

Curs 6

2011/2012

Tehnici moderne de proiectare a aplicatiilor web

CURS

I.	HTML si XHTML (recapitulare)	1 oră
II	CSS	2 ore
III	Baze de date, punct de vedere practic	1 oră
IV	Limbajul de interogare SQL	4 ore
V	PHP - HyperText Preprocessor	8 ore
VI	XML - Extended Mark-up Language si aplicatii	4 ore
VII	Conlucrare intre PHP/MySql, PHP/XML, Javascript/HTML	2 ore
VIII	Exemple de aplicatii	6 ore
	Total	28 ore

Hypertext PreProcessor

PHP

PHP - Concepte

- limbaj interpretat – compilat “on the fly” de interpretorul PHP de pe server
- poate fi integrat in HTML – utilizarea tipica
- un fisier sursa PHP este un fisier HTML (in general) cu sectiuni de cod PHP
 - interpretorul PHP cauta sectiunile pe care trebuie sa le interpreteze si interiorul lor proceseaza instructiuni ca fiind PHP
 - ce se gaseste in exteriorul acestor sectiuni este trimis spre server-ul web nemodificat
- **<?php ... ?>**
 - stil XML – implicit, disponibil intotdeauna, recomandat

Variante de integrare

- Toate variantele ofera aceeasi sursa HTML pentru browser
- E **recomandata** cea care lasa structura HTML nemodificata si doar datele dinamice sunt rezultatul procesarii
- Codul HTML + PHP e interpretat mult mai elegant in editoarele WYSIWYG

```
<h2>Rezultate comanda</h2>  
<?php echo '<p>Comanda receptionata</p>';?>
```

```
<h2>Rezultate comanda</h2>  
<p><?php echo 'Comanda receptionata';?></p>
```

```
<?php echo '<h1>Magazin online XXX SRL</h1>';?>  
<?php echo '<h2>Rezultate comanda</h2>';?>  
<?php echo '<p>Comanda receptionata</p>';?>
```

```
<?php  
echo '<h1>Magazin online XXX SRL</h1>';  
echo '<h2>Rezultate comanda</h2>';  
echo '<p>Comanda receptionata</p>';  
?>
```

PHP – tipuri de date

- tipul de date nu e decis de programator prin declaratia variabilei
- e decis de interpretor in functie de tipul de date stocat in variabila respectiva

```
<?php
echo $variabila ; // tip Null, neinitializat – valoare NULL (doar)
$variabila = "0"; // $variabila tip string (ASCII 48)
$variabila += 2; // $variabila tip integer (2)
$variabila = $variabila + 1.3; // $variabila tip float (3.3)
$variabila = 5 + "10 obiecte"; // $variabila tip integer (15)
$var2=5; // $var2 tip integer (5)
$variabila=$var2."10 obiecte"; // $variabila tip string "510 obiecte"
?>
```

PHP – Functii

- conceptual similare celor din C/C++
- functiile nu trebuie declarate inainte de a fi folosite
- numele functiilor este “case-insensitive”
- un mare numar de functii cu utilitate directa in aplicatiile web exista in bibliotecile PHP
- unele biblioteci trebuie activate in momentul configurarii PHP
 - `extension=php_gd2.dll` (linia 639) // pentru functii de procesare grafica de exemplu
 - `extension=php_mysql.dll` (linia 651) // pentru functii de acces la baze de date MySql

Elemente de control

- while
- do-while
- for
- switch
- return
- break
- goto

- Similare cu echivalentele C/C++

```
$i = 1;
while ($i <= 10) {
    echo $i++;
}
```

```
$i = 0;
do {
    echo $i;
} while ($i > 0);
```

```
for ($i = 1; $i <= 10; $i++) {
    echo $i;
}
```

```
switch ($i) {
    case 0:
        echo "i este 0";
        break;
    case 1:
        echo "i este 1";
        break;
    default:
        echo "i nici 1 nici 0";
        break;
}
```


Elemente de control

- `include()`
- `require()`
- `include_once()`
- `require_once()`

- pentru inserarea **SI** evaluarea fisierului folosit ca parametru
- folosite pentru a nu multiplica sectiunile de cod comune
- `require` opreste executia script-ului curent daca fisierul parametru nu este gasit
- `..._once()` verifica daca respectivul fisier a mai fost introdus si **nu** il mai introduce inca o data

