UNIVERZA V LJUBLJANI

FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Tavčar Leon

Analiza in primerjava PHP in JSP

DIPLOMSKO DELO

VISOKOŠOLSKI STROKOVNI ŠTUDIJSKI PROGRAM PRVE STOPNJE RAČUNALNIŠTVO IN INFORMATIKA

Ljubljana, 2012

UNIVERZA V LJUBLJANI

FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Tavčar Leon

Analiza in primerjava PHP in JSP

DIPLOMSKO DELO

VISOKOŠOLSKI STROKOVNI ŠTUDIJSKI PROGRAM PRVE STOPNJE RAČUNALNIŠTVO IN INFORMATIKA

Mentor: prof. dr. Matjaž Branko Jurič

Univerza
v Ljubljani

Fakulteta za računalništvo
in informatiko

Tržaška 25
1000 Ljubljana, Slovenija
telefon: 01 476 84 11
faks: 01 426 46 47
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 00146/2011
Datum: 02.09.2011

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **LEON TAVČAR**

Naslov: **ANALIZA IN PRIMERJAVA PHP IN JSP**
ANALYSIS AND COMPARISON OF PHP AND JSP

Vrsta naloge: Diplomsko delo visokošolskega strokovnega študija prve stopnje

Tematika naloge:

Proučite tehnologije za razvoj spletnih aplikacij. Opravite natančen pregled in analizo JSP in PHP. Primerjajte temeljne koncepte v PHP in JSP. Izdelajte spletno aplikacijo v izbrani tehnologiji.

Mentor:

prof. dr. Matjaž B. Jurič

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Dekan:

prof. dr. Nikolaj Zimic

Ljubljana, 2012

I Z J A V A O A V T O R S T V U

diplomskega dela

Spodaj podpisani/-a _____,

z vpisno številko _____,

sem avtor/-ica diplomskega dela z naslovom:

_____

_____

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal/-a samostojno pod mentorstvom (naziv, ime in priimek)

  _____

  in somentorstvom (naziv, ime in priimek)

  _____

- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela

- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki »Dela FRI«.

V Ljubljani, dne _____ Podpis avtorja/-ice: _____

Zahvala

Najprej bi se rad zahvalil svojim staršem za moralno in finančno podporo skozi vsa študijska leta.

Zahvaljujem se tudi mentorju, prof. dr. Matjažu Branku Juriču, za strokovne nasvete in pomoč.

Kazalo

Povzetek

Cilj diplomske naloge je primerjava dveh najpogosteje uporabljenih programskih jezikov za razvoj dinamičnih spletnih aplikacij. Danes se največ uporablja jezik PHP, ki je bil že izvirno zamišljen za vzdrževanje osebnih domačih strani in je nato prerasel v programski jezik s številnimi možnostmi, s katerim je mogoče upravljati velika spletna okolja. JSP je tehnologija Java, ki se danes vse več uporablja in ravno tako omogoča upravljanje velikih spletnih okolij.

Spletne aplikacije vse bolj izpodrivajo klasične aplikacije, saj jih ni potrebno nameščati na uporabnikove računalnike in omogočajo enostavno uporabo z brskalnikom, zato postajajo tehnologije za razvoj spletnih aplikacij vedno bolj pomembne in izpopolnjene. Konceptov, ki se uporabljajo pri razvoju, je veliko, zato je smiselno primerjati glavne.

Na praktičnem primeru spletne aplikacije so uporabljeni, za primerjavo, glavni koncepti spletnih tehnologij PHP in JSP, temeljni elementi, delo z nizi, delo z obrazci, delo z zbirko podatkov, piškoti in seje.

V zadnjem poglavju so opisana uporabljena razvojna orodja, spletna strežnika in zbirka podatkov, ki so nujni za razvoj dinamičnih spletnih aplikacij in so med najbolj uporabljenimi med razvijalci.

Abstract

The aim of the diploma thesis was to perform a comparison of the two most frequently used programming languages for developing dynamic Web applications. The PHP language which is most frequently used today was originally developed to maintain personal homepages and was later expanded into a programming language which provides numerous possibilities for managing large Web environments. JSP is a Java technology whose use is increasing that also enables the management of large Web environments.

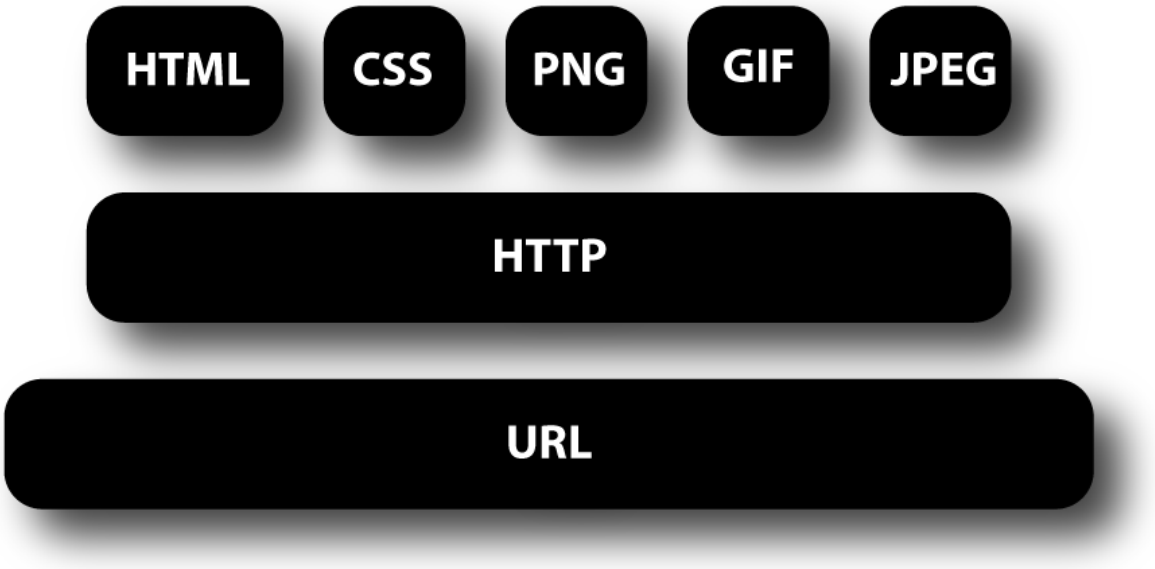
Web technologies are increasingly replacing classic applications for do not need to be installed on the user's computer and enable simple browser use. This has resulted in the growing importance and supplementation of Web application development technologies. Numerous concepts used in their development exist therefore a comparison of the key concepts would be wise.

The key concepts of PHP and JSP Web technologies, key elements, work with sets, forms and data collections and cookies and seeds were used to compare the two Web applications.

The last chapter describes the development tools required for the development of dynamic Web applications most widely used by developers: Web servers and data collections.

1 Uvod

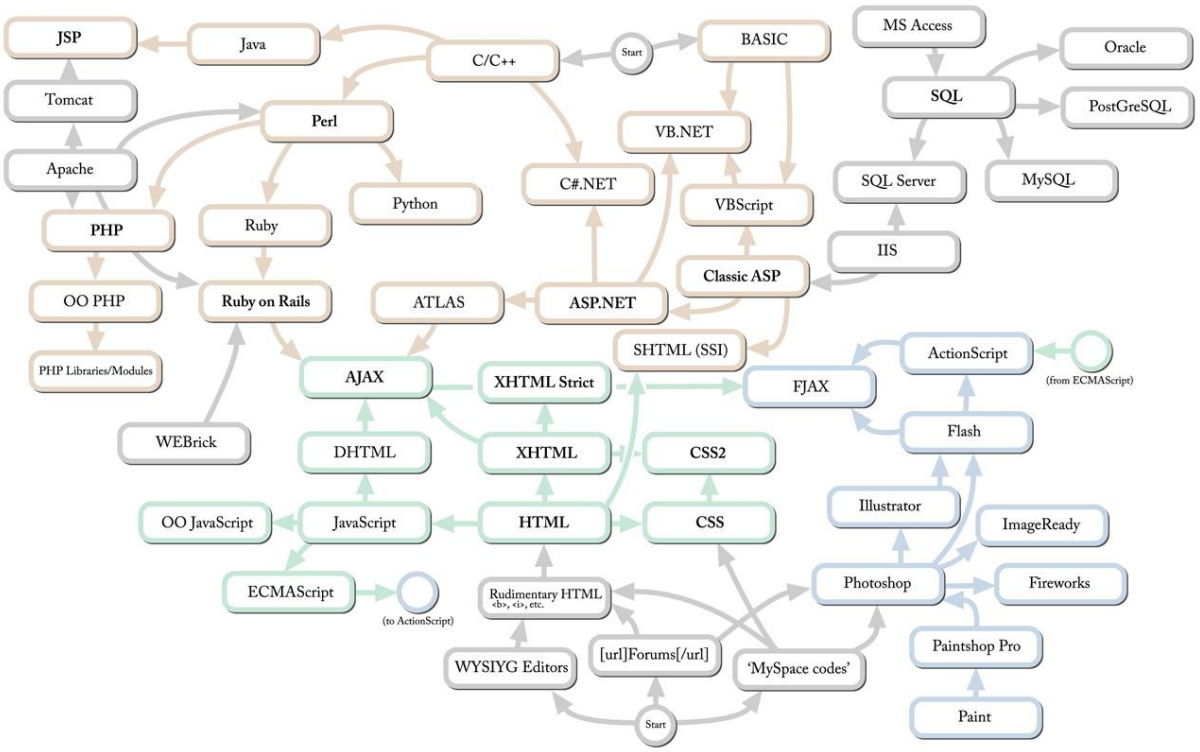
Predlog za splet, kot ga poznamo danes, je prvič uradno javnosti predstavil Tim Berners-Lee 12. novembra 1990. Nastala so vsa potrebna orodja za delujoč splet WorldWideWeb: spletni brskalnik in spletni strežnik. Namen spleta naj bi bilo čim več akademskih informacij brezplačno dostopnih vsakomur. Začele so se izdelovati spletne strani, ki so v računalništvu dokumenti s hiperbesedilom, ki jih prikaže brskalnik. Na spletni strani so lahko različne vsebine: besedilo, slike, povezave, zvočni in video posnetki. Danes je svetovni splet porazdeljen hipertekstni sistem, ki deluje v medmrežju. Slika 1 prikazuje arhitekturo svetovnega spleta, ki sloni na treh standardih, ki jih določa W3C: URL, HTTP in HTML [1].



Slika 1: Arhitektura svetovnega spleta (Vir: The Web according to W3C. Dostopno na: http://www.w3.org/Talks/2005/1227-W3C-22C3)

Čedalje bolj prihajajo do izraza spletne aplikacije. To so aplikacije, dostopne preko omrežja, ki postajajo čedalje bolj zapletene in zahtevne, tako s strani načrtovanja, kot izdelovanja. Velik pomen pri izdelovanju imajo tehnologije in programski jeziki, ki nam omogočajo izdelavo velikih spletnih okolij.

Pri načrtovanju spletnih aplikacij se pojavi vprašanje, katero tehnologijo uporabiti, kajti teh je veliko, kot je razvidno iz slike 2. Na spletu je veliko mnenj na temo izbire prave tehnologije. Večina jih prisega na PHP, ki je enostaven za učenje in odprtokoden. Po drugi strani imamo privržence objektno orientiranih jezikov, ki prisegajo na Javo.



Slika 2: Množičnost spletnih tehnologij (Vir: Web Technology Family Tree. Dostopno na: http://modernl.com/article/web-tech-family-tree)

V diplomskem delu raziskujemo razlike dveh tehnologij, ki sta med najbolj uporabljenimi tehnologijami za razvoj spletnih aplikacij.

V 2. poglavju smo predstavili razvoj današnjih spletnih aplikacij, kaj sploh so, kako so sestavljene, katere platforme obstajajo za nadaljnji razvoj, kam gre razvoj spleta in najpomembnejše tehnologije spleta 2.0.

Pregled tehnologij JSP in PHP opisujeta poglavji 3 in 4, kjer smo opisali osnovno delovanje in uporabo, skriptne elemente in direktive.

Koncepte tehnologij PHP in JSP opisujemo v 5. poglavju, kjer se nahaja tudi primerjava temeljnih elementov, delo z nizi, delo z obrazci, delo z zbirko podatkov, uporaba piškotkov in sej.

V 6. poglavju opisujemo praktično aplikacijo, v kateri smo preizkusili delovanje konceptov, spletne strežnike za poganjanje PHP in JSP, delo z zbirko podatkov MySQL, katera razvojna orodja so bila uporabljena in implementacijo.

Sklepno 7. poglavje povzema bistvene razlike in ugotovitve v diplomskem delu.

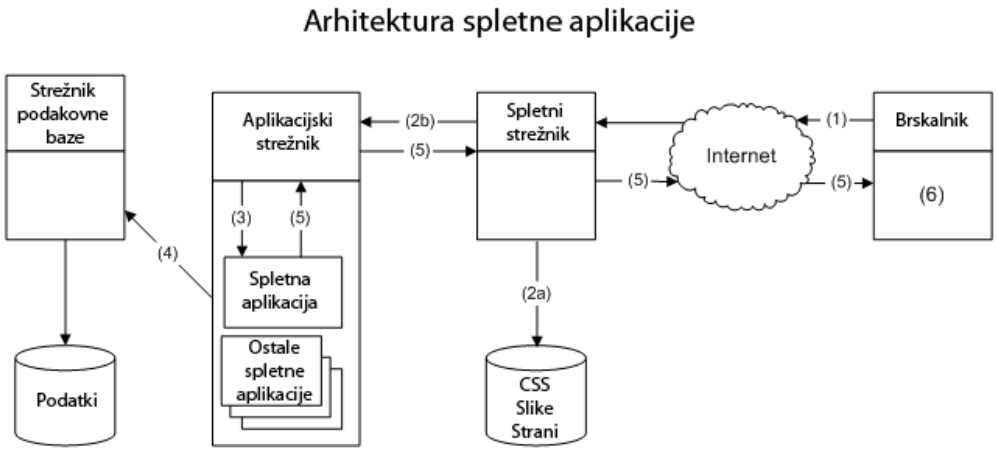
2 Razvoj spletnih aplikacij

2.1 Kaj so spletne aplikacije

Spletne aplikacije so aplikacije, ki so dostopne preko omrežja, kot sta internet ali intranet. Izvajamo lahko programsko opremo, ki jo gostimo v brskalnikovem nadzorovanem okolju (npr. Java applet) ali v brskalniku, ki podpira jezike, kot sta označevalni jezik HTML ali skriptni jezik JavaScript. Priljubljene so, ker nam omogočajo enostavno uporabo z brskalnikom in sposobnosti posodabljanja ter vzdrževanja spletnih aplikacij brez nameščanja programske opreme na potencialno tisoče odjemalskih računalnikov [2]. Družbeno mrežo sestavlja vse več spletnih orodij in platform, kjer lahko ljudje delijo svoje poglede, mnenja, misli in izkušnje. Aplikacije spleta 2.0 temeljijo na spletnem sodelovanju in izmenjavi informacij med uporabniki. Danes se aplikacije spleta 2.0 največ uporablja in so lahko sestavljene iz:

- blogov in mikroblogov,
- wikijev,
- označevanj spletnih strani z meta oznakami, ki opisujejo njihove vsebine in pripomorejo k lažjemu iskanju,
- spletnih mest za socialna omrežja,
- RSS naročanja in
- multimedijskih skupnosti (photo/video sharing)

Slika 3 prikazuje arhitekturo enostavne spletne aplikacije, kjer brskalnik pošlje spletnemu strežniku zahtevo po določeni vsebini. Strežnik pošlje spletno stran, predlogo in slike v primeru, ko gre za statične vsebine. Ko pa odjemalec zahteva dinamično vsebino, se zahteva posreduje aplikacijskemu strežniku, ki zahtevo preusmeri določeni spletni aplikaciji, ta pa zgradi odgovor. Podatki se večinoma hranijo v zbirki podatkov, ki je lahko nameščena lokalno na samem spletnem strežniku ali ločenem namenskem strežniku.



