STAGES PER STUDENTI DI SCUOLA MEDIA SUPERIORE

9 / 27 JUNE 2003

SESSION : INFORMATIC
SUBJECT : Conversion and optimization of dynamic HTML pages, starting from XML documents

STUDENT : Marco Mariani
INSTITUTE : Liceo Scientifico Statale "FARNESINA"
TUTOR : Igor Sfiligoi

27/06/03

PURPOSE OF THE STAGE

The purpose of the stage was to build dynamic HTML pages starting from .XML data files.

USED EQUIPMENT

During the whole course we used a computer with a Linux O/S (operating system) and an Internet connection; we also had the possibility to connect with other computers using a LAN connection. We used Emacs and Xemacs as program editors and Mozilla! or Netscape to run them; we also used a personal account where we could save our works.
None of us had ever worked with Linux, so we read "Linux in a Nutshell" by Jessica Perry and the Staff of O'Reilly (ISBN 1-56592-167-4) how to use this O/S: this book explained to us the main commands of Linux, like "ls" or
"cd", and how to work with programs.

After we learned the basic things of Linux, we started with the PHP programming course. In the beginning, we followed two interesting on-line lessons on the PHP language, which taught us the basic functions for writing a program.

PHP links:

http://web.tiscali.it/no-redirect-tiscali/antonio_corsi/index.htm

http://www.risorse.net/php/index.php

During the exercises, we found the book
"Guida a PHP (2nd edition)" by Tim Converse and Joyce Park (ISBN 8838643202) very useful because it explains the formulation of command strings; we also used on-line guides and tutorials.

Guides and Tutorials links

http://www.php.net

http://www.htmlgoodies.com/tutors/forms.html

In our opinion, these on-line manuals have been the best sources for finding of particular functions, because there were many explicit examples of the commands.
When we acquired the PHP language, we began to build programs using the XML language: the manual "XML la Guida Completa" by Heather Williamson (ISBN 8838642125) helped us to understand this language and its dynamics.

METHODOLOGY OF WORK

The course can be divided fundamentally in three parts that are equivalent to the three weeks of the course:

- On the first week we learned how to use the PHP language and its applications with the HTML format. We made some exercises under the supervision of the tutor, who left us free to try new commands, but who gave us many tips when we were in trouble. The first session of exercises helped us to get more familiarity with PHP and its functions. We had some problems with the functions "For", "Foreach" (see example 1) and with the use of the Arrays, but we solved them with the help of the tutor and of the manuals. These exercises have been very important because they have built the foundations from which we started to write all the other programs.

-During the second week, we used all the knowledge learned, with new techniques of representation like the construction of HTML tables and the structure of FORM (see example 2), for building Web pages connected between them. The data were usually taken from a .DAT file. Then, we started to write specific programs that were able to read data and to build HTML pages showing tables that contained the values of a .DAT file.
For large amount of data, we learned and used some commands that allow to separate the different tables and give the possibility to data chosen.
We also optimized the files created adding other functions like "Foreach", "Chop", "If-Else" and the Arrays, resulting in a more dynamic program.
We spent most of our time trying to create Multidimensional Arrays (see example 3).

-On the last week we wrote programs taking the data from an .XML file (see example 4): at the beginning our programs only wrote an .XML file as output, writing in it all the data. Once we learned to write and to read this new type of data, we understood that the following programs were only an adaptation of the preceding ones to the reading from .DAT files to .XML files.
In fact, with the help of our tutor, we have succeeded: first of all, to write a program that reads the XML language, then to write the same .XML file in a multidimensional Array, like the preceding exercises, and in the end to use this Array to develop a dynamic HTML table, giving the possibility of choosing which schemes or tables to show (see example 5).

RELATIONS AND FORMULAS USED

Html page

Tags

A fundamental concept we learnt during this stage focuses on the structure of the HTML pages.
In fact they are ordered by Tags.
For example an HTML page should always have tags like this

```
 <html>
   .......
   .......
 </html>
```

where dots mean what will be written in output.
This is very important to give an initial structure to the page. In a page there could be a lot of different Tags controlling the color, the position inside the text of the string that they contain.

For example, these tags write a string in title format

```
<h2>  .........  </h2>
```

These represent the opening of a table

```
<table> ...... .... </table>
```

All these structures start with <.....> and end with </......>.