Variabile globale

Variabile globale

- Variabilele globale (predefinite)
 - accesibile script-urilor PHP prin conlucrarea cu server-ul
 - Exemple:
 - `$_SERVER` — Server and execution environment information
 - `$_GET` — HTTP GET variables
 - `$_POST` — HTTP POST variables
 - `$_FILES` — HTTP File Upload variables
 - `$_REQUEST` — HTTP Request variables
 - `$_SESSION` — Session variables
 - `$_ENV` — Environment variables
 - `$_COOKIE` — HTTP Cookies

Interactiunea cu utilizatorul

- Datele introduse de utilizator in forme se regasesc (in functie de metoda aleasa pentru forma) in una din variabilele:
 - `$_POST` – method="post"
 - `$_GET` – method="get"
 - `$_REQUEST` – ambele metode
- variabilele sunt matrici cu **cheia** data de atributul **name** din forma HTML
 - `<input type="text" name="carti_cant" size="3" maxlength="3" />`
 - `$_POST['carti_cant']` contine valoarea introdusa

Includerea / controlul formei in fisierile PHP

Laborator – L4

- se creaza macar 3 pagini:
 - lista produse
 - formular comanda
 - rezultat
- forma paginilor:
 - tabel

culoare	IMAGINE	culoare
	Continut (cu alta culoare fundal)	

Laborator – L4 - rezultat

Magazin online Firma X SRL

Lista Produse

Nr.	Produs	Pret
1	Carti	100
2	Caiete	50
3	Penare	150
4	Stilouri	125
5	Creioane	25

[Comanda](#)

Magazin online Firma X SRL

Realizati comanda

Nr.	Produs	Pret	Cantitate
1	Carti	100	<input type="text" value="1"/>
2	Caiete	50	<input type="text" value="2"/>
3	Penare	150	<input type="text" value="1"/>
4	Stilouri	125	<input type="text" value="0"/>
5	Creioane	25	<input type="text" value="0"/>

Magazin online Firma X SRL

Rezultate comanda

Pret total (fara TVA): 350

Pret total (cu TVA): 416.5

Comanda receptionata la data: 17/03/2010 ora 08:24

Laborator – L4 – sursa

```
<body><?php
define('PRET_CARTE',100);
define('PRET_CAIET',50);
define('PRET_PENAR',150);
define('PRET_STILOU',125);
define('PRET_CREION',25);
?><h1>Magazin online Firma X SRL</h1>
<h2>Rezultate comanda</h2>
<p>Pret total (fara TVA): <?php
$pret=$_POST['carte_cant']*PRET_CARTE+$_POST['caiet_cant']*PRET_CAIET+$_POST['pena
r_cant']*PRET_PENAR+$_POST['stilou_cant']*PRET_STILOU+$_POST['creion_cant']*PRET_C
REION;
echo $pret;?></p>
<p>Pret total (cu TVA): <?php
$pret*=1.19;
echo $pret;?></p>
<p>Comanda receptionata la data: <?php echo date('d/m/Y')." ora ".date('H:i');?></p>
<?php
$matr
?>
```


Analiza critica

- design?
 - in aplicatiile web forma este importanta
 - nu trebuie sa fie inovativa ci familiara
 - "Don't make me think!"
- capacitatea de extindere?
 - mai multe produse
 - schimbare de pret

Design 1

```
<html>
<head>
<title>Magazin online Firma X SRL </title>
</head>
<body>
<table width="100%" border="0"
bgcolor="#CCFFFF">
<tr><td rowspan="2" width="*"></td>
<td width="600"></td>
<td rowspan="2" width="*"></td></tr>
<tr><td height="600" valign="top"
bgcolor="#FFFACC">
Continut
</td></tr>
</table>
</body>
</html>
```

culoare	IMAGINE	culoare
	Continut (cu alta culoare fundal)	



tabel

Design 1



Design 2

```
<html>
<head>
<title>Magazin online Firma X SRL</title>
</head>
<body bgcolor="#CCFFFF">
<table width="600" border="0" align="center">
<tr><td></td></tr>
<tr><td height="600" valign="top"
bgcolor="#FFFCC">
Continut
</td></tr>
</table>
</body>
</html>
```

culoare	IMAGINE	culoare
	Continut (cu alta culoare fundal)	



tabel

Design 2



Design 3

```
<html>
<head>
<title>Magazin online Firma X SRL</title>
<link rel="stylesheet" href="stil.css"
type="text/css" />
</head>
<body>
<div class="antet"></div>
<div class="continut">
Continut
</div>
</body>
</html>
```

```
body { background-color: #CCFFFF;}
.antet { background-image:url(images/antet.gif);
background-repeat:no-repeat;
height: 100px;
width: 600px;
margin-top: 0px;
margin-right: auto;
margin-left: auto;}
.continut{ background-color:#FFFFCC;
height: 600px;
width: 600px;
margin-top: 5px;
margin-right: auto;
margin-left: auto;}
```