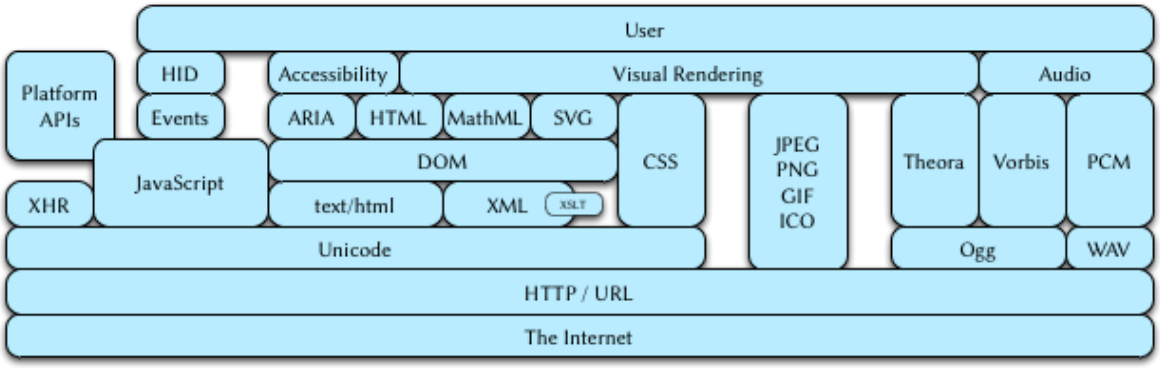
Slika 3: Primer arhitekture enostavne spletne aplikacije (Vir: AppThena. Dostopno na: http://www.appthena.com/developer/manual/architecture.html)

2.2 Katere platforme obstajajo

Spletne platforme sestojijo iz tehnologij, ki se v splošnem uporabljajo za obdelavo vsebine v spletu. Spletne platforme so lahko odprte (brezplačne) ali plačljive.

Slika 4 prikazuje brezplačne tehnologije, iz katerih sestojijo platforme in jih lahko vsi uporabljajo brez plačila licenčnine [3]. Seznam odprtih standardov, ki jih določa World Wide Web Consortium (W3C):

- HTML
- DOM
- CSS
- SVG
- MathML
- Spletni API-ji
- EcmaScript / JavaScript
- HTTP
- URI



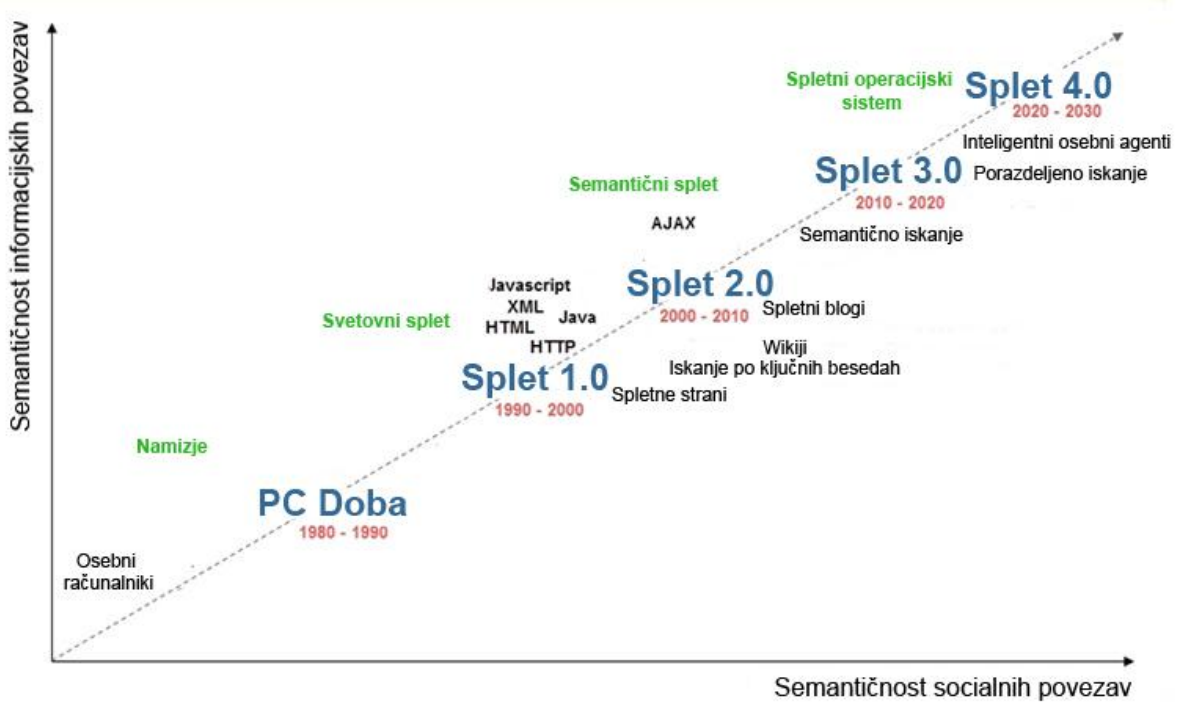
Slika 4: Diagram spletnih tehnologij, ki lahko sestavljajo platforme (Vir: (2009)The Web platform: what it is. Dostopno na: http://edward.oconnor.cx/2009/05/what-the-web-platform-is)

Aplikacije spleta 2.0 so lahko sestavljene iz različnih platform, le te pa so temelji, na katerih se gradijo aplikacije. Danes obstaja vrsta platform, ki jih v grobem delimo na bloge, mikrobloge, wikije, spletna mesta za socialna omrežja in sisteme CMS. Omogočajo hitro in enostavno izgradnjo spletnih portalov. V večini so platforme skupek skript, ki ga namestimo na strežnik, poženemo in urejamo preko brskalnika. Tipična primera blog platform sta Blooger in WordPress, obstaja pa še cela vrsta podobnih orodij, s katerimi razvijalci gradijo bloge. Wikije uporabljamo za prispevanje znanja z nekega področja, ki uporabnikom omogoča prosto ustvarjanje in urejanje spletnih strani s spletnim brskalnikom. Med prevladujočimi brezplačnimi najdemo TWiki, xWiki, MesiaWiki, MindTouch Core in Foswiki. Spletna mesta za socialna omrežja so strani, ki gradijo in odražajo socialne mreže ali socialne odnose med ljudmi, ki imajo npr. skupne interese in aktivnosti. CMS (Content

Management System) je sistem za upravljanje vsebin, ki omogoča urejanje in vzdrževanje vsebine enostavnih spletnih strani brez znanja označevalnega jezika HTML. Urednik spletne strani lahko tako samostojno spreminja besedila, slike in druge elemente spletne strani brez pomoči podjetja ali osebe, ki je stran izdelala.

2.3 Kam gre razvoj (v smer Web 2.0, 3.0, itd.)

Razvoj spleta prikazuje slika 5. Splet 1.0 je omogočal medsebojno povezovanje hiperbesedilnih dokumentov, ki so bili dostopni prek interneta. S spletnim brskalnikom so si uporabniki lahko ogledali pasivne spletne strani, ki so vsebovale besedilo, slike in druge večpredstavnostne vsebine in so bile med seboj povezane z hiperpovezavami. Naslednja verzija, ki se danes največ uporablja, je 2.0. Splet 2.0 uporablja enake tehnologije kot verzija 1.0, razlika je dejansko v tem, kako se aplikacije uporabljajo in ne, kako se izvajajo. Splet 2.0 torej omogoča uporabnikom interakcijo in sodelovanje drug z drugim, poleg tega pa vključuje še uporabnike kot aktivne člane, ki lahko sodelujejo pri ustvarjanju vsebin v virtualni skupnosti. Ponuja veliko več, kot je ponujal splet 1.0, ki je uporabnike omejil na pasivno gledanje vsebin, ki so bile ustvarjene za njih. Splet 2.0 vključuje socialno mreženje, bloge, wikije in spletne aplikacije. Naslednja verzija, ki se razvija, je Splet 3.0, ki mu pravijo tudi Semantični splet, le ta pa ni ločen splet, ampak nadgradnja obstoječega. Semantični splet niso spletne strani in povezave, temveč odnosi med podatki. To je skupek tehnoloških standardov (Unicode, URI, XML, RDF, RDFS, OWL), ki omogoča strojem razumeti semantične dokumente in podatke. Spletu 3.0 bo sledila verzija 4.0, ki bo nadgradnja semantičnega spleta, omogočala pa bo integracijo inteligentnih vezij in programske opreme v skoraj vsako napravo (od lista papirja do avtomobila) in povezavo teh naprav v omrežje (internet). Vsaka naprava bi tako lahko komunicirala z vsako (zmanjka piva in hladilnik javi to listku za nakupe, ki ga imamo na mobilnemu telefonu). Cilj tega je izdelati inteligentne naprave, da bodo služile človeku kot osebni asistenti. Po predvidevanjih bo do leta 2030 web 4.0 postal paralelen človeškim možganom. Za Splet 5.0 in nadaljnje še ni ravno jasne utemeljitve, kaj naj bi vseboval. Pojavljajo se govorice, da naj bi omogočal direktno komunikacijo človeških možganov z napravami. Na tak način bi lahko menjavali kanale na TV brez uporabe katerega koli pripomočka, se naučili programirat v Javi, tako kot se je Neo v Matrici naučil karateja, in postavljali vprašanja spletu ter dobivali odgovore v realnem času.



Slika 5: Časovna premica razvoja svetovnega spleta (Vir: (2011)Call My Agent: Evolution In Information Retrieval. Dostopno na: http://solutions.wolterskluwer.com/blog/2011/09/call-my-agent-evolution-in-information-retrieval)

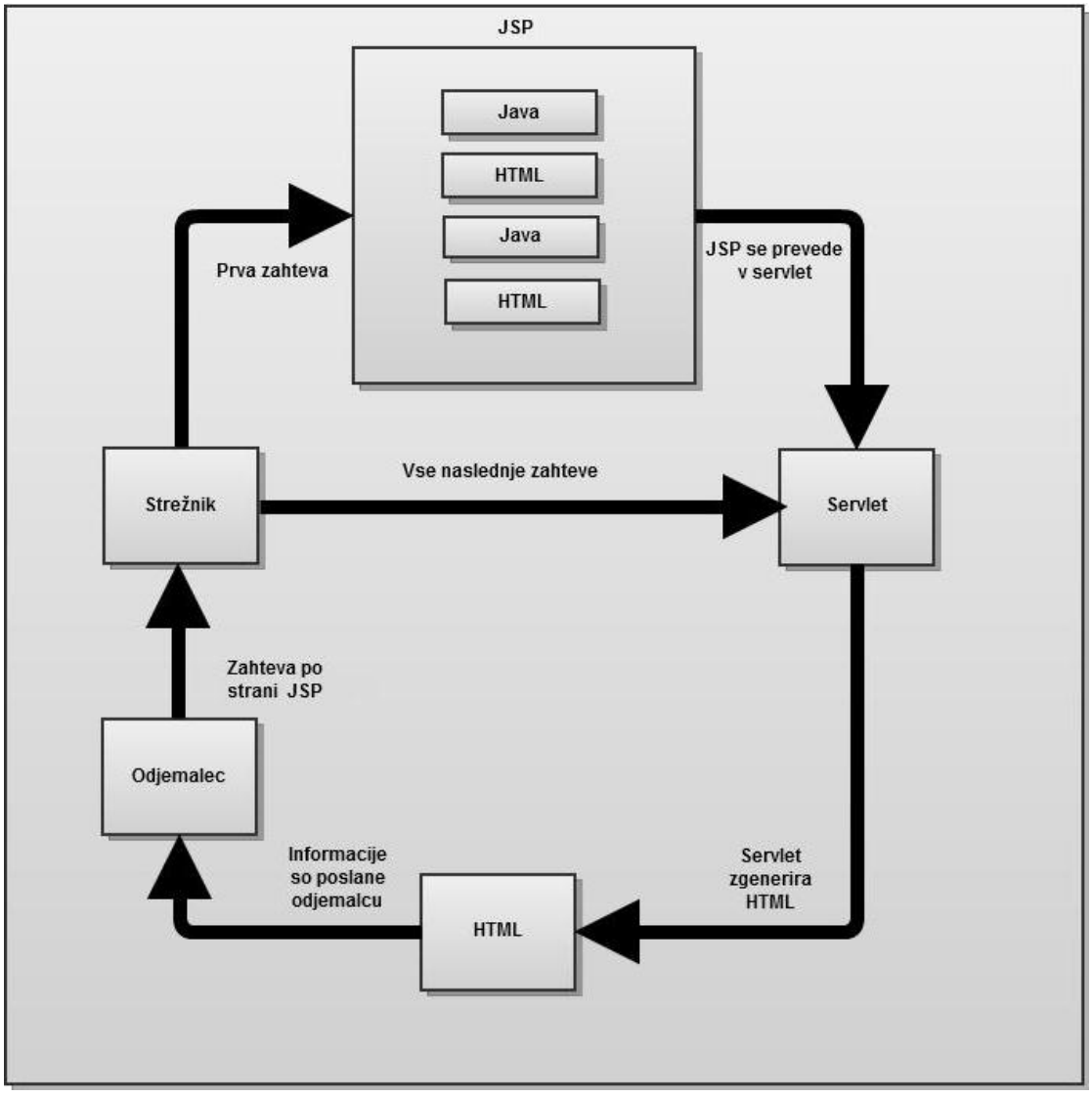
2.4 Najpomembnejše tehnologije za razvoj spletnih aplikacij

Najpogostejše uporabljene tehnologije pri razvoju spletnih aplikacij lahko delimo na dve vrsti tehnologij: tehnologije, ki se uporabljajo za razvijanje aplikacij na odjemalčevi strani, in tehnologije na strežniški strani. Najpomembnejše tehnologije, ki se uporabljajo na odjemalčevi strani, so JavaScript, Ajax, Adobe Flash in Dom. S prihodom spleta 2.0 se je razvila tehnologija Ajax (Asynchronous JavaScript and XML), ki omogoča spreminjanje vsebine dela spletne strani. Tehnologija Ajax tipično uporablja XML ali JSON (JavaScript Object Notation) podatkovno strukturo, ki jo strežnik pošlje odjemalcu. Ko so podatki preneseni, program Javascript uporabi Dom (Document Object Model) za dinamično osveževanje spletne strani. Najpomembnejše in hkrati največkrat uporabljene strežniške tehnologije so PHP, ASP.NET, JSP, Ruby, Perl, in Python. Vse tehnologije, razen ASP.NET, lahko poganjamo na različnih operacijskih sistemih, kar je prednost, saj so prenosljive. V diplomski nalogi sem izbral tehnologiji PHP in JSP. PHP sem izbral zaradi slovesa, da je enostaven za novince, ponuja številne napredne funkcije in se ga največ uporablja. JSP pa sem izbral, ker temelji na Java tehnologiji, ki smo jo obravnavali tekom šolanja, pa tudi zato, ker se ga veliko uporablja. Poleg tega so številne spletne platforme pisane tako v tehnologiji PHP kot v JSP.

3 Pregled JSP

JavaServer Pages (JSP) je tehnologija za izdelavo spletnih strani z dinamično vsebino, ki temelji na programskem jeziku Java.

Stran JSP je v osnovi dokument HTML, ki vsebuje kodo Java, uvedeno s posebnimi oznakami. Odseki so elementi JSP med oznakami, medtem ko je vse ostalo predloga, ki je neposredno posredovana do brskalnika. Nasprotno s statičnimi spletnimi stranmi JSP omogoča dinamično vsebino, ki se spreminja med izvajanjem, z uporabo spremenljivk. Ko spletni strežnik prejme zahtevo od odjemalca za določeno stran JSP, se predloga in elementi JSP združijo. Rezultat, ki se prevede in nato požene, je servlet. Potek generiranja spletne strani JSP prikazuje slika 6.