Composition of a HTML page

A HTML page is usually composed of two principal Tags:
<head> and <body>.
Head contains the title of the program and other information that won't be sowed in the page. Body contains all the functions that govern the working of the program.

A PHP Program

In a dynamic HTML page there are these kinds of tags

```
<?php
     ...... ... .
      ..  ......
?>
```

showing the beginning of a PHP text.
The first thing we have learnt about PHP was the command denomination: every PHP command end with ";".
As regards the writing of something in output

- If you want to show a string

ECHO "string"; or PRINT ("string");

- If you want to show a variable (in PHP called as $name_of_variable)

ECHO $name_of_variable; or PRINT ($name_of_variable);

- If you want to show an array

PRINT_R ($array);

To assign value to a variable you have to use some operator like "=" (value = $variable) or something more complex, as "For" or "ForEach".

Comments are strings that won't be show in output and in PHP are obtained with //....... or /*......*/.

To insert input when making an interactive page you must use FORMS and INPUT buttons so you can link pages.

The PHP language is ruled by control operator; for example If-Else.
More complex and useful are LOOPS. They are various: the most famous are "For", "ForEach" and "While".
Their function is to execute repetitive operations inside a text.
Further on Loop there are the Functions that allow to memorize some instructions and replay them without need to write them.

PHP has a huge library of commands, operators and functions.
During the stage we used File or Fopen to read files and Array or Explode to make table. If you want to work with XML files you have to know other operator like $xml_parser to read this kind of format.
With all these commands we finished our exercise.

RESULTS OF THE COURSE

We spent most of our time learning some of the principal notions about PHP because we hadn't learned them yet.
After we acquired this knowledge we started to write simple programs able to show tables in an HTML page.
From these exercises we moved on to dynamic and interactive pages, which supply users with the choices of which table to show.
Also the exercises with data files in XML format were more or less the same: all these exercises were based on creating multidimensional Arrays from which it is possible to extract the required information with speed and without difficulty: for example Arrays containing Sub Arrays "Schema" that are composed from Sub Arrays "Table" which contain "Field" divided in "Name", "Type" and "Nullable" that are the values appearing in the table.
The final result has been the creation of dynamic HTML page from an XML format data file.

```
1) FOREACH

<?php
 $a1 = array("abc"=>"def","ghk"=>"iuy");
 $a2 = array("zxc"=>"vbn","123"=>"456");
  $c1 = array("a1"=>$a1,"a2"=>$a2);
 $a12 = array("abc2"=>"def2","ghk2"=>"iuy2");
 $a22 = array("zxc2"=>"vbn2","1232"=>"4562");
  $c2 = array("a12"=>$a12,"a22"=>$a22);
   $d1 = array("c1"=>$c1,"c2"=>$c2);


foreach ($d1 as $e=>$f)
 { echo "<schema name=\"" .$e . "\">" . "\n";
   foreach ($f as $g=>$i)
    { echo "\t<table name=\"" . $g . "\">" . "\n";
      foreach ($i as $h=>$k)
      { echo "\t\t<field " . $h . "=\"" . $k . "\">\n";}
        echo "\t</table>\n";}
     echo "</schema>\n";}
?>
```

2) FORM

```
<html>
 <head>
  <title>    NuovoEsempio    </title>
 </head>
 <body>
   <p>
   Inserisci tre numeri a tua scelta e ne avrai delle relazioni
   </p>
  <FORM action="NuovoEsempioCorrelato.php" method="get">
   <p>
   Numero A
   <input type="NUMBER" name="a">
   <br>
   </p>
   <p>
   Numero B
   <input type="NUMBER" name="b">
   <br>
   </p>
   <p>
   Numero C
   <input type="NUMBER" name="c">
   <br>
   </p>
   <input type="submit" name="submit" value="INVIA">
   <input type="reset" name "reset" value="RIPRISTINA">
  </FORM>
 </body>
</html>
```

3) MULTIDIMENTIONAL ARRAY

```
<?php
$dat = file ('prova2finale.dat');
$primo =0;
$tabelle = array();
foreach ($dat as $a=>$c)
    { if ($a == "---\n")
      { $titolo = explode(".",chop($dat[$primo]));
        $schema = $titolo[0];
        $tabella = $titolo[1];
        $field = array();
        for ($b=$primo+1;$b<$a;$b++)
         { $field[] = explode (" ",$dat[$b]);
           $tabelle[$schema][$tabella] = $field;
           $primo = $a+1; }}
```

```
?>
```

