SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

OTVORENO RAČUNARSTVO

Vježbe

Autori:
Branko Mihaljević, Marin Orlić,
Mario Žagar, Igor Čavrak, Ivana Bosnić

Zagreb, 2011.

Naslov: Otvoreno računarstvo – Vježbe

Izdanje: 1. izdanje

Autori: dr.sc. Branko Mihaljević
dr.sc. Marin Orlić
prof.dr.sc. Mario Žagar
doc.dr.sc. Igor Čavrak
Ivana Bosnić, dipl.ing.

Recenzenti: prof.dr.sc. Željko Hocenski
prof.dr.sc. Danko Basch

Nakladnik: Sveučilište u Zagrebu Fakultet elektrotehnike i računarstva

Mjesto izdanja: Zagreb

Godina izdanja: 2011.

Odobrenje: Povjerenstva za znanstveno-nastavnu literaturu Sveučilišta u Zagrebu br. xx-x-xx/x-2011 od xx. prosinca 2011. – u procesu dobivanja

ISBN: xxx-xxxxx-x-x – u procesu dobivanja

CIP - Katalogizacija u publikaciji
Nacionalna i sveučilišna knjižnica – Zagreb

UDK xxx.xx(xxx.xx) – u procesu dobivanja

MIHALJEVIĆ, Branko i ostali
Otvoreno računarstvo – Vježbe
Udžbenici Sveučilišta u Zagrebu = Manualia Universitatis studiorum Zagrabiensis

ISBN: xxx-xxxxx-x-x – u procesu dobivanja

xxxxxxxxxx – u procesu dobivanja

SADRŽAJ

Predgovor

Pojam otvorenog računarstvo (engl. *open computing*) temelji se na principu i konceptu otvorenosti, gledano s tehnološke računarske strane na razini zapisa i pohrane informacija i podataka, na razini programske podrške, aplikacija i programa, na razini operacijskih sustava i platformi, te na kraju i na razini samih informacijskih sustava. No, ovaj tehnički pogled na otvorenost treba se svakako nadopuniti i drugim oblicima otvorenosti koji nisu izravno vezane uz tehnologiju, ali opet na kraju postaju i njen sastavni dio, a uključuju otvorenost razmišljanja, komunikacije, povezivanja, suradnje, pristupa i svih drugih oblika i aspekata ljudskog djelovanja. Iako ova dva oblika otvorenosti dolaze iz različitih znanstvenih područja, prvi iz područja tehničkih znanosti, a drugi iz društvenih, ili čak i humanističkih znanosti, oni su u svakodnevnom modernom životu ispunjenom sveprisutnim računarskim sustavima, uređajima i procesima isprepliću. Rezultat tih aktivnosti su informacijski sustavi, računala i sklopovlje, programska podrška, te načini povezivanja, komunikacije i interakcije današnjice, koji su u službi takvog modernog načina života. A otvoreno računarstvo p(r)oučava baš pogled na svu tu silnu tehničku snagu i svijet današnjice s aspekta otvorenosti.

Na preddiplomskom studiju Računarstva Fakulteta elektrotehnike i računarstva Sveučilišta u Zagrebu već se dugi niz godina predaje kolegij **Otvoreno računarstvo** koji ima cilj upoznati buduće inženjere i magistre računarstva, a djelomično i elektrotehnike, s konceptima otvorenih sustava i tehnologija na kojima se zasnivaju, te njihovom važnosti u računalnom svijetu. Koncept otvorenosti u računarstvu stasao je početkom sedamdesetih godina prošlog stoljeća s razvojem operacijskog sustava Unix, kada se i javljaju prva pitanja o otvaranju izvornog koda i njegovom dijeljenju, te prenosivosti. Kasnije se ti principi i dalje vežu uz Unix, ali se ideja širenja slobodne programske podrške, stvaranje otvorenih norma, pojava niza licencija otvorenog koda i općenito kultura otvorenosti šire u različitim pravcima. Pojam otvorenih sustava s vremenom mijenja svoje značenje i danas se najviše naglašava važnost suradnje, integracije i povezivanja različitih otvorenih sustava u funkcionalne cjeline, ali neovisno o tome potreba za otvorenim sustavima se ne mijenja.

Kolegij Otvoreno računarstvo stoga daje pregled otvorenosti u sustavima, sklopovlju i programskoj podršci s naglaskom na norme, njihovu svrhu i uporabu u svijetu raspodijeljenih informatičkih sustava i usluga. Po svom sadržaju kolegij Otvoreno računarstvo je vrlo specifičan s obzirom da prolazi kroz mnoge aspekte otvorenih sustava i tehnologija unutar studijskog programa Računarstva, te je obavezan predmet za modul Računalnog inženjerstva i izborni predmet za module Programskog inženjerstva te Telekomunikacije i informatika. Kako bi se studente na odgovarajući način podučilo tehničkim konceptima otvorenosti i njihovim primjenama, potrebno je osmisliti i izvesti praktični dio nastave u obliku laboratorijskih vježbi koje ih upoznaju s različitim oblicima otvorenih tehnologija, a posebno vezanim uz Web i Internet. Praktičan rad na izradi raspodijeljenih sveprisutnih dinamičkih aplikacija Weba zasnovanih na različitim modernim tehnologijama i jezicima, kao što su jezici HTML, CSS, JavaScript, XML, XSL, PHP i Java, nezaobilazan je dio takvog obrazovanja. Iz tog razloga su nastale laboratorijske vježbe, a posljedično i ovo djelo.

Na kraju predgovora želimo zahvaliti svima koji su na bilo koji način pomogli u izdavanju ovog djela i u oblikovanju vježbi.

Autori

Uvod

O kolegiju Otvoreno računarstvo

Otvoreno računarstvo kolegij je šestog semestra preddiplomskog studija Računarstva Fakulteta elektrotehnike i računarstva Sveučilišta u Zagrebu. Primarni cilj kolegija je upoznati studente s principima otvorenosti, konceptima otvorenih sustava te nizom tehnologija na kojima se danas ta otvorenost zasniva. No, sekundarni, ne manje važan, cilj je osvijestiti studente o važnosti otvorenosti u računarstvu i upoznati ih s otvorenim normama, licencijama i općenito otvorenom kulturom. U kolegiju se stoga sustavno proučava različite oblike i tehnologije otvorenosti od razina podataka, preko sklopovlja, operacijskih sustava, do programske podrške i cjelovitih informacijskih sustava zasnovanih na raspodijeljenim informatičkim uslugama, s neizostavnim naglaskom na otvorene norme.

Kolegij izvode nastavnici i suradnici grupe predmeta RASIP (Računalni sustavi i procesi) Zavoda za automatiku i računalno inženjerstvo pod vodstvom prof.dr.sc. **Marija Žagara** u tjednom opterećenju 3 sata predavanja i 1 sat vježbi, te 4 ECTS boda. Tijekom niza godina na kolegiju su, osim nositelja prof.dr.sc. Maria Žagara, u izvedbi i osmišljavanju gradiva, uključivo i laboratorijskim vježbama predstavljenim u ovom djelu, sudjelovali doc.dr.sc. **Igor Čavrak**, dr.sc. **Branko Mihaljević**, dr.sc. **Marin Orlić**, dr.sc. **Martin Žagar**, **Ivana Bosnić** dipl.ing. i **Tomislav Sečen** dipl.ing. Od akademske godine 2009./2010., kolegij se izvodi u trenutno aktualnoj kombinaciji nositelja i predavača prof.dr.sc. Maria Žagara, glavnog asistenta, koordinatora kolegija i izvođača vježbi dr.sc. Branka Mihaljevića, te asistenta i izvođača vježbi dr.sc. Marina Orlića.

Kolegij Otvoreno računarstvo se prvo izvodio unutar nastavnog programa FER-1 od 1998. godine, a zatim se u većinski prerađenom izdanju nastavio izvoditi u aktualnom nastavnom programu FER-2 po bolonjskom procesu u 3 ciklusa ukupnog trajanja 13 tjedana, a od akademske godine 2011./2012. u 2 ciklusa ukupnog trajanja 15 tjedana. Izvedba uključuje predavanja, pripremu za laboratorijske vježbe i domaće zadaće, laboratorijske vježbe, te međuispite i završne ispite. Kolegij je dizajniran kao kombinacija elemenata e-obrazovanja i klasične nastave. Većina navedenih elemenata odvija se u učionici, a sustavi za upravljanje učenjem podrška su izvođenju nastave u obliku e-obrazovanja.

Kolegij Otvoreno računarstvo ima povijest uporabe koncepata e-obrazovanja s ciljem boljeg približavanja gradiva studentima još od akademske godine 2003./2004. U prvim godinama izvođenja po starom programu FER-1, kada je broj studenata godišnje bio više od dvjestotinjak, velika je pozornost bila upućena prilagodbi nastavnih materijala u multimedijskom obliku **SMIL**[1] (audio, video, prezentacije, dodatna objašnjenja i poveznice). Danas se po primjeni tehnologija i sustava može podijeliti na dva dijela.