Slika 6: Model delovanja tehnologije JSP (Vir: (2011)JSP Overview. Dostopno na: http://blog.tangcs.com/2008/05/12/jsp-overview/)

Najbolje je strani JSP uporabljati v kombinaciji s servleti. S tem pristopom prevzamejo servleti obdelavo zahtev in poslovno logiko, strani JSP pa uporabniški vmesnik, kar bistveno olajša delo programerjev in izdelovalcev spletnih strani.

JSP nudi objekte JavaBean, ki so enostavni za vključevanje in upravljanje. Zrna se v večini primerov uporabljajo kot nosilci podatkov, ki jih želimo vključiti in prikazati na spletni strani. Pravi pomen zrn se pokaže, kadar poleg strani JSP uporabljamo tudi servlete. Servlet lahko obdela podatke in jih shrani v zrna. Stran JSP nato samo zrno naloži in podatke prikaže na spletni strani. Vse to lahko opravimo z nekaj enostavnimi akcijami JSP.

JSP nudi tudi uporabno značilnost, ki omogoča izdelavo značkovnih knjižnic po meri. To dosežemo z uporabo komponente JSTL (JavaServer Pages Standard Tag Library). JSTL omogoča definiranje lastnih akcij brez uvajanja kode Java, ki jih lahko nato uporabljamo v dokumentih JSP. Te akcije se v dokument JSP vstavljajo s pomočjo oznak XML, ki jih lahko sami definiramo. Ta koncept bistveno pripomore k boljši berljivosti dokumentov JSP, pomeni pa tudi, da lahko razvijalci spletnih strani uporabljajo bolj zapletene funkcionalnosti, brez potrebe znanja programiranja. Knjižnica Sql omogoča, da se JSTL tudi enostavno povezuje s podatkovno bazo preko vmesnika JDBC[4].

3.1 Sintaksa

3.1.1 Skriptni elementi

Obstajajo tri vrste skriptnih elementov. Prvi so skriptleti, ki so označeni z (<% ... %>) in omogočajo vključevanje kode Java v dokumente JSP. Drugi so izrazi, ki so označeni z (<%= ... %>) in omogočajo izpisovanje vrnjenih rezultatov, na drugi način je to mogoče doseči tudi z oznako (<%out.println(...) %>), ki uporablja izraz println() objekta out. Tretji elementi so deklaracije, ki jih uporabljamo za deklariranje spremenljivk in metod, ki jih pišemo med oznake (<%! ... %>) [4].

3.1.2 Direktive

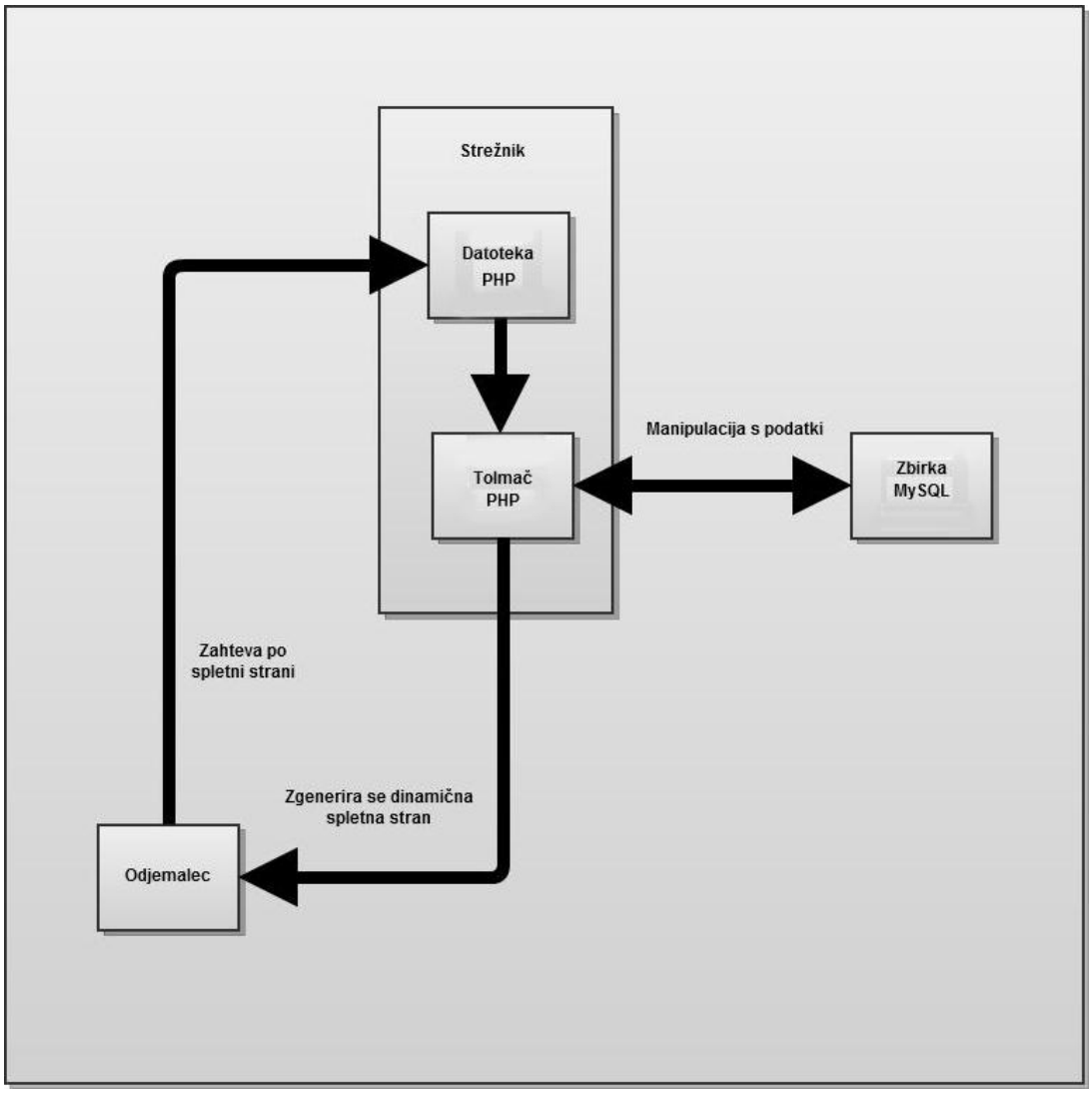
JSP lahko vsebuje tri vrste direktiv, ki se v JSP dokumentu začnejo z oznako (<@...>). Prva direktiva je page, ki jo uporabljamo za določanje nastavitev in atributov, ki se nanašajo na celoten dokument. Druga direktiva je include, ki jo uporabljamo za vključevanje drugih datotek v dokument JSP. Tretja pa je direktiva taglib, ki služi za deklariranje knjižnice prilagojenih oznak [4].

4 Pregled PHP

PHP (trenutno tričrkovni rekurzivni akronim za PHP Hypertext Preprocessor, izvirno pa Personal Home Page Tools, slovensko orodja za osebno spletno stran) je razširjen odprtokodni programski jezik, uporabljen za razvoj dinamičnih spletnih vsebin. Lahko ga primerjamo z Microsoftovim sistemom ASP, VBScript in JScript, Sun Microsystemovim sistemom JSP in Java ter sistemom CGI in Perl. Podoben je običajno strukturiranim programskim jezikom, najbolj jezikoma C in Perl. Izkušenim programerjem omogoča razvijanje zapletenih uporab brez dolgega učenja. Trenutno se uporabljata dve večji različici: 5.3.x in 5.2.x. Razvoj PHP4 se je končal 31. decembra 2007, njegovi kritični popravki pa so bili na voljo vse do 8. avgusta 2008.

PHP je bil napisan kot skupina CGI-programov v programskem jeziku C. Leta 1994 ga je napisal dansko kanadski programer Rasmus Lerdorf, da bi zamenjal nekaj skript, napisanih v Perlu, ki jih je na svoji spletni strani uporabljal za upravljanje.

PHP primarno uporabljamo na spletnem strežniku, kjer PHP izvorno kodo jemlje za vhod in generira izhod, ki je spletna stran. Poleg teka PHP omogoča tudi zaganjanje skript v ukaznem načinu in kreiranje grafičnih aplikacij. Slika 7 prikazuje potek generiranja spletne strani. Brskalnik poda zahtevo strežniku po spletni strani, nato tolmač združi podatke iz zbirke podatkov in datotek v dokument HTML, ki ga pošlje nazaj brskalniku [5].



Slika 7: Model delovanja tehnologije PHP (Vir: Virtual workshop. Dostopno na: http://www.keithjbrown.co.uk/vworks/php/php_p1.php)

4.1 Sintaksa

4.1.1 Skriptni elementi

Kodo PHP pišemo vedno znotraj oznake (<?php…?>), tudi ko definiramo funkcije in predmete, saj lahko prevajalnik le tako razume, da gre za kodo PHP. Če ima strežnik omogočeno, da prepozna tudi okrajšave, lahko pišemo kodo tudi med oznake (<?…?>), vendar se taka uporaba odsvetuje [5].

4.1.2 Direktive

V PHP-ju lahko spremenimo nastavitve tako, da spremenimo vrednosti atributov v datoteki php.ini ali v samem skriptu z uporabo funkcije ini_set(). Funkcija prejme dva argumenta: prvi je ime atributa, ki ga želimo nastaviti, in drugi vrednost. Obstajajo direktive za nastavljanje

same sintakse, omejevanje porabe spomina skriptov, učinkovitosti, ravnanje s podatki, poti in imenikov [6].

5 Primerjava konceptov

5.1 Temeljni elementi

5.1.1 Podatkovni tipi

5.1.1.1 PHP

PHP je prosto zapisljiv, kar pomeni, da podatkovne tipe izračunava, ko je posamezni spremenljivki dodeljen podatek. Slika 8 prikazuje vrednosti in podatkovne tipe, ki jih vračajo funkcije za preverjanje vrednosti spremenljivk in podatkovnih tipov.

	Funkcije				
Izrazi	gettype()	empty()	is_null()	isset()	Boolean if(x)
$x="";	string	TRUE	FALSE	TRUE	FALSE
$x=null	NULL	TRUE	TRUE	FALSE	FALSE
var $x;	NULL	TRUE	TRUE	FALSE	FALSE
$x is undentified	NULL	TRUE	TRUE	FALSE	FALSE
$x=array();	array	TRUE	FALSE	TRUE	FALSE
$x=false;	boolean	TRUE	FALSE	TRUE	FALSE
$x=true;	boolean	FALSE	FALSE	TRUE	TRUE
$x=1;	integer	FALSE	FALSE	TRUE	TRUE
$x=52;	integer	FALSE	FALSE	TRUE	TRUE
$x=0;	integer	TRUE	FALSE	TRUE	FALSE
$x=-1;	integer	FALSE	FALSE	TRUE	TRUE
$x="1";	string	FALSE	FALSE	TRUE	TRUE
$x="0";	string	TRUE	FALSE	TRUE	FALSE
$x="-1";	string	FALSE	FALSE	TRUE	TRUE
$x="php";	string	FALSE	FALSE	TRUE	TRUE
$x="true";	string	FALSE	FALSE	TRUE	TRUE
$x="false";	string	FALSE	FALSE	TRUE	TRUE

Slika 8: Tabela podatkovnih tipov PHP

Za spremenljivke, ki še niso bile inicializirane, uporablja vrednost NULL, kar pomeni, da jim vrednosti še niso bile dodeljene. Vrsto katere koli spremenljivke ugotovimo z ukazom gettype(), ki kot argument prejme spremenljivko, kot rezultat pa vrne niz, ki predstavlja ustrezen tip. Spremenljivki lahko s funkcijo settype() nastavimo želeni podatkovni tip. Podamo ji dva argumenta, spremenljivko in ključno besedo podatkovnega tipa [7].

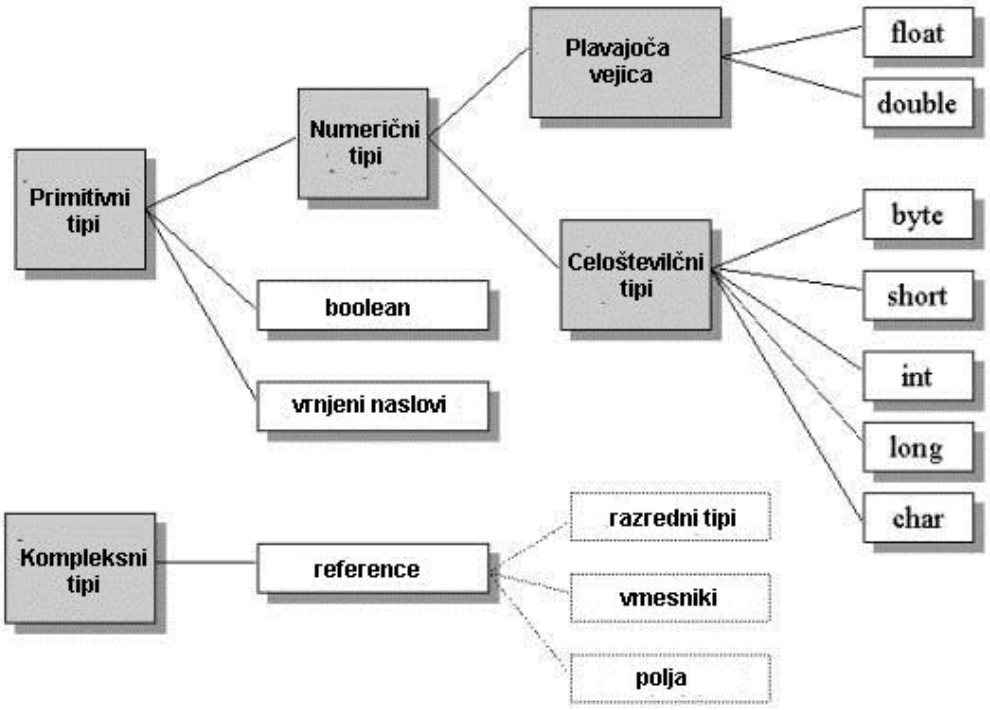
```
//deklariramo spremenljivko tipa double
$spr=4.15;
//spremenimo tip spremenljivke v string
settype($spr, string);
```

```
//gettype vrne tip spremenljivke
gettype($spr);
//rezultat je string
```

5.1.1.2 Java

Slika 9 prikazuje podatkovne tipe. Java pozna dve bistveno različni skupini podatkovnih tipov:

- Primitivni podatkovni tipi, ki dejansko hranijo podatke. Vsak primitivni tip ima ustrezno ključno besedo, s katero ga označimo v izvorni kodi. Java jih pozna osem: byte, short, int, long, float, double, boolean in char.
- Sklicni podatkovni tipi, ki hranijo zgolj naslov podatkov oziroma bolj zapletenih podatkovnih struktur. Z njimi se sklicujemo na podatke.



Slika 9: Diagram podatkovnih tipov Java (Vir: The Java Virtual Machine. Dostopno na: http://www.artima.com/insidejvm/ed2/jvm3.html)

Vsaka spremenljivka mora imeti predpisan podatkovni tip, zato pravimo, da je Java tipiziran jezik. Prevajalnik zahteva uporabo pravega podatkovnega tipa v določenih programskih elementih, ker bo drugače javil napako [8].

//deklariramo spremenljivko primitivnega podatkovnega tipa integer

int a=5;

5.1.1.3 Primerjava

Pri PHP-ju se tip spremenljivke določi šele, ko spremenljivki določimo vrednost. To, da je PHP prosto zapisljiv jezik, nam pri pisanju daljših skriptov lahko povzroča težave, saj ko spremenljivki priredimo neko vrednost določenega tipa, ni nujno da bo ta ostala takega tipa tekom celotnega časa izvajanja. Java pa je strogo tipiziran jezik, zato bo, ko je spremenljivka enkrat deklarirana, ostala takega tipa, torej nespremenjena.

