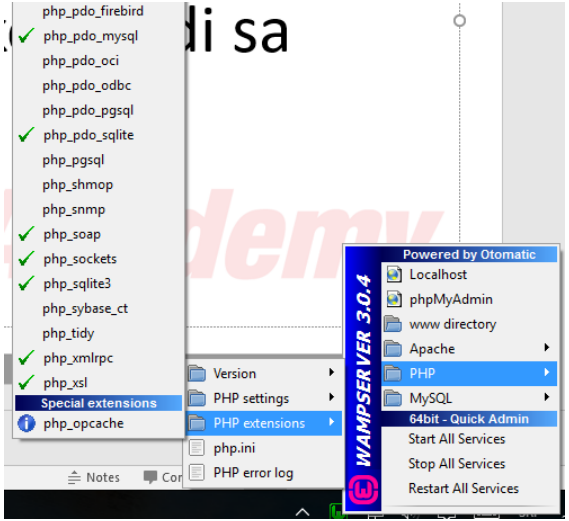
Objekti sa rad sa bazama podataka

Advanced PHP

ITAcademy

PDO

- PDO (Php Data Object) je php biblioteka koja nudi jedinstveni interfejs u radu sa drugim bazama podataka, pored MySQL-a
- Mora biti uključena biblioteka koja radi sa određenom bazom podataka(



PDO

- Uspostavljanje i zatvaranje konekcije

```
<?php
    $pdo= new PDO("mysql:host=localhost;dbname=test", "root", "");
    if(!$pdo)
        echo "Greska";
    else
        echo "Uspeh";
   /*
    * rad sa bazom
    */
   unset ($pdo);//$pdo=null;
   ?>
```

PDO

- Izvršavanje upita bez resultset-a

```
$brojUnetihRedova=$pdo->exec("INSERT INTO korisnici VALUES (null, 'Bosko')") or
die("Neuspeo upit.<br>".print_r($pdo->errorInfo(), true));
echo "Broj unetih redova: ".$brojUnetihRedova;

$brojIzmenjenihRedova=$pdo->exec("UPDATE korisnici SET ime='Pera' WHERE ime='Bosko'");
echo "Broj izmenjenih redova: ".$brojIzmenjenihRedova;

$brojObrisanihRedova=$pdo->exec("DELETE FROM korisnici WHERE ime='Pera'");
echo "Broj obrisanih redova: ".$brojObrisanihRedova;
.......
echo "Poslednji dodat id: ".$pdo->lastInsertId();
......
```

PDO

- Izvršavanje upita sa resulset-om

```
$rez=$pdo->query("SELECT * FROM korisnici");//Upit za izvršavanje
..........
foreach($rez as $red)
    echo $red[0]." ".$red['ime']."<br>";
Ili

$pomRez=$rez->fetchAll(PDO::FETCH_ASSOC);
foreach($pomRez as $k=>$v)
    echo "Red: ".$k.": id=".$v['id'].", ime=
".$v['ime']."<br>";
```

PDO

```
class Korisnik
{
    public $id;
    public $ime;
};
$rez=$pdo->query("SELECT * FROM korisnici");
$pomRez=$rez->fetchAll(PDO::FETCH_CLASS,"Korisnik");
foreach($pomRez as $k)
    echo "id=".$k->id.", ime= ".$k->ime."<br>";
```

PDO

- Priprema upita i parametara

```
$sql=$pdo->prepare("INSERT INTO korisnici VALUES(null, :ime)");
$ime="Mile";
$sql->bindParam(":ime", $ime);
$sql->execute();
```

PDO

- Rukovanje transakcijama

```
$pdo->beginTransaction();
$pdo->exec("INSERT INTO korisnici VALUES (null,
'Stefan')");
$rez=$pdo->query(("SELECT * FROM korisnici WHERE
ime='Stefan'"));
foreach($rez as $red)
 echo $red['id']." ".$red['ime']."<br>";
$pdo->rollBack();//Baza mora da bude InnoDB (ne
MyISAM)
```

SQLite

- SQLite je mala , portabilna, efikasna i relativno brza baza podataka
- Koristimo je kad želimo da rukujemo sa umerenom količinom podataka i kada nam nisu neophodne jake performanse
- Ne zahteva nikakav sistem za skladištenje, već se smešta u običnu datoteku
- Potrebno je da se uključi biblioteka sa rad sa SQLite

SQLite - konekcija

```
$db = new SQLite3("mojaBaza.sqlite");
if(!$db)
{
    echo "Greska!!!";
    exit();
}
else echo "Succes!<hr>";
```

- Ukoliko datoteka već postoji, ostvariće se konekcija na nju, neće se izbrisati postojeća i otvarati nova

SQLite – kreiranje baze

- Postoje razvojna okruženja za rad sa SQLite bazama, ali najčešći (i najjednostavniji) način rada je sa SQL upitima

```
$db->exec("CREATE TABLE korisnici (id integer primary key, ime
varchar(255))");
.........
$db->exec("INSERT INTO korisnici VALUES (null, 'Petar')");
echo "Dodat id: ".$db->lastInsertRowID()."<br>";
$db->exec("INSERT INTO korisnici VALUES (null, 'Bosko')");
echo "Dodat id: ".$db->lastInsertRowID()."<br>";
$db->exec("INSERT INTO korisnici VALUES (null, 'Jovan')");
echo "Dodat id: ".$db->lastInsertRowID()."<br>";
.........
```

SQLite - upiti

```php
$rez=$db->query("SELECT * FROM korisnici;");
while($red=$rez->fetchArray())
    echo "id: ".$red[0]." ime=".$red['ime']."<br>";
```

