WebDiP drugi kolokvij

WebDiP_06

Proces dizajna:

- Prvi korak u dizajniraju Web mjesta je definiranje ciljeva
- Bez jasno postavljenje misije i svrhe projekt može zalutati, zapeti ili nastaviti nakon prikladne točke kraja.
- Pažljivo planiranje i jasna svrha ključni su u izgradni Web mjesta, posebno kad a se posao obavlja u timu.

Planiranje Web mjesta je proces s dva dijela:
1. Sakupljanje partnera, analiza potreba i clijeva, rad na dotjerivanju vlastitih planova
2. Kreiranje dokumenta sa specifikacijom koji sadrži detalje o tome **što** se misli raditi i **zašto**, koja **tehnologija** je potrebna, koji **sadržaji** su potrebni, **trajanje** procesa, **troškovi, kako** će se **procjeniti** rezultati provedenih napora.

Web razvojni tim: web projekt nije projekt jedne osobe nego cijelog tima.
Ključne vještine za web razvojni tim su: strategija i planiranje, upravljanje projektom, informacijska arhitektura i dizajn korisničkog sučelja, grafički dizajn za web, Web tehnologija, produkcija web mjesta.

Garret-ovih 9 stupova uspješnih web timova !!!!!!!
1. Istraživanje korisnika: korisniku usmjeren dizajn znači razumjeti što korisnici trebaju, kako oni razmiđljaju i kako se ponašaju. Istraživanje pruža sirova promatranja koja hrani taj pogled na ljude koje treba posluživati web mjesto.
2. Strategija mjesta: određivanje vlastitih ciljeva za mjesto može biti teško. Počinje od zajedničkog razumijevanja svrhe web mjesta za organizaciju, kako će se postaviti prioriteti različitih ciljeva web mjesta.
3. Tehnološka strategija: web mjesta su tehnološki složena i s vremenom postaju sve zamršenija. Identificiranje strategije bitno je da se izbjegnu skupe pogreške.
4. Strategija sadržaja: sadržaj je obično razlog zašto korisnici dolaze na web mjesto. Koji sadržaj treba ponuditi da se zadovolje očekivanja korisnika. Koliko je sadržaj prikladan i koji oblik treba imati. Prije nego se proizvede sadržaj, treba odgovoriti na temeljna pitanja strategije sadržaja.
5. Apstraktni dizajn: informacijska arhitektura i dizaj interakcije prevode strateške ciljeve u konceptualni okvir za krajnje korisničko iskustvo.
6. Tehnološka implementacija: izgradnja tehničkih sustava uključuje mnogo tečkog rada i specijalizirana znanja: jezici i protokoli, programiranje i pronalaženje i ispravljanje pogrešaka, testiranje i pročišćavanje.

7. Prozivodnja sadržaja: nije dovoljno znanje koji sadržaj je potreban. Treba znati kako ga proizvesti. Skupljanje sirovih informacija, pisanje i uređivanje, ,određivanje uredničkog radnog tijeka i odobravanje, sve su djelovi proizodnje.
8. Konkretni dizajn: prije nego što apstraktni dizajn postane potpuno realizirano korisničko iskustvo, potrebno je odrediti specifične detalje sučelja, navigacije, informacijskog dizajna i vizualnog dizajna. Ovo područje bitno je za kreiranje konačnog proizvoda.
9. Upravljanje projektom: mjesto koje zajedno povezuje sve taktičke kompetencije kao i motor koji vodi projekt prema završteku, upravljanje projektom zahtjeva visokospecijalizirani skup vještina za sebe. Zanemarivanje tog područja često završi u prekoračenju rokova i troškova.

Informacijska arhitektura
Opisuje cjelokupne konceptualne modele i opće dizajne koji se koriste za planiranje, strukturiranje i sastavljanje web mjesta.
Svako web mjesto ima informacijsku arhitekturu, ali i tehnike informacijske arhitekture posebno su važne kod velikih, složenih web mjesta gdje su primarni ciljevi:

- Organizirati sadržaj web mjesta u taksonomije i hijerarhije informacija
- Prenijeti konceptualne prikaze i cjelokupnu organizaciju web mjesta timu za dizajn i klijentima
- Postaviti standardne i specifikacije za html semantičko označavanje, te formatiranje i rad s tekstualnim sadržajem
- Dizajnirati i implementirati standarde i strategije za optimizaciju pretraživanja.

Osnovni koraci u organiziranju informacija:

- Inventura sadržaja: što se ima, što je potrebno
- Uspostavljanje hijerarhijskog izgleda za sadržaj i kreiranje kontroliranog rječnika tako da se glavni sadržaj, struktura mjesta i elementi navigacije uvijek mogu konzistentno identificirati
- Cjepanje – podjeli sadržaje u logičke cjeline s konzistentnom modularnom strukturom
- Nacrtaj dijagrame koji prikaziki strukturu mjesta i grubo skiciraj stranice s listom glavnih navigacijskih veza
- Analiziraj sustav kroz interaktivno testiranje organizacije sa stvarnim korisnicima

Struktura web mjesta treba odgovarati očekivanim korisnicima i mentalnim modelima koji oni stvaraju
Mogu biti: slijedovi, hijerarhije, paučine.
Slijedovi: najjednostavniji oblik, može biti kronološko, logički slijed aktivnosti, abecedno.
Odgovara za trening ili edukacijska mjesta.
Složenija web mjesta mogu biti organizirana u logičke slijedove s time da svaka stranica u slijedu može imati jednu ili više stranica za digresije, usputnih informacija ili informacija na drugim web mjestima.
Hijerarhije: odgovara organizaciji složenijih web mjesta. Počinju osnovnom stranicom koja sadrži veze na stranice izbora podkategorija.
Paučine: postavlja granice za korištenje informacija tako da veze dopuštaju slobodan tok ideja.

Prezentacija informacijske arhitekture:
Rad na informacijskoj arhitekturi bit će uspješniji ako se vizualizira.
Pojam traženja puta za opis svog koncepta okolišne čitljivosti kojeg čine elementi izgrađenog okoliša koji nam omogućavaju da se uspješno snalazimo kroz složene prostore kao što su gradovi.
Traženje puta ima 4 temeljne komponente:

- Orijentacija
- Odluke o rutama: mogu li pronaći put
- Mentalno povezivanje: da li su moja iskustva dovoljno konzistentna i razumljiva
- Zatvaranje: moglu li prepoznati da sam stigao na pravo mjesto.

Princip za traženje puta na web mjestu:

- **Putanje** kreiranje konzistentne, dobro označene navigacijske putove
- **Područja** kreiraj jedinstvene ali povezane identitete za svako područje
- **Čvorovi** ne zbunjuju korisnika s previše izbora na osnovnoj stranici ili na glavnim izborničkim stranicama
- **Znakovi** koristi konzistentne znakove u navigaciji web mjesta i grafiku za orijentaciju korisnika.

