HTML/CSS...etc...

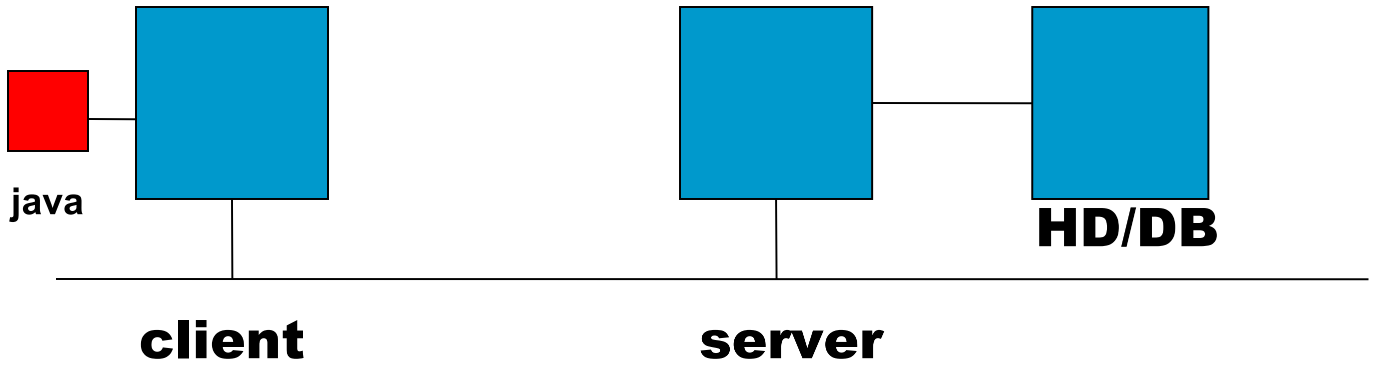
- Future is now : HTML5 ;-)
- class versus id and . versus #
- div versus span
- Special characters
  - Conf file server side or
  - HTML code &eacute;... etc... or
  - Meta tag (see example)

```
<META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=ISO-8859-1">
```

- Internal link (label and ref)
- And a lot more …

Javascript
(client side programming)

- Dynamic model



- Included into HTML
- Exec client side
- Animation, tparameter testing and a lot more (Google)

Conclusion : the red box !

Main lines

- Inclusion : specific markup

  <script>…**</script>**
- Main parameters :
  - Text type : **type**
  - Used language : **language**
- OO programming style…
- Call/execution :
  - Simple inclusion or ...
  - HTML event

1st example ;-)

```
<html>
<head>
  <title>My first Javascript</title>
</head>
<body>
<script language="JavaScript" type="text/JavaScript">
document.write("Hello everybody!");
document.close();
</script>
</body>
</html>
```

- Exercise: center ;-)
- Create alert... with alert(...)
- Pb: we want to control when -> events

With an HTML event !

```
<html>
<head>
   <title>My second Javascript for a simple computation</title>
<script type="text/javascript" language="JavaScript">
   function compute(){ //function definition
   var x=1 ; var y = 2; //user defined variables
   var result = x + y;
   document.write("the result is " + result + "<br>");
   document.close();}
</script>
</head>
<body>
Click on the following button to run the script !
     <input type="button" value="calculate" OnClick="compute()">
     <!--call associated to an event -->
</body>
</html>
```

- **Pb: on veut nous même entrer les paramètres -> HTML FORM**

With forms!

```
<script language="javascript" type="text/javascript">
function go(){
alert("hello "+maforme.votreprenom.value + " " +
   maforme.votrenom.value)
  }
</script>
</head>
<body>
<form name="maforme">
Enter your surname : <input type="text" name="votrenom"><br/>
Enter your firstname : <input type="text" name="votreprenom">
<input type="button" value="clic here" OnClick="go()">
</form>
</body>
</html>
```

Pb: write inside the page?

Solution: getElementById()

Write inside the page

```
<script language="javascript...>
function go() {
  document.getElementById('myarea').innerHTML = "Hello
   " + maforme.votreprenom.value + " " +
   maforme.votrenom.value;
  document.getElementById('mazone').innerHTML +=
   '<br/>';
}
</script></script>
</head>
<body>
<form ... as usual </form>
<div id=«myarea» ></div><!-- preserve space -->
</body>
</html>
```

A basic calculator (1)!

```
<form name="mycalc"><!-- the form has basically 3 parameters  -->
parameter 1 : <input type="text" name="ope1" value="0"><br>
parameter 2 : <input type="text" name="ope2" value="0"><br>
your choice of operator : <select name="operator" size="3" >
   <option value="1">+
   <option value="2">*
   <option value="3">/
   </select>
<br>
<input type="button" name="button" value="go"
   onClick="compute(mycalc)">
<!-- mycalc equivalent to this.form -->
<br>
The result of your calculation is : <br>
<input type="number" name="result" value="0"><br>
<input type="reset" name="stop" value="reset"> <!-- add a reset button
   -->
</form>
```

A basic calculator (2)!

```
<script language="javascript">
function add(form){ // a form parameter
var a=parseFloat(form.ope1.value);//transform text into real
   value
var b=parseFloat(form.ope2.value);
var result=a+b;
form.result.value=result; //met le resultat au bon endroit
}
....etc pour les autres opérateurs

function compute(form){
if (form.operator.value =="1") {add(form);}
else if (form.operator.value == "2") {mult(form);}
else div(form);
}
```