5.1.2 Operatorji in izrazi

Operatorji so simboli ali nizi znakov, ki v povezavi z operandi izvedejo dejanje in pogosto proizvedejo novo vrednost. Operatorji:

- Določevalni operator
- Aritmetični operatorji
- Operator spajanja
- Primerjalni operatorji
- Logični operatorji

5.1.2.1 PHP

Določevalni operator vzame vrednost desnega operanda in jo priredi levemu operandu. Sestavljen je iz enega znaka (=).

//spremenljivki a priredimo vrednost
$a=5;

Aritmetični operatorji: seštevanje (+), odštevanje (-), množenje (*), deljenje (/), modul (%). Operator spajanja je ena pika (.). Če sta oba operanda niza, spoji desni operand z levim. Ne glede na podatkovne tipe operandov so ti obravnavani kot nizi in rezultat je vedno niz. Primerjalni operatorji (==,!=,===,>,>=,<,<=) izvajajo preizkuse na svojih operandih. Vrnejo logično vrednost true, če je preizkus uspešen, in false, če ni. Logični operatorji (||,or,xor,&&,and,!) so kombinirani s primerjalnimi operatorji [7].

5.1.2.2 Java

Določevalni operator vzame vrednost desnega operanda in jo priredi levemu operandu. Sestavljen je iz enega znaka (=).

//spremenljivki a priredimo vrednost
int a=5;

Aritmetični operatorji: seštevanje (+), odštevanje (-), množenje (*), deljenje (/), modul (%). Operator spajanja je znak (+). Če sta oba operanda niza, spoji desni operand z levim.

Primerjalni operatorji (==,!=,>,>=,<,<=) izvajajo preizkuse na svojih operandih. Vrnejo logično vrednost true, če je preizkus uspešen, in false, če ni. Za primerjanje dveh nizov moramo v Javi uporabiti funkcijo equals (), ki primerja dva niza in vrne true, če sta niza enaka, in false, če nista [8].

5.1.2.3 Primerjava

Iz slike 10 je razvidno, da je uporaba določevalnih in aritmetičnih operatorjev pri obeh jezikih enaka. Operator spajanja je v Javi znak (+) pri PHP-ju pa znak (.). Pri logičnih operatorjih je razlika ta, da ima PHP poleg znakovnih operatorjev (&&,||) vgrajene še operatorje (or, and), katerih funkcija je enaka, vendar je razlika v prednostnem vrstnem redu. Operator Xor v PHP-ju pišemo kar xor, Java pa ima za to namenjen znak (^). Primerjalni operatorji so tudi v večini enaki, z razliko, da ima PHP še operator identičnost (===), ki ga v verziji PHP 5 uporabljamo za ugotavljanje, ali vsebujeta spremenljivki isti predmet, v PHP 4 pa primerja lastnosti dveh predmetov.

		PHP		JSP	
Operatorji in izrazi		Operatorji	Uporaba	Operatorji	Uporaba
Določevalni operator		**=**	$a = 5;	**=**	a = 5;
Aritmetični operatorji	Seštevanje	**+**	$c = $a + $b;	**+**	c = a + b;
	Odštevanje	**-**	$c = $a - $b;	**-**	c = a - b;
	Množenje	*****	$c = $a*$b;	*****	c = a * b;
	Deljenje	**/**	$c = $a / $b;	**/**	c = a / b;
	Modul	**%**	$c = $a % $b;	**%**	c = a % b;
Operator spajanja		**.**	$c = $a . $b;	**+**	'c = a + b;
Primerjalni operatorji	Enakovrednost	**==**	$a == $b	**==**	a == b
	Neenakovrednost	**!=**	$a != $b	**!=**	a != b
	Identičnost	**===**	$a === $b	**-**	-
	Večji od	**>**	$a < $b	**>**	a > b
	Večji od ali enako	**>=**	$a >= $b	**>=**	a >= b
	Manjši od	**<**	$a < $b	**<**	a < b
	Manjši od ali enako	**<=**	$a <= $b	**<=**	a <= b
Logični operatorji	Ali	**||,or**	$a || $b, $a or $b	**||**	a || b
	In	**&&,and**	$a && $b, $a and $b	**&&**	a && b
	Ekskluzivni ali	**xor**	$a xor $b	**^**	a ^ b
	Negacija	**!**	!$b	**!**	!b

Slika 10: Tabela operatorjev in izrazov PHP in JSP

5.1.3 Nadzor izvajanja

Zančni in pogojni programski stavki nadzirajo potek izvajanja programa. Programerju omogočajo, da v program vplete odločitveno logiko, ki se prilagaja podatkom. Odločitve pa so neločljivo povezane z logičnimi izjavami. Logične izjave:

- Izjava if
- Izjava else
- Izjava switch
- Izjava while
- Izjava do ... while
- Izjava for

5.1.3.1 PHP

Izjava if je način nadzora izvajanja stavka. Znotraj oklepajev zapišemo logične izjave, ki so bodisi resnične ali neresnične. V primeru, da so izjave neresnične, lahko z izjavo else preusmerimo tok izvajanja.

```
//preverimo pravice, ki jih ima uporabnik
if($uporabnik->getPravice()=="stranka"){
        header( 'Location: naredisiizgled.php' );
}
//če so pravice administratorja, preusmerimo tok izvajanja
else if($uporabnik->getPravice()=="admin"){
      header( 'Location: Admin/adminzacetna.php' );
 }
```

Izjava switch ima za vrednost le en izraz in izvaja različno kodo glede na rezultat tega izraza. Struktura izjave while je videti enaka izjavi if, le da se ta blok kode ponavlja, dokler je pogoj resničen. Izjava do ... while je podobna izjavi while. Bistvena razlika je v tem, da se blok kode vsekakor enkrat izvede in šele na koncu bloka preverja, ali je pogoj resničen ali neresničen. Izjava for omogoča, da zanko zapišemo v eni vrstici. Izjave znotraj for-a so ločene s podpičjem, kjer prva izjava inicijalizira števec, druga preverja pogoj, tretja pa povečuje ali zmanjšuje števec [7].

5.1.3.2 Java

Pri Javi je koncept popolnoma enak PHP.

```
//preverimo pravice, ki jih ima uporabnik
if(uporabnik.getPravice().equals("stranka"))
        response.sendRedirect("naredisiizgled.jsp");
//če so pravice administratorja, preusmerimo tok izvajanja
else if(uporabnik.getPravice().equals("admin"))
       response.sendRedirect("/Admin/adminzacetna.jsp");
```

5.1.3.3 Primerjava

Izjava if je pri obeh tehnologijah način nadzora izvajanja stavka. Pri obeh tehnologijah zapišemo znotraj oklepajev logične izjave, ki so bodisi resnične ali neresnične. V primeru, da so izjave neresnične, lahko z izjavo else preusmerimo tok izvajanja. Strukture izjav in uporaba while, do … while, for in switch stavkov so enake pri obeh tehnologijah.

5.1.4 Funkcije

Funkcije so osrednji del dobro organiziranega skripta in omogočajo lahko branje kode in ponovno uporabo. Nobenega večjega projekta si ne bi mogli predstavljati brez njih. Funkcija je samozadosten blok kode, ki ga lahko kličemo. Funkcijam lahko posredujemo vrednosti, ki jih te nato obdelujejo in na koncu lahko vrnejo rezultat.

5.1.4.1 PHP

Funkcijo pokličemo z njenim imenom in v oklepajih podamo morebitne argumente. Določimo jo z izjavo function, imenom in nizom oklepajev. Če funkcija zahteva argumente, moramo postaviti v oklepaje imena spremenljivk. Te spremenljivke so zapolnjene z vrednostmi, ki so posredovane funkciji.

```
//določitev funkcije z argumentom
public function setIme($ime) {
        $this->ime = $ime;
}
```

Če želimo, da funkcija rezultat vrne, uporabimo izjavo return in nato spremenljivko, ki jo želimo vrniti kot rezultat. Spremenljivka, določena v funkciji, ostane le za to funkcijo. Privzeto iz funkcije ne moremo imeti dostopa do spremenljivk, ki so bile določene drugje.

```
//določitev funkcije, ki vrne rezultat
public function getIme() {
        return $this->ime;
}
```

Ko želimo dostopati iz funkcije do zunanjih spremenljivk, damo izjavo global pred spremenljivko. S tem povemo, da želimo uporabiti globalno spremenljivko. Spreminjanje spremenljivke v funkciji z izjavo global pomeni, da se bo spremenil izvirnik. Rezultat spremenljivk, ki so deklarirane znotraj funkcije, izgine, ko funkcijo odrabimo. V primeru, da želimo ohraniti rezultat, uporabimo izjavo ststic pred spremenljivko. Če so funkcije znotraj razreda deklarirane kot statične, pomeni, da se funkcija inicijalizira samo enkrat in ni potrebno vedno delati instanc razreda. PHP omogoča posredovanje sklicev na spremenljivke funkcijam, ki jim pošlje kot argument sklic na spremenljivko, na kateri katera koli sprememba spremeni vrednost izvirne spremenljivke. Za posredovanje sklicev uporabimo znak & pred deklaracijo spremenljivke, znotraj oklepajev funkcije. Funkcije lahko izdelamo kot anonimne funkcije, ki imajo to lastnost, da so hranjene v spremenljivkah. Za njihovo izdelovanje uporabimo funkcijo create_function(), ki zahteva argumente v dveh nizih. Prvi argument mora

vsebovati z vejicami ločen seznam argumentov spremenljivke. Drugi argument mora vsebovati osrednji del funkcije. Za preizkus, ali funkcija sploh obstaja, uporabimo funkcijo function_exists(), ki prejme, kot argument, ime funkcije in vrača true, če obstaja, false pa, če ne obstaja. Za omejevanje dostopa uporabljamo izjave public, private in protected. Funkcije, ki so tipa public, so dostopne vsem, private so dostopne samo znotraj razreda, protected pa so dostopne le znotraj paketa PHP. Če omejitve dostopa ne napišemo, je privzeta izjava public [7].

5.1.4.2 Java

Ker je Java predmetno usmerjen programski jezik, ne pozna samostojnih podprogramov ali funkcij. Zaključna opravila morajo biti vsebovana v funkciji. Funkcija je funkcijski podprogram, ki pripada nekemu razredu in določa del njegovega obnašanja. Funkcijo deklariramo z imenom, ki mu sledi par oklepajev, med katere navedemo deklaracijo morebitnih argumentov, ki podrobneje določajo njeno delovanje. Par oklepajev moramo zapisati tudi, če funkcija nima argumentov. Klic funkcije mora ravno tako imeti par oklepajev za imenom funkcije.

```
// določitev funkcije z argumentom, ki mora biti določenega tipa
public void setIme(String ime) {
        this.ime = ime;
}
```

Ker so funkcije funkcijski podprogrami in lahko nastopajo v izrazih, jim moramo določiti tip, ki določa vrsto vrnjenega podatka. Podatek vrnemo s stavkom return, ki je lahko kjer koli v funkciji.

```
//določitev funkcije, ki vrne rezultat
public String getIme() {
        return ime;
}
```

Če funkcija ne vrača rezultata, moramo to označiti z izjavo void. Iz funkcij vidimo lokalne spremenljivke, ki so bile deklarirane znotraj funkcije, in globalne spremenljivke, ki so deklarirane zunaj funkcij. Če so funkcije znotraj razreda deklarirane kot statične, pomeni, da se funkcija inicjalizira samo enkrat in ni potrebno vedno delati instanc razreda. Za omejevanje dostopa uporabljamo izjave public, private in protected. Funkcije, ki so tipa public, so dostopne vsem, private so dostopne samo znotraj razreda, protected pa so dostopne le znotraj paketa Java. Če omejitve dostopa ne napišemo, je privzeta izjava protected [8].

5.1.4.3 Primerjava

Razlika pri osnovni deklaraciji funkcije je ta, da moramo pri Javi navesti podatkovni tip, ki ga funkcija vrača in v primeru, da funkcija ne vrača ničesar, dati izjavo void. PHP z izjavo function, brez deklaracije tipa, lahko vrača kateri koli podatkovni tip z izjavo return. Dostop do globalnih spremenljivk je podoben, le da moramo pri PHP-ju povedati, da gre za globalno spremenljivko z izjavo global pred samo spremenljivko. Oba jezika poznata izjavo static pred

osnovno deklaracijo funkcije, ki pomeni, da se funkcija inicjalizira samo enkrat. Prav tako poznata izjave za določanje dostopa public, private in protected. Razlika je pri prednastavljenih dostopih, saj če ne napišemo ničesar, je pri PHP privzeto public, pri Javi pa se lahko dostopa do funkcije samo znotraj paketa.

5.1.5 Polja

Polja so posebne spremenljivke, ki omogočajo shranjevanje želenega števila vrednosti v eno spremenljivko. Vsaka vrednost je v polju indeksirana.

5.1.5.1 PHP

Dostop do posameznega elementa je mogoč neposredno iz njegovega indeksa. Indeks elementa polja je lahko bodisi število bodisi niz. Privzeto so elementi polja indeksirani po številkah, začenši z 0. Polju lahko priredimo vrednosti s konstruktom array() ali rabo praznih oglatih oklepajev []. Konstruktu array() podamo kot argument seznam vrednosti, ločenih z vejicami, in je uporaben, ko želimo polju prirediti več vrednosti naenkrat.

```
//določanje polja z uporabo oznake array
$uporabniki=array("Borut","Tine");

//določanje polja z uporabo oznake polja
$uporabniki[]="Borut";
$uporabniki[]="Tine";
```

Povezljiva polja naredimo, če pred vsako vrednost argumenta damo še ime indeksa v narekovajih in znaka =>.

```
//določanje povezljivega polja z uporabo oznake array
$uporabniki=array("ime"=>"Borut","priimek"=>"Mahne");
```

Velikost polja pridobimo s funkcijo count(), ki za argument dobi polje in vrne število elementov v polju. Zadnji element pridobimo s funkcijo end(), ki kot argument zahteva spremenljivko polja. Za prehod čez polje je najboljši način izjava foreach(), ki deluje tako za indekse, ki so številčni, kot za tiste z nizi. Funkcija print_r() sprejme katero koli spremenljivko in vrne informacije o vsebini in strukturi argumentov. Če tej funkciji posredujemo polje, dobimo seznam elementov polja in vse strukture, kot so objekti ali polja [7].

5.1.5.2 Java

Polja v Javi so zbirka več podatkov istega tipa. Ker je Java močno tipiziran jezik, je v polje mogoče shraniti le podatke enega tipa. Tipi so lahko primitivni ali sklicni. Pri deklaraciji polja najprej napovemo spremenljivko ustreznega tipa, potem pa zasežemo ustrezen pomnilniški prostor. Pri deklaraciji spremenljivke, ki bo hranila polje podatkov, uporabimo par oglatih oklepajev ([]), ki jih lahko pripišemo bodisi simboličnemu imenu bodisi sami ključni besedi. Z deklaracijo smo deklarirali spremenljivko, ki bo hranila naslov bodočega polja. Izjava new

izdela novo instanco želenega polja in vrne njegov naslov. Polju moramo vnaprej povedati, kakšna bo dolžina.

//določimo tabelo desetih števil
int tabela=new int[10];
Dolžine kasneje ne moremo več spreminjati. Vsi elementi v polju so indeksirani z zaporednimi številkami, začenši z 0. Ob zaseganju pomnilnika je privzeta vrednost za primitivne tipe številk 0, znakov '\u0000' in (false) resnica. Za elemente sklicnih tipov je privzeta vrednost null, kar pomeni neveljaven naslov. Vsak objekt tipa polje ima funkcijo length(), ki vrača dolžino polja. Čez polje se sprehodimo z uporabo zanke for, kateri damo kot pogoj izvajanja dolžino polja in indeks povečujemo od 0 naprej s korakom 1 [8].