Prvi dio je osnovna uporaba FER-ovog **sustava za upravljanje sadržajem**, tzv. „**Quilt CMS**“[2] te FER-ovog **sustava za upravljanje učenjem Moodle**[3]. Na FER-ovom CMS-u je službena stranica kolegija na kojoj se nalaze sve obavijesti, forum za diskusije te repozitorij

[1] SMIL - Synchronized Multimedia Integration Language, http://www.w3.org/TR/smil/
[2] FER e-Campus - Quilt CMS (Content Management System), http://www.fer.unizg.hr/cms
[3] FER Moodle Learning Management System (LMS), http://moodle.fer.hr/

prezentacija u obliku PDF. Stranica kolegija na Moodle LMS-u sadrži prezentacije objavljene i na sustavu **SlideShare**[4], što doprinosi dostupnosti sadržaja. Osim toga, s ove stranice dostupna su i cjelokupna multimedijska predavanja kolegija po starom programu FER-1 u obliku SMIL, jer iako se dio sadržaja unaprijedio, materijali su i dalje vrlo korisni, a oblik SMIL primjer je otvorene norme, što je korisno za objašnjavanje problematike kolegija.

Drugi dio usmjeren je prema uporabi alata namijenjenih kvalitetnijem učenju, čiji ciljevi nisu mogli biti postignuti samo uporabom LMS-a. Naglasak je stavljen na probleme dostupnosti i zajedničkog uređivanja nastavnih materijala sustavom **WikiPres**[5], te automatizirane provjere i usporedbe studentskih rješenja vježbi sustavom **ORVViS**[6] (Otvoreno Računarstvo – Validacija, Verifikacija i Simulacija), kao pomoć studentima u provjeri valjanosti izrađenih rješenja vježbi. Predajom rješenja vježbi na Moodle LMS-u pokreće se automatska analiza predanog rješenja vježbe te se studente obavještava o ispravnosti rješenja. Sustav automatski elektroničkom poštom studentima šalje informacije o pogreškama pri validaciji, a testiranje se može ponavljati do roka predaje vježbe. Na konzultacijama i vježbama studenti razgovaraju s asistentima o vlastitoj ideji rješenja vježbe te prikazuju svoju izvedbu rješenja. Osim što nastavno osoblje po isteku roka predaje vježbe automatski dobiva sažetak ispravnosti predanih rješenja, sustav objedinjuje i podršku za provjeru plagijata korištenjem tzv. sustava **Sherlock**[7], koja uspoređuje datoteke rješenja svih studenata kroz više godina i traži sličnosti kôda, što se učinkovito koristi za provedbu pravila „poštene igre“ i sprječavanje predstavljanja tuđeg rada kao svojega.

Predavanja se na kolegiju objavljuju pod licencijom **Creative Commons**[8] BY-NC-ND (Attribution – NonCommercial – NoDerivativeWorks), počevši s akademskom godinom 2007./2008. Kao posljedicu ove ideje Fakultetsko Vijeće FER-a odlučilo je 2008. godine licenciju Creative Commons prihvatiti kao preporuku za sve nastavne materijale oblikovane na FER-u. Za studente kolegija nastavni materijali na internom sustavu WikiPress izuzeti su od pravila o zabrani prerada, što je dodatni korak prema željenom obliku licencije Creative Commons BY-NC-SA (prerada uz dijeljenje pod istim uvjetima). Inače, svi elementi predavanja izrađeni su u alatu **OpenOffice.org**[9] korištenjem otvorenih norma zapisa podataka.

Kolegij Otvoreno računarstvo je također dobio i prvu nagradu kao najbolji e-kolegij na Sveučilištu u Zagrebu za akademsku godinu 2009./2010.

O vježbama

Djelo „Otvoreno računarstvo – vježbe“ iz kolegija Otvoreno računarstvo objedinjuje sve upute za uspješno savladavanje koraka između teorijskih osnova i primjera predstavljenih na predavanjima, te praktičnih samostalnih studentskih uradaka. Djelo je nastalo kao rezultat višegodišnjeg osmišljavanja i izvođenja laboratorijskih vježbi na predmetu Otvoreno računarstvo te ustustavljivanjem pojedinačnih opisa vježbi u cjelinu.

[4] SlideShare, http://www.slideshare.net/
[5] WikiPres, RASIP, FER, UNIZG, http://wiki.rasip.fer.hr/or/
[6] ORVViS: Otvoreno Računarstvo – Validacija, Verifikacija i Simulacija, RASIP, FER, UNIZG
[7] The Sherlock Plagiarism Detector, http://www.cs.su.oz.au/~scilect/sherlock/
[8] Creative Commons, http://creativecommons.org/
[9] OpenOffice.org, http://www.openoffice.org/

Osnovna ideja laboratorijskih vježbi je upoznavanje studenata s različitim oblicima otvorenih tehnologija, a posebno vezanim uz Web, kroz praktičan rad na izradi dinamičke heterogene aplikacije Weba na neku od odabranih tema. Ukupno šest vježbi zadano je u nepotpuno definiranom obliku, kako bi ih studenti sami oblikovali ovisno o svojoj kreativnosti i inovativnosti. Na kraju semestra sve vježbe objedinjavaju se u jedinstveno funkcionalno rješenje aplikacije Weba zasnovano na otvorenim tehnologijama. Dodatni izazov pri rješavanju zadataka je velika količina tehnologija, razvojnih okruženja i jezika na koje se treba brzo priviknuti, uključivo jezike HTML, CSS, JavaScript, XML, XSL, PHP i Java.

U sljedećem poglavlju objašnjena je izvedba vježbi u okviru kolegija Otvoreno računarstvo, uključivo sve preduvjete i uvjete za uspješno savladavanje vježbi u sklopu studija FER-a, kao i niz pravila vezanih uz aktualni sustav predavanja rješenja i ocjenjivanja. Svi čitatelji koji nisu studenti upisani na navedenom kolegiju, a zanimaju ih konkretne vježbe, mogu ukratko pročitati ovo poglavlje i krenuti dalje na same vježbe, dok studente kolegija, baš suprotno, upozoravamo da pažljivo i detaljno pročitaju sljedeće poglavlje kako bi razumjeli sustav bodovanja i uvjete, uključivo obavezno prisustvo, pripremu, postavljanje na poslužitelj, predaju i samo ocjenjivanje. S obzirom da postoji uzročnost i slijednost laboratorijskih vježbi, čitatelju koji nije student kolegija također se ne preporuča preskakanje pojedine vježbe, jer se svaka sljedeća vježba gradivom i praktičnim uratkom nadovezuje na prethodnu vježbu, a rezultati prethodnih laboratorijskih vježbi koriste se u narednima, te na kraju sve vježbe zajedno čine funkcionalnu cjelinu.

Izvedba laboratorijskih vježbi u okviru predmeta Otvoreno računarstvo

U sklopu kolegija Otvoreno računarstvo održava se šest laboratorijskih vježbi. Laboratorijske vježbe održavaju se tijekom nastavnih ciklusa, načelno **svaki drugi tjedan** (kad nisu pripremna predavanja) i traju **dva sunčana sata** u terminima koji se naknadno određuju u skladu s izradom rasporeda nastave i satnice FER-a. Svakoj laboratorijskoj vježbi prethodi priprema za laboratorijske vježbe i/ili domaća zadaća, koje se izvode samostalno.

Laboratorijske vježbe služe kao praktična iskustva u usvajanju znanja iz gradiva predmeta. Zadatak vježbe koji se rješava povezan je s prethodnom pripremom za laboratorijsku vježbu i domaćom zadaćom. Postoji uzročnost i slijednost laboratorijskih vježbi, pa se tako svaka laboratorijska vježba gradivom i praktičnim uratkom nadovezuje na prethodnu vježbu, u cilju postepene nadogradnje znanja. Rezultati prethodnih laboratorijskih vježbi koriste se u narednima, u cijelosti ili kao bitan dio novog rješenja, te je potrebno dobro poznavanje gradiva prethodnih vježbi kako bi se izradilo potpuno funkcionalno rješenje određene laboratorijske vježbe i posljedično kolokviralo vježbu.

Na svakoj predaji vježbi se, usmeno i/ili pismeno, provjerava i ocjenjuje znanje vezano uz laboratorijsku vježbu i domaću zadaću, čime student može vježbu uspješno kolokvirati.

Obaveznost vježbi i uvjet za pristup završnom ispitu

Vježbe su **obavezne** i **predaja svih vježbi je uvjet za potpis**, bez obzira na broj postignutih bodova na vježbi. Kako bi se uopće pristupilo pismenom dijelu završnog ispita, prije toga trebaju biti predane sve laboratorijske vježbe. Dodatno, kako studenti ne bi završili kolegij bez barem minimalnih praktičnih iskustava, da bi se pristupilo pismenom dijelu završnog ispita potrebno je sakupiti **minimalno 50% bodova iz laboratorijskih vježbi**.