Design 3



Template

- Sablon
- controlul simultan al formei pentru toate paginile din site
- separarea aplicatiei de forma

Exemplu – design 2

- sectiunile repetabile pot fi mutate intr-un fisier separat si introduse cu require()
- se identifica zonele comune

```
<html>
<head>
<title>Magazin online Firma X SRL</title>
</head>
<body bgcolor="#CCFFFF">
<table width="600" border="0" align="center">
<tr><td></td></tr>
<tr><td height="600" valign="top"
bgcolor="#FFFFCC">
Continut
</td></tr>
</table>
</body>
</html>
```

Lista produse – cu design2

```
<html>
<head>
<title>Magazin online Firma X
SRL</title>
</head>
<body bgcolor="#CCFFFF"><?php
define('PRET_CARTE',100);
define('PRET_CAJET',50);
define('PRET_PENAR',150);
define('PRET_STILOU',125);
define('PRET_CREION',25);
?><table width="600" border="0"
align="center">
<tr><td></td></tr>
<tr><td height="600" valign="top"
bgcolor="#FFFFCC">
<h1>Magazin online Firma X SRL</h1>
```

```
</td></tr>
</table>
</body>
</html>
```

```
<?php require('antet.php');?>
<h2>Lista Produse</h2>
<table border="1">
...
</table>
<?php require('subsol.php');?>
```

Lista produse – cu design2

Magazin **Firma X SRL**

Magazin online Firma X SRL

Lista Produse

Nr.	Produs	Pret
1	Carti	100
2	Caiete	50
3	Penare	150
4	Stilouri	125
5	Creioane	25

[Comanda](#)

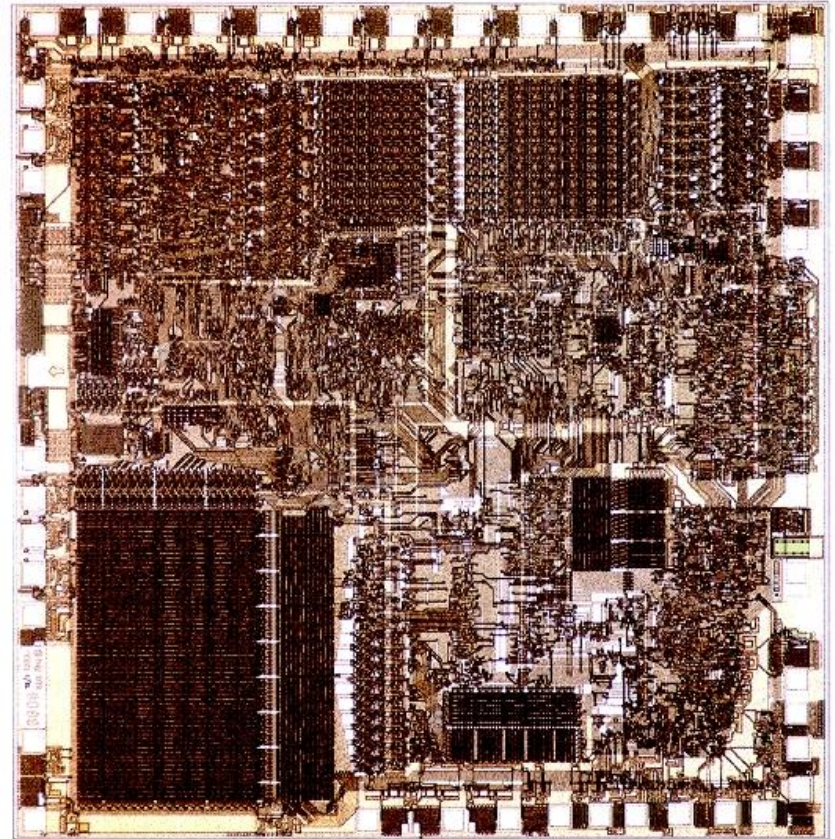
Avantajul lucrului cu sabloane

- viteza de dezvoltare a aplicatiei
- separare clara a formei de aplicatie
- forma unitara
 - “don’t make me think”
- modificarea simultana a formei pentru toate paginile din site
- posibilitatea definirii datelor comune intr-un singur fisier
 - `define('PRET_CARTE',100);`