5.1.5.3 Primerjava

Pri delu s polji je Java strogo tipiziran jezik, kjer mora biti vsako polje vnaprej določenega tipa in določene velikosti. Pri PHP ni potrebno določiti velikosti in ne tipa. Indeksiranje polja je pri Javi striktno z zaporednimi številkami, ki se začnejo z 0. Pri PHP so lahko indeksi nizi ali števila, ki niso nujno zaporedna. PHP ima že vgrajene funkcije, ki omogočajo delo s polji. Polja lahko tako enostavno združujemo skupaj, dodajamo ali odstranjujemo elemente in razvrščamo elemente v polju. Java ima za delo z objekti namenjene generične tipe. Med temi je tudi ArrayList, ki ne zahteva vnaprej določene dolžine. Za shranjevanje objektov ni potrebno določiti, kakšnega tipa so, zato lahko shranjujejo različne tipe objektov. ArrayList-u lahko vnaprej določimo tip, ki ga bo hranil tako, da med znaka (<>) podamo tip, ki ga bo ArrayList hranil. Poleg tega ima ArrayList za delo s polji vgrajene funkcije, ki omogočajo lažje delo s shranjenimi elementi.

5.1.6 Predmeti

S pomočjo predmetov je mogoče hitreje sestavljati zanesljive programe. Vsak predmet ima svoje lastnosti in obnašanje, ki ga nadzirajo funkcije. Predmet je zaprt zavitek spremenljivk in funkcij, ki je sestavljen iz posebne predloge, ki se imenuje razred.

5.1.6.1 PHP

Če želimo izdelati predmet, moramo najprej zasnovati predlogo, iz katere ga bomo konkretizirali. Ta predloga je zasnovana kot razred in jo moramo deklarirati z izjavo class in nato z imenom razreda. Poljubno število primerkov predmeta izdelamo tako, da uporabimo izjavo new. Razred ima dostop do razrednih spremenljivk, imenovanih lastnosti. Določimo jih lahko kjer koli v osrednjem delu razreda, vendar jih zaradi jasnosti določimo na vrhu. Lastnost je lahko vrednost, polje ali drug predmet. Spremenljivko določimo z izjavo var, ki je v PHP 4 nujna. PHP 5 pa omogoča omejevanje dostopa tako, da namesto izjave var uporabimo eno od naslednjih izjav:

- Public vidno vsem
- Private na voljo samo razredu, ki ga vsebuje

- Protected na voljo samo razredom in podrazredom, ki ga vsebujejo

Če se odločimo, da bomo uporabili omejevanje dostopa s private ali protected, razredu napišemo še funkcije za nastavljanje vrednosti spremenljivk. Ravno tako kot spremenljivkam, omogoča PHP5, da tudi funkcijam omejimo dostop. Razred ima lahko en konstruktor, ki prejme argumente. Vrednosti argumentov priredimo razrednim spremenljivkam tako, da damo izjavo $this in znaka -> pred ime razredne spremenljivke, kateri želimo prirediti vrednost [7].

```
//predloga za izdelavo razreda
class Fuga {
   //deklaracija razrednih spremenljivk
   private $fuga_id;
   private $rgb;
   private $fuga_odstranjena;
  //konstruktor razreda, ki je lahko samo eden
   public function Fuga($fuga_id, $rgb, $fuga_odstranjena){
      $this->fuga_id=$fuga_id;
      $this->rgb=$rgb;
      $this->fuga_odstranjena=$fuga_odstranjena;
   }
  //razredne funkcije
   public function getFuga_id() {
      return $this->fuga_id;
   }
  public function getRgb() {
      return $this->rgb;
  }
 public function isOdstranjena() {
      return $this->fuga_odstranjena;
 }
 public function setFuga_id($ploscica_id) {
      $this->fuga_id = $ploscica_id;
 }
 public function setRgb($rgb) {
      $this->rgb = $rgb;
 }
 public function setOdstranjena($fuga_odstranjena) {
      $this->fuga_odstranjena = $fuga_odstranjena;
 }
}
```

5.1.6.2 Java

V Javi ni mogoče napisati tudi najbolj preprostega programa, ne da bi se ukvarjali z razredi. Preprosti programi vsebujejo le en razred, ki uporablja bogato knjižnico razredov, ki so na

voljo v izvajalnem okolju. Predlogo razreda deklariramo z izjavo class, ki mu sledi ime razreda. To ime ponazarja predmetno spremenljivko. Predmetna spremenljivka hrani le sklic na morebitni predmet. Razred ima dostop do razrednih spremenljivk, ki so deklarirane v razredu in so določenega tipa. Tako spremenljivkam kot funkcijam v razredu lahko omejimo dostop z izjavami public, private in protected. Razred vsebuje vsaj en konstruktor, ki ne vsebuje argumentov, tudi če ga ne eksplicitno deklariramo. Razred lahko vsebuje poljubno število konstruktorjev, saj zna Java izbrati pravega iz števila in tipa argumentov [8].

```
//predloga za izdelavo razreda
public class Fuga {
  //deklaracija razrednih spremenljivk
   private int fuga_id;
   private String rgb;
   private boolean fuga_odstranjena;
   //konstruktorji razreda, katerih je lahko v Javi več
   public Fuga(String rgb){
      this.rgb=rgb;
      this.fuga_odstranjena=false;
   }
   public Fuga(String rgb, boolean fuga_odstranjena){
      this.rgb=rgb;
      this.fuga_odstranjena=fuga_odstranjena;
   }
   public Fuga(int fuga_id, String rgb, boolean fuga_odstranjena){
      this.fuga_id=fuga_id;
      this.rgb=rgb;
      this.fuga_odstranjena=fuga_odstranjena;
   }
   //razredne funkcije
   public int getFuga_id() {
      return fuga_id;
   }
   public String getRgb() {
      return rgb;
   }
   public boolean isOdstranjena() {
      return fuga_odstranjena;
   }
   public void setFuga_id(int ploscica_id) {
      this.fuga_id = ploscica_id;
   }
   public void setRgb(String rgb) {
      this.rgb = rgb;
   }
```

```
    public void setOdstranjena(boolean fuga_odstranjena) {
        this.fuga_odstranjena = fuga_odstranjena;
    }
}
```

5.1.6.3 Primerjava

Tako Java kot novejše verzije PHP omogočajo predmetno-usmerjeno programiranje. Java je veliko bolj striktno predmetno usmerjen jezik, saj celoten koncept temelji na predmetih. Oba omogočata omejevanje dostopa do spremenljivk in funkcij. Izdelava predloge razreda je po principu enaka. Java pa z razliko od PHP omogoča, da imamo lahko več konstruktorjev.

5.2 Delo z nizi

Nizi znakov so v programih zelo pogosti. Predstavljamo si jih lahko kot polje znakov.

5.2.1 PHP

Nize združujemo enostavno z znakom (.).

Za izpisovanje nizov ponuja PHP dve funkciji print in echo, ki jima podamo niz kot argument. Funkcija printf zahteva nizovni argument, ki je niz za nadzor oblikovanja. Ta niz vsebuje navodila za oblikovanje, kako prikazati dodatne argumente. Specifikacija pretvorbe v nizu za nadzor oblikovanja se začne z znakom (%) in določa, kako bo ustrezen argument obravnavan. V niz za nadzor izvajanja lahko vključimo poljubno število specifikacij pretvorbe, če posredujemo enako število argumentov. Na popolnoma enak način deluje funkcija sprintf, le da rezultata ne izpiše, temveč rezultat vrne in ga lahko priredimo spremenljivki ali zapišemo v datoteko.

Ker so nizi organizirani kot polja in imajo znake indeksirane, lahko nize preiskujemo. Pri preiskovanju nam PHP nudi pomoč z več funkcijami. Funkciji strlen() podamo niz, vrne pa nam celoštevilsko vrednost dolžine niza. Funkcija strstr() preizkusi ali obstaja niz v nizu. Zahteva dva argumenta: izvorni niz in podniz, ki ga želimo najti v njem. Vrne true, če niz obstaja, ali false, če ne obstaja. Funkcijo strpos() uporabljamo za iskanje indeksa, kjer se niz v nizu začne. Če niz ne obstaja, vrne false, če pa obstaja, vrne celoštevilsko vrednost, ki je indeks, kjer se niz v nizu začne. Zahteva dva argumenta: izvorni niz in podniz, ki ga želimo najti v njem, in tretji opcijski argument, ki je celoštevilčna vrednost indeksa, od katerega naprej naj išče niz. Funkcija substr() vrne del niza, ki temelji na začetnem indeksu in dolžini dela, ki ga iščemo. Zahteva dva argumenta: izvorni niz in začetni indeks. Tretji argument je opcijski in pove dolžino od začetnega indeksa.

Za zamenjevanje nizov uporabljamo funkcije substr_replace(), str_replace(), strtoupper() in strtolower(). Funkcija substr_raplace() omogoča, da del niza, ki ga izvlečemo, zamenjamo. Funkcija str_replace() zamenja vse primerke niza v drugem nizu. Funkciji strtoupper() in strtolower() zamenjata vse črke v velike oziroma majhne črke.

Zelo uporabna funkcija je explode(), ki razbije niz v polja. Funkcija explode() zahteva dva argumenta: ločevalni niz, ki ga želimo uporabljati za razbitje izvornega niza, in izvorni niz sam. Tretji izbirni argument je celoštevilčna vrednost, ki določa, koliko je največje število delov, na katere je mogoče razbiti niz [7].

5.2.2 Java

V Javi so nizi predmeti, tipa String, zato so funkcije, ki jih kličemo, del predmeta. Nize primerjamo s funkcijo equals(), ki nam vrne true, če sta niza enaka, in false, če nista. Čeprav Java za primerjavo nizov omogoča uporabo dvojnih enačajev (==), je tu nevarnost, ker na tak način ne primerjamo vsebine, temveč sklice predmeta. Za združevanje nizov uporabljamo znak (+).

Za izpisovanje nizov uporabljamo funkciji print in println. Funkcija println izpisuje nize vsakega v svojo vrstico. Java ima tako kot PHP funkcijo printf, ki jo uporabljamo na enak način.

Nizi so predmeti, za katere v javanski knjižnici obstaja cela vrsta metod. Pogosto se sprašujemo po dolžini niza. Vrne ga funkcija length(). Če je niz zapisan kot polje znakov, dolžino niza pove lastnost length brez oklepajev. Posamezen znak niza lahko izluščimo s funkcijo charAt(), ki ji posredujemo indeks želenega znaka. Funkcija indexOf() vrne celoštevilčno vrednost indeksa, kjer se znak ali podniz prične v nizu. Če želimo iz niza izvleči podniz, uporabimo funkcijo substring(). Prvi argument določa indeks znaka, kjer se podniz prične, drugi pa indeks prvega znaka, ki ga ne želimo več.

Zamenjevanje nizov omogočajo funkcije replace(), replaceAll in replaceFirst(). Prva zahteva dva argumenta: znak, ki ga želimo zamenjati, in znak, s katerim ga želimo zamenjati. Funkcija zamenja vse pojavitve znaka. Druga je enaka funkciji replace(), le da zahteva niza namesto znakov. Tretja pa zamenja le prvo pojavitev niza. Funkciji toUpperCase() in toLowerCase() zamenjata vse črke v velike oziroma majhne.

Funkcija split() razbije niz v polja. Split() zahteva ločevalni niz, ki ga želimo uporabljati za razbitje izvornega niza [8].

5.2.3 Primerjava

Oba jezika imata vrsto funkcij za delo z nizi. Java nize obravnava kot predmete tipa String. Vsak tak predmet ima različne funkcije za delo z nizi. PHP ima nize realizirane kot polja, kjer lahko do posameznega znaka dostopamo z uporabo indeksa. Nize primerjamo v Javi s funkcijo equals(), ker želimo primerjati vsebino predmetov in ne sklicev predmetov. Združevanje nizov je podobno, razlika je operator, ki je v Javi znak (+), pri PHP pa (.). Uporaba funkcij print() in printf() je pri obeh jezikih enaka. Java ima še dodatno funkcijo println(), ki posamezne nize izpisuje vsakega v svoji vrstici. Oba imata funkcije za preiskovanje, zamenjavo in razbijanje nizov.

5.3 Delo z obrazci

V svetovnem spletu so obrazci HTML glavni način, s katerimi je mogoče posredovati precejšen del informacij uporabnikov strežnika. PHP je zasnovan tako, da pridobiva in dela z informacijami, ki so posredovane z obrazci HTML.

5.3.1 PHP

Vsi uporabnikovi vnosi, ki se posredujejo strežniku, se shranjujejo v superglobalne spremenljivke. Superglobalne spremenljivke so polja, ki se zapolnijo samodejno, omogočajo takojšen dostop do podatkov v njih in so na voljo od kjer koli. Uporabnikove vnose nam hranijo tri polja. Prvo polje $_GET nam hrani vnose, ki so bili poslani z obrazcem, ki ima za pošiljanje HTTP-metodo get. Drugo polje $_POST hrani vnose, ki so bili poslani po metodi post. Tretje pa je $_REQUEST, pri katerem nam ni potrebno vedeti, katera metoda pošiljanja je bila uporabljena. Vsebuje namreč združeno vsebino polj $_GET, $_POST in polje $_COOKIE, ki hrani piškotke. Pri uporabi obrazca enctype="multipart/form-data" se vsi podatki ravno tako shranijo v superglobalne spremenljivke. Dodatno je sedaj na voljo še polje $_FILES, ki shrani vse uporabnikove datoteke. Datoteke se prenesejo z začasnim imenom v mapo tmp, ki začasno hrani datoteke. $_FILES je dvodimenzionalno polje, ki hrani: ime prenesene datoteke, pot do začasne datoteke, velikost datoteke, kodo napake in vrsto prenesene datoteke. Datoteko lahko prenesemo v poljubno mapo z uporabo funkcije move_uploaded_file(), ki zahteva dva argumenta, vir in cilj, in jo premakne iz enega v drugega [7].

```
//deklariramo spremenljivke, ki hranijo stanje uporabnikove izbire
$slika_ime;$slika_width; $slika_height;$slika_top;$slika_left; $st_ploscic_visina;
$st_ploscic_sirina; $stranka_id; $ploscica_ploscica_id; $fuga_fuga_id;$fuga_sirina;
$obdelano;
//preverimo, če obstaja seja za uporabnika
if(isset ($_SESSION['uporabnik'])){
        $uporabnik=$_SESSION['uporabnik'];
        //preverimo pravice uporabnika
        if($uporabnik->getPravice()=="stranka"){
          //preverimo, ali slika ustreza zahtevam

           if((($_FILES["slika"]["type"]=="image/gif")||($_FILES["slika"]["type"]==
"image/jpeg")||($_FILES["slika"]["type"] == "image/jpeg"))&&($_FILES["slika"]["size"] <
10000000)){
//sliko premaknemo iz začasnega odlagališča v želeno mapo
move_uploaded_file($_FILES["slika"]["tmp_name"],"Slike/".$_FILES["slika"]["name"]);
           }
           //iz polja request preberemo uporabnikove vnose in jih priredimo spremenljivkam
           $slika_ime=$_FILES["slika"]["name"];
           $slika_width=Utilities::odstraniPx($_REQUEST["width"]);
           $slika_height=Utilities::odstraniPx($_REQUEST["height"]);
```

```
            $slika_top=Utilities::odstraniPx($_REQUEST["top"]);
            $slika_left=Utilities::odstraniPx($_REQUEST["left"]);
            $st_ploscic_visina=$_REQUEST["stevilo_ploscic_visina_st"];
            $st_ploscic_sirina=$_REQUEST["stevilo_ploscic_sirina_st"];
            $stranka_id=$uporabnik->getUporabnik_id();
            $ploscica_ploscica_id=$_REQUEST["velikost_ploscice"];
            $fuga_fuga_id=$_REQUEST["barve_fug"];
            $fuga_sirina=$_REQUEST["fuga"];
            $obdelano=0;
            //naredimo nov predmet povpraševanje, ki ga nato shranimo v zbirko podatkov
            $povprasevanje=new Povprasevanje(-1, $slika_ime, $slika_width, $slika_height,
$slika_top, $slika_left,$ st_ploscic_visina,$st_ploscic_sirina, NULL, $obdelano,$stranka_id,
$ploscica_ploscica_id, $fuga_fuga_id);
            DBHelper::dodajPovprasevanje($povprasevanje);
            //ko je povpraševanje uspešno dodano v zbirko, naredimo preusmeritev
            header('Location: narediSiIzgled.php');
        }
    }
```

5.3.2 Java

Uporabnikovi vnosi, ki se posredujejo strežniku, se shranjujejo v zahtevane objekte, ki so navadno nizi. Vsak JSP ima vgrajen predmet request, ki vsebuje informacije o zahtevi brskalnika. Preko tega predmeta lahko pridobimo vrednosti spremenljivk, ki so bile poslane preko metod get in post. Metodo lahko preverimo s funkcijo getMethod(), ki vrne get ali post, odvisno od metode, ki je bila uporabljena. Vrednosti spremenljivk pridobimo s funkcijo getParameter(), kot argument pa zahteva ime spremenljivke, ki jo želimo. Funkcija getParameter() vrne null, če spremenljivka ne obstaja, v nasprotnem primeru pa vrednost, ki jo hrani.