Dizajn sučelja
Direktne veze na određenu stranicu unutar web mjesta ili putem pretraživača
Smjestiti određene elemente na mjesta na kojima ih korisnik očekuje
Izbjegavati stranice u kojima nema veza
Struktura web mjesta
Dizajn web mjesta na temelju eksplicitne i jednoznačne primjene CSS-a
Uvijek se počinje s radom na predlošcima internih stranica, a na kraju s početnom
Predlošci stranica
S ozbirom na mogući sadržaj stranica može se pripremiti više varijanti na bazi broja stupaca.
S obzirom na mogući sadržaj stranica ili funkcionalnosti može se pripremiti varijanta s minimalnim sučeljem.
Dizajn stranice
Važno je prilagoditi dizajn prema osobinama korisničkog uređaja.
Slijed elemenata u dokumentu treba pratiti njihovu važnost.
Vizualni dizajn temelji se na funkcionalnim područjima.
Konzistentnost je važna osobina dobrog dizajna.
Dizmenzije stranice, prostor stranice (širina manja od ukupne širine preglednika)

Screen Resolutions (03.2018.)		
1	1366x768	18.39%
2	1024x768	15.19%
3	1280x800	10.85%
4	1280x1024	7.51%
5	1440x900	5.99%
6	1920x1080	5.55%
7	320x480	4.73%
8	1600x900	3.59%
9	768x1024	3.41%
10	1680x1050	3.34%

WebDiP_07

Evolucija Web dizajna

1. generacija – glavni cilj bio je prisustvo na webu uz izgradnju web mjesta, a klijenti će sami doći te se uglavnom nije posebna briga posvećivala klijentima pa nisu poznati podaci o načinu izvršvanja web mjesta s gledišta klijenta

2. generacija – glavni cilj bilo je oglašavanje da se postoji na webu uz složeni web dizajn kojeg nije bilo jednostavno razumijeti, a sve zbog brzog razvoja uslijed velike konkurencije

3. generacija – web dizajn usmjeren na korisnika kako bi se postigla jednostavnost korištenja, zadovoljavajuće performanse, povjerenje i opće zadovoljstvo web mjestom.

Pristupačnost i korisnost

Odnosi se na osiguranje ekvivalentnog korisničkog iskustva za osobe s poteškoćama, uključujući osobe koje imaju oštećenja koja proizlaze iz starosne dobi.

Korisnost se odnosi na dizajn proizvoda tako da bude djelotvoran, učinkovit i zadovoljavajući.

Ben Shneiderman : 8 zlatnih pravila dizajn sučelja: teži konzistentnosti, omogući čestim korisnicima da koriste prečice, ponudi informativnu povratnu vezu, dizajniraj dijalog da ima logički slijed od početka do kraja, ponudi jednostavno rukovanje s pogreškama, dopusti jednostavno vraćanje akcija, podrži interno mjesto kontrole, smanji punjenje kratkoročne memorije.

Jakob Nielsen : 10 principa za dizajn korisničkog sučelja: vidljivost statusa sustava, poklapanje između sustava i relanog svijeta, korisnička kontrola i sloboda, konzistentnost i sloboda,

konzistentnost i norme, prevencija pogrešake, prepoznavanje umjesto sjećanje, fleksibilnost i efikasnost korištenja, estetski i minimalistički dizajn , pomoći korisniku u prepoznavanju, dikagnosticiranju i oporavku iz pogrešaka, pomoć i dokumentacija.

Rane faze dizajna i prototipa: koriste se 3 dizajnerska pomagala: mapa mjesta, ilustrirani scenarij, shematski prikazi.

Uzorci web dizajna

Uzorci prenose poglede na probleme u dizajnu tako da obuhvaćaju suštinu problema i njihovih rješenja u sažetom obliku. Detaljno opisuju problem, razlog za rješenje, kako primijeniti rješenje i neke od ustupaka u primjeni rješenja.

Uzorci web mjesta: žanr web mjeta, kreiranje navigacijskog okvira, kreiranje snažne početne stranice, pisanje i upravljanje sadržajem, izgradnja povjerenje i kredibilnosti, osnovno e-poslovanje, napredno e-poslovanje, pomoć klijentima da završe poslove, dizajniranje efektivnog izgleda stranice, stvaranje brzog i relevantnog pretraživanja web mjesta, stvaranje jednostavne navigacije, ubrzavanje web mjesta, mobilni web.

WebDiP_08

Pojam autorski alat odnosi se na široku paletu softvera koji se koristi za kreiranje Web sadržaja, uključujući:

- Alati za uređenje posebno dizajnirani da proizvode Web sadržaj
- Alati koi nude opciju pohranjivanja materijala u Web formatu
- Alati koji transformiraju dokumente u Web formate
- Alati koji proizvode multimedijske elemente, posebno kada su namijenjeni za korištenje na Webu
- Alati za upravljanje web mjestom ili izdavaštvom web mjesta, uključujući alate koji automatsko generiraju dinamičke sadržaje iz BP, alate za neposrednu konverziju i alate za izdavaštvo Web mesta
- Alati za upravljanje izgledom

W3C smjernice:

1. podržavati pristupačne autorske prakse
2. generirati standardno označavanje
3. podržavati kreiranje pristupačnog sadržaja
4. pružati puteve provjere i ispravljanje nepristupačnog sadržaja.
5. integrirati pristupačna rješenja u opći „look and feel“
6. promovirati pristupačnost u pomoći i dokumentaciji
7. osigurati da je autorski alat pristupačan autorima s tjelesnim oštećenjima

WebDiP_14

Programiranje na strani poslužitelja što se može očekivati: pristup do komandi operacijskog sustava, pristup do datotečnog sustava, pristup do baze podataka, povezivanje s drugim servisima, dinamičko kreiranje sadržaja stranice

Koji se sutavi temelje na PnSP : B2C, B2B, B2G, G2C

Mogući problemi kod PnSP: sigurnost poslužitelja, podataka na poslužitelju i lokalne mreže

Vrste programskih jezika: kompilatorski, interpreterski, interpreterski/kompilatorski, interpreterski/kompilatorski (nakon 1. izvršenja) ugrađen u HTML, prošireni HTML

CGI – Common Gateway Interface

- Sučelje kojim se proširuje funkcionalnost poslužitelja
- Metoda kojom poslužitelj može prikupljati podatke iz baze podataka, dokumenata i drugih programa i prezentirati te podatke preko Weba

Potrebne osobine da bi se koristio za CGI: pisanje na standardni izlaz, čitanje sa standardnog ulaza, čitanje varijable okoline
CGI može biti pisan u gotovo bilo kojem programskom jeziku.
Pitanje je kompromisa poželjnih osobina: prenosivost, ugrađene osobine, brzina izvršavanja, sigurnost
Na početku se najčešće koristio Perl jer ima visoku razinu za prve dvije osobine, ali je interpreterski jezik zbog čega je spor.

ASP

- Razvijeno u Microsoft-u
- Skriptirajuće oznake

```
<% i %>
```

–označava da se kod izvršava na poslužitelju

- `@LANGUAGE`

–određuje se skriptirajući jezik (default VBScript)

```
–<% @LANGUAGE = "VBScript" %>
```

- Potrebno je i moguće konfigurirati poslužitelj da provodi operacije na određeni i/ili zahtjevani način.
- Problem/prednost je stroga veze na Windows platformu (IIS i PWS).
- Postoji i nezavisni Web poslužitelj pod Linux-om koji je ASP usmjeren (Chil!iSoft)

C#

- Proizveden u Microsoft-u (see-sharp)
- Izveden iz C/C++ (čitaj Java)
- Razvijen s ciljem:
  - brzog učega
  - ponovnog korištenja
  - komponentnog razvoja
- Temeljni jezik .NET tehnologije
- Komplira se pod .NET SDK ili Visual Studio.NET
- Može se povezati na većinu baza podataka i koristiti razne servise
- Može se koristiti za razvoj stolnih i Web aplikacija

PHP

- Autor: Rasmus Lerdorf, 1994. g.
- Sličan C jeziku
- Skriptirajuće oznake

```
<?php i ?>
```

–označava da se kod izvršava na poslužitelju

- Open Source
- Od verzije 4.0 može biti integriran u Apache Web server
- Može se povezati na većinu baza podataka i koristiti razne servise

Perl

- Autor: Larry Wall, 1987. g.
- Sličan C jeziku
- Na početku kreiran za korištenje na UNIX-u, danas se može instalirati na svim važnijim platformama
- Open Source
- Može se povezati na većinu baza podataka i koristiti razne servise
- Na početku Web boom-a bio je najrašireniji jezik za Web programiranje
- mod_Perl može biti integriran u Apache Web server

Java servlet

- Razvijeno u Sun Microsystems
- Sličan C++ jeziku
- Temelji se na virtualnom stroju (JVM)
- Posebne klase kojima se postiže obrada HTML zahtjeva
- Open Source
- Može se povezati na većinu baza podataka (JDBC) i koristiti razne servise
- Skalabilano rješenje

JSP

- Razvijeno u Sun Microsystems
- Skriptirajuće oznake

```
<% i %>
```

–označava da se kod izvršava na poslužitelju

- `@language`

–određuje se skriptirajući jezik (default Java)

```
–<%@ page language = "java" %>
```

- Potrebno je i moguće konfigurirati poslužitelj da provodi operacije na određeni i/ili zahtjevani način.