Structuri repetitive – matrici

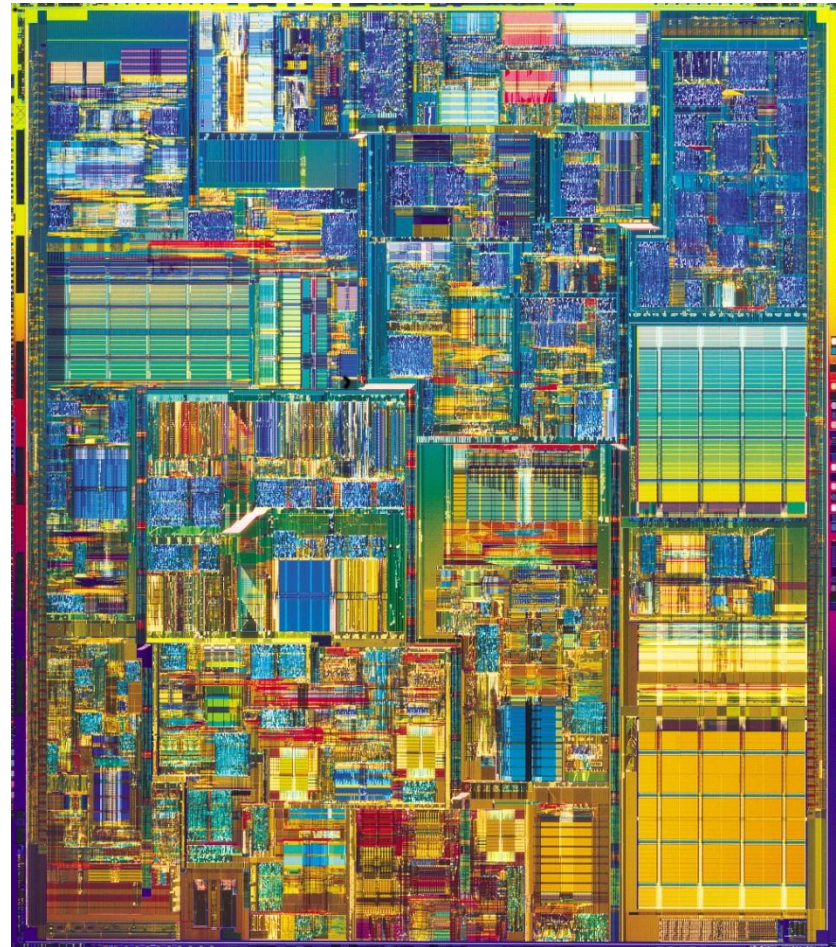
Impresionant?

- Intel® 8086
- 29.000 tranzistoare pe CPU
- 1978
- 1 MB date
- 4.7 MHz



Impresionant?

- Intel® Itanium® processors (Tukwila)
- 2009
- 2 miliarde tranzistoare pe CPU
- 16 EB date (16 G GB)
- > 3 GHz



Concepte

- Efectuare foarte rapida a unui numar **mic** de instructiuni, de **complexitate redusa**, repetate de un numar foarte mare de ori
- Programare: coborarea rationamentului la nivelul de **complexitate redusa**, cu obtinerea performantei prin structuri repetitive simple efectuate rapid.
- Operatii repetitive / date repetitive

Matrici in PHP

- matricea este tipul de variabila care asociaza **valori** unor **chei**
- spre deosebire de C, Basic, **cheile nu sunt** obligatoriu numere **intregi**, pot fi si **siruri**
- implicit cheile sunt intregi succesivi (pentru fiecare element adaugat) si primul element este 0.
- definirea unei perechi cheie / valoare
 - cheie => valoare
- definirea unei matrici
 - `$matr = array("definirea perechilor chei/valori")`

Matrici in PHP

```
$matr = array(1, 2, 3, 4, 5);
```

```
$matr[0]=1
```

```
$matr[1]=2
```

```
$matr[2]=3
```

```
$matr[3]=4
```

```
$matr[4]=5
```

```
$matr = array('a' => 1, 'b' => 2, 3, 4, 5);
```

```
$matr['a']=1
```

```
$matr['b']=2
```

```
$matr[0]=3
```

```
$matr[1]=4
```

```
$matr[2]=5
```

```
$matr= array (  
    "fruits" => array("a" => "orange", "b" => "banana", "c" => "apple"),  
    "numbers" => array(1, 2, 3, 4, 5, 6),  
    "holes" => array("first", 5 => "second", "third")  
);
```