Za prenos datotek uporabljamo obrazce tipa enctype="multipart/form-data". Obdelovanje take vrste obrazcev s predmetom request je precej naporno opravilo, še posebej, če prenašamo več datotek. Delo si olajšamo z uporabo paketa fileUpload, ki ga ponuja podjetje Apache za strežnik Tomcat. Paket fileUpload vključimo v našo aplikacijo in imamo na voljo predmete za prenos datotek. Predmet ServletFileUpload ima funkcijo parseRequest(), ki ji podamo predmet request, ki nam ga razčleni v seznam spremenljivk. Čez seznam se sprehodimo z uporabo iteratorja. Za vsak element seznama s funkcijo isFormField() preverimo, ali je element posredovan kot uporabnikov vnos ali datoteka. Vsak element v seznamu ima svojo vrednost in ime. Če je element datoteka, jo lahko shranimo v želeno mapo z uporabo predmeta FileItem, ki drži vsebino datoteke in predmeta File, le ta pa nam pove pot do želene mape. Pot do mape, ki se nahaja na strežniku, dobimo z uporabo predmeta request, ki ima funkcijo getContextPath(), ta pa nam vrne pot, od koder je bil servlet zagnan [9].

```
//deklariramo spremenljivke, ki hranijo stanje uporabnikove izbire
```

```
<%!String slika_ime; int slika_width, slika_height,slika_top, slika_left,s t_ploscic_visina,
st_ploscic_sirina, stranka_id, ploscica_ploscica_id, fuga_fuga_id,fuga_sirina; boolean
obdelano;%>
   <%
      Uporabnik uporabnik = null;
//preverimo, ali obstaja seja za uporabnika
if (session.getAttribute("uporabnik") != null) {
                    uporabnik = (Uporabnik) session.getAttribute("uporabnik");
//preverimo pravice uporabnika
if(uporabnik.getPravice().equals("stranka")){
//preverimo ali je uporabljeni obrazec enctype="multipart/form-data"
if (ServletFileUpload.isMultipartContent(request)){
                DiskFileItemFactory factory = new DiskFileItemFactory();
                ServletFileUpload upload = new ServletFileUpload(factory);
                List items = upload.parseRequest(request);
                Iterator iter = items.iterator();
                while (iter.hasNext()) {
                   FileItem item = (FileItem) iter.next();
                   String name=item.getFieldName();
                   String value=item.getString();
                  //preverimo ali je podatek uporabnikov vnos v obrazec
                   if (item.isFormField()) {
        if(name.equals("height"))slika_height=Utilities.odstraniPx(value);
        if(name.equals("width"))slika_width=Utilities.odstraniPx(value);
        if(name.equals("top"))slika_top=Utilities.odstraniPx(value);
        if(name.equals("left"))slika_left=Utilities.odstraniPx(value);
        if(name.equals("fuga"))fuga_sirina=Utilities.odstraniPx(value);
        if(name.equals("sirina_fuge"))fuga_sirina=Utilities.odstraniPx(value);
        if(name.equals("stevilo_ploscic_sirina_st"))st_ploscic_sirina=Utilities.odstraniPx(val
        ue);
        if(name.equals("stevilo_ploscic_visina_st"))st_ploscic_visina=Utilities.odstraniPx(val
        ue);
        if(name.equals("barve_fug"))fuga_fuga_id=Utilities.odstraniPx(value);
        if(name.equals("velikost_ploscice"))ploscica_ploscica_id=Utilities.odstraniPx(value);
        ;
 } else {
        if(name.equals("slika")){
                ServletContext context = session.getServletContext();
                String realContextPath = context.getRealPath(request.getContextPath());
                File saveTo = new File(realContextPath+"/Slike/"+item.getName());
                item.write(saveTo);
                slika_ime=item.getName();
                      }
```

```
                    }
                }
            }
//v zbirko dodamo nov predmet povpraševanja
DBHelper.dodajPovprasevanje(new Povprasevanje(slika_ime, slika_width, slika_height,
slika_top, slika_left, st_ploscic_visina, st_ploscic_sirina, false, uporabnik.getUporabnik_id(),
ploscica_ploscica_id, fuga_fuga_id));
//ko je povpraševanje uspešno dodano v zbirko, naredimo preusmeritev
response.sendRedirect("naredisiizgled.jsp");
return;
            }
    }
```

5.3.3 Primerjava

PHP uporablja superglobalne spremenljivke, ki hranijo vse uporabnikove vnose in datoteke, ki so bile poslane s strani brskalnika. Tako nam omogoča enostaven dostop do njih. Pri Javi hrani uporabnikove spremenljivke predmet request, ki ravno tako ponuja enostaven dostop do njih, vendar se zadeva zaplete pri prenosu datotek, zato si olajšamo delo z uporabo paketa fileUpload, ki nam pomaga razčleniti poslane podatke.

5.4 Delo z zbirko podatkov

MySQL ja bila od nekdaj zbirka podatkov, za katero so se odločali razvijalci PHP-ja, ker jo lahko dobimo v tako imenovani triadi WAMP ali LAMP, ki pomeni Windows ali Linux, Apache, MySQL in PHP. Namestitev triade je enostavna in imamo takoj na voljo strežniško storitev za zbirko podatkov, na katero se lahko priključijo uporabniki iz istega ali oddaljenih računalnikov, ki imajo določene pravice. Zbirka podatkov je prav tako dobavljiva za večino današnjih sistemov kot samostojna aplikacija brezplačno. Za upravljanje zbirke z grafičnim vmesnikom lahko uporabimo eno izmed orodij, ki so prav tako brezplačna, phpMyAdmin. Model zbirke lahko izdelamo z MySQL Workbench, ki nam nato izdela zbirko z uporabo metode forward engener. Namesto zbirke MySQL podatkov bi lahko uporabili katero koli drugo zbirko, med katerimi se največ uporabljajo:

- Oracle Enterprise Server
- Oracle DB2
- Microsoft SQL Server
- Sybase Adaptive Server Enterprise (ASE)
- Teradata Database
- ADABAS
- FileMaker
- Microsoft Access

5.4.1 PHP

Ko želimo delati z zbirko podatkov, se moramo najprej povezati s strežnikom. PHP zato nudi funkcijo mysql_conect(), ki prejme do pet argumentov. Prvi trije so nizi: ime gostitelja, uporabniško ime in geslo. Četrti izbirni argument je logična vrednost, če je nastavljena na true, vsak klic mysql_connect() vrne novo povezavo. Drugače ta funkcija vrne trenutno odprto povezavo za vse klice, ki uporabljajo iste argumente. Če te argumente izpustimo, funkcija predvideva, da je gostitelj localhost, ter da geslo in uporabniško geslo nista bila nastavljena, kar ni priporočljivo. Mysql_connect() vrne povezavo vira, če je povezava uspešna, katero lahko shranimo v spremenljivko in jo nato uporabimo za delo z zbirko podatkov. Z ukazom mysql_select_db() izberemo, s katero zbirko podatkov želimo delati. Če to izpustimo, se predvideva, da bo uporabljen vir tisti, ki ga je vrnila zadnja povezava. Funkcija vrne true, če zbirka obstaja in imamo dostop. Pri odkrivanju napak nam pomaga funkcija die(), ki skript prekine. Kot argument ji lahko podamo bolj informativno besedilo, ki nam pomaga pri odkrivanju napak. Niz napak nam vrne funkcija mysql_error(), številko napake pa mysql_errorno(). Znak @ nam služi za preprečevanje prikaza opozoril, ki nam jih vračata mysql_connect() in mysql_select_db(). Za izvedbo stavkov uporabimo funkcijo mysql_query(), ki kot niz prejema SQL stavke. Če je SQL ukaz uspešno izveden funkcija, vrne true v primeru, da gre za vrsto poizvedb insertali delete, ko pa gre za poizvedbo select, vrne vir kot rezultat, ki ga preberemo z ukazom mysql_fetch_array(), le ta pa prejme dva argumenta, vir rezultata in tip želene tabele, ki je lahko asociativna, numerična ali oboje. Vsak klic te funkcije vrne vrstico v tabeli. Število vrstic, ki so bile vrnjene kot rezultat select stavka, preberemo s funkcijo mysql_num_rows(). Ko gre za ukaz update, mysql_query() vrne rezultat, ki ga preberemo z ukazom mysql_affected_rows(), in vrne, koliko vrstic je bilo dejansko spremenjenih [7].

```
//funkcija, ki se kliče vsakokrat, ko zahtevamo povezavo z zbirko
public static function povezi(){
    //nastavitve za povezavo z zbirko
    $server="localhost";
    $username="root";
    $password="";
    $database_name="mydatabasev1.0";
    //spremenljivka vsebuje povezavo z bazo
    $link=mysql_connect($server, $username, $password);
   //preverimo, če je povezava uspela
    if(! $link){
       die("Povezava z bazo spodletela".  mysql_error());
    }
   //povemo, katero zbirko želimo uporabljati
   @mysql_select_db($database_name, $link)
         or die("Povezava z bazo ".$database_name." je povzrocila ".  mysql_error());
    return $link;
  }
```

```
//primer funkcije za dodajanje uporabnika v zbirko podatkov
public static function dodajUporabnika(Uporabnik $uporabnik){
        $id_pravice=DBHelper::vrniIdPravice($uporabnik->getPravice());
        if($id_pravice!=-1){
           $link=DBHelper::povezi();
           $sql="INSERT INTO
uporabnik(ime,priimek,naslov,postna_stevilka,kraj,email,telefon_kontaktni,uporabnisko_ime,
geslo,mailing_list,pravice_pravice_id) VALUES('".$uporabnik->getIme()."', '".$uporabnik-
>getPriimek()."', '".$uporabnik->getNaslov()."', '".$uporabnik->getPostnaStevilka()."',
'".$uporabnik->getKraj()."', '".$uporabnik->getEmail()."', '".$uporabnik-
>getKontaktniTelefon()."', '".$uporabnik->getUporabniskoIme()."', '".$uporabnik-
>getGeslo()."', ".$uporabnik->isMailingL().", $id_pravice)";
           mysql_query($sql,$link);
           mysql_close();
        }
     }
```

5.4.2 Java

Za delo z zbirko podatkov v Javi uporabljamo namenjen gonilnik JDBC (Java DataBase Connectivity), ki standardizira dostop do relacijskih podatkovnih baz. Dobimo ga kot API na straneh podjetja Oracle v obliki datoteke JAR, ki jo vključimo v projekt. Nudi nam vzpostavitev povezave z večino zbirk podatkov, izvajanje povpraševanj SQL in strukturira rezultat povpraševanj v objekt. Pri uporabi JDBC najprej naredimo razred, ki pove, kateri gonilnik želimo uporabiti z ukazom Class.forName(), kateremu podamo niz gonilnika kot argument. Za delo z gonilniki skrbi razred DriverManager, ki ima metodo getConnection in vrne povezavo, ki jo priredimo spremenljivki. Metodi podamo tri argumente Url, uporabniško ime in geslo. Za poizvedbe SQL lahko uporabimo tri razrede Statment, PreparedStatment, CallableStatment. Proti napadu, imenovanemu SQL injection, ima PreparedStatment vgrajeno zaščito, zato najraje uporabimo tega. PreparedStatment ima tri metode za izvedbo povpraševanj: metode executeUpdate, ki jo uporabljamo za poizvedbe tipa update, insert in delete, executeQuery za stavke select in execute za stavke create in drop. ExecuteQuery nam vrne rezultat kot objekt, ki ga beremo z metodami razreda ResultSet-a. Najbolj običajna metoda je next(), ki nam ob vsakem klicu vrne naslednjo vrstico iz rezultata. Vrednosti preberemo z ustreznimi metodami get+podatkovni tip. Tem metodam lahko podamo argumente kot indeks v tabeli ali niz, ki predstavlja ime stolpca. Povezavo moramo nato zapreti z ukazom connection.close(). Za lovljenje napak, ki utegnejo nastati zaradi odsotnosti gonilnikov, pridobitvi povezave, izvajanje povpraševanj ali zapiranju povezave je Java striktna in zahteva uporabo blokov try catch[10].

```
//deklariramo globalno spremenljivo tipa Connection
private static Connection connection;
//funkcija, ki se kliče vsakokrat, ko zahtevamo povezavo z zbirko
public static Connection povezi() {
```

```
        boolean povezi=false;
        try {
          //izberemo, kateri gonilnik želimo uporabljati
           Class.forName("com.mysql.jdbc.Driver");
           String povezava="jdbc:mysql://localhost/mydatabasev1.0";
           try {
             //vzpostavimo povezavo z zbirko podatkov
              connection=DriverManager.getConnection(povezava,"root","");
              connection.setAutoCommit(false);
              povezi= true;
           //v primeru napake jo izpišemo
           } catch (SQLException e) {
              e.printStackTrace();
           }
        }
        //v primeru napake gonilnika izpišemo napako
        catch (ClassNotFoundException cnfe) {
           System.out.println("Error loadnig driver: "+cnfe);
        }
       //funkcija vrne spremenljivko, ki hrani povezavo z zbirko podatkov
        return connection;
           }


//primer funkcije za dodajanje uporabnika v zbirko podatkov
public static boolean dodajUporabnika(Uporabnik uporabnik) throws Exception{
        boolean dodan=false;
        int id_pravice=vrniIdPravice(uporabnik.getPravice());
        //preverimo, če obstaja taka pravica v zbirki
        if(id_pravice!=-1){
          //klic funkcije povezi, ki vzpostavi povezavo z zbirko
          povezi();

String sql = "insert into
uporabnik(ime,priimek,naslov,postna_stevilka,kraj,email,telefon_kontaktni,uporabnisko_ime,
geslo,mailing_list,pravice_pravice_id) values (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)";
//namesto zgornjih vprašajev bo PreparedStatment vstavilspremenljivke
PreparedStatement statement = connection.prepareStatement(sql);
            statement.setString(1,uporabnik.getIme());
            statement.setString(2,uporabnik.getPriimek());
            statement.setString(3, uporabnik.getNaslov());
            statement.setString(4,uporabnik.getPostnaStevilka());
            statement.setString(5,uporabnik.getKraj());
            statement.setString(6, uporabnik.getEmail());
```

```
    statement.setString(7, uporabnik.getKontaktniTelefon());
    statement.setString(8, uporabnik.getUporabniskoIme());
    statement.setString(9, uporabnik.getGeslo());
    statement.setBoolean(10,uporabnik.isMailingL());
    statement.setInt(11,id_pravice);
    Uporabnik preveri = null;
   //preverimo ali uporabnik že obstaja
    preveri = vrniUporabnika(uporabnik.getUporabniskoIme(), uporabnik.getGeslo());
    if(preveri==null){
       //če uporabnik ne obstaja, ga dodamo
          statement.executeUpdate();
       dodan=true;
    }
                else{
                        System.out.println("Uporabnik ze obstaja!");
                }
}
else{
   System.out.println("Taka pravica ne obstaja!");
}
try {
  //preverimo, ali je povezava vzpostavljena, da jo lahko zapremo
   if (connection != null)
            connection.close();
} catch (SQLException e) {
       e.printStackTrace();
}
return dodan;
   }
```

5.4.3 Primerjava

Oba jezika nudita učinkovite gonilnike in funkcije za delo z zbirko podatkov. Razvijalci PHP so se od nekdaj večinoma odločali za zbirko podatkov MySQL, zato je PHP še nekoliko bolj prilagojen in enostaven za delo z zbirko podatkov MySQL. Java se tudi pri uporabi zbirke podatkov pokaže kot striktni programski jezik. Za izvajanje povpraševanj nad zbirko podatkov zahteva uporabo treh funkcij, vsako za svojo vrsto izvajanja povpraševanj, medtem ko PHP reši vsa povpraševanja s funkcijo mysql_query(). Javanski razred PreparedStatment nam nudi zaščito pred napadom sql injection, kateri znake, ki jih uporabnik vpiše ali posreduje strežniku in bi utegnili biti ukazi zbirki podatkov, ustrezno označi, da jih tudi zbirka podatkov prepozna kot take. Podobno kot Java ima tudi PHP funkcijo mysql_real_escape_string(), kateri kot argument podamo niz, ki ga je uporabnik poslal strežniku. Funkcija vrne nov niz, ki ima vse kritične znake označene tako, da jih zbirka podatkov ne obravnava kot ukaze.