Ruby on Rails

- Aplikacijski okvir za razvoj Web aplikacija, napisan u programskom jeziku Ruby
- Autor RoR: David Heinemeier Hansson, 2004. g.
- Autor Ruby-a: Yukihiro "Matz" Matsumoto, 1995. g.
- Inspiriran Perl-om s objektno orijentiranim osobinama koje sliče Smalltalk-u
- Interpreterski jezik a postoje razne implementacije Rubinius, JRuby, IronRuby i sl.
- Može se povezati na većinu baza podataka i koristiti razne servise

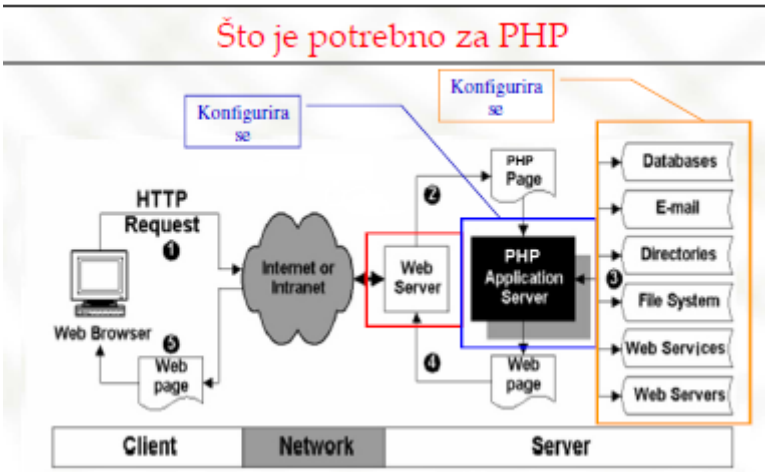
WebDiP_15

PHP je skriptni jezik za programiranje na strani poslužitelja. Za njegovo izvođenje potreban je Web poslužitelj s ugrađenom instalacijom PHP modula ili kao CGI binary.

PHP Hypertext Preprocessor

Što može PHP?

- Preuzmati podatke iz formi
- Generirati dinamičke web stranice
- Slati i primati cookie
- Rad s bazama podataka
- Podržava i druge protokole
- Moguće je korštenje XML
- Generiranje PDF i drugih formata



Povijest PHP-a
Autor Rasmus Lerdorf, 1994

Izlazak iz HTML-a i ulazi u PHP <?php ……. ?>

Imena varijabli se mogu kreirati po sljedećem pravilu:
$[a-zA-Z_\x7f-\xff][a-zA-Z0-9_\x7f-\xff]*

PHP Ne zahtjeva deklariranje tipa podataka. Varijablu možemo deklarirati pmoću naziva vez pridruživanja vrijednosti, koja u tom slučaju nema pridruženi tip i tretira se kao prazna varijabla.

Jednostruki navodnici doslovno prikazuju tekst dok dvostruki provode zamjenu varijabli s njihovim vrijednostima.

Koristi kao i C++ tipove komentara, nije dozvoljeno ugnjezditi višelinijske komentare

Razvoj PHP aplikacija može biti jednsotavnije i brže ako se provodi strukturiranje sadržaja Web stranice s ciljenm stvaranje predložaka stranica.

Provodi se na dva načina: include i require
Razlika je u tome što require generira fatalnu pogrešku kada datoteka ne postoji dok include samo upozorenje.

PHP podržava sljedeće tipove podataka: arrays, floating, integer, object, string.

Nizovi se u PHP ponašaju:

- Kao vektori odnosno indeksirani nizovi
- Kao hash tablice odnosno asocijativni nizovi

Prepoznavaju se po []

Niz možeo kreirati koristeći array() funkciju ili možemo eksplicitno pridružiti vrijednost svakom elementu niza. Možemo koristiti i prazne zagrade pa će element biti automatski dodan na kraj zagrade.

Jednodimenzionalni nizovi:

asort() – sortira niz i održava redoslijed pridruživanja
arsort() – sortira niz u obrnutom redoslijedu i zadržava redoslijed pridruživanja
ksort() – sortira niz po ključu
rsort() – sortira niz u obrnutom redoslijedu (od najvećeg prema najmanjem
sort() – sortira niz od najmanjeg prema najvećem elementu
uasort(), usort(), uksort() - sortiraju nizove po funkcijama koje im se zadaju
print_r() – ispisuje informaciju u varijabli (ne samo za niz) u čitljivom obliku
var_dump() – ispisuje informaciju o varijabli (ne samo za niz)

Višedimenzionalni nizovi

Za svaku novu dimenziju dodamo novi [indeks] ili [ključ].

Možemo miješati numeričke i asocijativne ključeve.

Višedimenzionale nizove ne možemo direktno referencirati unutar stringova.

Indeksirani nizovi

Prvi način:
```
$a[0] = "Varaždin";
$a[1] = "Čakovec";
$a[2] = "Zagreb";
```

Drugi način:
```
$a = array("Varaždin", "Čakovec", "Zagreb");
```

Asocijativni nizovi

Prvi način:
```
$b["color"] = "red";
$b["taste"] = "sweet";
$b["name"] = "apple";
```

Drugi način:
```
$b = array(  "color" => "red",
             "taste" => "sweet",
             "name" => "apple");
```

Funkcija array() može biti ugnježđena kod višedimenzionalnih nizova

Objekti
Da bi instancirali klasu u varijablu koriste se operator new

PHP ne zahtjeva deklariranje tipa podataka, postupak se provodi automarski na bazi vrste operatora tako da se uvijek pretvara u viši tip podatka.
Ako se strogo želi koristiti određeni tip podataka to se čini funkcijom

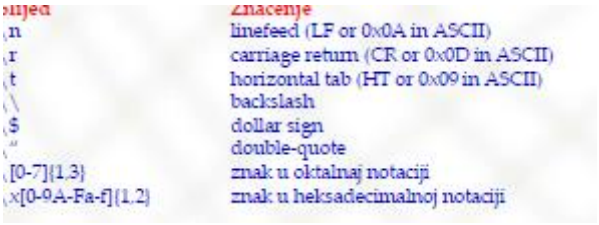
Int settype (string var, string type)

Pretvara varijablu var u tip podataka type,
Funkcija vraća true ako je promjean uspješna u suprotnom vraća false.
Suprotno radi funkcija string gettype (mixed var).
Ako se strogo želi ispitati određeni tip podataka to se čini funkcijom : is_array(), is_double(), is_float(), is_real(), is_long(), is_int(), is_integer(), is_string(), is_object().

Provjera postojanja varijable, njeno brisanje i ima li vrijednost: isset(), unset(), empty()

String

Označavamo ih navodnicima. Ako koristimo obične „“ varijable unutar stringa će biti zamijenjene pravim vrijednostima. Drugi način za označavanje stringova su jednostruki navodni '. string se može tretirati kao indeksirani niz znakova. Za specijalne znakove koristi se \.