Priprema za laboratorijsku vježbu i izrada rješenja

Priprema za laboratorijsku vježbu podrazumijeva da je potrebno **samostalno proučiti određeni dio gradiva, izraditi domaću zadaću i izraditi rješenje zadatka**, te tako izvesti cjelovitu pripremu za laboratorijsku vježbu. Student se tako prvo detaljno upoznaje s gradivom i zadatkom laboratorijske vježbe kako bi mogao izraditi rješenje. Stoga se priprema izvodi prije termina vježbe, i to samostalno**,** od svakog pojedinog studenta. Pripreme služe za bolje usvajanje gradiva i proširenje tema obrađenih na predavanjima korištenjem praktičnih primjera. Zadatak za pripremu opisan je uz pojedinu laboratorijsku vježbu, a u svakom zadatku je detaljno objašnjenje točno što je (koje datoteke i kakvog sadržaja) potrebno izraditi u cilju rješenja vježbe.

Gradivo priprema laboratorijskih vježbi i domaćih zadaća ulazi u ispitno gradivo sljedeićih provjera:

- provjera znanja na laboratorijskoj vježbi
- nenajavljenih kratkih provjera znanja
- međuispita te završnog pismenog i usmenog ispita

Postavljanje rješenja vježbe na poslužitelj

Rješenje vježbe treba se **postaviti na poslužitelj** (*upload*) prije dolaska na laboratorijsku vježbu. Postavljanje rješenja vježbe na poslužitelj je **preduvjet** pristupu laboratorijskoj vježbi, a time kolokviranju, odnosno i dobivanju bodova. To znači da student treba rješenje vježbe u odgovarajućem obliku postaviti na sustav Moodle, pod svojim korisničkim računom. Sustav Moodle dohvatljiv je poveznicom sa stranice kolegija. Vježba se postavlja na poslužitelj kao jedna datoteka u obliku arhive ZIP koja u sebi sadrži sve datoteke potrebne za ispravan rad vježbe postavljene u određenu strukturu zadanu u samoj vježbi.

Sadržaj datoteka postavljenih na poslužitelj **uspoređuje se** zbog potencijalnih pokušaja prepisivanja. Datoteke se automatski, te po potrebi dodatno ručno, uspoređuju s rješenjima studenata ove i svih prethodnih akademskih godina. Svaki uočeni slučaj sličnosti će se detaljno proučiti te će po potrebi biti prijavljen Povjerenstvu za stegovnu odgovornost studenata.

Vježbu je potrebno postaviti na poslužitelj **do određenog roka**, odnosno datuma i vremena zadanog kod pojedine vježbe na sustavu Moodle, a koji je studentima i posebno istaknut u obavijesti o pojedinoj vježbi na stranici kolegija, što je u načelu oko 24 sata prije termina vježbe.

Naziv datoteke u obliku ZIP treba sadržavati JMBAG, prezime, ime i broj vježbe, odvojene podvlakom (*underscore*). *Primjer*: 0036432456_Ivic_Ivo_1.zip

Brisanje već poslane datoteke i ponovno postavljanje obnovljene inačice na poslužitelj dozvoljeno je neograničeni broj puta, sve do roka predaje. Tako poslana i postavljena datoteka smatra se „nacrtom“ i nad njom se izvodi automatska provjera, svaki put kad je ponovno postavljena.

Nakon što student izradi završnu inačicu rješenja vježbe, treba je obavezno postaviti na poslužitelj i dodatno **označiti kao završnu predaju** odabirom gumba „Predaj na ocjenjivanje“ (*Send for marking*). Takva datoteka se više ne smatra „nacrtom“, već završnom predajom rješenja, nakon kojeg datoteku više nije moguće brisati, niti zamijeniti novom inačicom. Sustav omogućuje predaju vježbe i poslije roka predaje, ali takva predaja imat će za posljedicu umanjenje broja bodova koji se može postići, kao što je kasnije objašnjeno. U slučaju da je postavljena nedovršena inačica rješenja vježbe i, prije krajnjeg roka, slučajno odabrana oznaka završne predaje, potrebno se je obratiti nastavniku kako bi se status vratio na „nacrt“ i kako bi se omogućila ponovna završna predaja.

Automatska validacija i verifikacija rješenja

Kao dodatna pomoć prilikom svakog postavljanja vježbe na poslužitelj koristi se sustav **ORVViS** (Otvoreno Računarstvo – Validacija, Verifikacija i Simulacija). Sustav ORVViS služi za automatiziranu provjeru postavljenog rješenja i izvještavanje studenata o uočenim pogreškama i upozorenjima te studenti nemaju izravnu interakciju s njim, osim što ih sustav izvještava putem elektroničke pošte.

Provjera se izvodi prvenstveno na sintaksnoj razini te se provjerava postojanje resursa, postojanje pojedinih dijelova rješenja i ispravna uporaba sintakse rješenja. Sustav ORVViS značajno štedi vrijeme otkrivanja pogrešaka u rješenju, jer upućuje studenta na određeni resurs s opisom greške koja je uočena ili upozorava o korištenju neprimjerenog dijela rješenja. Izvještaj o postavljenom rješenju se svakom studentu u roku od pola sata šalje elektroničkom poštom na njegovu službenu fakultetsku adresu. Napominjemo da sustav ORVViS ne ocjenjuje rješenje već samo uočava potencijalne pogreške i upozorava studenta. Po isteku roka za postavljanje rješenja sustav ORVViS dodatno nastavniku šalje statističko izvješće o postavljenim rješenjima.

Sustav ORVViS zamišljen je kao dodatna pomoć studentima, još uvijek je u testnoj fazi i ne garantira potpunu ispravnost rješenja i uočavanje svih grešaka. S obzirom da je sustav konstantno u nadogradnji i testiranju, moguće je da povremeno nije dostupan ili da odgovor sustava kasni, za što se treba obratiti nastavniku. Također, ako student nije dobio nikakav odgovor od sustava, postoji mogućnost da je rješenje postavljeno na nepravilan ili neprimjeren način te zbog toga nije čitljivo sa strane sustava, za što se isto treba obratiti nastavniku.

Problemi – izostanci, nadoknade, kašnjenja i umanjenje bodova

Nepoštivanje pravila za laboratorijske vježbe može za posljedicu imati **zabranu prisustvovanja laboratorijskoj vježbi ili umanjenje bodova**.

Ako se rješenje vježbe ne postavi na poslužitelj onda se studentu zabranjuje prisustvovanje laboratorijskoj vježbi. U slučaju predaje samo „nacrta“ rješenja vježbe u predviđenom roku, ali sa sadržajno funkcionalnim rješenjem, student može predati vježbu za maksimalno 50% bodova. Kašnjenje predaje završnog rješenja vježbe do 3 sata nakon roka omogućava predaju vježbe za maksimalno 75% bodova. Prema navedenome, ako postoji funkcionalno rješenje, bolje ga je označiti za ocjenjivanje i do 3 sata nakon roka, nego ostaviti samo „nacrt“, jer je tako moguće dobiti više bodova. Kašnjenje predaje završnog rješenja vježbe do 6 sati nakon roka omogućava predaju vježbe za maksimalno 50% bodova. Kašnjenje predaje završnog rješenja vježbe do 12 sati nakon roka omogućava predaju vježbe za maksimalno 25% bodova. Kašnjenje predaje završnog rješenja vježbe više od 12 sati nakon roka omogućava predaju vježbe za maksimalno 0 bodova, ali uz naknadnu detaljnu provjeru i usporedbu sličnosti rješenja. No, ponavljamo, ako rješenje nije uopće ni u kojem obliku postavljeno na poslužitelj do 1 sat prije termina vježbe, studentu se zabranjuje prisustvovanje laboratorijskoj vježbi, te može doći tek na termin sljedeće vježbe.

Zbog neispravnog imenovanja datoteke i njene strukture, odnosno sadržaja ili ako datoteka u obliku ZIP sadrži neprimjereni, djelomični ili pogrešni sadržaj studentu će se umanjiti bodovi u količini razmjernoj težini greške, a ako je vježba zbog nepotpunog ili pogrešnog sadržaja nefunkcionalna zabranit će se prisustvovanje na laboratorijskoj vježbi, prema odluci nastavnika.

U slučaju **neprisustvovanja** vježbi, iz bilo kog razloga (uključivo opravdani izostanak zbog zdravstvenih ili nekih drugih opravdanih razloga, kao i neopravdani izostanak), student treba doći **na prvi sljedeći mogući termin** laboratorijskih vježbi predati vježbu s koje je izostao. Ako je izostanak učestao, odnosno događa se nekoliko puta zaredom (npr.

dugotrajna bolest), onda je potrebno u dogovoru s nastavnikom dogovoriti posebne predaje vježbe izvan termina, najčešće na Zavodu, kako se ne bi izgubilo pravo na prolaz predmeta. Prilikom dolaska na sljedeći termin laboratorijske vježbe vrijede sva pravila i rokovi za postavljanje rješenja na poslužitelj, kao i za prethodni termin.