5.5 Piškotki

Ker je HTTP protokol brez stanja, je vsaka stran, ki jo uporabnik prenese s strežnika, ločena povezava. Za prenašanje stanja med stranmi nam služijo piškotki, ki hranijo določene majhne količine podatkov na odjemalčevi strani. Po standardu lahko strežnik zahteva, da je na odjemalcu shranjenih do 20 piškotkov z največjo velikostjo 4 Kb. Vsak piškotek je sestavljen iz imena, vrednosti, roka trajanja, poti in podatkov o strežniku, ki je zahteval, da se piškotek shrani. Nastavljene piškotke lahko bere samo izvorni strežnik, kar nam zagotavlja zasebnost uporabnikov. Uporabnik lahko nastavi brskalnik tako, da zavrača vse piškotke, zato se ne smemo zanašati na piškotke kot na osnovni element pri načrtovanju spletnih aplikacij.

5.5.1 PHP

Piškotek lahko v PHP nastavimo na dva načina. Prvi je, da uporabimo ukaz header, ki pošlje odjemalcu HTTP glavo, kot argument pa podamo zahtevo Set-Cookie: ime_piškotka=vsebina; expires=cas_preteka; path=pot . Drugi način, ki naredi to ravno tako, je funkcija setcookie(), katere rezultat je glava Set-Cookie. Funkcija setcookie() mora biti poslana pred kakršno koli drugo vsebino in sprejema ime piškotka, njegovo vrednost, rok trajanja, pot, domeno in vrednost 1 ali 0, ki pove, ali bo piškotek poslan samo z varno povezavo. Vsi argumenti so izbirni, razen prvega. Pot pove, katerim stranem bo piškotek poslan. Ker želimo, da bo piškotek poslan samo domeni strežnika, ki gosti skript, uporabimo spremenljivko $SERVER['SERVER_NAME']. Za brisanje piškotka je najbolje, da uporabimo enako funkcijo, kot ko smo piškotek nastavili za pretečenim rokom trajanja time()-60 [7].

```
//če hoče uporabnik, da ostane prijavljen, mu naredimo piškotek
if($zapomniSiMe!=""){
        //nastavimo piškotek, ki ima rok trajanja 30 dni
        setcookie("uporabniskoIme", $uporabniskoIme, time()+60*60*24*30);
        setcookie("geslo", $geslo, time()+60*60*24*30);
}
```

Ko želimo piškotek prebrati, nam zahteva vrne vse piškotke, ki so v domeni strežnika in se shranijo v superglobalno spremenljivko _COOKIES, ki predstavlja asociativno tabelo.

```
//preverimo, če sta imeni piškotkov nastavljeni
if(isset($_COOKIE['uporabniskoIme'])&&isset($_COOKIE['geslo'])){
        //spremenljivkam nastavimo vrednost piškotkov
        $upurabniskoIme=$_COOKIE['uporabniskoIme'];
        $geslo=$_COOKIE['geslo'];
}
```

5.5.2 Java

Piškotek naredimo z uporabo razreda Cookie, nato pa z response.addCookie(cookie), ki zgenerira glavo HTTP in posreduje odjemalcu zahtevo po namestitvi piškotka. Konstruktor razreda Cookie prejema dva argumenta, ime in vrednost piškotka. Objekt ima dodatne funkcije za pridobivanje in nastavljanje nastavitev domene, poti in roka trajanja.

```
//če hoče uporabnik ostati prijavljen, mu naredimo piškotek
if(zapomniSiMe!=null){
        //naredimo nov predmet tipa Cookie in mu nastavimo ime ter vrednost
         Cookie uporabniskoIme= new Cookie("uporabniskoIme",
uporabnik.getUporabniskoIme());
              //nastavimo piškotek, ki ima rok trajanja 30 dni
              uporabniskoIme.setMaxAge(60*60*24*30);
              response.addCookie(uporabniskoIme);
              Cookie geslo= new Cookie("geslo", uporabnik.getGeslo());
              geslo.setMaxAge(60*60*24*30);
              response.addCookie(geslo);
           }
```

Za pridobitev nastavljenih vrednosti piškotkov skrbi ukaz request.getCookies, ki vrne tabelo piškotkov. Vsak piškotek je v tabeli zapisan z njegovim imenom in vrednostjo, tako da se moramo najprej iterirati čez vse piškotke in primerjati, kateri ima želeno ime z ukazom getName(), šele nato pa lahko pridobimo vrednost z ukazom getValue() [11].

```
//funkcija, ki poišče piškotek z imenom geslo, če obstaja
public static String getcookieGeslo(Cookie[] cookie){
    String value=null;
    //sprehodimo se čez vse piškotke
    for(int i=0; i<cookie.length;i++){
     //če je ime trenutnega piškotka geslo, nastavimo spremenljivki value vrednost piškotka
     if(cookie[i].equals("geslo"))
          value=cookie[i].getValue();
    }
    return value;
  }
```

5.5.3 Primerjava

Pri Javi imamo več dela v primerjavi s PHP, ker se moramo za pridobitev piškotka sprehoditi čez vse piškote in iskati, kateri ima iskano ime. Za nastavljanje piškotov je koncept enak, le razlika je v sintaksi, kjer moramo v Javi najprej narediti nov objekt, ki ga pošljemo odjemalcu.

5.6 Seje

Seja (sessions) je koncept, ki uporabnikom spletne aplikacije priskrbi edinstven identifikator, ki ga je mogoče uporabiti od dostopa do dostopa za pridobivanje podatkov, ki so povezani s tem id-jem. Ko uporabnik odpre stran, se mu dodeli nova seja, v primeru, da seja že obstaja, vse spremenljivke, ki so bile povezane s sejo, postanejo na voljo kodi uporabnika. Najpogosteje se seje uporablja za prenos uporabniških podatkov med stranmi spletne aplikacije, zagotavljajo pa tudi varovanje dostopa do posamezne strani, ki zahteva določene pravice. Seje se shranjujejo v pomnilniku strežnika. Privzeto se odjemalcu pošlje piškotek, ki

vsebuje id seje. Nato na vsaki strani preverjamo, ali obstaja na odjemalcu piškotek, ki ima id seje in pa, če ima ta uporabnik v seji kakšne podatke, ki jih potrebujemo. Druga možnost je, da pošiljamo id seje kot parameter med stranmi.

5.6.1 PHP

Po privzetih nastavitvah se seje v PHP ne začnejo samodejno. Začnemo jih z ukazom session_start() v vsakem dokumentu, kjer so seje potrebne. Nastavitve lahko spremenimo tako, da se seja začne samodejno, s tem, da nastavimo vrednost session.auto_start na 1 v datoteki php.ini. Takoj, ko se seja začne, imamo dostop do id-ja seje uporabnika, in sicer z ukazom session_id(), ki nam omogoča nastavljanje in pridobivanje id-ja seje. Ko je skript prvič zagnan, se id zgenerira in ostane enak, vse dokler se seja ne prekine. Seja se privzeto prekine, ko uporabnik zapre brskalnik. To vedenje lahko spremenimo tako, da spremenimo nastavitev session.cookie_lifetime v datoteki php.ini ali uporabimo funkcijo v samem skriptu session_set_cookie_params, ki ji kot parameter podamo čas v sekundah [7].

```
//funkcijo session_set_cookie_params napišemo pred funkcijo session_start
session_set_cookie_params(300);
session_start();

//dodajanje uporabnika v sejo
$_SESSION['uporabnik']=$uporabnik;
```

5.6.2 Java

Pri privzetih nastavitvah se seje začnejo samodejno, kajti vsak servlet vsebuje objekt PageContext, ki med drugim skrbi za upravljanje s sejami in drugimi atributi v povezavi s samim servletom. V primeru, da ne želimo, da se seje generirajo samodejno, moramo to nastaviti na vsaki strani z ukazom <%@ page session="false" %>, ki ga pogosto uporabljamo pri iskanju napak, v primeru, da smo pomotoma poizkušali uporabiti spremenljivko seje. V sejo shranimo želene objekte enostavno z ukazom session.setAtribute, ki mu podamo edinstveno ime objekta in sam objekt [12]. Primer dodajanja uporabnika v sejo:

```
//Naredimo predmet uporabnik, ki hrani podatke o uporabniku
 Uporabnik uporabnik = DBHelper.vrniUporabnika(uporabniskoIme, geslo);
 //Seji dodamo predmet uporabnik
session.setAttribute("uporabnik", uporabnik);
 //Seji nastavimo čas preteka seje v sekundah
 session.setMaxInactiveInterval(300);
```

Sejo prekinemo z ukazom session.invalidate().

5.6.3 Primerjava

Seja kot koncept je pri obeh enaka. Razvijalci so se pri posameznem jeziku odločili za različne nastavitve, ki pa jih pa lahko poljubno nastavimo in priredimo po potrebah, ki jih

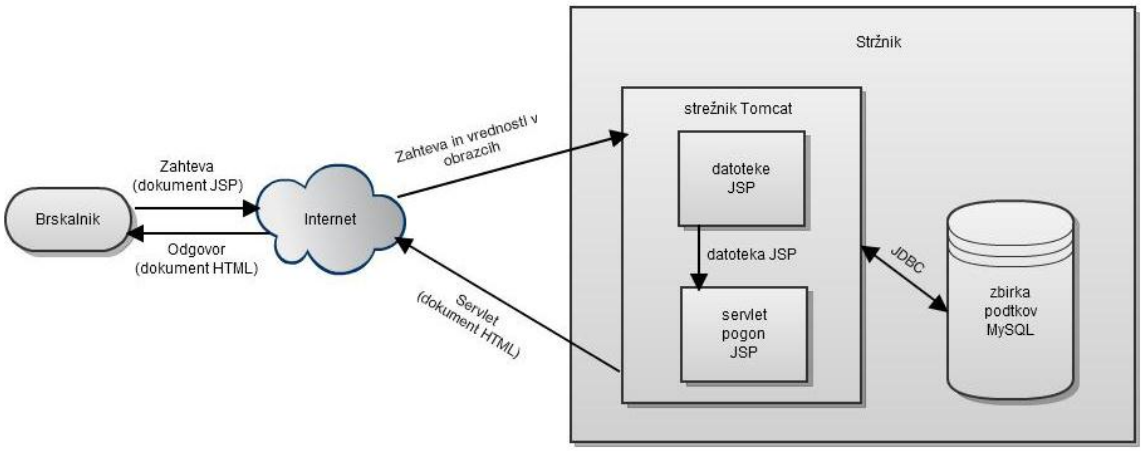
zahteva določen projekt. Pri obeh, tako v PHP-ju, kot JSP-ju, se lahko shranjujejo objekti. PHP seje shranjuje kot datoteke v določeno mapo z imenom id-ja, ki jo lahko nastavljamo z opcijo session.save_path. JSP pa shranjuje seje kot objekte v pomnilniku.

6. Praktična aplikacija

6.1 Spletni strežnik

Za razvoj spletnih aplikacij potrebujemo spletni strežnik, ki si ga lahko namestimo na domači računalnik ali zakupimo gostovanje na katerem od ponudnikov na spletu. Za namestitev strežnika na domačem računalniku potrebujemo operacijski sistem, na katerega namestimo namensko programsko opremo. Za delo s PHP-jem obstaja tako imenovana triada WAMP oziroma LAMP, odvisno od operacijskega sistema, s katero namestimo PHP, MySQL in Apache. Za izvedbo praktične aplikacije sem uporabil različico WAMP 5.3.5. Sama namestitev je zelo enostavna in ne potrebuje nobenega predznanja. Privzeta mapa strežnika je www, ki se nahaja v mapi wamp. Delovanje sem preizkusil tako, da sem naredil datoteko Test.php in vanjo vpisal izraz *phpinfo();*. Shranil sem jo v mapo www, v brskalnik podal naslov localhost/Test.php in že se je prikazala stran, ki prikazuje konfiguracijo in nameščene komponente. Razvojno okolje NetBeans sem nastavil tako, da je mapa z datotekami projekta v privzeti mapi strežnika in da tam tudi shranjuje nove datoteke. Razvojno okolje je bilo pripravljeno za nadaljnji razvoj.

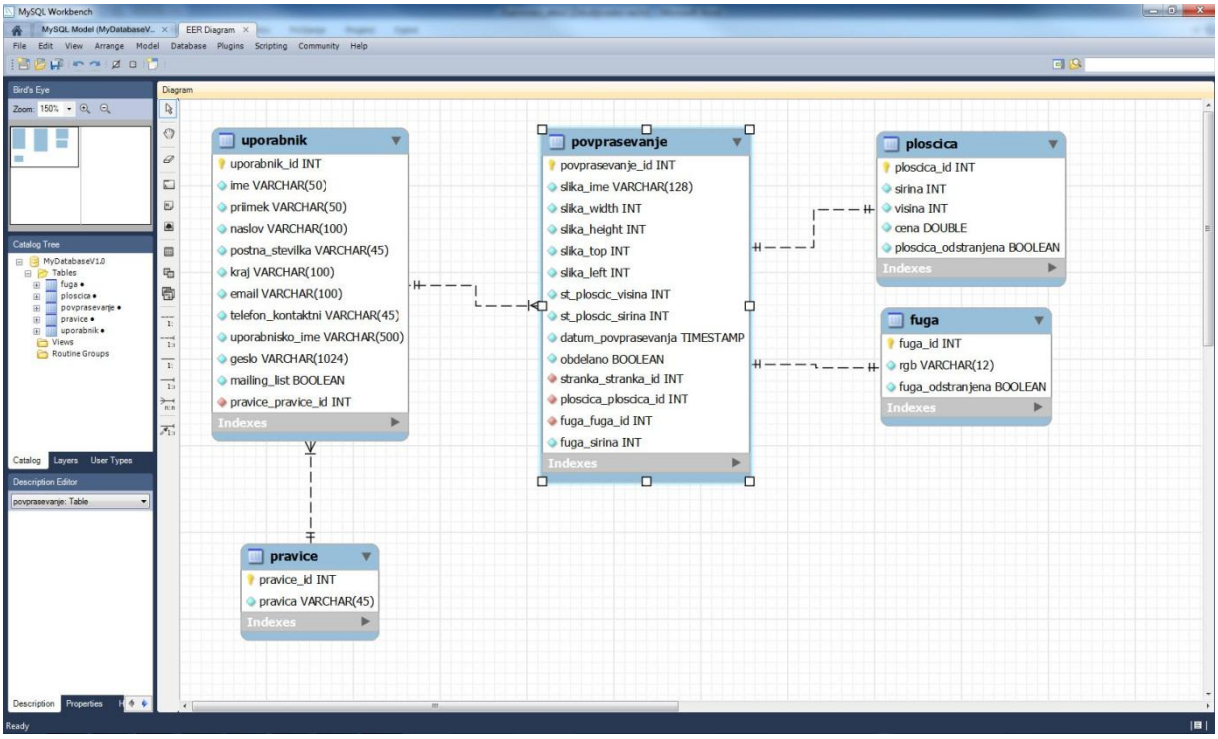
Med strežniki za poganjanje strani JSP sta najpogosteje uporabljena Tomcat in Glassfish. NetBeans ima prednastavljen strežnik Glassfish, zato sem ga na začetku tudi uporabljal. Ko sem hotel prenašati datoteke na strežnik, sem naletel na težavo z uporabo obrazca tipa *enctype='multipart/form-data'*. Med iskanjem rešitve sem opazil, da so informacije za delo z strežnikom Glassfish veliko bolj skope v primerjavi s strežnikom Tomcat, za katerega sem dobil namige takoj. Zamenjal sem strežnik Glassfish s strežnikom Tomcat in namestil verzijo Apache commons fileupload 1.2.2, za katero je priročnikov za uporabo na spletu dovolj. Slika 11 prikazuje uporabljene spletne komponente pri izdelavi aplikacije JSP. Brskalnik pošlje parametre v obrazcu strežniku, ki jih skupaj z datoteko JSP posreduje pogonu servlet. Strežnik Tomcat pošlje servlet nazaj brskalniku, ki sedaj prikazuje posodobljene podatke.