...ijed	Značenje
\n	linefeed (LF or 0x0A in ASCII)
\r	carriage return (CR or 0x0D in ASCII)
\t	horizontal tab (HT or 0x09 in ASCII)
\\	backslash
\$	dollar sign
\"	double-quote
\[0-7]{1,3}	znak u oktalnaj notaciji
\x[0-9A-Fa-f]{1,2}	znak u heksadecimalnoj notaciji

String u više redova, oznaka <<<

```
<?php
        $str = <<<EOD
                iza 3 znaka
                kojom označa
                početak i kr
EOD;
?>
```

Unaprijed definirane varijable

_COOKIE	asocijativni niz vrijednosti privremeno spremljenih za korisnika u lokalnom spremištu preglednika (HTTP cookies)
_ENV	asocijativni niz vrijednosti koje se odnose na okolinu u kojoj se izvršava skripta
_FILES	asocijativni niz stavki koje su kreirane prilikom prenošenja datoteke tj. HTTP file upload
_GET	asocijativni niz vrijednosti proslijeđenih skripti preko HTTP GET metode
_POST	asocijativni niz vrijednosti proslijeđenih skripti preko HTTP POST metode
_SERVER	asocijativni niz vrijednosti koje se odnose na HTTP zaglavlja, putanje, lokacije skripte
_SESSION	asocijativni niz vrijednosti koje se odnose na poslužitelju

Vrijednost varijable može i sama postati varijabla

```php
<?php
  $a = "Hello";
  $$a = "world";    // sadržaj varijable a je nova
                    pod nazivom $Hello kojoj se
                    vrijednost "world"
  echo "$a <br>";          // ispisuje: Hello
  echo "$$a <br>";         // ispisuje: $Hello
  echo "${$a} <br>";       // ispisuje: world
  echo "$Hello <br>";      // ispisuje: world
  echo "$a ${$a} <br>";    // ispisuje: Hello world
  echo "$a $Hello";        // ispisuje: Hello world
?>
```

Konstante
Definiramo sa define() funkcijom
Predefinirane su: TRUE, FALSE, __FILE_ (ime skripte), __LINE__ (linija u skripti), PHP_VERSION, E_ERROR, E_PARSE, E_ALL, PHP_OS, E_WARNING, E_NOTICE

Define(„POZDRAV“, „Hello world“);
Echo POZDRAV;

U PHP-u postoje sljedeće vrste operatora: aritmetički, pridruživanja, uspoređivanja, na razini bitova, za kontrolu grešaka, izvršavanja, za povećavanje i smanjivanje, logički za rad sa string.

== jednako
=== identično

PHP podržava samo jedan operator tipa, a to je instanceof – instanca klase
Dva string operatora **.** operator spajanja **.=** pridruživanja stringova

Foreach za indeksirane nizove **foreach ($arr as $i)**
Foreach za asocijativne nizove **foreach($arr as $k => $vr) echo „ $k = $vr “**

Funkcije možemo definirati na sljedeći način: bez argumenata, s argumentima(prijenos vrijednosti bez promjene) s argumentima(s promjenom), s pretpostavljenim vrijednostima argumenata.
Func_num_args() broj argumenata
Func_get_arg() preuzimanje argumenta
Funkcije mogu vraćati vrijednost, nit, referencu

Dinamičko definiranje naziva funkcije koja se poziva dobije se tako da varijabla sadrži ime funkcije pa tu varijablu možemo koristiti da pozovemo funkciju.

4 vrste vidljivosti varijabli:

- Ugrađene superglobalne varijable koje su vidljive bilo gdje u skripti
- Glabalne varijable definirane su u skripti izvan funkcija i vidljive su kroz cijelu skriptu ali ne u funkcijama
- Varijable definirane u funkcijama su lokalne varijable funkcije

- Ako je u funkciji potrebno pristupiti globalnoj varijali tada se ona definira u funkciji kao globalna koristi se global $var
- Lokalne varijable mogu zadržavati vrijednost kroz više poziva funkcije ako su definirane kao statičke lokalne varijable koristi se static $var

Superglobalne varijable: $GLOBALS, $_SERVER, $_GET, $_POST, $_FILES, $_COOKIE, $_SESSION, $_REQUEST, $_ENV

Objektna orijentacija u PHP-u
veza je definirana ulogama
klasa -klasa
metoda -funckija
objekt -referenca prema nekom podatku unutar klase

klase uključuju se u skriptu
Konstruktor mora imati isti naziv kao i klasa 5.3.2
Može imati naziv __construct(…)
Destruktor __destruct(void)

Rad s datumom i vremenom
Checkdate – validacija gregorijanskog datuma
Date – formatira lokano vrijeme/datum
Getdate – daje informaciju u datumu vremenu
Gettimeofday – daje važeće vrijeme
Gmdate – formatiram GM lokano vrijeme
Gmmktime -postavlja UNIX timestamp za GMT datum
Gmstrftime – formatira GMT lokano vrijeme prema lokalnim postavkama
Localtime – daje lokalno vrijeme
Microtime – daje UNIX timestamp s kirosekundama
Mktime – daje UNIX timestamp za datum
Strftimee – formatira vrijeme prema lokanim postavkama
Strtotime – parsira engleski tekstualni opis datuma u UNIX timestamp
Time – daje važeći UNIX timestamp

Kodovi za date funkciju

```
d,j    - dan u mjesecu (01-31, 1-31)
D,l    - dan u tjednu (Mon-Sun, Monday-Sunday)
F,M    - mjesec u godini (January-December, Jan-Dec)
g,h    - sat u danu u 12 satnom formatu (1-12, 01-12)
G,H    - sat u danu u 24 satnom formatu (0-23, 00-23)
m,n    - mjesec u godini (01-12, 1-12)
s      - sekunde u minuti (00-59)
t      - ukupan broj dana u mjesecu
U      - ukupan broj sekundi od 01.01.1970.
w      - dan u tjednu (0-6)
y      - godina u 2 brojnom formatu
Y      - godina u 4 brojnom formatu
z      - dan u godini (0-365)
```

CGI varijable okoline

DOCUMENT_ROOT	korijenski direktorij na poslužitelju
HTTP_COOKIE	kolačići koje je postavio korisnik
HTTP_HOST	naziv računala
HTTP_REFERER	URL stranice koja je pozvala skriptu
HTTP_USER_AGENT	tip korisnikovog preglednika
HTTPS	"on" ako je skripta zvana preko sigurnog poslužitelja
PATH	putanja na kojoj je izvršen poslužitelj
QUERY_STRING	lista parametara (kod GET poziva)
REMOTE_ADDR	IP adres korisnika
REMOTE_HOST	naziv računala korisnika (ako se može dobiti) ili IP adresa
REMOTE_PORT	port preko kojeg je korisnik povezan na poslužitelj
REMOTE_USER	korisničko ime (ako je stranica zaštićena putem poslužitelja
REQUEST_METHOD	GET ili POST
REQUEST_URI	putanja do zahtjevanog dokumenta ili skripte (relativno do korijenskog direktorija dokumenta)
SCRIPT_FILENAME	puni naziv važećeg sktipte
SCRIPT_NAME	putanja do važeće skripte (relativno do korijenskog direktorija dokumenta)
SERVER_ADMIN	email adres webmastera
SERVER_NAME	puno ime poslužitelja
SERVER_PORT	port kojeg prisluškuje poslužitelj
SERVER_SOFTWARE	software koji se koristi na poslužitelju (npr Apache/2.0.52)

WWW poslužitelj automatski postavlja asocijativni niz $_SERVER za svaki CGI – nova verzija 4.1.0.

Platforma određuje koje su varijable okoline.

Za prijenos podataka iz formulara također se koristi varijabla okoline **QUERY_STRING** u koju su upisani podaci svih kontrola iz formulara. Skripta sama obrađuje tu varijablu ali sada je jednostavnije jer se znaformat po kojem su upisani podaci. NIJE sigurna metoda slanja podataka jer se podaci šalju kao dio URL-a! u PHP postoji asocijatini niz $_GET koji sadrži podatke.
WWW poslužitelj provodi određeno pretvaranje nekih znakova u adresnoj liniji:

- Alfanumeričke prenosi bez promjene
- Praznine pretvra u +
- Ostale znakove pretvra u oblik %HH gdje je HH ASCII kod znaka u bazi 16.