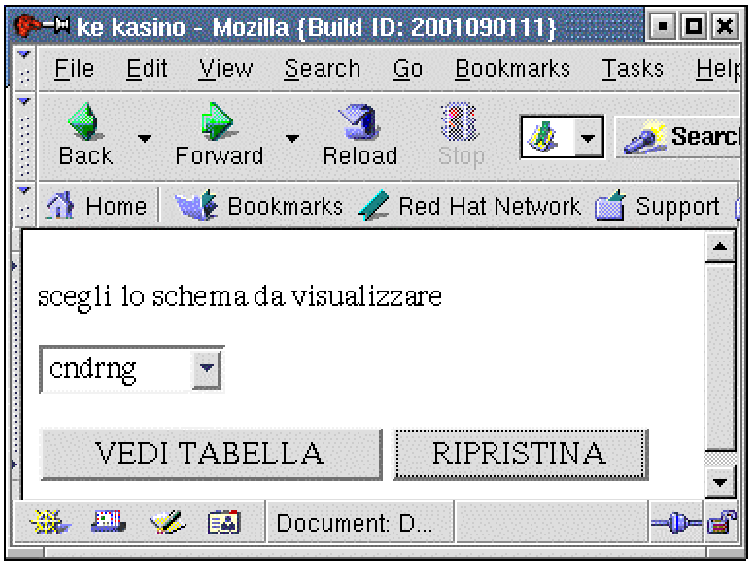
4) FILE.XML

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE dbfdesc PUBLIC "-//Sfiligoi Igor//DTD dbfields//EN" "dbfields.dtd">
<dbfields>
<schema name="cndrng">
     <table name="dc_bads">
           <field name="start_of_validity" type="integer" nullable="No"  />
           <field name="end_of_validity" type="integer" nullable="Yes"  />
           <field name="layer" type="integer" nullable="No" />
           <field name="wire" type="integer" nullable="No"  />
           <field name="status" type="integer" nullable="No"  />
     </table>
     <table name="dchtrg_calib">
           <field name="start_of_validity" type="integer" nullable="No"  />
           <field name="end_of_validity" type="integer" nullable="Yes"  />
           <field name="slope_t0d" type="double" nullable="No"  />
           <field name="slope_t1d" type="double" nullable="No"  />
           <field name="slope_tcrd" type="double" nullable="No"  />
           <field name="slope_t2d" type="double" nullable="No"  />
           <field name="offset_t0d" type="integer" nullable="No"  />
           <field name="offset_t1d" type="integer" nullable="No"  />
           <field name="offset_tcrd" type="integer" nullable="No"  />
           <field name="offset_t2d" type="integer" nullable="No"  />
     </table>
     <table name="emc_bads">
           <field name="start_of_validity" type="integer" nullable="No" />
           <field name="end_of_validity" type="integer" nullable="Yes"  />
           <field name="subdet_id" type="integer" nullable="No"  />
           <field name="module" type="integer" nullable="No"  />
           <field name="plane" type="integer" nullable="No"  />
           <field name="column" type="integer" nullable="No"  />
           <field name="side" type="integer" nullable="No"  />
           <field name="board_id" type="integer" nullable="No"  />
           <field name="status" type="integer" nullable="No"  />
     </table>
</schema>
<schema name="cndrun">
     <table name="dc_eff">
           <field name="run_nr" type="integer" nullable="No"  />
           <field name="layer" type="integer" nullable="No"  />
           <field name="hw_eff" type="double" nullable="No"  />
           <field name="hw_eff_err" type="double" nullable="No"  />
           <field name="sw_eff" type="double" nullable="No"  />
           <field name="sw_eff_err" type="double" nullable="No"  />
     </table>
</schema>
<schema name="crossref">
     <table name="torta_run">
```