Slika 11:Prikaz spletnih komponent JSP

6.2 Zbirka podatkov MySQL

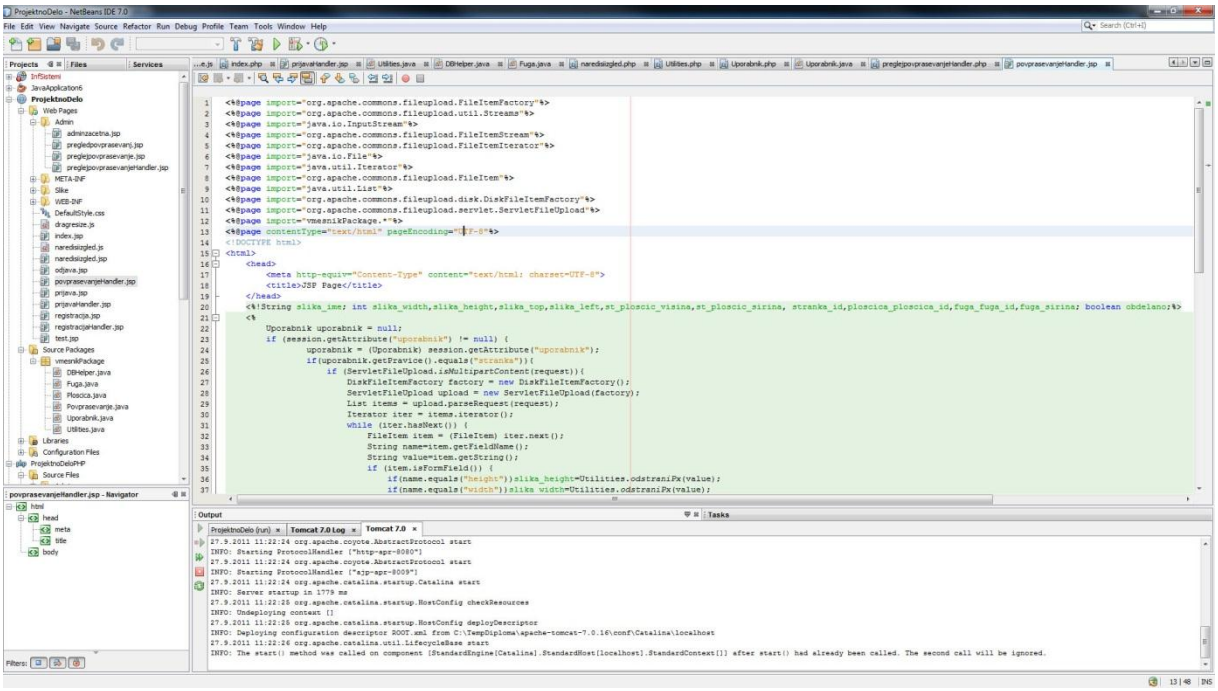
MySQL ja bila od vedno zbirka podatkov, za katero so se odločali razvijalci PHP, ker jo lahko dobimo v tako imenovani triadi WAMP ali LAMP, kar pomeni Windows ali Linux, Apache, MySQL in PHP. Namestitev triade je enostavna in imamo takoj na voljo strežniško storitev za zbirko podatkov, na katero se lahko priključijo uporabniki iz istega ali oddaljenih računalnikov, ki imajo določene pravice. Zbirka podatkov je prav tako dobavljiva za večino današnjih sistemov kot samostojna aplikacija brezplačno. Za upravljanje zbirke z grafičnim vmesnikom lahko uporabimo eno izmed orodij, ki so prav tako brezplačna, phpMyAdmin. Model zbirke sem izdelal z MySQL Workbench, ki nam izdela zbirko z uporabo metode forward engener. Slika 12 prikazuje grafični vmesnik orodja MySQL Workbench, ki je enostaven in intuitiven.



Slika 12: Izgled razvojnega okolja MySQL Workbench

6.3 Razvojna orodja

Za razvijanje aplikacij PHP so najbolj znana orodja ZendStudio, PHPEdit, Notepad++, phpDesigner, Bluefish, Context Editor, Eclipse, NetBeans in NuSphere PhpED. Za razvijanje aplikacij JSP pa Eclipse, NetBeans, JEdit, JBuilder X, IntelliJ IDEA, JDeveloper, SlickEdit in JEdit. Med njimi sta Eclipse in NetBeans, ki sta zelo priljubljena in omogočata pisanje skriptov PHP, skriptov JSP in aplikacij Java. Namestitev orodja NetBeans je enostavna in že vključuje funkcije za razvoj aplikacij. Slika 13 prikazuje izgled uporabniškega vmesnika. Že privzete nastavitve omogočajo, da naredimo nov projekt JavaWEB za izdelavo strani JSP, ga poimenujemo in poženemo. Odpre se brskalnik in že vidimo napis Hello World!. Privzeto NetBeans uporablja strežnik Glassfish, ki ga lahko zamenjamo s Tomcatom in nastavimo vsak projekt posebej, katerega želimo uporabljati.



Slika 13: Razvojno orodje NetBeans

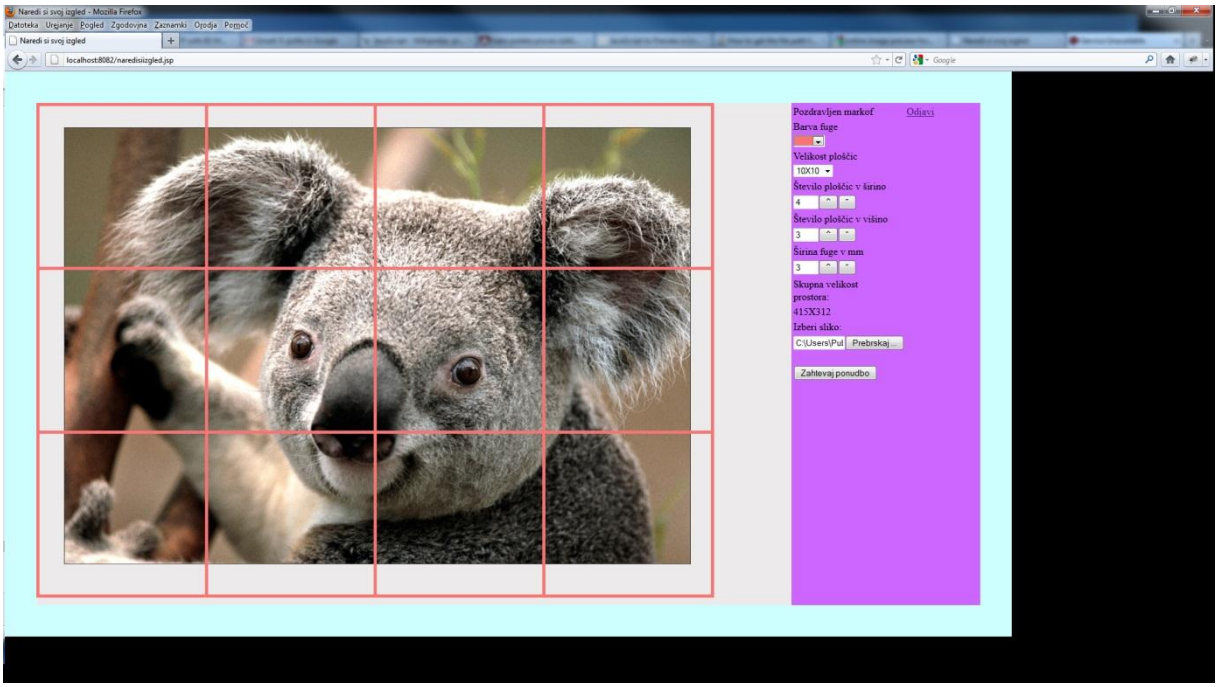
Poganjanje skriptov PHP z orodjem NetBeans poteka nekoliko drugače, ker moramo nastaviti:

- mapo datoteke php.exe, ki nam omogoča poganjanje skriptov PHP;
- korensko mapo strežnika Apache in naslov url za dostop do skriptov.

Med razvojem aplikacij včasih naletimo na težave pri poganjanju kode, zato se nam splača nastaviti tudi razhroščevalnik, ki nam pomaga pri odkrivanju napak. Namestil sem XDebug 2.1.5, ki je razširitev za PHP in nastavil NetBeans, da ga uporablja pri razhroščevanju. Tako imamo pregled nad potekom programa.

6.4 Implementacija

Implementacija praktične aplikacije je, poleg strežnika in strežniške kode, zahtevala tudi izdelavo uporabniškega vmesnika. Ideja je bila, da si uporabniki izberejo velikost keramičnih ploščic, barvo fuge, naložijo poljubno sliko in si prilagodijo izgled. Za izdelavo uporabniškega vmesnika, kot ga prikazuje slika 14, sem uporabil JavaScrip, ki je objektni skriptni jezik za izdelavo interaktivnih spletnih strani, ki ga izvaja brskalnik.



Slika 14: Izgled uporabniškega vmesnika v brskalniku Firefox

Tako lahko uporabniki izbirajo velikost in barvo fug med ploščicami, velikost keramičnih ploščic in si naložijo izbrano sliko, ki jo nato lahko povečujejo, zmanjšujejo in premikajo. Ko si uporabnik izbere sliko, se ta pokaže v vmesniku in se, šele ko uporabnik potrdi, pošlje strežniku, kjer pa nastane težava, saj različni spletni brskalniki izpostavijo različne objekte za uporabo. Ko si uporabnik izbere sliko, z uporabo obrazca enctype="multipart/form-data" in polja <input type="file" name="ime">, jo brskalnik Firefox shrani lokalno in spremeni vir slike, tako dobimo značko HTML <img width="100" height="100" src="moz-filedata:13a6a556-db09-4ce9-b3d9-4158032ac1c7">. Že samo v različnih verzijah brskalnika Firefox so ukazi JavaScript različni. Drugi parametri slike, kot sta velikost in pozicija, se v aplikaciji zapišejo v skrite značke HTML in se pošljejo po metodi post strežniku, ko to uporabnik zahteva s potrditvijo. Strežnik parametre prebere in zapiše v zbirko podatkov tako, da je skrbniku še enkrat omogočen predogled, preden pošlje ponudbo.

7. Sklep

Cilj diplomske naloge je bil pregled in primerjava dveh med najbolj uporabljenimi tehnologijami za razvoj, izdelavo in vzdrževanje spletnih okolij. Tako PHP kot JSP sta močni tehnologiji, ki razvijalcem ponujata funkcionalnosti, ki jih zahteva današnji splet. Primerjati je bilo potrebno vse glavne koncepte programskih jezikov, ki omogočajo izdelavo dinamičnih spletnih strani.

Sam pristop do programiranja je med JSP in PHP različen, ker je Java popolnoma objektno orientiran programski jezik, ima striktno sintakso in vnaprej točno določene podatkovne tipe, zaradi česar postane koda bolj čista in berljiva. Tudi PHP ima v zadnjih verzijah podporo objektnemu programiranju, vendar ima manj striktno sintakso kot Java. Pri PHP se podatkovni tipi določajo, ko spremenljivki določimo vrednost, kar zna privesti do napak v daljših skriptih, ker se podatkovni tip spremenljivk lahko skozi izvajanje spremeni.

Dodatne funkcije, kot so pošiljanje elektronske pošte in nalaganje datotek na strežnik, so pri Javi omogočene preko knjižnic, ki jih moramo uvoziti dodatno, so zapletene in je potrebno več kode za osnovne funkcionalnosti kot pri PHP.

Delo z zbirko podatkov je, če uporabljamo MySQL, bistveno lažje s PHP, ker je bil že v začetku zasnovan skupaj z zbirko MySQL in so vse funkcionalnosti že vključene v osnovno namestitev PHP, koda pa je enostavna in kratka. Pri Javi se uporablja standardno knjižnico JDBC, ki vsebuje gonilnike za vse zbirke in je po potrebi lažje menjati zbirko podatkov.

Obe tehnologiji učinkovito uporabljata seje in piškotke. PHP shranjuje vse spremenljivke tako sej kot piškotkov v superglobalne spremenljivke, do katerih imamo dostop z asociativnimi imeni spremenljivk, ki nam nekoliko olajšajo delo. Pri Javi hranita piškotke in seje ločena predmeta, do spremenljivk pa dostopamo preko funkcij, ki jih imata. Do vrednosti spremenljivke, shranjene v piškotku, lahko pridemo le tako, da se sprehodimo čez vse spremenljivke in iščemo asociacijo z želenim imenom spremenljivke.

Razvijalci manjših spletnih aplikacij večinoma uporabljajo PHP zaradi enostavnosti, medtem ko se JSP veliko uporablja za izdelavo večjih spletnih okolij, ki zahtevajo interakcijo z različnimi platformami, še posebej, če temeljijo na okolju Java. Za obe tehnologiji obstaja vrsta razvojnih okolij, ki so enostavno dobavljiva, veliko pa jih je tudi brezplačnih.

Kazalo slik

Literatura

[1] Splet. Dostopno na:
http://sl.wikipedia.org/wiki/Splet

[2] Web application. Dostopno na:
http://en.wikipedia.org/wiki/Web_application

[3] The Open Web Platform. Dostopno na:
http://www.w3.org/wiki/Open_Web_Platform

[4] JavaServer Pages. Dostopno na:
http://sl.wikipedia.org/wiki/JavaServer_Pages

[5] PHP. Dostopno na:
http://sl.wikipedia.org/wiki/PHP

[6]List of php.ini directives. Dostopno na:
http://php.net/manual/en/ini.list.php

[7] Z. Matt, Naučite se PHP v 24 urah, Ljubljana: Pasadena, 2004, pogl. 4, 5, 6, 7, 8, 9, 10, 13, 19, 20.

[8] M. Uroš, F. Borut, Java 2, Ljubljana: Pasadena, 2004, 1. del pogl. 4, 5.

[9] Handling Form-based File Upload with Java Servlet or JSP. Dostopno na:
http://www.developershome.com/wap/wapUpload/wap_upload.asp?page=jsp

[10] JDBC Introduction. Dostopno na:
http://download.oracle.com/javase/tutorial/jdbc/overview/index.html

[11] Working With Cookies. Dostopno na:
http://download.oracle.com/javase/tutorial/networking/cookies/index.html

[12] Working with JSP Sessions. Dostopno na:
http://www.exforsys.com/tutorials/jsp/jsp-session-object-methods.html