Samo asocijativni niz

```php
$rez=$db->query("SELECT * FROM korisnici;");
while($red=$rez->fetchArray(SQLITE3_ASSOC))
    echo "id: ".$red['id']." ime=".$red['ime']."<br>";
```

Samo numerički niz

```php
$rez=$db->query("SELECT * FROM korisnici;");
while($red=$rez->fetchArray(SQLITE3_NUM))
    echo "id: ".$red[0]." ime=".$red[1]."<br>";
```

SQLite – uništavanje konekcije

- unset ($db);

SQLite i PDO

```
$pdo=new PDO("sqlite:mojaBaza.sqlite");
$rez=$pdo->query("SELECT * FROM korisnici");
foreach($rez as $red)
    echo "id: ".$red['id']." ime: ".$red['ime']."<br>";
```

Klase i objekti

Advanced Java Script

Agenda

- Prototype objeltni model
- Kreiranje i instanciranje klasa
- Prototipovi
- Događaji

Prototype objektni model

- Ne postoji nasleđivanje
- Funkcionalnost se dopunjuje putem prototipova

Kreiranje klasa u JS

- Putem funkcija:

```
function MojaKlasa(){}
```

- Dodavanje polja

```
function MojaKlasa()
{
    this.x=10;
}
```

Kreiranje klasa u JS

- U zavisnosti od JS verzije na računaru

```
class MojaKlasa{
    constructor(){
        this.x=5;
        this.y=6;
    }
}
var obj=new MojaKlasa();
alert(obj.x+" "+obj.y);
```

- Može da se dogodi da ne radi na svim računarima

Instanciranje objekta klase

```
var mk=new MojaKlasa();
alert(mk.x);
```

- Atributi klase se mogu instancirati u prilikom kreiranja objekta bez da su prethodno definisani

```
var mk={x:20, y:30};
alert(mk.x+" "+mk.y);
```

Instanciranje objekta klase

- Naknadno dodavanje polja

```
function MojaKlasa()
{
    this.x=10;
}


var mk=new MojaKlasa();
mk.y=35;
alert(mk.y);
```

Kreiranje klasa u JS

- Kreiranje parametrizovanog konstruktora

```
function MojaKlasa(prom)
{
    this.x=prom;
}
```

- Opciona parametrizacija u konstruktoru

```
function MojaKlasa2(prom1, prom2)
{
    this.x=prom1;
    this.y=(prom2==null)?0:prom2;
}
```

- Automatsko preuzimanje parametara

```
function MojaKlasa3()
{
    for(i in arguments)
        console.debug(arguments[i]);
}
var mk1=new MojaKlasa3(2,3,4,5,6,7);
```

Kreiranje klasa u JS

- Dodavanje metoda

```
function MojaKlasa()
{
     this.mojaMetoda=mojaMetoda;
     function mojaMetoda()
     {
          alert("Aktivirana mojaMetoda");
     }

     this.mojaMetoda2=function(){
          return "Aktivirana mojaMetoda2";
     }
}
var obj=new MojaKlasa();
 obj.mojaMetoda();
 alert(obj.mojaMetoda2());
```

Kreiranje klasa u JS

- Naknadno dodavanje metoda

```
function MojaKlasa()
{
    this.mojaMetoda=mojaMetoda;
}
function mojaMetoda()
{
    alert("Aktivirana mojaMetoda");
}

var obj=new MojaKlasa();
 obj.mojaMetoda();
```

Primer

- Napraviti klasu korisnik koja ima polja
  - Id
  - Ime
  - Prezime
  - Datum rođenja
  - Jmbg
  - Email
- Klasa mora posedovati parametrizovan konstruktor
- Klasa mora posedovati metod toString, koja vraća sva polja korisnika u vidu stringa

Rešenje

```
function Korisnik(id, ime, prezime, dr, jmbg, email)
{
    this.id=id;
    this.ime=ime;
    this.prezime=prezime;
    this.dr=dr;
    this.jmbg=jmbg;
    this.email=email;
    this.toString=toString;
    function toString(){
        return "id="+this.id+"\nime="+this.ime+"\nprezime="+this.prezime+
"\ndatum rodjenja="+this.dr+"\njmbg="+this.jmbg+"\nemail="+this.email;
    }
}
var obj=new Korisnik(1, "Bosko", "Bogojevic", "20.01.1976", "11111111111",
"email@gmail.com");
alert (obj.toString());
```

Kreiranje i rukovanje događajima

- Događaji
  - Registrovane promene statusa unutar klasa
  - Aktiviraju se ručno, na zahtev instance klase
  - Svaki događaj aktivira funkciju koja je predviđena za njega
  - Događaji se vezuju za hendlere prilikom instanciranja klase

```
function Cal(x,y)
{
     this.x=x;
     this.y=y;
     this.saberi=saberi;
     this.onCalculate=function(){}

     function saberi()
     {
          this.onCalculate();
          return this.x+this.y;
     }
}
var c=new Cal(5,4);
c.onCalculate=function(){alert(„Sabiranje je izvrseno")};
alert(c.saberi());
```

Prototipovi

- Omogućavaju da se, postojećoj klasi, doda funkcionalnost

```
Klasa.prototype.novaFunkcija=function(){
        alert(„nova funkcija“);
}
```

Problem

- Postoji klasa Array koja radi sa nizovima. Dodati novu funkcionalnost izbacivanja parnih brojeva iz niza metodom „removeEven“

Rešenje

```
Array.prototype.removeEven=function()
{
    var velicina=this.length;
    for(var i=0;i<velicina;i++)
        if(this[i]%2==0)
                this.splice(i--, 1);
}
var niz=[1,2,3,4,5,6,7,8,9];
alert(niz);
niz.removeEven();
alert(niz);
```