

Facultatea Calculatoare, Informatica si Microelectronica
Universitatea Tehnica a Moldovei

Medii Interactive de Dezvoltare a Produselor Soft
Lucrarea de laborator#2

GUI Development

Autor:
Lungu Dan

lector asistent:
Victor Gojin

lector superior:
Radu Melnic

1. Scopul lucrării de laborator

Realizarea unui simplu GUI calculator.

2. Mersul lucrării de laborator

2.1 Cerințele

1.Realizează un simplu GUI calculator care suportă următoare funcții: +, -, /, *, putere, radical,inversare semn(+/-), operații cu numere zecimale.

2.Divizarea proiectului în două module - Interfața grafică(Modul GUI) și Modulul de bază(Core Module).

2.2 Analiza Lucrării de laborator

În lucrarea dată am folosit programarea în PHP pentru a crea un GUI Calculator.

In primul rind am creat un Class cu denumirea Calculator in care am declarat 2 variabile pentru cifre. Am creat 2 functii: una pentru operator si alta pentru rezultat. Pentru a afisa operatorii, cifrele si rezultatele am folosit limbajul HTML si CSS.

```
<?php
$rezult = "";
class calculator
{
    var $a;
    var $b;

    function operatie($oprator)
    {
        switch($oprator)
        {
            case '+':
                return $this->a + $this->b;
```

```

        break;

        case '-':
            return $this->a - $this->b;
            break;

        case '*':
            return $this->a * $this->b;
            break;

        case '/':
            return $this->a / $this->b;
            break;

        case 'sqrt':
            return sqrt($this->a);
            break;

        case '^':
            return pow($this->a, $this->b);
            break;

        default:
            return "Error";
    }
}

function rezultat_1($a, $c)
{
    $this->a = $a;
    return $this->operatie($c);
}

function rezultat_2($a, $b, $c)
{
    $this->a = $a;

```

```

        $this->b = $b;

        return $this->operatie($c);
    }
}

$ca = new calculator();
if(isset($_POST['submit']))
{
    if(isset($_POST['nr2'])== NULL)
    {
        $rezult = $ca->rezultat_1($_POST['nr1'],$_POST['operatie']);
    }else{
        $rezult = $ca->rezultat_2($_POST['nr1'],$_POST['nr2'],$_POST['operatie']);
    }
}

?>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

<head>

<style type="text/css">

.box {
    border: 5px solid #434343;
    background: gray;
    padding: 10px;
    display: block;
    width: 18%;
    position: absolute;
    left: 50%;
    right: 50%;
    transform: translate3d(-50%, 50%, 0);
    border-radius: 5px;
}

input[type=button], input[type=reset], input[type=submit]
{

```

```

background: -webkit-linear-gradient(left, #000000 ,#434343); /* For Safari 5.1 to 6.0 */
background: -o-linear-gradient(right,#000000 ,#434343); /* For Opera 11.1 to 12.0 */
background: -moz-linear-gradient(right,#000000 ,#434343); /* For Firefox 3.6 to 15 */
background: linear-gradient(to right, #000000 ,#434343); /* Standard syntax */

border: none; color: white; padding: 14px 30px; text-decoration: none; margin: 4px 2px;
cursor: pointer; position: relative;border-radius: 10px;display: block;}

input[type=button], input[type=reset], input[type=submit]:hover
{
background: -webkit-linear-gradient(left, #434343 ,#000000); /* For Safari 5.1 to 6.0 */
background: -o-linear-gradient(right,  #434343 ,#000000); /* For Opera 11.1 to 12.0 */
background: -moz-linear-gradient(right, #434343 ,#000000); /* For Firefox 3.6 to 15 */
background: linear-gradient(to right,  #434343 ,#000000); /* Standard syntax */

border: none; color: white; padding: 14px 30px; text-decoration: none; margin: 4px 2px;
cursor: pointer; position: relative;border-radius: 10px;display: block;}

input[type=number], input[type=text]{
    border:1px solid #434343;
    width: 100%;
    height: 10px;
    padding: 12px 20px;
    margin: 8px 0;
    box-sizing: border-box;
    border-radius: 5px;
    display: block;
}

.rezultat {
position: relative;
width:100px;

}

.sub{
position: relative;
left: 50px;
}

td {

```

```
color: white;
```

```
font-size: 16px;
```

```
}
```

```
</style>
```

```
<script>
```

```
function maxLengthCheck(object) {  
    if (object.value.length > object.max.length)  
        object.value = object.value.slice(0, object.max.length)  
}
```

```
</script>
```

```
<script src="http://code.jquery.com/jquery-1.9.1.js"></script>
```

```
</head>
```

```
<form method="post">
```

```
    <div class="box">
```

```
<table align="center">
```

```
    <tr>
```

```
        <td>Nr.1</td>
```

```
        <td><input type="number" name="nr1" placeholder="Numarul 1.." autocomplete='off'  
oninput="maxLengthCheck(this)" min = "1" max = "9999999999" required></td>
```

```
    </tr>
```

```
    <tr>
```

```
        <td>Nr.2</td>
```

```
        <td><input type="number" id="nr2" name="nr2" placeholder="Numarul 2.." autocomplete='off'  
oninput="maxLengthCheck(this)" min = "1" max = "9999999999" required></td>
```

```
    </tr>
```

```
    <tr>
```

```
        <td>Operatia</td>
```

```

        <td><select id="operatie" name="operatie">

            <option value="+">    +    </option>

            <option value="-">    -    </option>

            <option value="*">    *    </option>

            <option value="/">    /    </option>

            <option value="sqrt">    sqrt    </option>

            <option value="^">    ^    </option>

        </select></td>

    </tr>

    <script>

    $('#operatie').change(function(e){

    if ($('#operatie').val() == 'sqrt'){

        $('#nr2').css('display','none');

        $('#nr2').prop('required',false);

    }else{

        $('#nr2').show();

        $('#nr1').prop('required',true);

        $('#nr2').prop('required',true);

    }

    });

    </script>

    <tr><td>Rezultat</td><td><input readonly type="text" value="<?php echo $rezult; ?>"
placeholder="Rezultat..."></td></tr>

    <tr><td><input type="submit" name="submit" value="    =    "></td>

    <td><input type="reset" value="Reset"></td></tr>

</table>

</div>

</form>

```

3. Screen-uri



The image shows a simple calculator interface with a dark gray background. It features four input fields on the left side, each with a label and a corresponding text box on the right. The labels are 'Nr.1', 'Nr.2', 'Operatia', and 'Rezultat'. The text boxes contain 'Numarul 1..', 'Numarul 2..', '+ ▼', and 'Rezultat...' respectively. Below the input fields, there are two buttons: a dark gray button with an equals sign '=' and a dark gray button with the text 'Reset'.

4. Concluzie

În urma efectuării lucrării de laborator, am învățat cum să realizez un simplu GUI calculator care suportă următoarele funcții: +, -, /, *, putere, radical, inversare semn(+/-), operații cu numere zecimale. Am aplicat programarea în PHP, mi-am aprofundat cunoștințele.