```
            <field name="run_nr" type="integer" nullable="No"  />
            <field name="start_of_validity" type="integer" nullable="No"  />
            <field name="insert_time" type="timestamp" nullable="No"  />
      </table>
 </schema>
 <schema name="daq">
      <table name="test1">
            <field name="filename" type="varchar(64)" nullable="No"  />
            <field name="type" type="char(1)" nullable="No"  />
            <field name="data" type="integer" nullable="No"  />
            <field name="id" type="char(13)" nullable="No"  />
      </table>
 </schema>
 </dbfields>
```

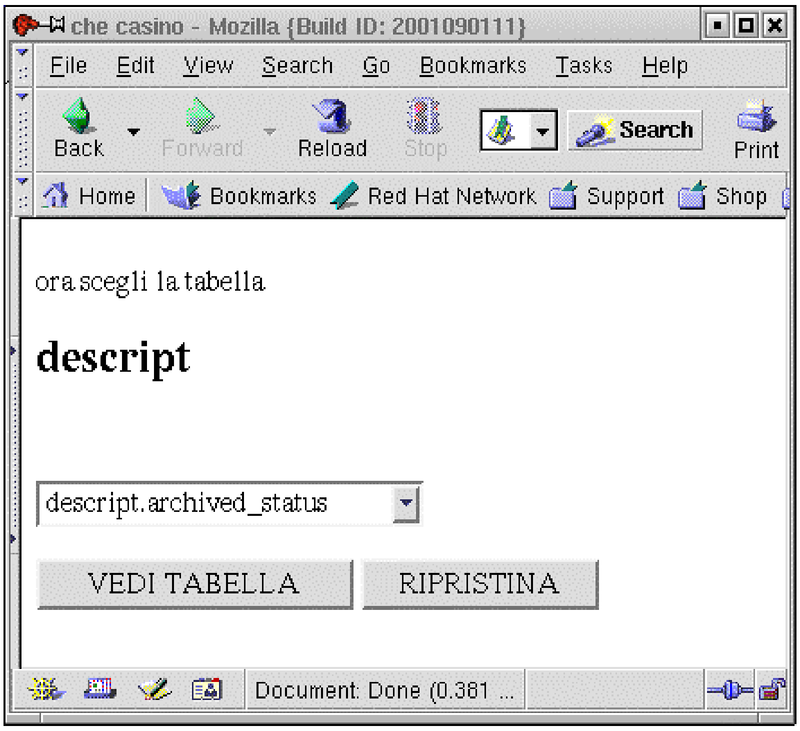
5) DYNAMIC AND INTERACTIVE HTML PAGE

In this page you can choose a “schema”.


scegli lo schema da visualizzare

cndrng

In the second page you can chose the table (tabella).

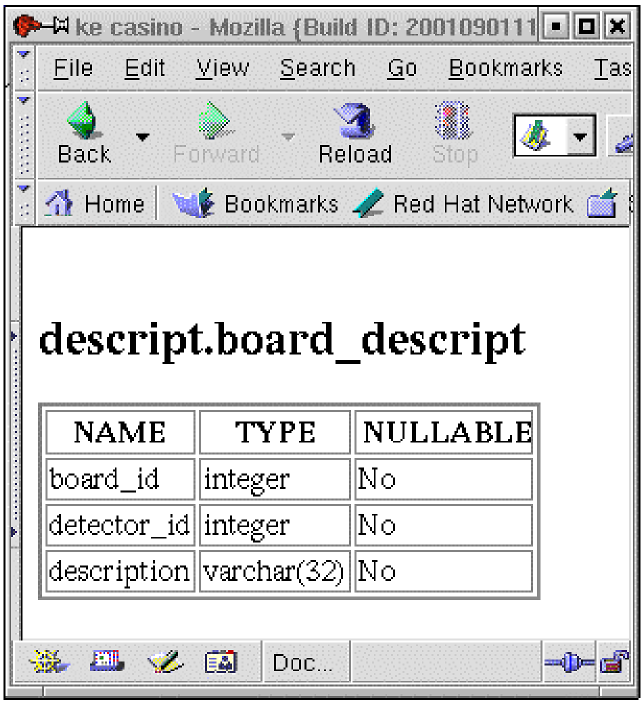

che casino - Mozilla {Build ID: 2001090111}
File Edit View Search Go Bookmarks Tasks Help
Back Forward Reload Stop Search Print
Home Bookmarks Red Hat Network Support Shop
VEDI TABELLA
RIPRISTINA
Document: Done (0.381 ...

ora scegli la tabella

descript

descript.archived_status

In the third page you have your table.


ke casino - Mozilla {Build ID: 2001090111
File Edit View Search Go Bookmarks Tas
Back Forward Reload Stop
Home Bookmarks Red Hat Network
Doc...

descript.board_descript

NAME	TYPE	NULLABLE
board_id	integer	No
detector_id	integer	No
description	varchar(32)	No