Za prijenos podataka iz formulara koristi se standardni ulaz. Skripta čita podatke sa standardnog ulaza i puni varijable podacima iz formulara. Sigurnija metoda slanja podataka od GET.
Prijenos podataka iz formulara različit je za pojedine vrste kontrole kao što su checkbox, radio button i sl kada se koriste u grupi.

Cookie
Programiranje na strani klijenta i na strani poslužitelja moraju imati više dodirnih točaka-sučelja putem koojih se ostvaruje njihova komunikacija. Prije svea to su standardni ulaz i izlaz na strani poslužitelja kojima se primaju podaci od klijenta i kojima se šalju podaci klijentu. Druga važna točka su varijale okoline koje služe strani poslužitelju za prikupljanje podataka o klijentu i njegovom radnom okruženju. Personalizacija se obavlja na sttrani klijenta, a identfikator korisnika se zapisuje kao vrijednost određenog cookie-a.

Upisivanje:
Int setcookie (string name [,string value [, int expire [, string path [, string domain [, int secure]]]]])

WebDiP_16

Rad s datotekama

Aplikacijska logika najčešće traži da WWW poslužitelj pohranjuje i čita podatke iz datoteke. Danas se sve više koristi zapis u XML formatu.

Postoje glavne funkcije: fopen, fclose, fread, file, fwrite, flock, fseek (pozicioniranje)

Datoteka može biti otvorena za jednu ili više vrsta operacija:

$fp = fopen(„podac.txt“, „r“) čitanje

$fp = fopen („podaci.txt“, „w+“); upisivanje, prepisivanje

$fp = fopen („podaci.txt“, „a+“) dodavanje na kraj

Upisivanje u datoteku : fwrite ($fp, $ime);

Čitanje datoteke, određeni broj znakova : fread ($fp, filesize ($fn));

Čitanje datoteke cijela u niz file ($fn)

Zaključavanje datoteke:

Simultao korištenje datoteke od strane 2 ili više skripata koji može dovesti do nekonzistentnosti podataka datoteke ukoliko se ne ograniči pristup samo jednoj skripti u datom vremenu. To znači da ostale skripte trebaju čekati dok prva ne završi svoj rad s datotekom. Zaključavanje datoteke bit će pouzdano rješenje samo u slučaju kada sve skripte traže zaključavanje datoteke. Ukoliko jedna skripta ne koristi taj princip nego direktno pristupa podacima, cijeli koncept gubi smisao jer u slučaju paralelnog rada neće postojati mehanizam međusobnog isključivanja. Ovisi o OS-u.

Flock (broj_veze_datoteke, operacija);

Uglavnom se koriste dvije operacije zaključavanja:

Ekskluzivno zaključavanje (LOCK_EX)

Otključavanje ranijeg zaključavanja (LOCK_UN)

Pozicioniranje u datoteci

Kod otvaranja datoteke važeća pozicija datoteke je njen početak. Moguće je realizirati datoteku sa zapisima fiskne duljine kod koje je moguće ažuriranje podataka. Za to je potrebno postaviti važeću poziciju u datoteci na početak zapisa koji se ažurira i nakon toga provesti zapis podataka. Drugi slučaj kod kojeg je potrebno primijeniti pozicioniranje u datoteci odnosi se na zaključavanje datoteke. Kod poziva se zaključavanja datoteke može se čekati dok druga skripta ne završi svoj posao, a ona može promijeniti neke podatke u datoteci. Ako se dodaje na kraj datoteke potrebno je postaviti važeću poziciju u datoteci na njen novi kraj. Ovisi o OS-u.

Fseek(broj_veze_datoteke, omak, [. Odakle]);

Pomak je broj bajtova, a odakle može biti:

SEEK_SET početak datoteke, SEEK_CUR važeća pozicija u datoteci, SEEK_END kraj dat.

obrada pogrešaka u radu s datotekama

u PHP u slučaju pojavljivanja pogreške u nekoj liniji se nastavlja sa sljedećom kao da nije došlo do pogreške. Potrebno je uključiti obradu pogrešaka kod većine U/I operacija s datotekama. Za obradu pogreške U/I funkcija se povezuje operatorom **or** s funkcijom za obradu pogreške.

Slanje e-mail poruke

Predložaka za funkciju mail:

Predložak za funkciju mail:

```
bool mail (string to, string subject, string message
        [, string additional_headers
        [, string additional_parameters]])
```

Da bi se mogla slati email poruka putem funkcije mail(...) potrebno je podesiti u konfiguracijskoj datoteci (php.ini) postavku za SMTP poslužitelja i PORT.
To radi administrator sustava! Kod XAMPP-a postoji Mercury email poslužitelj pa se nakon kreiranja korisnika može osnovnom konfiguracijom PHP-a slati email poruka na Marcury.

```
$mail_to = $_POST["email"];
$mail_from = "From: WebDiP@foi.hr";
$mail_subject = $_POST["subjekt"];
$mail_body = $_POST["tekst"];

if(mail($mail_to, $mail_subject, $mail_body, $mail_from))
{
        echo("Poslana poruka za: '$mail_to'!");
} else {
        echo("Problem kod poruke za: '$mail_to'!");
}
?>
```

Geneiriranje HTML stranica

Kada se podaci nalaze u datoteci poželjno je obaviti čišćenje od nepotrebnih znakova

To rade funkcije **$varijabla – htmlspecialchars(trim($varijabla));**

```
<?php
$fn = "podaci/ADRESAR.TXT";
$fcontents = file ($fn);
while (list ($line_num, $line) = each ($fcontents)) {
        $value = htmlspecialchars(trim($line));
        print "<tr><td>$value</td><td><a href=\"mailto:$value\">Šalji poruku za:
$value</a></td></tr>\n";
}
?>
</table><br>
```

datoteka je strukturirana tako da svaki zapis sadrži dva podatka emial adresu prezime i ime odvojena znakom #.

Kod pripreme podataka za ispis potrebno je odvojiti dijelove zapisa u zasebne podatke.

Razina izvještaja o pogreškama

Int error_reporting ([int level])

Funkcijom **error_reporting** može se postaviti ili utvrditi važeća razina izvještavanja o pogreškama u skriptama.

Rukovanje s pogrečkama

mixed set_error-handler (callback error_handler [, int error_types])

funkcijom **set_error_handler** može se preusmjeriti izvršavanje skripte na definiranu funkciju za rukovanje s pogreškama u slučaju određene razine pogreške u skriptama.

Pretpostavljena razina izvještavanja postavljena je varijablom **error_reporting**.

Funkcija za rukovanje s pogreškama ima sljedeći format:

```
handler ( int errno, string errstr [, string errfile [,
int errline [, array errcontext]]] )
```

errno – razina pogreška koja se desila
errstr – poruka pogreške
errfile – naziv datoteke u kojoj se desila pogreška
errline – broj linije u kojoj se desila pogreška
errcontext – vektor varijabli koje postoje u pogledu dešavanja pogreške.
NE smiju se mijenjati!

Operatorom **@** može se ignorirati pogreška u izrazu koji slijedi operator.

Generiranje pogrešaka

Bool trigger_error (string error_msg [, int error_type])

Funkcijom **trigger_error** može se generirati pogreška željenje poruke i razine te preusmjeriti izvršavanje skripte na postavljenog rukovatelja pogrešaka.

```
<?php
include_once 'postavke.php';

set_error_handler('obradaPogresaka');

function obradaPogresaka($errno, $errstr, $errfile, $errline,
$errcontext) {
  echo "Desila se pogreška kod izvršavanja!<br>";
  echo "Datoteka: $errfile<br>";
  echo "Linija: $errline<br>";
  echo "Opis: $errstr<br>";
  echo "Kod: $errno<br>";
  exit;
}
?>
```

evidencija pogrešaka

bool error_log (string message [, int message_type [, string destination [, string extra_headers]]]

funkcijom **error_log** može se evidentirati pogreška u dnevniku web poslužitelja u dnevniku web poslužitelja, poslati na TCP port ili u vlastitu datoteku.

