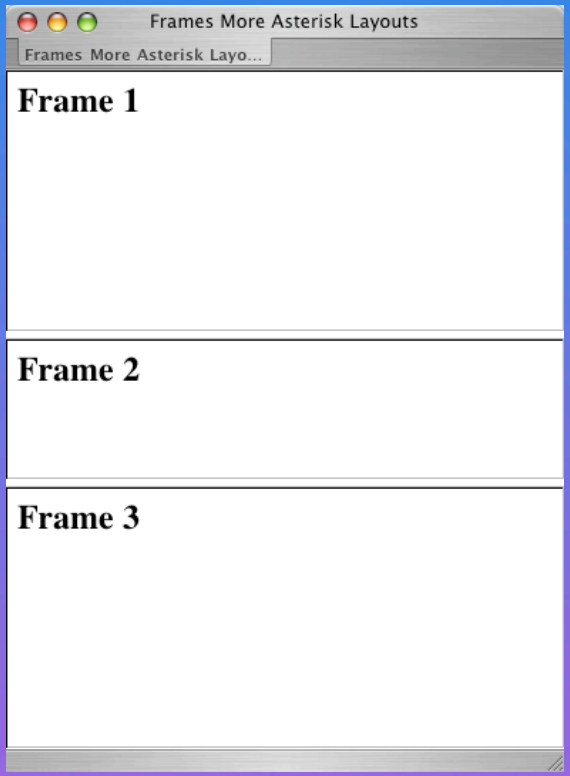
# Yet Another Layout Example

Here's a even fancier layout example

- You may also use the asterisk for more than one row- or column-attribute value. In that case, the corresponding rows or columns equally divide the available space.
- For example, frame_ast3.html:

```
<frameset rows="*,100,*">
```

This creates a 100-pixel tall row in the middle of the frameset and equal-sized rows above and below it.







# Multiple Asterisk Sizes

If you precede the asterisk with an **integer value**

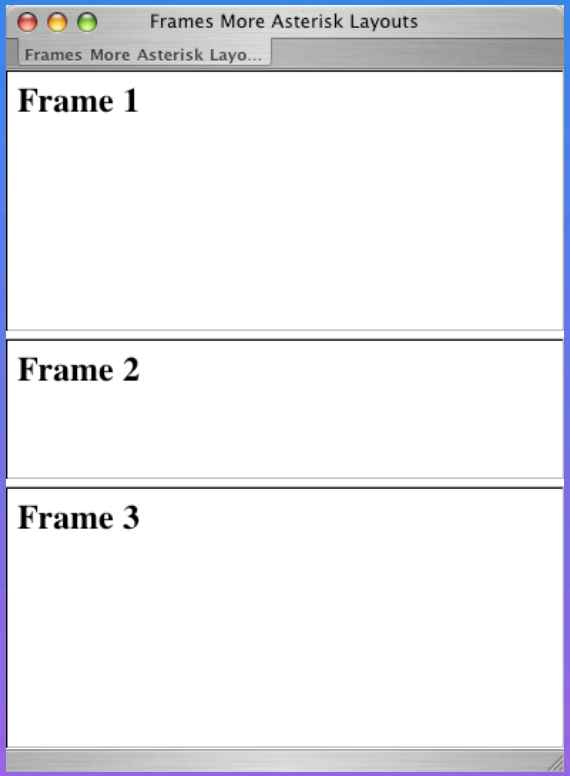
- The corresponding row or column gets proportionally more of the available space.

For example, `frame_ast4.html`:

```
<frameset cols="10%,3*,*,*">
```

This creates four columns:

- The first column occupies 10 percent of the overall width of the frameset.
- The browser then gives the second **three-fifths** of the remaining space, and
- The third and the fourth are each given **one-fifth** of the remaining space.



## Advantages of Using Asterisks

Using asterisks, especially with the numeric prefix,

- Makes it easy to divide up the remaining space in a frameset.

**NOTE**:

- Users manually resize the individual frame document's columns and rows,
- Hence change the relative proportions each frame occupies in their frames display.
- To prevent this (if desired/necessary), set the `noresize` attribute for the frame tag.