Parameters checking

- Objective: check before sending
  - To not overload the server
  - To not overload the network !
- Simple method:
  - HTML forms
  - For each parameter, validity checking
  - Event to control execution

Basic example

```
function check1(s){
var OK=true;
if (s=="")  {alert("put your name inside the
box");OK=false;}
else {alert("OK")};
return OK
} @@@@@ BODY
<form name="myform" action="mailto:grichard@bite.ac.uk"
onSubmit="return check1(document.myform.myname.value)">
put your name inside the box : <input type="text"
name="myname" size="20">
<input type="submit" value="send">
</form>
```

JavaScript focus function

- **goal**: to improve the UX ;-)
  - Parameter checking
  - Focus on the first wrong parameter in a form
- **How**: using the `focus()` function
- **Algo**:
  - If notOK(param.value) {param.focus()}

Exercise

1. Use it for your previous HTML forms

2. Try the function select() !

3. Show me !

Exercise

1. Use it for your previous HTML forms

2. Try the function select() !
3. HTML5 <input required... > allows to force
4. not a standard (safari,etc.)

Style modification

- **DIV** tag: block inside a page
- A lot of parameters:
  - Visibility: hidden, visible
  - Position: absolute, relative
  - Z-index: like a stack of div

Everything can be modified with JavaScript

Simple example

```
.layer1 {
position: relative;
margin-top: 45px;
margin-left: 20px;
font: bold 50px arial;
z-index: 2;
}

.layer2 {
position: relative;
margin-top:-50px;
margin-left: 5px;
font: italic 80px arial;
color: red;
z-index: 1;
}
```

Z-index with JavaScript

- **How to access style's parameters:**
  - **Depends on the Web browser**
  - **Not easy ;-)**
  - **But powerful animation**
- **Layer defined with** `div` **and** `id`
- **4<Netscape<6:**
  - `document.layers.div_name`
- **4<IE<5**
  - `document.all.div_name`
- **IE>5 and N>6**
  - `document.getElementById(« div_name »)`

An example with Mozilla

- We use:

  `document.getElementById(« div_name »)`
- Function show/hide: `visibility` parameter
  - Show example14-visibility.html + code
  - Modify visibility (see next slide)
- Function OnTop/OnBottom: `z-index`
  - Show example9 again
  - Modify z-index

The script

```
function handleClick() { //for firefox
 if (document.getElementById('button2').style.visibility !=
"hidden")
    {
    document.getElementById('button2').style.visibility =
"hidden";
    document.getElementById('but1').value = "Show Other
Button";
  }
  else {
    document.getElementById('button2').style.visibility =
"visible";
    document.getElementById('but1').value = "Hide Other
Button";
  }
}
```

The call

(2 div - 2 buttons - Show example14)

```
<body>
<div ID="button1">
<INPUT id="but1"" TYPE="button" VALUE="Hide Other
Button" onclick="handleClick()">
</div>

<div id="button2" STYLE="position:relative; left:100;">
<INPUT TYPE="button"  VALUE="Hide Me"
onclick="handleClick()">
</div>

</body>
```

How to move

- **We use (with Mozilla)**

  `document.getElementById(« div_name »)`

- **Parameters left and top**
  - **Show example9**
  - **`style.left= x` (means xinit = xinit +x)**
  - **`style.top= y` (means yinit = yinit + y)**

Practical uses

- banners,
- Games, (show amazing example)
- Menus,
- Etc...
- One limit = your imagination!

HTML+CSS+JavaScript

=

Dynamic HTML (DHTML)

Cookies

- Small text files (<4KB)
  - Name
  - Value
  - Expiry date (plus domain)
- Usefull pour UX ;-)
- 3 actions: create – read – delete
- String `document.cookie` associe a la page
- Create:
  - `document.cookie=″nom=gilles″`
  - `document.cookie=″nom=gilles;expires=...”`
- Read : string analysis `document.cookie`
- Delete : allocate past date

Cookies example

- **function viewCookie()** {

```
if (document.cookie.length>0)//if we have some cookies
alert(document.cookie);
}
```

- **function createCookie(name,value,days)** {

```
var expires;
if (days) {
var date = new Date();
date.setTime(date.getTime()+(days*24*60*60*1000));
expires = '; expires="+date.toGMTString();
}         else expires = "";
document.cookie = name+"="+value+expires;
}
```

Cookies example

- **function viewCookie()** {

```
if (document.cookie.length>0)//if we have some cookies
alert(document.cookie);
}
```

- **function createCookie(name,value,days)** {

```
var expires;
if (days) {
var date = new Date();
date.setTime(date.getTime()+(days*24*60*60*1000));
expires = '; expires="+date.toGMTString();
}          else expires = "";
document.cookie = name+"="+value+expires;
}
```

OO philosophy example

- no class - only object
- object inherit from object
- A simple example

```
function myClass(type) {
                this.type=type; this.a="";
                this.geta= function(){ //get method
                               return this.a;
                          }
               this.seta= function(s){  //set method
                this.a=s.toUpperCase();
                          }    }
function createObject(s){
var o= new myClass();
o.seta(s);
return o;
}
```

Conclusion

- OO philosophy
- Easy integration with HTML
- Modular programming (functions, classes)
- A lot of info on the web
- Jquery : a nice library (do not reinvent the wheel!)
- Not enough... since :
  - No action server side
  - Visible code (security) (option : minify!)
  - No influence on web page modularity