Afisarea matricilor

```
echo "<pre>";  
print_r ($matr);  
echo "</pre>";
```

```
$matr= array (  
"fruits" =>  
array("a" => "orange", "b" => "banana", "c" => "apple",  
"ultim"),  
"numbers" =>  
array(1, 2, 3, 4, 5, 6),  
"holes" =>  
array("first", 5 => "second", "third")  
);  
echo $matr;  
echo "<pre>";  
print_r ($matr);  
echo "</pre>";
```

```
Array  
Array  
(  
  [fruits] => Array  
  (  
    [a] => orange  
    [b] => banana  
    [c] => apple  
    [0] => ultim  
  )  
  [numbers] => Array  
  (  
    [0] => 1  
    [1] => 2  
    [2] => 3  
    [3] => 4  
    [4] => 5  
    [5] => 6  
  )  
  [holes] => Array  
  (  
    [0] => first  
    [5] => second  
    [6] => third  
  )  
)
```

Chei

- Chei numerice
 - implicite
 - similare celorlalte limbaje de programare
 - dificil de utilizat (trebuie retinuta valoarea logica a unei anumite chei numerice)
- Chei sir
 - claritate mai mare
 - eficienta numerica mai mica
 - matricile au un index numeric intern, implicit ascuns, accesibil prin functii :
index => cheie => valoare

Elemente de control

- `for` – util dacă la definirea matricilor sunt folosite cheile numerice implicite (numere întregi)
- `do ... while` și `while` se pot folosi împreună cu funcții specifice caracteristice matricilor `next()`, `prev()`, `end()`, `reset()`, `current()`, `each()`
- `foreach` - elementul de control al iteratiilor cel mai potrivit pentru chei alfanumerice

Elemente de control – foreach

- `foreach (array_expression as $key => $value) statement`
- `foreach (array_expression as $value) statement`
- iterarea prin fiecare element al matricii
- la fiecare element variabila declarata in instructiune **\$key** ofera acces la cheia curenta iar variabila **\$value** ofera acces la valoarea asociata
- `foreach()` lucreaza cu o **copie** a matricii deci matricea originala nu va fi modificata prin schimbarea variabilelor `$key` si `$value`

Elemente de control – foreach

```
$matr = array (  
    "fruits" => array("a" => "orange", "b" => "banana", "c" => "apple", "ultim"),  
    "numbers" => "in loc de numere",  
    "holes" => "in loc de ce era"  
);  
foreach ($matr as $scheie => $continut)  
    echo "matr[".$scheie."]=".$continut."<br />";
```

```
matr[fruits]=Array  
matr[numbers]=in loc de numere  
matr[holes]=in loc de ce era
```

Matrici – functii utile

- `current ($matr)` – **returneaza** elementul indicat de indicele intern al matricii (`~v[i]`)
- `next ($matr)` – incrementeaza indicele intern si **returneaza** valoarea stocata acolo (`~v[++i]`)
- `prev ($matr)` – decrementeaza indicele intern si **returneaza** valoarea stocata acolo (`~v[--i]`)
- `end($matr)` – muta indicele intern la ultimul element si **returneaza** valoarea stocata acolo (`~i=N-1;v[i]`)
- `reset($matr)` – muta indicele intern la primul element si **returneaza** valoarea stocata acolo (`~i=0;v[i]`)

Matrici – functii utile

- `sort($matr)` – ordoneaza in ordine crescatoare a **valorilor** o matrice, cheile sunt sterse si recreate
 - `$fruits = array("lemon", "orange", "banana", "apple");`
`sort($fruits);`
 - `fruits[0] = apple, fruits[1] = banana, fruits[2] = lemon, fruits[3] = orange`
- `rsort($matr)` – similar, descrescator

Matrici – functii utile

- `asort($matr)` ordoneaza in ordine crescatoare a **valorilor** o matrice, cheile sunt pastrate, inclusiv asocierea cheie => valoare
 - `$fruits = array("d" => "lemon", "a" => "orange", "b" => "banana", "c" => "apple");`
`asort($fruits);`
 - `c = apple, b = banana, d = lemon, a = orange`
- `arsort($matr)` – similar, descrescator

Matrici – functii utile

- `ksort($matr)` ordoneaza in ordine crescatoare a **cheilor** o matrice, cheile sunt pastrate, inclusiv asocierea cheie => valoare
 - `$fruits = array("d" => "lemon", "a" => "orange", "b" => "banana", "c" => "apple");`
`ksort($fruits);`
 - a = orange, b = banana, c = apple , d = lemon
- `krsort($matr)` – similar, descrescator

Exemplu utilizare matrici

Exemplu

- In exemplul anterior utilizarea matricilor va aduce urmatoarele avantaje:
 - codul va fi mai concis
 - codul va fi mai general (valabil si pentru 5 produse si pentru 1000)
 - scalabilitate crescuta (se pot adauga usor produse)