# Nesting frameset Tags

Framsets may be nested inside each other:

- You can create some elaborate browser displays with a single frameset, but the frame layout can be
  - Unimaginative — too regular (not always a bad thing)
  - Restrictive — inhibits a logical layout of frames
- HTML allows the creation nested frames and other more complex layouts with **multiple frameset tags**
  - Nested within a **top-level frameset** in the **frame** document.

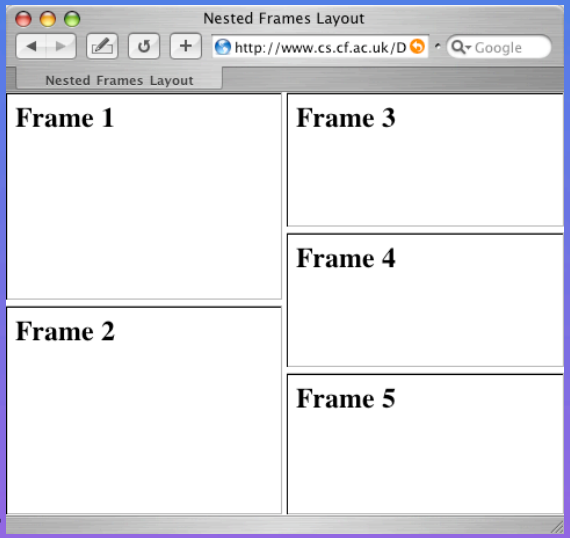





# Simple Nested Frameset Example

For example:

- Create a layout of two columns,
  - The first column with two rows and
  - The second column with three rows,
- By nesting two `frameset` tags with row specifications within a top-level frameset that specifies the columns we can readily achieve such a layout, `frame_nest.html`:




```
<frameset cols="50%,*">
  <frameset rows="50%,*">
    <frame src="frame1.html">
    <frame src="frame2.html">
  </frameset>
  <frameset rows ="33%,33%,*">
    <frame src="frame3.html">
    <frame src="frame4.html">
    <frame src="frame5.html">
  </frameset>
</frameset>
```

# JavaScript and Frames

Frames and JavaScript:

- Can be combined on the same page
- Can develop interesting interactive pages.
- Often a site (set of pages/frames)
  - Have links in one frame and
  - Wish to output in another frame
  - See Earlier Example, `new_frame.html`
- A mix of Frames and JavaScript:
  - Can provide easy movement through complex data/layout.






# Example: Building A Colour Picker

This example is taken from the recommended course book:

- **Web Programming** by Chris Bates

The **simple colour picker**:

- **Shows how we can customise HTML a little more also**
- **Also shows how to process a HTML form in JavaScript**
- We construct a framset of two frames.
  - The upper one contains a form which is used for data gathering.
    - Enter data in form
    - Can set colour (by name) of Background Color, Text Color, Table Headings and Table Data.
    - **We meet with HTML table in Next Section** — ignore finer details of HTML tables here for now.
  - The lower frame shows the result of the colour selections
    - The lower frame is created directly by JavaScript code.






# The Colour Picker Example Output

# The Colour Picker Example Frameset

The code for the frameset is very simple, col_pick_frames.html:

```
<html>
<head>
<title>colour picker</title>
</head>
<frameset rows="40%,*">
<frame name="topone" src="./cols.html">
<frame name="botone" src="./blank.html">
</frameset>
</html>
```

The names of the frames will become important shortly






# The Colour Picker Example Initial Bottom Frame

- Eventually we write to the bottom frame
- Initially it is set blank HTMLL page
- For good browser support best to use blank HTML code:

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






# The Colour Picker Example Top Frame

The top frame is simple enough.