Ako je razlog izostanka opravdan, na sljedećem terminu je vježbu moguće predati uz dobivanje maksimalnog broja bodova. Opravdani izostanak se dokazuje odgovarajućom pisanom potvrdom (npr. liječničkom ispričnicom) ili drugom dokumentacijom.

Neopravdani izostanak s vježbe znači da se vježba može predati na prvom sljedećem terminu vježbe, ali maksimalno za 0 bodova i to uz potpuno pridržavanje roka za postavljanje na poslužitelj (bez kašnjenja) i bez drugih većih grešaka pri postavljanju i predaji. S obzirom da ne postoji mogućnost umanjenja navedenih 0 bodova, onda će se pri predaji naročito pomno paziti da ne postoje neke druge greške ili nedostatci u rješenju, a ako su isti uočeni student neće moći predati vježbu. Student će u tom slučaju trebati ponovno dolaziti na svaki sljedeći mogući termin laboratorijskih vježbi sve dok ne preda vježbu za maksimalno 0 bodova. Ovo je navedeno samo kao ekstremni primjer za koji se nadamo da u praksi nikad neće postojati.

Vježbe se **obavezno predaju redoslijedom kojim su zadane**. To znači da se uvijek predaje prva vježba iz slijeda koju student dotad nije predao. Nema mogućnosti preskakanja predaje određene vježbe iz slijeda. Studentima se omogućava da iznimno na jednom terminu laboratorijskih vježbi mogu predati dvije vježbe, naravno ako su zadovoljeni svi preduvjeti oko postavljanja na poslužitelj i ispravnosti rješenja vježbe.

Predaja vježbe

Na svakoj laboratorijskoj vježbi prilikom predaje demonstrira se vlastito rješenje zadatka. Način demonstracije rješenja načelno je moguć na fakultetskom poslužitelju, na lokalnom računalu u prostoriji ili na vlastitom prijenosnom računalu, ovisno o zadatku i karakteru same vježbe, te preduvjetima za predaju vježbe u smislu instalirane podrške (alati, poslužitelj i sl.). Prilikom demonstracije rješenje vježbe treba biti potpuno i funkcionalno. Iako se neka manja odstupanja u funkcionalnosti rješenja mogu tolerirati što rezultira umanjenjem bodova, veća odstupanja u funkcionalnosti i djelomična rješenja neće se uopće ni ocjenjivati, već će se student, zbog neprimjerene pripremljenosti i nepotpunog rješenja, uputiti na ponavljanje vježbe u sljedećem terminu, s bodovanjem kao kod neopravdanog izostanka.

Na svakoj laboratorijskoj vježbi detaljno se provjerava znanje vezano uz gradivo same vježbe i praktični uradak te se ocjenjuje s 4% ukupnih bodova na kolegiju. Nastavnik pismeno i/ili usmeno ispituje razumijevanje zadatka, prikaz rada rješenja i gradivo vezano uz vježbu te domaću zadaću. Na vježbi nastavnik može zadati dodatne praktične i teoretske zadatke za provjeru znanja. Student po uspješnom kolokviranju vježbe, ovisno o prikazanom znanju i vještinama, dobiva ocjenu vježbe u obliku bodova, i to cijelog broja od 0 do 4 boda.

Kada student odgovara gradivo za manji broj bodova, očekuje se potpuno poznavanje gradiva kao za puni broj bodova. Ako je potrebno iz prethodno navedenih razloga umanjiti bodove, onda se ocjena, odnosno bodovi iz ukupno pokazanog znanja relativno skaliraju

naniže ovisno o maksimalnom postotku bodova koji se može postići. Ako se uoči da student ne poznaje gradivo u dovoljnoj mjeri da kolokvira vježbu, uputiti će ga se na ponavljanje vježbe u sljedećem terminu, s bodovanjem kao kod neopravdanog izostanka. Ako se uoči da student nije samostalno riješio vježbu, uputit će ga se na ponavljanje vježbe u sljedećem terminu, s bodovanjem kao kod neopravdanog izostanka, uz dodatnu prijavu Povjerenstvu za stegovnu odgovornost studenata.

Inačice vježbe

Iako su zadaci vježbe, u skladu s gradivom koje se obrađuje, konceptualno isti, sa, zadatak vježbe postoji u osam inačica, s različitim područjima primjene rješenja zadatka. Student izrađuje rješenja svih vježbi isključivo za njemu pridruženu inačicu. Inačica vježbe se pridružuje slučajnim odabirom na početku semestra. U svakoj vježbi koristi se dio informacija iz opisa inačica, pa treba pratiti objašnjenja zadatka u svakoj pojedinoj vježbi. Raspodjela inačica bit će definirana u početnim tjednima semestra i objavljena kao vijest na stranici kolegija.

Sve inačice prikazane su i objašnjenje na kraju u zadnjem poglavlju **Prilog - inačice laboratorijskih vježbi**. Inačice su označene slovom od A do H kako slijedi:

- **Inačica A: DVD-teka**
- **Inačica B: knjižnica**
- **Inačica C: sustav dokumenata**
- **Inačica D: telefonski imenik**
- **Inačica E: evidencija djelatnika**
- **Inačica F: prodaja CD-a**
- **Inačica G: popis literature**
- **Inačica H: evidencija računalne opreme**

Značenje stupaca u opisu inačica je sljedeće:

- **REDNI BROJ** – označava redni broj podatka u strukturi
- **NAZIV** – označava naziv podatka u strukturi
- **HIJERARHIJSKA RAZINA** – označava razinu hijerarhije podatka u odnosu na korijenski element podataka (atributi se smatraju podrazinom elementa)
- **BROJNOST** – označava može li se element pojaviti samo jednom (1) ili više puta (N) unutar istog nadređenog elementa
- **ELEMENT ili ATRIBUT** – označava hoće li se podatak prikazivati kao element ili atribut
- **OBAVEZNOST POSTOJANJA** – označava obaveznost (nužnost) postojanja (barem jednog) podatka (elementa ili atributa)
- **VRIJEDNOSTI** – označava sadrži li element podatak
  - DA – element sadrži (standardno upisanu) tekstualnu vrijednost
  - SKUP – element sadrži podatak isključivo iz skupa dozvoljenih vrijednosti
  - NEMA – element ne sadrži podatak, već se podaci opcionalno nalaze u podelementima i atributima
- **PRIMJER**: SLOBODAN UPISA VRIJEDNOSTI ili SKUP DOZVOLJENIH VRIJEDNOSTI – označava primjer tekstualnog podatka ili prikazuje skup dozvoljenih vrijednosti podatka

  - Primjer tekstualnog podatka (ako u stupcu VRIJEDNOSTI piše DA) – npr. „tekst“
  - Skup dozvoljenih vrijednosti (ako u stupcu VRIJEDNOSTI piše SKUP) – npr. za ocjenu je 1, 2, 3, 4, 5
- **ELEMENT U OBRASCU** na stranici za pretraživanje – označava kako će se izvesti unos podatka po kojem će se pretraživati, odnosno pomoću kojeg elementa za pretraživanje će se ostvariti unos
  - POLJE ZA UNOS – unosno polje za tekstualni podatak
  - KVADRATIĆ ZA IZBOR (*checkbox*) – polje za odabir jedne ili više vrijednosti iz skupa dozvoljenih vrijednosti ili oznaka za postojanje/nepostojanje podatka
  - KRUŽIĆ ZA ODABIR (*radio*) – polje za odabir samo jedne vrijednosti iz skupa dozvoljenih vrijednosti
  - IZBORNIK ZA VIŠESTRUKI ODABIR (*multiple select*) – polje za odabir više (ili samo jedne) vrijednosti iz skupa dozvoljenih vrijednosti

Nagrada ORscar

Ako se do kraja semestra pokaže da pojedini studentski uratci rješenja laboratorijskih vježbi po svojem izgledu i funkcionalnosti pripadaju u kategoriju tzv. „naj“ rješenja (najbolje, najfunkcionalnije, najljepše, najčistije, najzanimljivije …), organizirat će se Natjecanje za nagradu ORscar za odabir najboljih rješenja laboratorijskih vježbi. O konačnom pobjedniku iste odlučivat će zajednički svi studenti na kolegiju i nastavnički žiri, a podijelit će se i simbolične nagrade pobjednicima. Natjecanje se inicijalno provodi svake godine, ali se od njega može odustati ako, prema mišljenju nastavničkog žirija, ne postoje uratci dovoljno dobri da bi konkurirali za nagradu.