Exemplu

- fiecare produs e caracterizat de:
 - nume
 - pret
 - (eventual) descriere
 - cantitate comandata
- putem folosi unul din attribute ca si cheie (numele in exemplu)
- se poate controla (prin atributul name = "") structura variabilei globale `$_POST`

Matrice produse

- una din structurile posibile

```
$produse = array ( 'carte' => array ("descr" => "mai multe pagini scrise", "pret" => 100, "cant" => 0),  
                  'caiet' => array ("descr" => "mai multe pagini goale", "pret" => 50, "cant" => 0),  
                  'penar' => array ("descr" => "loc de depozitat instrumente", "pret" => 150, "cant" => 0),  
                  'stilou' => array ("descr" => "instrument de scris albastru", "pret" => 125, "cant" => 0),  
                  'creion' => array ("descr" => "instrument de scris gri", "pret" => 25, "cant" => 0)  
                );
```

- se urmareste obtinerea unei structuri clare
 - usor de modificat/adaugat date
 - usor de utilizat
- daca definitia se introduce in fisierul antet va fi accesibila in toate fisierele individuale

Crearea listei de produse

Magazin

Firma X SRL

Magazin online Firma X SRL

Lista Produse

Nr.	Produs	Descriere	Pret
1	Carte	mai multe pagini scrise legate	100
2	Caiet	mai multe pagini goale legate	50
3	Penar	loc de depozitat instrumente de scris	150
4	Stilou	instrument de scris albastru	125
5	Creion	instrument de scris gri	25
1	Carte	mai multe pagini scrise legate	100
2	Caiet	mai multe pagini goale legate	50
3	Penar	loc de depozitat instrumente de scris	150
4	Stilou	instrument de scris albastru	125
5	Creion	instrument de scris gri	25
Comanda			

Crearea listei de produse

```
<?php require('antet.php');?>
<h2>Lista Produse</h2>
<table border="1">
<tr bgcolor="#cccccc"><td>Nr.</td><td width="150">Produs</td><td width="150">Descriere</td><td
width="50">Pret</td></tr>
<?php
$index=1;
foreach ($produse as $prod => $detalii)
    {
        echo "<tr><td>".$index."</td><td>".ucfirst(strtolower($prod))."</td><td>".$detalii['descr']."</td><td
align='center'">".$detalii['pret']."</td></tr>";
        $index++;
    }
?>
<?php
$index=1;
foreach ($produse as $prod => $detalii)
    {?>
<tr><td><?php echo $index;?></td><td><?php echo ucfirst(strtolower($prod));?></td><td><?php echo
$detalii['descr'];?></td><td align="center"><?php echo $detalii['pret'];?></td></tr>
<?php $index++;
    }?>
<tr><td colspan="4" align="center"><a href="formular.php">Comanda</a></td></tr></table>
<?php require('subsol.php');?>
```

Subdivizare \$_POST

- atributul **name** in forma devine **cheie** in matricea globala \$_POST
 - `<input type="text" name="carti_cant" size="3" maxlength="3" />`
 - `$_POST['carti_cant']` contine valoarea introdusa
- realizand atributul name ca matrice, se obtine in \$_POST o "submatrice" care grupeaza elementele input
 - `<input type="text" name="cant[carti]" size="3" maxlength="3" />`
 - `$_POST ['cant'] ['carti']` contine valoarea introdusa

Crearea formei de comanda

```
<?php require('antet.php');?>
<h2>Realizati comanda</h2>
<form action="rezultat.php" method="post">
<table border="0">
<tr bgcolor="#cccccc"><td>Nr.</td><td width="150">Produs</td><td width="50">Pret</td><td
width="15">Cantitate</td></tr>
<?php
$index=1;
foreach ($produse as $prod => $detalii)
    {?>
<tr><td><?php echo $index;?></td><td><?php echo ucfirst(strtolower($prod));?></td><td
align="center"><?php echo $detalii['pret'];?></td><td><input name="<?php echo
"cant[".$prod."];?>" type="text" value="0" size="3" maxlength="3" /></td></tr>
<?php $index++;
    }?>
<tr><td colspan="4" align="center"><input type="submit" value="Trimite" /></td></tr>
</table>
</form>
<?php require('subsol.php');?>
```

Crearea listei de produse

Magazin

Firma X SRL

Magazin online Firma X SRL

Realizati comanda

Nr.	Produs	Pret	Cantitate
1	Carte	100	<input type="text" value="0"/>
2	Caiet	50	<input type="text" value="0"/>
3	Penar	150	<input type="text" value="0"/>
4	Stilou	125	<input type="text" value="0"/>
5	Creion	25	<input type="text" value="0"/>