- Simple HTML
- Call a JavaScript: `picker.js`
- Use `onClick()` action to trigger bottom frame display (by `picker.js` on submit






# The Colour Picker Example Top Frame HTML Code

```
<html>
<head>
  <script language=javascript src="picker.js">
  </script>
</head>

<body bgcolor=white text=red>
<h1 align=center>Chris's HomeBrew Colour Picker</h1>

<form>
  <table align="center" border=0 cellpadding=5>
  <tr>
  <td colspan=4 align=center>
  <h2>Enter Colour Values in the Boxes</h2>
  </td>
  </tr>
  <tr>
  <td>
  <h3>Background Colour</h3>
  </td>
  <td>
  <input type="textfield" size=16 name=bgcol
   value=white>
  </td>
  <td>
```






```
<h3>Text Colour</h3>
</td>
<td>
<input type="textfield" size=16 name=fgcol
 value=black>
</td>
</tr>
<tr>
<td>
<h3>Table Headings</h3>
</td>
<td>
<input type="textfield" size=16 name=thcol
 value=black>
</td>
<td>
<h3>Table Data</h3>
</td>
<td>
<input type="textfield" size=16 name=tdcol
 value=black>
</td>
</tr>
<tr>
<td colspan=2 align=center>
<input type="submit" value="Show It!!"
 onClick="ShowIt()">
```






```
  </td>
  <td colspan=2 align=center>
  <input type="reset" value="Reset It">
  </td>
  </tr>
  </table>
</form>
</body>
</html>
```






Back
Close

# The Colour Picker Example JavaScript Code: `picker.js`

## The Interesting Bit – At Last !!

The operation of code is fairly simple:

- Use JavaScript `frame` class — **which we've not seen until now**.
- Use JavaScript `form` class.
- Single function called `ShowIt()` does the job.
- Form colour values don't need to be passed as parameters.
  - Available to the script through the `frameset` itself.
  - Frames given the names when created `topone` and `botone`,
  - Script is part of the document that is being displayed in frame `topone`
  - Access colour values through this






- Create two local variables:

```
var topbox = document.forms[0].elements;
var bottombox = parent.frames['botone'].document;
```

`topbox` : refers to all items in the upper frame –

- Document only has one `form`
- Start of the forms array on position zero.
- Values of form stored in `elements` array.

`bottombox` : refers to all items in the lower frame –

- This how we are going to communicate info from the top to bottom frame.
- Reference the `bottombox` and `write()` data to it.






# Getting Information from the Form

Once the documents have been correctly aliased the values can be extracted from the form.

- We refer the `topbox` variable:
- For example to get the background colour off the form. We do
  ```
  var bg = topbox.bgcol.value;
  ```
- Note `bgcol` is the name associated with this form element.
- Other form items similarly treated.






# Setting up the bottom frame

Having extracted the parameters the coloured sample page can now be created.

- First the frame has to be opened — otherwise we cannot write to the frame:
  `bottombox.open();`
- Next we create the HTML for the page:
  - Need to make the `<BODY> ...</BODY>` pair of tags
  - To set up background and foreground colours.
  - Use `write()` to the `bottombox`:

```
bottombox.write("<body bgcolor="
                  + bg
                  + " text="
                  + fg
                  + ">\n");
bottombox.write("</body>");
```

- Also the visible content (a table) in similar way
- Finally we `close` the document.
  `bottombox.close();`
  - **Essential**: The point at which the HTML gets sent to the frame
  - The page gets displayed.






# picker.js Full Source Code

The full code for picker.js:

```
function ShowIt( ){

var topbox = document.forms[0].elements;
var bottombox = parent.frames['botone'].document;

// first extract the values from the form
var bg = topbox.bgcol.value;
var fg = topbox.fgcol.value;
var thc = topbox.thcol.value;
var tdc = topbox.tdcol.value;

// now build the new page
bottombox.open();
bottombox.write("<body bgcolor="
                + bg
                " text="
                + fg
                + ">\n");
bottombox.write("<h1 align=center>The Result Is:</h1>");
bottombox.write("<table align=center border=2"
                + "cellpadding=4 cellspacing=4>\n<tr>"
                + "<th>Plain Heading</th>"
                + "<th bgcolor="
```






```
                    + thc
                    + ">Coloured Heading</th>"
                    + "</tr>"
                    + "<th>Plain Data</th>"
                    + "<th bgcolor="
                    + tdc
                    + ">Coloured Data</th>"
                    + "</tr>"
                    + "</tr>\n</table>");
bottombox.write("</body>");
bottombox.close();
}
```

**EXERCISE:** amend the `picker.js` script to check for valid colour data form entry?