Vrste:
0 – poruka se šalje PHP sistemskom dnevniku
1 – poruka se šalje kao email na adresu
2 - poruka se šalje na vezu PHP debugera
3 – poruka se dodaje na kraj datoteke

RAD S BAZAMA PODATAKA

Postoje tri načina korištenja BP MySQL

- Mysql označeno kao deprecated pa treba izbjegavati
- Mysqli poboljšana verzija objektno orijentirana

- PDO MySQL apstraktni sloj za BP da se može mijenjati BP uz minimalnu promjenu programskog koda

Poboljšana verija donosi: OO sučelje, podrška za pripremljene instrukcije, podrška za višestruke instrukcije, podrška za transakcije, dopunjuje mogućnosti debug-a, podrška za ugrađeni poslužitelj
Može se koristiti PO i OO način.

PO: prvi korak veza prema poslužitelju, drugi korak postavljanje skupa znakova
OO: prvo veza prema poslužitelju pomoću objekta klase mysqli.
Kod OO korštenja koristi se objekt klase mysqli te se na njemu izvršavaju funkcije.

SQL skup naredbi: DDL, DML, DCL, TCL

Imenički servisi – LDAP
LDAP definira kako klijenti trebaju pristupiti do podataka na poslužitelju. On ne određuje kako podaci trebaju biti spremljeni na poslužitelju.
Organiziran je u stablo pod nazivom DIT **Director Information Tree.** Svaki list u DIT-u naziva se element (entry). Prvi element u DIT-u naziva se korijenski element. Element se sastoji od DN **Distingunished Name** i bilo kojeg broja parova atribut/vrijednost. DN je naziv elementa i mora biti jednoznačan. LDAP poslužitelj podržavaju uputnice na druge LDAP direktorije tako da jedan LDAP poslužitelj može pretražiti milijun elemenata temeljem jednog zahtjva korisnika.

Imenički servisi kao što je LDAP sve više se koriste zbog centralizacije autentifikacijskih podataka, a time i uklanjanja redundancije podataka za lozinku. Tada se koriste kao dodatak bazi podataka koja u tablici sadrži skup korisnika kojima se dozvoljava rad, a može se i u LDAP uvesti atribut koji sadrži domene za koje korisnik ima pristup.
Prednosti LDAP je što omogućava kreiranje vlastite sheme a ujedno je podržana autentikacija i autorizacija - na razini atributa. LDAP ima ugrađen vrlo snažan sigurnosni model korištenja ACL za zaštitu podataka na poslužitelju i podržava SSL – Secure Socket Layer protokol.

LDAP imenički servis omogućava pretraživanje po raznim kriterijima te izlistavanje izabranih atributa za zadovoljene element. Može se ispitivati vrijednost izabranog atributa.

WebDiP_17

Autentikacija

Korištenje svih ili samo određenih dijelova aplikacije treba biti dozvoljeno samo ograničenom i poznatom skupu korisnika. Svaki korisnik identificira se svojim korisničkim imenom koje je zaštićeno pripadajućom lozinkom.

Web autentikacija može biti provedena : internom autentikacijom Web poslužitelja – Apache s datotekom, vlastitom autentikacijom s formularom, bazom podataka i pohranom privemenih podataka u session.

Interna autentikacija Apache poslužitelja s datotekom

Konfiguracijom Web poslužitelja može se dopustiti autentikacija od određenog dijela stabla direktorija. Cijeli sustav autentikacije temelji se na datoteci **.htaccess**. U njoj se određuju konfiguracija autentikacije. Da bi se mogla primijeniti autentikacija metodom .htaccess potrebno je u datoteci konfiguracijom httpd.conf postaviti **AllowOverride AuthConfig**

Da bi autentikacija bila od koristi potrebno je kreirati datoteku korisnika i upisivati korisnike i njihove pripadajuće lozinke. Koristi se komanda htpasswd. Kreiranje datoteke provodi se opcijom -c i koristi se samo kod prvog korisnika.

Autentikacija Apache poslužiteljem

Kada se zatraži prikaz bilo kojeg dokumenta s tog direktorija odnosno bilo kojeg direktorija koji ga slijedi tada Web poslužitelj pokrene postupak autentikacije korisnika i prikazuje se maska za unos podataka. Podaci vrijede sve dok se ne zatvori preglednik u kojem je obavljena autentikacija.

Nakon uspješne autentikacije Web poslužitelj upisuje podatke o korisniku u varijablama okoline

Korisnik: **$_SERVER[„PHP_AUTH_USER“]**

Dodavanje korisnika putem Web aplikacije

Zbog specifićnosti načina dodavanja korisnika tu se može raditi o potencijalnom sigurnostnom propustu jer se izvršava poziv funkcije sustava **shell_exec** iz PHP skripte za izvršavanje komande httpasswd

Autentikacija korištenjem sesije

Session je privremeno spremište podataka koje se nalazi na poslužitelju i vezano je uz rad pojedinačnog korisnika. Spremljeni podaci u sesiji dostupni su kroz više različitih stranica jednog korisnika aplikacije koji koristi određeni preglednik.

Sesija se kreira pozivom funkcije session_start() koja se mora izvrštiti prije bilo kojeg HTML sadržaja u skripti. Podaci se upisuju i preuzimaju putem varijable **$_SESSION**, koja je asocijativni niz. U sesiju se mogu spremati jednostavni tipovi podataka kao i složeni. Brisanje podataka sesije provodi se session_unset(). Brisanje sesije session_destroy().

Gašenjem i ponovnim pokretanjem poslužitelja brišu se sve sesije.

Autorizacija

Provođenje pojedinih operacija u svim ili samo određenim dijelovim aplikacije treba biti dozvoljeno samo ograničenom i poznatom skupu autentičnih korisnika.

Autorizacija se treba provoditi za svaki zahtjev korisnika čime se usporava rad poslužitelja. Uobičajno je da se podaci autorizacije upisuju u bazu podataka. Zbog toga je praktično podijeliti web aplikaciju na javni i privatni dio, pri čemu se samo privatni dio štiti autorizacijskim mehanizmom. Autorizacija se može korisiti i za filtriranje podataka/akcija u nekoj skripti.

Složenija autorizacija može se provoditi na sličan način kao što je ACL Access Control Lists. To znači da se svakom korisniku grupi određuju koje operacije smije obavljati, odnosno kojim stranicama smije pristupiti.

Metode autorizacije u odnosu na skriptu mogu biti statičkog i dinamičkog tipa.

Kod statičkog tipa sama skripta sadrži podatke o zahtjevnom elementu autorizacije te ih provjerava u odnosu na korisnikove atribute koji su obično upisani u tablicama.

Kod dinamičkog tipa skripta provjerava dozvole izvođenja na temelju svog ID ili naziva u odnosu na podatke iz tablica.