Trimite

Prelucrarea comenzii

```
<?php require('antet.php');?>
<h2>Rezultate comanda</h2>
<p>Pret total (fara TVA):
<?php
$pret=0;
$afis="";
foreach ($_POST['cant'] as $prod => $cant)
    {
        $pret += $cant*$produse[$prod]['pret'];
        $afis .= "+".$cant."x".$produse[$prod]['pret'];
    }
echo $pret;
?>
<p>Obtinut astfel: <?php echo $afis;?></p>
<p>Pret total (cu TVA): <?php echo $pret*1.19;?></p>
<p><?php
echo "<pre>";
print_r ($_POST);
echo "</pre>";
?>
</p>
<p>Comanda receptionata la data: <?php echo date('d/m/Y')." ora ".date('H:i');?></p>
<?php require('subsol.php');?>
```

Prelucrarea comenzii

Magazin

Firma X SRL

Magazin online Firma X SRL

Rezultate comanda

Pret total (fara TVA): 600

Obtinut astfel: +2x100+2x50+2x150+0x125+0x25

Pret total (cu TVA): 714

```
Array
(
    [cant] => Array
        (
            [carte] => 2
            [caiet] => 2
            [penar] => 2
            [stilou] => 0
            [creion] => 0
        )
)
```

Comanda receptionata la data: 17/03/2010 ora 13:55

Scrierea datelor pe disc

- Pentru a oferi posibilitatea beneficiarului aplicatiei (vanzator) sa poata adauga/sterge/modifica produse
 - din interfata browser
 - fara sa aiba cunostinte de programare
- E necesar ca matricea **\$produse** sa fie creata in timpul rularii plecand de la un suport extern de date, accesibil pentru scriere vanzatorului
- Ulterior se va implenta aplicatia ce utilizeaza baze de date – momentan se vor scrie datele pe disc

Utilizarea fisierelor – Functii

- `pointer = fopen(cale,mod)` deschide un fisier pentru operatii descrise de "mod". Se returneaza un pointer spre fisier de tip resursa care va fi folosit la operatiile urmatoare
- `fwrite(pointer,date)` – scrie datele in fisier (date – de tip string)
- `string = fread(pointer,cantitate)` citeste "cantitate" octeti din fisier
- `$matr = file(cale)` deschide fisierul identificat cu "cale" si citeste fiecare linie (incluzand \n) intr-un element distinct in matrice. \$matr de tip array, matrice de siruri

Crearea fisierului

```
$produse = array ( 'carte' => array ("descr" => "mai multe pagini scrise legate", "pret" => 100, "cant" => 0),  
                  'caiet' => array ("descr" => "mai multe pagini goale legate", "pret" => 50, "cant" => 0),  
                  'penar' => array ("descr" => "loc de depozitat  
instrumente de scris", "pret" => 150, "cant" => 0),  
                  'stilou' => array ("descr" => "instrument de scris  
albastru", "pret" => 125, "cant" => 0),  
                  'creion' => array ("descr" => "instrument de scris  
gri", "pret" => 25, "cant" => 0)  
                );  
$handle = fopen("produse.txt", "wb");  
foreach ($produse as $prod => $detalii)  
    fwrite($handle,$prod."\t".$detalii['descr']."\t".$detalii['pret']."\t".$detalii['cant']."\r\n");
```

Crearea fisierului

- crearea initiala se poate face prin modificarea o singura data a fisierului antet.php existent astfel incat sa scrie datele pe disc

```
$produse = array ( 'carte' => array ("descr" => "mai multe pagini scrise", "pret" => 100, "cant" => 0),  
                  'caiet' => array ("descr" => "mai multe pagini goale", "pret" => 50, "cant" => 0),  
                  'penar' => array ("descr" => "loc de depozitat instrumente", "pret" => 150, "cant" => 0),  
                  'stilou' => array ("descr" => "instrument de scris albastru", "pret" => 125, "cant" => 0),  
                  'creion' => array ("descr" => "instrument de scris gri", "pret" => 25, "cant" => 0)  
                );  
$handle = fopen("produse.txt", "wb");  
foreach ($produse as $prod => $detalii)  
    fwrite($handle,$prod."\t".$detalii['descr']."\t".$detalii['pret']."\t".$detalii['cant']."\r\n");
```

Citirea fisierului pentru crearea matricii

```
$matr=file("produse.txt");  
echo "<pre>";  
print_r($matr);  
echo "</pre>";  
foreach ($matr as $linie)  
{  
    $valori=explode("\t",$linie,4);  
    $produse[$valori[0]]=array("descr" => $valori[1], "pret" => $valori[2], "cant" => $valori[3]);  
}
```