Pitanja i problemi

Sva pitanja **od općeg interesa** u vezi pojedine laboratorijske vježbe trebaju se uputiti na diskusijsku grupu - **forum** na stranici kolegija. Nastavnici će odgovarati na pitanja čim budu u mogućnosti. Ako se radi **o osobnom problemu ili pitanju**, ono se **elektroničkom poštom upućuje koordinatoru kolegija**, **isključivo s vlastite fakultetske adrese** elektroničke pošte (inače se smatra neželjenom elektroničkom poštom – *spam*), uz **naslov** poruke (*subject*) koji započinje s „**[OR]**“.

Za sva ostala pitanja obratite se izravno nastavniku, elektroničkom poštom ili osobno.

1. laboratorijska vježba: Izrada stranica Weba i obrasca

Cilj vježbe „Izrada stranica Weba i obrasca“ je upoznavanje s načinima izrade stranica Weba korištenjem otvorenih normi i pratećih jezika. Stranice Weba, osim što su osnovni dio svakog sjedišta Weba, čine i osnovno korisničko grafičko sučelje aplikacija Weba, što će biti korišteno i u drugim vježbama. Stoga se u ovoj vježbi prvo upoznaje s općeprihvaćenim označnim jezikom HTML za izradu dokumenata stranica Weba i jezikom predložaka stila CSS za oblikovanje stranica Weba i definiranje njihovog izgleda. U vježbi se detaljnije upoznaje s osnovnim elementima i oznakama jezika HTML, uključujući izradu listi, tablica i obrazaca te primjenom stilova pomoću jezika CSS. Poseban naglasak je stavljen na izradu stranice s obrascem pretrage koji će se koristiti u kasnijim vježbama.

Za bolje razumijevanje predstavljenog, a pogotovo u slučaju izostanka s predavanja, preporuča se detaljno proučiti materijale s predavanja pod nazivom „Oznake“.

Priprema za vježbu

U cilju pripreme za vježbu potrebno je **proučiti osnovne dijelove označnog jezika HTML** s naglaskom na:

- sintaksu i druga pravila jezika HTML
- osnovnu strukturu dokumenata u jeziku HTML – zaglavlje, tijelo
- definiranje dodatnih podataka o dokumentu – oznaka `<meta>`
- definiranje kodne stranice sadržaja dokumenta
- deklaraciju tipa dokumenta za provjeru pravila – oznaka `DOCTYPE`
- povezivanje s vanjskom datotekom predložaka stila u jeziku CSS – oznaka `<link>`
- definiranje odjeljaka – oznaka `<div>` te atributi `id` i `class`
- označavanje naslova i podnaslova – oznake `<h1>` do `<h3>`
- razdvajanja teksta u dokumentu – oznaka odjeljka `<p>` i novog reda `<br>`
- isticanje teksta oblicima slova – oznake `<b>` i `<i>`
- umetanje slika u dokument – oznaka `<img>`
- poveznice (između stranica i na elektroničku poštu) – oznaka `<a>`
- osnove izrade tablica – oznake `<table>`, `<tr>`, `<th>` i `<td>` te atributi `colspan` i `rowspan`
- izradu pobrojanih i nepobrojanih lista – oznake `<ul>`, `<ol>` i `<li>`
- osnovu izrade obrazaca – oznaka `<form>` i različite inačice oznake `<input>`

Osim toga potrebno je **proučiti osnovne dijelove jezika predložaka stila CSS** s naglaskom na:

- sintaksu i druga pravila jezika CSS
- odabir (*selector*) obzirom na elemente, razrede (klase) i identifikatore
- razmještaj (*layout*) elemenata na stranici
- definiranje značajki elemenata kao što su boje, pozadina, vrsta i veličina slova, praznine oko rubova i razmaci, a naročito u odnosu na:
  - odjeljke teksta
  - liste
  - tablice
  - poveznice (*links*)

Zadatak za vježbu

Ovisno o pridruženoj inačici zadatka, potrebno je **izraditi sjedište Weba s dvije stranice** koje će predstavljati osnovno korisničko sučelje za sve predstojeće laboratorijske vježbe. Stoga je, istovremeno s upoznavanjem sa zadatkom vježbe, potrebno proučiti i inačicu zadatka koja vam je pridružena (u skladu s obavijesti na stranici predmeta). Inačice zadatka dane su na kraju u poglavlju **Prilog - inačice laboratorijskih vježbi**, a stranice Weba koje treba izraditi razlikuju se ovisno o inačici zadatka. U ovom zadatku opisani su svi zajednički zahtjevi i postupci u oblikovanju stranica Weba koji vrijede za sve inačice.

Napomena: Radi lakšeg uočavanja dijelova oznaka u jeziku HTML preporučuje se korištenje nekog od tekstualnih uređivača koda koji imaju podršku za bojanje sintakse programskih i označnih jezika, kao što su jEdit[10], Notepad2[11], Notepad++[12] i slično.

U izvedbi zadatka, kako bi rješenje bilo u skladu s osnovnom idejom vježbi, sljedilo određenu razinu uniformnosti u oblikovanju i izvedbi, te bilo izvedeno u skladu s određenim normama, nužno je potrebno pridržavati se određenih uputa i zahtjeva. Ti zahtjevi su zajednički nad svim stranicama Weba svih inačica i vrijede za sve zadatke, a uključuju sljedeće:

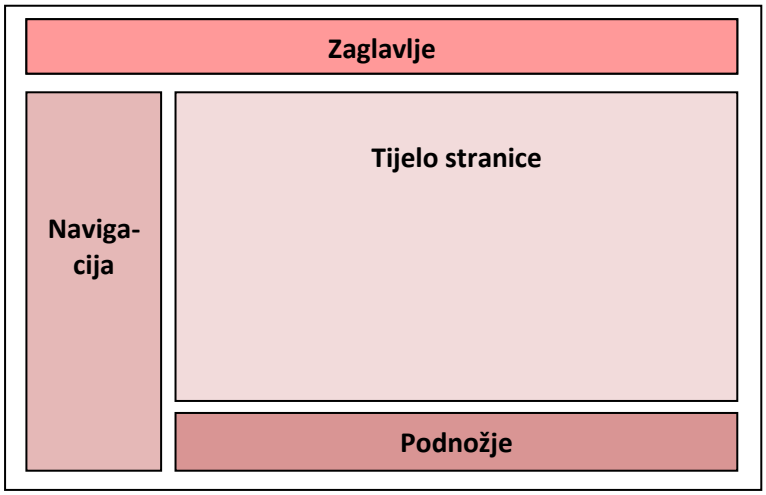
- treba koristiti naredbu obrade **DOCTYPE** i deklaraciju tipa dokumenta **HTML 4.01 Strict**[13]
- stranice Weba trebaju biti napisane u normi za zapis tekstualnih dokumenata (kodnoj stranici) **UTF-8**
  - prilikom snimanja paziti da dokument bude **pohranjen** u odgovarajućem obliku zapisa - oblik UTF-8 bez *byte-order-mark*, BOM[14]
  - u dokumentu oznakom postaviti **pretpostavljenu kodnu stranicu** koju će preglednik automatski koristiti
- **raspored** elemenata treba izvesti **pomoću predložaka stila u jeziku CSS** korištenjem atributa `id` i `class`
- za predloške stila treba koristiti **povezivanje s** jedinstvenom **vanjskom datotekom CSS**
- raspored elemenata treba izvesti **korištenjem područja** oznakom `<div>` (a ne pomoću tablica)
- postoje **obavezna područja stranica** koja su uvijek uniformno definirana na svim stranicama Weba (trebaju se izvesti se korištenjem područja oznakom `<div>` u skladu s uputom o rasporedu područja na sljedećoj stranici):
  - **zaglavlje stranice** (*header*) – treba sadržavati **osnove informacije** o sjedištu uključivo i:
    - **naslov** sjedišta Weba
    - naslovnu **sliku**
    - **poveznicu** izvedenu nad naslovnom slikom koja treba voditi na početnu stranicu sjedišta Weba

[10] jEdit – Programmer's Text Editor, http://jedit.org/
[11] Notepad2, http://www.flos-freeware.ch/notepad2.html
[12] Notepad++, http://notepad-plus-plus.org/
[13] HTML 4.01 Specification - http://www.w3.org/TR/html401
[14] UTF-8 (bajtno orjentirani oblik kodiranja Unicode) i BOM http://unicode.org/faq/utf_bom.html

- **navigacija** – treba biti **vertikalna** (poveznice složene jedna ispod druge) i treba sadržavati barem sljedeće **poveznice**:
  - poveznicu na početnu stranicu sjedišta
  - poveznicu na stranicu za pretraživanje
  - poveznicu na sjedište RASIP (http://www.rasip.fer.hr)
  - poveznicu na sjedište FER (http://www.fer.unizg.hr), uz otvaranje u **novom** prozoru
  - poveznicu na adresu elektroničke pošte autora vježbe
  - prijedlog: napraviti navigaciju korištenjem lista i njihovom prilagodbom kroz CSS
- **tijelo stranice** (*body*) – glavni dio za sadržaj, sadrži druge sadržajne elemente ovisno o namjeni stranice
- **podnožje** (*footer*) – sadrži osnovne informacije o sjedištu, primjerice ime i prezime autora stranice, uz eventualan logotip/sliku i slično

Napomena: Alternativno je moguće korištenje drugih oblika navigacije u obliku izbornika i slično (npr. korištenjem jezika JavaScript), ali na vlastitu inicijativu i odgovornost studenta, bez službene podrške od strane nastavnika.