Metode autorizacije:

- Autentikacijska temelji se na vrsti korisnika: anonimni i prijavljeni. Skripta provjerava zapis prijavljivanja u sesiji.
- Na temelju uloge: korisnicima se dodjeljuju uloga. Skripta provjerava zahtjevanu ulogu za izvršavanje skripte u odnosu na ulogu korisnika. (KORISNIK-ULOGA)
- Na temelju uloga: KORISNIK_ULOGA, KORISNIK, ULOGA
- Na temelju ovlaštenja KORISNIK_SKRIPTA, KORISNIK, dodjeljuje izvršavanje
- Na temelju akcija: KORISNIK_AKCIJA, KORISNIK, AKCIJA, dodjeljue se akcija
- Mješovite dodjeljuju se izvršavanja, akcije

Dnevnik rada – log

Postoje dva načina praćenja rada korisnika: Apache log, vlastita evidencija
Apache log prema potrebama postavlja se u datoteci httpd.conf format zapisa podataka
Uobičajno je da se podaci za dnevnik rada upisuju u bazu podataka, čime se otvara mogućnost jednostavnijih upita, odnosno pretraživanja.
Obično se upisuju: vrijeme, url, adresa računala, korisnik, status.

WebDiP_18

File upload

Proces preuzimanja datoteke treba biti pod posebnom kontrolom zbog mogućih problema koji mogu proizaći iz njega. Prije svega misli se na zatrpavanje poslužitelja raznim smećem. Zatim narušavanje sigurnosti poslužitelja kada se napada na sustav kanalizira kroz preuzetu datoteku.

PHP oogućava da se postave dvije granice za veličinu datoteke koja se preuzima : na razini sustava i na razini aplikacije (MAX_FILE_SIZE u obrascu)

Funkcije datotečnog sustava

U radu aplikacija može se javiti potreba za preuzimanjem datoteke s izabranog direktorija na poslužitelju ili dinamički prikaz slikovnih datoteka kao galerija s izabranog direktorija na poslužitelju.

Funkcije: is_file, is_dir, is_executable, is_writable, is_readable, opendir, readdir, fileperms, mkdir

Virutalno vrijeme sustava

Vrijeme poslužitelja ne može se mijenjati po potrebi, a kod testiranja nema vremena za čekanje da prođe zadano trajanje. Jedino rješenje je uvođenje virtualnog vremena sustava koje se temelji na zadanom pomaku vremena. Koristi se u svakoj akciji u kojoj je potrebno vrijeme poslužitelja.

Postupak pripreme i korištenja virtualnog vremena sastoji se od sljedećih koraka: unos pomaka vremena, preuzimanje zadanog pomaka vremena te se upis u privatni dio aplikacije, postavljanje virtualnog vremena sustava.

Sigurnost web aplikacija

Sigurnosne točke: web poslužitelj, aplikacijski poslužitelj, poslužitelj baze podataka, ostali, komunikacijski kanal.

Open Web Application Security Project je svjetska neprofitna organizacija koja je usmjerena na poboljšanje sigurnosti programske podrške.

Misija OWASP je da učini programsk podršku vidljivom tako da pojedinici i organzacije u svijetu mogu donositi odluke na bazi informacija o stvarnim sigurnosnim rizicima programske podrške

OWASP ne odobrava ili preporučuje komercijalne proizvode ili usluge ostavljajući da zajednica bude neovisna o dobavljaču.

Metodologija procjene rizika: 1. identificiranje rizika, 2. faktori za procjenu vjerojatnosti, 3. faktori za procjenu napada, 4. utvrđivanje ozbiljnosti rizika, 5. odlučivanje što popraviti, 6. prilagođavanje svog modela procjene rizika

WebDiP_19

Mjerenje opterećenja Web mjesta

Simulacijom broja korisnika, tempa njihovog rada i izbora dokumenata koje će učitavati

Za složene situacije može se formirati grupa raspodjeljenih računala koja će provoditi aktivnosti pod zajedničkom kontrolom raspodjeljenog sustava

Osnove testiranja opterećenja Web sustava

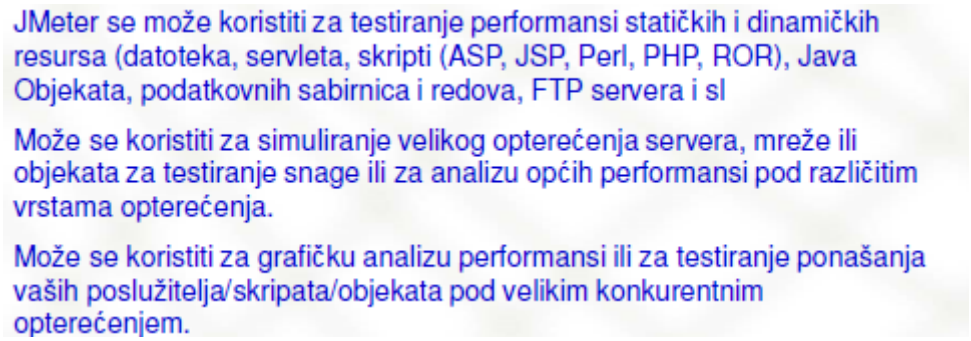
Testiranje performansi kroz aktivnosti:

- Identificiranje testne okoline: utvrđuje se fizička okolina za testiranje i produkcijska okolina te alati i tesursi koji su na raspolaganju timu za testiranje
- Identificiranje kriterija prihvatljivosti performasni: utvrđuje se ciljne vrijednosti i ograničenja za vrijeme odgovora, propustnost i korištenje resursa. Vrijeme odgovora odnosi se na korisničku domenu, propusnost spada u poslovnu domenu, a korištenje resursa je sistemska domena.
- Planiranje i dizajn testova: utvrđuju se ključni scenariji, određuju se varijabilnosti za predstavljanje korisnika i kako će se one simulirati, određuju se testni podaci i definiraju mjerenja koja će se prikupljati.
- Konfiguriranje testne okoline: priprema testne okoline, alata i potrebnih resursa za izvršavanje svake strategije kako pojedine osobine i komponente sustava budu na raspolaganju
- Implementacija dizajna testa: definiranje testova u skladu s dizajnom testa
- Izvršavanje testa: provođenje i nadziranje testova. Slijedi validacija testova, testiranje podataka i kolekcija rezultata.
- Analiza rezultata, izrada izvještaja, ponavljanje testa: konsolidacija podataka iz testova, njihovo analiziranje kroz više razina i grupa.

Kategorije testiranja performansi:

- **Testiranje performansi** : utvrđuje se ili provjerava brzina, skalabilnost, stabilnost kao osobine sustava ili aplikacije koja se testira. Performanse koje se razmatraju su vrijeme odziva, propustnost, razina korištenja resursa koji se odnose na ciljeve performansi koji su postavljeni za sustav ili aplikaciju.
- **Testiranje opterećenja (Load test)** utvrđuju se ili provjeravju osobine performansi sustava ili aplikacije koja se testira, kada je podređena radnim opterećenjem korištenja i koje se očekuju za vrijeme normalnog izvršavanja poslova
- **Testiranje naprezanja (Stress test)** utvrđuju se ili provjeravaju osobine performansi sustava ili aplikacije koja se testira kada su uvjeti ispod očekivanih za normalan rad sustava, pod kojim uvjetima će aplikacija prestati s radom, kako će prestatai i koji se indikatori mogu nadzirati kako bi se upozorilo na nadolezeći prekid.
- **Test kapaciteta (Capacity test)** utvrđuje se koliko korisnika ili transakcija može sustav opsluživati da još uvijek ispunjava ciljeve performansi.

Okruženje rada Web sustava: server-čeoni dio, poslužitelj BP-pozadinski, email i LDAP pomoćni



Ideficiranje kriterija prihvatljivosti performansi

Vrijeme odziva koje se odvija u širokopojansom kao preporučenom načinu spajanja na Internet, tada se može reći da je prihvatljivo u granicama od nekoliko sekundi pri čemu se više od 10 sekundi smatra neprihvatljivo.

Ni jedan poslužitelj ne bi smio preći granicu od 75% korištenja procesora podataka tijekom vršnog opterećenja.

Rad sustava za upravljanje bazom podataka tijekom vršnog opterećenja mora biti takav da korištenje procesora ne pređe 50-75% kako bi se osiguralo dovoljno vremena procesora za ostale konkurentne poslove.