Array

```
(  
    [0] => carte          mai multe pagini scrise legate  100    0  
    [1] => caiet         mai multe pagini goale legate   50    0  
    [2] => penar         loc de depozitat instrumente de scris 150    0  
    [3] => stilou        instrument de scris albastru    125    0  
    [4] => creion        instrument de scris gri 25      0
```

Laborator

Laborator 5

- Sa se continue magazinul virtual cu:
 - produsele sunt grupate pe categorii de produse
 - sa prezinte utilizatorului o lista de grupe de produse pentru a alege
 - sa prezinte utilizatorului o lista de produse si preturi in grupa aleasa
 - lista de produse si preturi se citeste dintr-un fisier
 - se preia comanda si se calculeaza suma totala
 - se creaza o pagina prin care vanzatorul poate modifica preturile si produsele

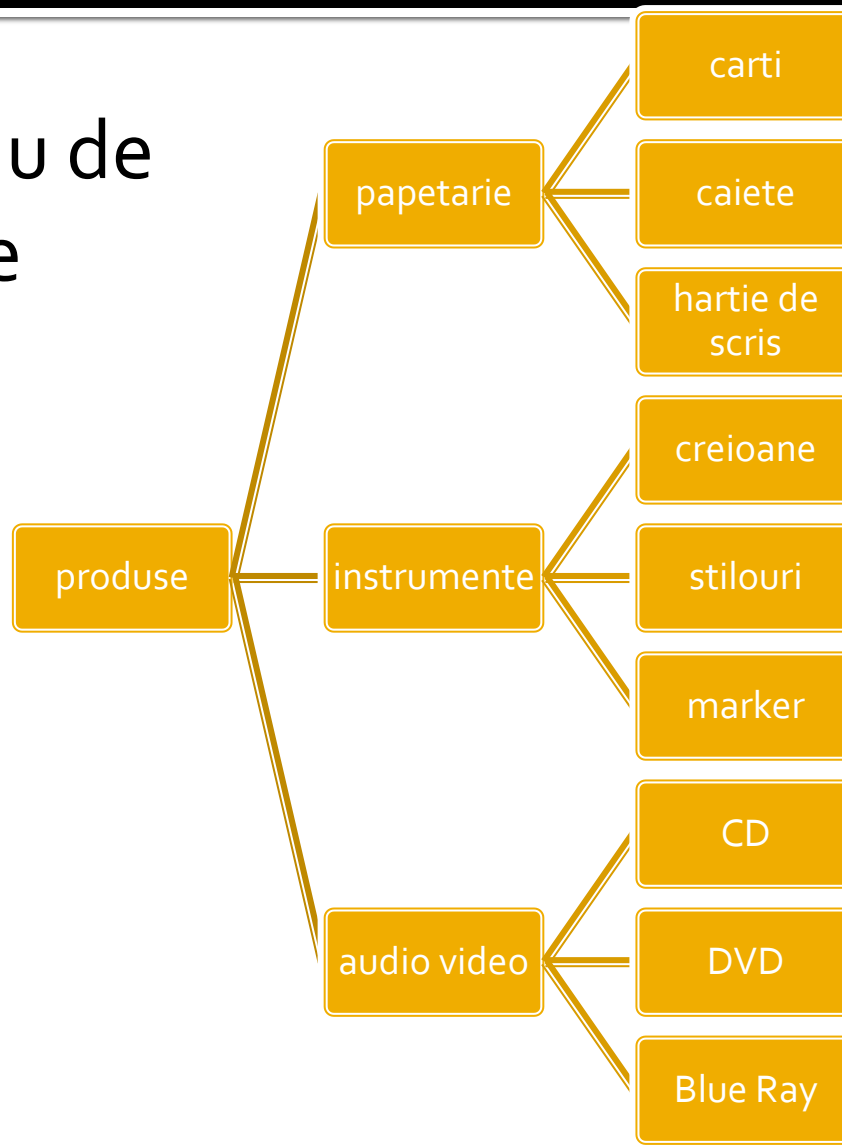
Laborator 5

- se lucreaza cu matrici
- forma paginilor:
 - exemplu alaturat, prin CSS
- exemplu de grupare pe pagina urmatoare

culoare	IMAGINE	culoare
	Continut (cu alta culoare fundal)	
	Copyright (cu alta culoare fundal)	

Laborator 5

- exemplu de grupare



Contact

- Laboratorul de microunde si optoelectronica
- <http://rf-opto.etti.tuiasi.ro>
- rdamian@etti.tuiasi.ro