- **raspored** (*layout*) područja na stranici:



- promjena svojstava izgleda – dizajn i izgled elemenata trebaju biti izvedeni **pomoću jezika CSS**, a potrebno je prikazati oblikovanje barem sljedećih elemenata:
  - veličine, vrste i boje slova
  - tipa oblikovanja slova (npr. **podebljana slova**, *kurziv*)
  - položaja elemenata na stranici
  - boje pozadine elemenata
  - okvira elemenata (*border*)
  - margina i ispuna (*margin, padding*)
  - liste za navigaciju
  - grafičke promjene elementa pri prelasku mišem (npr. odabir (*selector*) *a:hover* za uljepšavanje navigacije)

Napomena: Sve što ovdje nije navedeno ili definirano moguće je proizvoljno oblikovati. To znači da je moguće koristiti i druge napredne elemente i tehnike oblikovanja stranica, ali isključivo na vlastitu inicijativu i odgovornost studenta, bez podrške od strane nastavnika.

Prilikom izrade stranica obratite pažnju na uređenost stranice kao cjeline u smislu dizajna, izgleda, slaganja boja i slično. Prilikom izrade stranice nužno je pridržavanje normi pisanja dokumenata u jeziku HTML. Pridržavanje normi potrebno je prije predaje rješenja provjeriti uz pomoć alata za validaciju kao što su:

- W3 Markup Validation Service – http://validator.w3.org
- WDG HTML Validator – http://htmlhelp.com/tools/validator

Prva stranica: Početna stranica

Potrebno je izraditi **početnu stranicu** sjedišta Weba naziva `index.html` prema navedenim zajedničkim zahtjevima za sve stranice. Unutar tijela stranice potrebno je napisati **uvodne informacije** (naslov i nekoliko rečenica) o sjedištu Weba, u skladu s inačicom zadatka koju treba izraditi. Taj sadržaj dodatno treba sadržavati i **poveznicu na stranicu za pretraživanje**. Unutar uvodnog dijela potrebno je postaviti i **proizvoljnu sliku** vezanu za inačicu koja se izrađuje.

Dodatno, u sadržaju tijela stranice treba postojati **barem jedna tablicu s barem tri kolone i tri retka**, te **barem jedna pobrojana i jedna nepobrojana lista**. Grafičko uređenje tablica i listi, te sadržaj istih su proizvoljni, ali opet u skladu s inačicom vježbe.

Napomena: Prilikom odabira slika koje umećete u stranicu potrebno je obratiti pažnju na autorska prava i licencije. Slike s drugih stranica, iako se mogu, često se **ne smiju** umnožavati i koristiti. No, mnoga sjedišta Weba sadrže slike koje je dozvoljeno koristiti za vlastite potrebe, pa se treba pronaći takve slike primjerene za korištenje u rješenju vježbe.

Druga stranica: Stranica za pretraživanje

Potrebno je izraditi **stranicu** sjedišta Weba **za pretraživanje** naziva `obrazac.html` pomoću koje će se u idućim vježbama pretraživati podatke. I ova stranica treba zadovoljavati navedene zajedničke zahtjeve za sve stranice. Dodatno, u području tijela stranice ova stranica sadrži obrazac pomoću kojega će biti moguće filtriranje i prikaz odabranih podataka, odnosno pretraživanje po različitim parametrima zadanim u obrascu.

Tijelo stranice treba sadržavati naslovni dio i **obrazac**. Obrazac treba izraditi **u obliku tablice**, u kojem se u jednoj koloni nalazi opis polja, a u drugoj upisno polje ili jedan ili više elemenata za unos. Svojstva i izgled tablice, odnosno njenih redova i ćelija potrebno je definirati pomoću jezika CSS. **Polja obrasca ovise o inačici zadatka koju izrađujete.**

U ovoj vježbi su za definiranje polja obrasca **bitni sljedeći stupci** u opisu inačica:

- REDNI BROJ – označava redni broj podatka u strukturi
- **NAZIV** – označava naziv podatka u strukturi
- **VRIJEDNOSTI** – označava sadrži li element podatak
  - DA – sadrži (standardno upisanu) tekstualnu vrijednost
  - SKUP – sadrži podatak isključivo iz skupa dozvoljenih vrijednosti

  - NEMA – ne sadrži podatak, već se podaci opcionalno nalaze u podelementima i atributima
- **PRIMJER: SLOBODAN UPISA VRIJEDNOSTI ili SKUP DOZVOLJENIH VRIJEDNOSTI** – označava primjer tekstualnog podatka ili prikazuje skup dozvoljenih vrijednosti podatka
  - Primjer tekstualnog podatka (ako u stupcu VRIJEDNOSTI piše DA) – npr. „tekst“
  - Skup dozvoljenih vrijednosti (ako u stupcu VRIJEDNOSTI piše SKUP) – npr. za ocjenu je 1, 2, 3, 4, 5
- **ELEMENT U OBRASCU na stranici za pretraživanje** – označava kako će se izvesti unos podatka po kojem će se pretraživati, odnosno pomoću kojeg elementa za pretraživanje će se ostvariti unos
  - POLJE ZA UNOS – unosno polje za tekstualni podatak
  - KVADRATIĆ ZA IZBOR (*checkbox*) – polje za odabir jedne ili više vrijednosti iz skupa dozvoljenih vrijednosti ili oznaka za postojanje/nepostojanje podatka
  - KRUŽIĆ ZA ODABIR (*radio*) – polje za odabir samo jedne vrijednosti iz skupa dozvoljenih vrijednosti
  - IZBORNIK ZA VIŠESTRUKI ODABIR (*multiple select*) – polje za odabir više (ili samo jedne) vrijednosti iz skupa dozvoljenih vrijednosti

Napomena: Iz navedenog je vidljivo da u opisu inačica stupac ELEMENT U OBRASCU zapravo označava način prikaza pojedinog elementa u obrascu za pretraživanje na stranici.
Primjer: podatak IME se prikazuje kao POLJE ZA UNOS, a podatak SPOL se prikazuje kao dva KVADRATIĆA ZA IZBOR (*checkbox*).

Obrazac na stranici treba sadržavati, u skladu opisom inačica i mogućim podacima:

- **barem 5 polja za unos teksta** parametara za pretraživanje
- **barem 2 grupe kvadratića za izbor** (*checkbox*) s ponuđenim vrijednostima
- **barem 1 grupu kružića za odabir** (*radio*) s ponuđenim vrijednostima
- **barem 1 izbornik za višestruki odabir** (*multiple select)*
- **gumb za slanje** podataka na poslužitelj (*submit button*)
- **gumb za poništavanje** podataka u obrascu (*reset button*)

Napomena: U ovoj vježbi još ne postoji pozadinski proces obrade podataka s obrasca, pa obrazac trenutno nema funkcionalnost. Ovaj obrazac ćemo iskoristiti u sljedećim vježbama.

Napomena: Prilikom izrade rješenja vježbe potrebno je podržati korisničko sučelje na hrvatskom jeziku, uključivo sve hrvatske grafeme, odnosno dijakritičke znakove.

Predaja vježbe

Vježba se predaje prikazivanjem (demonstracijom) obje stranice u pregledniku na lokalnom računalu.

Napomena: Za prikaz rješenja koristit će se trenutno najzastupljeniji preglednici: Internet Explorer, Firefox, Chrome, Safari i Opera.

Kao što je već spomenuto, stranice Weba moraju biti zapisane u datotekama u obliku dokumenata HTML:

- `index.html`
- `obrazac.html`

datoteka dizajna mora biti zapisana u vanjskoj datoteci CSS

- `dizajn.css`

dok slike i ostale datoteke mogu biti proizvoljnih naziva i u proizvoljnim mapama.

Datoteka u obliku ZIP koja se predaje na poslužitelj mora u glavnoj mapi sadržavati sve tri navedene datoteke, dok ostale datoteke mogu biti u proizvoljnim mapama.

Napomena: Izrađene stranice bi trebale, ako već ne identično, onda barem vrlo slično izgledati u različitim preglednicima. Tijekom predaje vježbe je moguće da se traži prikaz u više različitih preglednika i dorada rješenja ako se u nekom od njih stranica ne prikazuje ispravno („raspadne se“).