Planiranje i dizajn testa

Razrada scenarija testiranja. Izvor podataka može biti dnevnik rada postojećeg sustava iz kojeg se izvlače podaci o slijedu rada korisnika, vremenima između poziva pojedinih web stranica.

Korištenjem sustava putem Jmeter HTTP Proxy modula mogu se prikupiti podaci o slijedu rada korisnika, vremenoima između poziva pojednih web stranica.

Razlika u radu korisnika sustava: anonimni korisnik, prijavljeni

Kod svake aktivnosti koja se odnosi na rad Web aplikacije bilo je važno utvrditi status njenog završetka kako bi se mogla voditi evidencija uspješnosti rada.

Optimizacija Web sustava: dodavanje poslužitelja za BP, podešavanje parametara poslužitelja BP, reorganizacija sheme BP, redundacija podataka kako bi se smanjilo ili ukinulo povezivanje tablica prilikom SQL upita..

WebDiP_20

Arhitektura velikih Web/Internet sustava

Postavljanje arhitekture može se realizirati kroz uzorke.

Uzorak je rješenje za problem koji se pojavljuje u specifičnom kontekstu.

Nefunckijske karakteristike: raspoloživost (koliko je sustav korisnicima raspoloživ), performanse (vrijeme odgovora na zahtjev), skalabilnost (prilagođavanje rada sustava kod povećanja broja korisnika), sigurnost(stavljane dovoljne zaštite), upravljivost(mogućnost nadgledanja i

promjene ponašanja sustava u radu), održavanost(jednostavno uklanjanje problema ili nadogradnja dijelova sustava), fleksibilnost(mogućnost stvaranja nove verzije sustava), protabilnost(kako jednostavno sustav može migrirati na nove okoline)

Performanse sustava: performanse, raspoloživost, kratkoročna skalabilnost

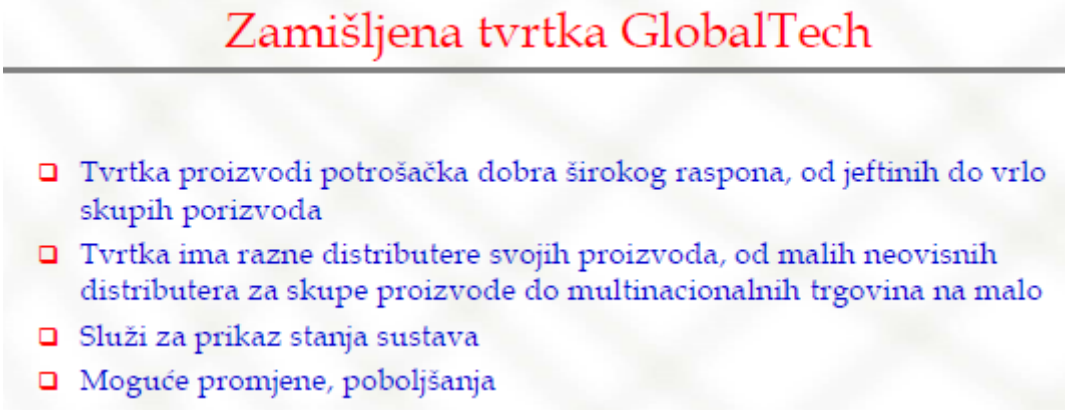
Kontrola sustava: sigurnost, upravljivost, održavanost

Evolucija sustava: fleksibilnost, portabilnost, dugoročna skalabilnost

Klasifikacija uzoraka za Internet bazirane sustave velike snage: temeljni uzorci, uzorci performansi sustava, uzorci kontrole sustava, uzorci evolucije sustava

Arhitektonski pogledi:

- Logički pogled opisuje komponente i softverske elemente sustava. On sastavlja softverske elemente sustava u logičke jedinice i opisuje odnose između tih jedinica.
- Infrastrukturni pogled pokazuje sklopovske uređaje koje koristi sustav i kako su ono povezani.
- Instalacijski pogled pokazuje kako su logičke komponente pridružene različitim sklopovskim elementima sustava.

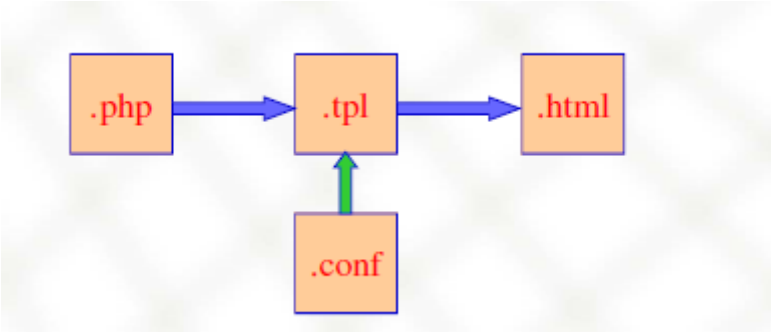


WebDiP_21

SMARTY

Sustav predložaka za PHP, služi za odvajanje aplikacijske logike i sadržaja od njegove prezentacije

Specifičnost je kompiliranje predloška tako da se iz njega kreira PHP skripta




```
{naziv_funkcije arg1="vrijednost1" arg2="vrijednos2"...}
npr.
{include file="header.tpl"}
```

Varijable z PHP-a `{$naziv_varijable}`

Asocijativni nizovi `{$naziv_varijable.kljuc}`

Indeksirani nizovi `{$naziv_varijable[indeks]}`

Objekti `{$naziv_varijable->atribut}`

Varijable iz konfiguracijske datoteke `{#naziv_varijable#}`

Modifikator varijabli mogu se primijeniti na varijablama, funkcijama ili stringovima. Počinu znakom **|**, a njegovi parametri se odvajaju znakom **:**

Ugrađene funkcije: config_load, foreach, section, if, elseif, else, literali, php,

Dodatne funkcije: assign, counter, html_options

WebDiP_22

XML dobro formiran dokument prema XML specifikaciji, validan dokument – ispravno formirani zadovoljava zahtjeve i specifkiacije iz DTD ili XML scheme

XML dokumnet: prolog dokumenta, elementi, atributi, entiteti, notacije, CDATA blokovi, instrukcije procesiranja, komentari

Korištenje XML dokumenta: obrada lokalne datoteke-dokumenta ili primljenih podataka-dokumenta od poslužitelja: SAX ili DOM

SAX: obrada dokumenta temelji se na događajima, postoje funkcije za rad s XML dokumentom

DOM: obrada dokmenta temelji se na kreiranju hijerarhijskog stabla koje predstavlja XML dokument, primjenjuje se objektna orijentacija tako da postoje klase sa svojim atributima i metodama za rad s XML dokumentom

- XSL – eXtensive Stylesheet Language - vrlo snažan jezik stilova koji je posebno dizajniran za XML dokumente
- XSL stilske upute su pravilno formirani XML dokumenti
- Postoje 2 različite vrste XML aplikacija:
  - XSL Transformacije (XSLT) – predstavlja opis transformacije ulaznog XML dokumenta iz jednog formata u izlazni XML dokument drugog formata
  - XSL Formatting Objects (XSL-FO) – predstavlja opis kako će stranice izgledati kada budu prezentirane čitatelju
- postoji cijeli XSL jezik (varijable, petlje, brojači, uvjeti,...)

- RSS - Really Simple Syndication
- Temelji se na XML 1.0
- Ideja RSS temelji se na bržem pristupu do podataka koji se često mijenjaju, a obično se objavljuju putem Web aplikacija (CMS i sl)
- Posebno kada se radi o permanentnom objavljivanju ažurnih podataka (vijesti, burzovni podaci i sl).
- Koriste se posebni alati za čitanje RSS-a (Sage kod Forefoxa)

WebDiP_23

Generiranje izlaznih datoteka u raznim formatima kao što je PDF postaje sve poželjnije zbog jednostavnijeg ispisa.