# E-business

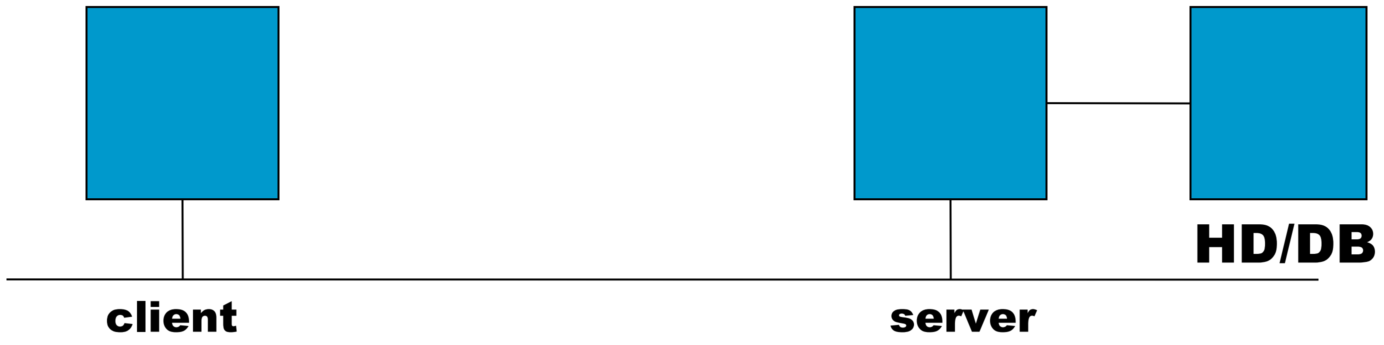
e for remote !

- e-X : X=
  - learning,
  - government,
  - banking,
  - surgery, …
- Main interface: **web** …

Conclusion : how does it work ?

## Standard Web services model

- Static model

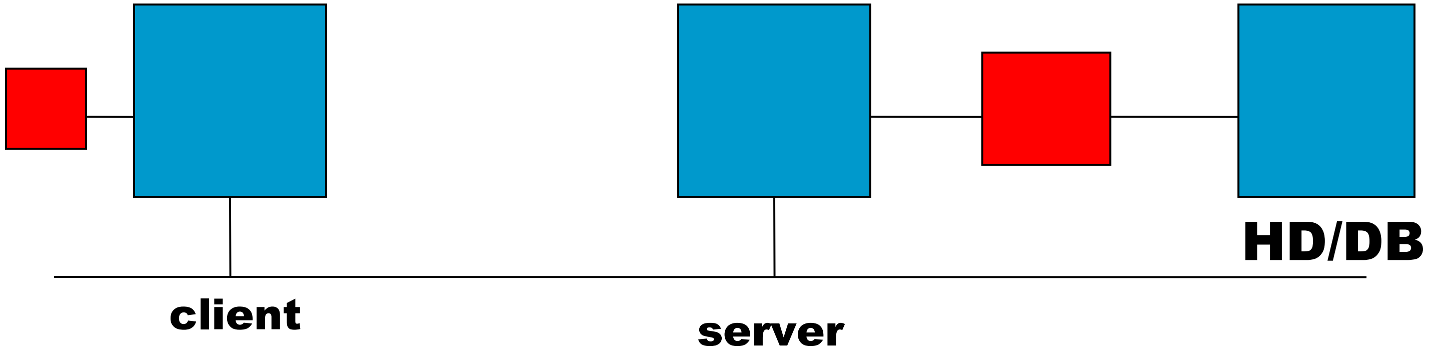


- + : simple, efficient, easy to learn and to understand,…
- - : limited interactivity, boring to learn (HTML/CSS), …

Conclusion : need a new one !

# Another Web model

- Dynamic model



- + : illimited interactivity, challenging to learn,…
- - : not so simple, security issues, …

Conclusion : the red boxes !

# Software

- Tools : 2 types
  - Design -> standard & specification tools
  - Implementation -> programming languages

- Programming languages : 2 sides
  - Server side
  - Client side
  - Both sides ;-)  (AJAX)

Conclusion : plan of the unit !

