# computing: nature, power and limits—robotics applications (cis1.0)
# fall 2006—lecture # B.4
# monday 25-sep-2006

## today

advanced HTML

- special characters
- style sheets
- making things dynamic
  - image maps
  - forms
  - javascript

NOTE: MIDTERM EXAM IS ON THURSDAY OCTOBER 12

## special characters

- forcing spaces: ` `
- accents and other special characters
  (there are many more — see the on-line tutorial called "advanced HTML"):

| symbol | code |
|---|---|
| copyright | `&copy;` |
| registered trademark | `&reg;` |
| trademark | `&#8482;` |
| less than | `<` |
| greater than | `>` |
| ampersand | `&` |
| long dash | `&#8212;` |
| double quotes | `&quot;` |
| é | `&eacute;` |
| è | `&egrave;` |
| ö | `&ouml;` |
| ñ | `&ntilde;` |

## style sheets

- text color and background color:

```
<style type="text/css">
  body { color: black; background: white; }
</style>
```

- linking to a separate style sheet (in another file):

```
<link type="text/css" rel="stylesheet" href="style.css">
```

  and the "style file" looks like this:

```
/* style.css - a simple style sheet */
body {
  color: black; background: white;
}
```

- margins, left and right indents:

```
body { margin-left: 10%; margin-right: 10%; }
h1 { margin-left: -8%;}
h2,h3,h4,h5,h6 { margin-left: -4%; }
p { text-indent: 2em; margin-top: 0; margin-bottom: 0; }
```

- white space above and below:

```
h2 { margin-top: 8em; margin-bottom: 3em; }
```

- fonts:
  - styles:

```
em { font-style: italic; font-weight: bold; }
strong { text-transform: uppercase; font-weight: bold; }
```

  - text-transform can be: uppercase, lowercase
  - font families: Verdana, Garamond, "Times New Roman", sans-serif, e.g. `body { font-family: Verdana, sans-serif; }`
- divisions:

  - borders
  - colors
  - name the divisions using "class"
  - example:
    *in the style file:*

```
div.box { border: solid; border-width: thin; width: 100% }
```

    *in the html file:*

```
<div class="box">
The content within this DIV element will be enclosed
in a box with a thin line around it.
</div>
```

- link colors:

```
:link { color: rgb(0, 0, 153) }  /* for unvisited links */
:visited { color: rgb(153, 0, 153) } /* for visited links */
a:active { color: rgb(255, 0, 102) } /* when link is clicked */
a:hover { color: rgb(0, 96, 255) } /* when mouse is over link */
```

## making things dynamic

- giving the user control to navigate pages
  - links
  - image maps
- giving the user control to change page content
  - forms
  - javascript

## image maps

```
<img src="mymap.jpg" width=300 usemap="#mymap">
<map name="mymap">
<area shape="rect" coords="0,  0,100,100" href="blue.html">
<area shape="rect" coords="100,0,200,100" href="purple.html">
<area shape="rect" coords="200,0,300,100" href="grey.html">
</map>
```

- origin is upper left corner (pixel)
- rect: left-x, top-y, right-x, bottom-y
- circle: center-x, center-y, radius
- poly: x1,y1, x2,y2, ... xn,yn

## forms

- overview
  - HTML forms provide a way to create *dynamic* web pages—
    i.e., the user has control over what appears
  - forms are linked with some type of *program* that runs either on the *server-side* or the *client-side*
  - we will learn a little bit of Javascript, which runs on the *client* side
- **form** tag:
  – appears in the body of an HTML file

```
<form name="form1">
.
.
.
</form>
```

- **input** tag:
  – contains **input** components that allow the user to enter information

```
<input type="button"
       name="mybutton"
       value="click me"
       onclick="alert('hello')" />+

<input type="text"
       name="myname" />+
```

- **javascript** *functions* handle user input, when user clicks on a `button`
  – in the example above, the javascript function `alert()` is "called", or "run" or "invoked"