Ispitno gradivo vježbe

Ispitno gradivo uključuje sve navedeno u pripremi za vježbu, te detaljno razumijevanje izrađenog rješenja i snalaženje u prepravcima istog prema uputama nastavnika.

Primjeri pitanja:

- Čemu služi definiranje kodne stranice dokumenta?
- Zašto se svojstva elemenata definiraju pomoću jezika CSS?
- Koje su prednosti povezivanja s vanjskom datotekom CSS?
- Kako možete promijeniti izgled navigacije, npr. iz okomite u vodoravnu?
- Kako se definira različita boja pozadine za parne i neparne retke u tablici?
- Kako ćete promijeniti boju i podcrtati naslov pri prelasku mišem preko poveznica?

Poveznice i literatura

- **W3Schools: HTML Tutorial** - http://www.w3schools.com/html
- HTML Code Tutorial - http://www.htmlcodetutorial.com
- HTML Dog: HTML Beginner Tutorial - http://www.htmldog.com/guides/htmlbeginner
- HTML 4.01 Specification - http://www.w3.org/TR/html401
- HTML 4.01 / XHTML 1.0 Reference - http://www.w3schools.com/tags
- **W3Schools: CSS Tutorial** - http://www.w3schools.com/css
- HTML Dog: CSS Beginner Tutorial - http://www.htmldog.com/guides/cssbeginner
- HTML Dog: HTML and CSS Tutorials - http://www.htmldog.com/guides
- Forms - http://www.w3.org/TR/REC-html40/interact/forms.html
- **The W3C Markup Validation Service** - http://validator.w3.org
- WDG HTML Validator – http://htmlhelp.com/tools/validator
- The W3C CSS Validation Service - http://jigsaw.w3.org/css-validator
- stock.xchng – the leading free stock photography site – http://www.sxc.hu

2. laboratorijska vježba: Strukturiranje, validacija i transformacija podataka

Cilj vježbe „Strukturiranje, validacija i transformacija podataka“ je upoznavanje sa strukturiranjem informacija u općeprihvaćenom obliku označnih jezika, te validacijama strukturiranih podataka i transformacijama u druge oblike. U sklopu vježbe se upoznaje s označnim jezikom XML i strukturiranim oblikom zapisa podataka u jeziku XML. Osim toga upoznaje se s problematikom dobre oblikovanosti i valjanosti podataka, a pomoću iste i s provjerom ispravnosti dokumenta pomoću gramatike. U drugom dijelu vježbe upoznaje se s transformacijama podataka uporabom jezika XSL. Dodatno, u ovoj vježbi se upoznaje i s razlikama jezika XHTML i HTML.

Priprema za vježbu

U cilju pripreme za vježbu potrebno je **proučiti osnovne dijelove označnog jezika XML** s naglaskom na:

- osnove jezika XML i zaglavlje - `<?xml..?>`
- korištenje elemenata i atributa
- definicije dobre oblikovanosti i valjanosti
- načine uključivanja DTD-a u XML dokumente `<!DOCTYPE ...>`

Dodatno, potrebno je **proučiti osnove gramatike za provjeru ispravnosti podataka u obliku XML** s naglaskom na:

- definiranje elemenata u DTD-u (sadrže neparsirani tekst, druge elemente, itd.)
- kod uključivanja drugih elemenata obratiti pažnju na BNF notaciju prilikom specifikacije brojnosti sadržaja (*, +, ?, |).
- načine definiranja atributa
  - proizvoljne vrijednosti
  - koji poprimaju vrijednosti iz skupa dozvoljenih vrijednosti
  - koji su obavezni te koji se podrazumijevaju ili su fiksni (#REQUIRED, #IMPLIED, #FIXED)

Osim toga, za drugi dio vježbe potrebno je **proučiti osnovne transformacije podataka u obliku XML korištenjem jezika XSL** s naglaskom na:

- osnove jezika XSL
- uključivanje stilova XSL u dokumente u obliku XML
- definiranje i primjenu predložaka u jeziku XSL
- postavljanje prostora imena i izbjegavanje konflikta imenovanja
- iteriranje po skupu elemenata korištenjem `for-each` i sličnih oznaka jezika XSL
- provjeru uvjeta XSL oznakom `if`
- dohvaćanje vrijednosti elemenata iz oblika XML jezikom XSL
- postavljanje atributa u izlaznom dokumentu jezikom XSL

Zadatak za vježbu

Potrebno je **izraditi gramatiku DTD** za provjeru odgovarajućeg strukturiranog zapisa podataka u XML dokumentu te dodatno **izraditi testni XML dokument** koji služi za ispitivanje primjerenosti izrađenog DTD dokumenta pretpostavljenoj strukturi podataka. Struktura podataka zadana je **inačicom zadatka** koji vam je pridružen. Dokument mora biti **dobro**

oblikovan i valjan. DTD dokument mora sadržavati sve navedene podatke u pridruženoj inačici.

Napomena: XML Schema (XSD) je suvremenija i složenija alternativa gramatike DTD. Schema omogućava detaljniji opis podataka koji su pohranjeni u XML dokumentu. Ovu vježbu možete izraditi i tako da umjesto DTD-a upotrijebite XSD za opis gramatike dokumenta. XSD nudi veće mogućnosti, ali je i složeniji, pa se korištenje XSD-a prepušta na vlastitu inicijativu i odgovornost, bez podrške od strane nastavnika. No, nastavnici će potencijalno, ako se ukaže potreba i ako budu u mogućnosti, pomoći odgovorima na pitanja i savjetima na forumu stranice kolegija.

U drugom dijelu vježbe za izrađeni XML dokument izraditi **transformaciju u jeziku XSL** koja će podatke iz XML dokumenta prebaciti u drugi dokument u obliku, a koji će otvaranjem u pregledniku prikazati podatke u obliku **tablice** unutar stranice Weba definirane dizajnom iz prethodne vježbe.

Pri izradi gramatike i testnog dokumenta u opisu inačica možemo uočiti niz stupaca koje imaju utjecaj na strukturu XML dokumenta:

- HIJERARHIJSKA RAZINA – označava razinu hijerarhije podatka u odnosu na korijenski element podataka (atributi se smatraju podrazinom elementa)
- BROJNOST – označava može li se element pojaviti samo jednom (1) ili više puta (N) unutar istog nadređenog elementa
- ELEMENT ili ATRIBUT – označava hoće li se podatak prikazivati kao element ili atribut
- OBAVEZNOST POSTOJANJA – označava obaveznost (nužnost) postojanja (barem jednog) podatka (elementa ili atributa)
- VRIJEDNOSTI – označava sadrži li element podatak
  - DA – sadrži (standardno upisanu) tekstualnu vrijednost
  - SKUP – sadrži podatak isključivo iz skupa dozvoljenih vrijednosti
  - NEMA – ne sadrži podatak, već se podaci opcionalno nalaze u podelementima i atributima
- PRIMJER: SLOBODAN UPISA VRIJEDNOSTI ili SKUP DOZVOLJENIH VRIJEDNOSTI – označava primjer tekstualnog podatka ili prikazuje skup dozvoljenih vrijednosti podatka
  - Primjer tekstualnog podatka (ako u stupcu VRIJEDNOSTI piše DA) – npr. „tekst“
  - Skup dozvoljenih vrijednosti (ako u stupcu VRIJEDNOSTI piše SKUP) – npr. za ocjenu je 1, 2, 3, 4, 5

Napomena: Dobro oblikovani XML dokument treba imati samo jedan korijenski element.

Uočite u opisu inačica da zadnja **četiri elementa ili atributa** nisu definirani i nose naziv „PROIZVOLJAN ODABIR“, te ih je potrebno **proizvoljno definirati**, no tako da s preostalim elementima i atributima čine smislenu logičku cjelinu. Na vježbi će se povjeravati prikladnost odabira tih elemenata i atributa u kontekstu cijele inačice. Očekuje se da su ta četiri podatka različitih tipova (što elementi, što atributi), te da dočaravaju i promišljanje o cjelokupnoj strukturi i logično je nadopunjuju.

Napomena: Na vježbi će se uspoređivati potencijalno prepisivanje takvih proizvoljnih tipova unutar rješenja iste kao i različitih inačica. Osnovna ideja proizvoljnog odabira je da se osmisle vlastiti elementi ili atributi koji pokazuju raznolike mogućnosti zapisa u obliku XML.

Testni XML dokument koji će se koristiti za provjeru valjanosti izrađenom gramatikom treba **sadržavati barem pet podataka prve razine**, koje treba proizvoljno izraditi (npr. pet osoba u imeniku i slično). Osim toga dokument treba biti zapisan u **kodnoj stranici UTF-8** i treba u odnosu na pridruženu gramatiku biti **valjan**. Provjeru valjanosti dokumenta možete izvesti pouzdanim alatom za po izboru.