# Our targets

- Specifications tools
  - UML
  - **HTML/CSS**
  - **XML**
- Programming tools
  - Server side : **cgi scripting with C, Perl**
    **Java, ASP (.net technology)**
    **PHP**
  - Client side : **javascript**
    **vbscript**
- Programming style : AJAX
  - **Synchronous versus asynchronous**

## Build a web site…easy

- Do not reinvent the wheel ;-)))
  - **Choose a nice one**
  - **Download it**
  - **Improve it**
- Obviously…

**Have to understand the code…**

## HTML

- **HTML**: HyperText Markup Language
- Derived from **SGML**:Standard Generalised…
- Defines **structure** + **layout** of web doc.
- The most successful language!
- A web site = a set of HTML pages

  (from 1 to 10000…)

## HTML document

- Text file with:
  - Text defining the informative content (what)
  - Tags defining the layout (how)
- Editor: text editor ;-)
- Starting tag <tag>
- Closing tag </tag>
- Empty tag <tag/>
- Two parts: <**head>** and <**body>**

## Global structure

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

## MAIN PROCESS

- Introduction of a new tag then…
- Live testing then…
- Improvements !

1. Basic (center, strong,font, etc...)
2. Anchor: <a ...> ... </a>
3. Images: <img ...>
4. Tables: <table ...>...</table>
5. Etc...;-)

## Your know-how...now

- Basic web site with HTML
- Simple but effective !
- With
  - images
  - tables (menus)
- Modern and attractive look
  - Sans serif font
  - Uniform font
- Googling for more details ;-)

## BUT…

- HTML… fantastic BUT
  - Lack of flexibility
    - » Ex.: change font for a whole web site…
  - Lack of interactivity (static web sites)
    - » Ex. : flights Toulouse-Tokyo
  - Lack of animation
    - » Ex. : change font onmouseover
- 2 big ideas to improve flexibility
  1. CSS files → externalize format info
  2. DIV tag → breakdown a page

## The first big idea !

- Separate formatting information
  - Formattage = CSS file
  - CSS= Cascading Style Sheet
  - Structure+content = HTML file
- Past version: 1 HTML file
- New version: 1HTMl(5) file + 1CSS file

## Learn again :-(

1. Syntax HTML (done;-)
2. How to link HTML to CSS
3. Syntax CSS
4. BUT

- CSS file = text file
- Editor : text editor (as usual)
- Syntax CSS != syntax HTML ;-))))

# Link HTML/CSS

- **New tag LINK**

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

- **Where: head of the document**
- **Mandatory for**
  - **Flexibility: 1 CSS -> several HTML**
  - **Attractiveness ;-)**
- **Still to be done : `style.css`**

## General syntax

- CSS = set of rules
- one rule =

```
Selector {
Property1: value1;
Property2: value2;
 .....
}
```

  - Selector = HTML markup
  - Property = markup parameter
  - Value = value allocated to the parameter

## Example

```
body {
font-family: arial,verdana;
color: blue;
font-size: 10px;
font-weight:normal;
/* visibility:hidden; joking! */
/* background-image:
   url(«im/myface.gif2); */
}
```

- **We test live! (**`margin for instance`**)**
- **About colors !**
- **Editor needed;-) (Linux : quanta, Win : a lot !)**

## Basic animation ...

- Define the link behaviour

```
a {
font-weight:normal;
text-decoration:none (or overline or underline)
}
```

- Events: style link to the mouse position!

```
a:hover {
text-decoration: underline;
}

a:visited {
font-color:red;
}
```

## 2nd big idea ;-)

- Standard Structure of a web page
  - Leader with menu
  - Left menu
  - Page centre for information
  - Footer
- Idea: breakdown a webpage into set of divisions

# DIV markup

- **Structure of a web page**

```
<html> <head>...</head>
<body>
<div class='header'>...</div>
<div class='leftMenu'>...</div>
<div class='content'>...</div>
<div class='footer'> ... </div>
</body>
</html>
```

- **one CSS rule per class (class/id:unique)**
  - **grouping**
  - **maximal modularity ( almost maximal !)**
  - **DIV order irrelevant;-)**

## An other example!

```
body { /* we define the body as we want */
font-family: Verdana,sans-serif;
font-size: 20px;
margin-left: 100px; /*we can move any DIV */
margin-right: 100px;
margin-top: 100px;
text-align: justify;
}
.header {/* . for class BUT # for id */
font-size:13px;
color: grey;
/* margin-top:100pt; */
}
.leftMenu {
text-align:right
}
.footer {
font-size:15pt;
color:yellow;
}
```

# Conclusion

- CSS + DIV give Flexibility (a bit!)
- HTML5 : mandatory to have CSS
- New first line : <!DOCTYPE html>
- CSS3 : a lot of new features
- Still missing
  - What if we want to modify a menu ?
  - Full Interactivity
    - client side -> JavaScript
    - server side -> PHP/ASP/CGI script
    - mix -> AJAX