U drugom dijelu vježbe potrebno je izraditi **transformaciju u jeziku XSL** za dohvat podataka iz XML dokumenta koji će popuniti retke **tablice u HTML obliku** unutar novoizrađenog dokumenta. Rezultat XSL transformacije treba biti **XHTML dokument**, odnosno HTML dokument zapisan u XML obliku, koji, među ostalim, sadrži navedenu tablicu.

Napomena: Uočite razlike između zapisa dokumenta u obliku XHTML i u obliku HTML.

Ispravnost prikaza XHTML dokumenta **provjerit** će se prikazom u nekom od **preglednika** na način da se pozove XML dokument nad kojim će preglednik automatski provesti XSL transformaciju kako je specificirano u zaglavlju dokumenta (objašnjeno kasnije).

Dodatno, na obje stranice iz 1. laboratorijske vježbe treba **unutar izbornika navigacije postaviti poveznicu na novoizrađenu stranicu**, odnosno točnije, na XML datoteku sa sufiksom `.xml`, koja u sebi sadrži poziv XSL transformacije sa sufiksom `.xsl`.

Novoizrađena stranica treba slijediti dizajn stranica iz 1. laboratorijske vježbe (zaglavlje, navigacija itd.), dok se tablica prikazuje u tijelu stranice. To znači da je unutar XSL datoteke, osim same transformacije, odnosno oznaka dohvata podataka, potrebno uključiti i sve ostale dijelove HTML stranice (npr. definiciju kodne stranice, vanjskog dokumenta CSS itd.), koji će se, isto kao i u 1. laboratorijskoj vježbi, koristiti kod prikaza u pregledniku.

Prilikom izrade tablice, uzmite u obzir **preglednost** prikaza podataka. U slučaju da je prikazivanje svih podataka nepregledno (ne stanu svi stupci na zaslon), izaberite podskup podataka za prikazivanje i prikažite samo njih.

Napomena: Ako je prikaz podataka nepregledan bolje je izostaviti neke manje bitne stupce (elemente, atribute) nego ih „nagurati“ na stranicu. Također, dvoredni prikaz pojedinog zapisa nije baš pregledan ni preporučljiv.

Tablica s prikazom podataka treba imati **zaglavni redak** (*header row*) s nazivom pojedinih podataka (stupaca).

Napomena: Prilikom izrade rješenja vježbe potrebno je podržati podatke na hrvatskom jeziku koji sadrže hrvatske grafeme, odnosno dijakritičke znakove.

Predaja vježbe

Vježba se predaje prikazivanjem (demonstracijom) niza dokumenata u uređivaču i pregledniku na lokalnom računalu. Datoteke koje treba predati su sljedećih naziva:

- `podaci.xml`
- `gramatika.dtd`
- `pretvorba.xsl`

Pod predajom vježbe podrazumijeva se:

- prikaz DTD datoteke uz objašnjenje pojedinih dijelova dokumentu
- predočavanje testnog XML dokumenta
- dokazivanje valjanosti testnog XML dokumenta (alatom po izboru)
- promjene u gramatici i testnom XML dokumentu te ponovnu provjeru valjanosti testnog dokumenta
- predočavanje XSL transformacije testnog XML dokumenta u pregledniku
- eventualne promjene u testnom XML dokumentu ili definiciji XSL transformacije te ponovno provođenje transformacije

U ovoj su vježbi potrebne i druge datoteke iz 1. laboratorijske vježbe. Stoga na poslužitelj Moodle treba postaviti datoteku sa ZIP arhivom koja sadrži sve datoteke potrebne za prikaz rješenja. Datoteka u obliku ZIP koja se predaje na poslužitelj mora u glavnoj mapi sadržavati sve tri navedene datoteke dok ostale datoteke mogu biti u proizvoljnim mapama.

Provjera valjanosti XML dokumenta

Kao što je već rečeno, izbor alata kojim se vrši provjera valjanosti je proizvoljan. Uvjet je da je korištenje alata legalno. Predlažemo sljedeće alate, odnosno načine provođenja provjere:

- Na stranici predmeta u repozitoriju datoteka nalazi se **Alat za validaciju XML-a**. Za pokretanje alata morate imati instaliran Java Virtual Machine (JVM) verzije 1.5 ili novije. Ako već nemate instaliranu podršku za potrebe izvođenja instalaciju okruženja Java Runtime Enviroment (JRE) možete preuzeti sa sljedeće adrese: http://www.oracle.com/technetwork/java/javase/downloads/index.html
- Na računalu s operacijskim sustavom Windows možete koristiti jednostavan i besplatan **alat AltovaXML 2011 Community Edition**. Instalacijsku datoteku možete preuzeti s adrese: http://www.altova.com/download/altovaxml/xml-processor-community.html
- U predavanjima se nalazi izvorni kod i primjer korištenja vrlo jednostavne **PHP skripte** koja se može koristiti za provjeru valjanosti XML dokumenta. Za korištenje ove skripte, trebate imati instaliran PHP (inačica 5.2.5 ili noviji) za operacijski sustav koji koristite na računalu. PHP možete preuzeti s adrese: http://www.php.net

Transformacija XML dokumenta

Transformacija dokumenta treba se provoditi korištenjem XSL procesora koji podržava XSL transformacije, najjednostavnije u jednom od preglednika.

Napomena: Za prikaz rješenja koristit će se trenutno najzastupljeniji preglednici: Internet Explorer, Firefox, Chrome, Safari i Opera.

Kako bi preglednik znao provesti XSL transformaciju XML dokumenta, u XML dokumentu je to potrebno označiti na sljedeći način:

```
<?xml-stylesheet type="text/xsl"
href="datoteka_s_opisom_transformacije.xsl"?>
```

Neke oznake u XSL datoteci

XSL datoteka je valjani XML dokument čiji je korijenski element `stylesheet` iz prostora imena `xsl`, a zapisuje se na sljedeći način:

```
<xsl:stylesheet
xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0"> ...
</xsl:stylesheet>
```

Postavljanje tipa izlaznog dokumenta (DOCTYPE) izvodi se umetanjem oznake `xsl:output` s odgovarajućim vrijednostima atributa :

```
<xsl:output method="xml" indent="yes"
doctype-system="http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd"
doctype-public="-//W3C//DTD XHTML 1.0 Strict//EN"/>
```

Primjena predloška na pojedinu strukturu elemenata izvodi se umetanjem oznake `xsl:template`, a pokreće se s:

```
<xsl:template match="/"> ... </xsl:template>
```

gdje izraz u atributu `match` omogućava odabir elementa na koji se predložak primjenjuje.

Napomena: Upis znaka"/" označava sve elemente.

Prostor imena oznaka u obliku oznaka jezika HTML može se postaviti na elementu `<html>` na sljedeći način:

```
<html xmlns="http://www.w3.org/1999/xhtml">
```

Iteriranje po podelementima nekog elementa izvodi se oznakom `for-each` koja je slična `for` petlji u programskim jezicima, i to na sljedeći način:

```
<xsl:for-each select="element"> ... </xsl:for-each>
```

Dohvaćanje sadržaja elementa XML-a izvodi se oznakom `value-of`:

```
<xsl:value-of select="naziv" />
```

Dodavanje atributa elementu izlaznog dokumenta potrebno je posebno navesti. Dodavanje atributa `href` elementu poveznice `<a>` i postavljanje sadržaja za slanje elektroničke pošte može se izvesti kako slijedi:

```
<a>
   <xsl:attribute name="href">
       mailto:
       <xsl:value-of select="ime" />
       .
       <xsl:value-of select="prezime" />
       @fer.hr
   </xsl:attribute>
   Elektronička pošta za:
   <xsl:value-of select="ime" />
   <xsl:value-of select="prezime" />
</a>
```

Ispitno gradivo vježbe

Ispitno gradivo uključuje sve navedeno u pripremi za vježbu, te detaljno razumijevanje izrađenog rješenja i snalaženje u prepravcima istog prema uputama nastavnika.

Primjeri pitanja:

- Što je XML, a što gramatika?
- Što znače pojedine stavke u zaglavlju XML dokumenta?
- Kako je sve moguće uključiti provjeru gramatike u XML dokument?
- Što znači dobro oblikovan, a što valjan dokument?
- Kako se u deklariraju elementi, a kako atributi elemenata?
- Kako se pišu komentari?
- Kako odlučiti da li bi nešto trebalo biti atribut ili element?
- Kako se XML dokumentu pridružuje XSL predložak stila?
- Kako se korištenjem jezika XSL može stvoriti popis svih zapisa (podataka) odjednom?
- Kako se korištenjem jezika XSL može stvoriti popis svih elemenata istog roditelja?
- Kako se korištenjem jezika XSL može stvoriti popis svih elemenata čiji atribut ima neku vrijednost?
- Kako bi drugom bojom označili elemente s određenim značajkama?

Poveznice i literatura

- **W3Schools XML Tutorial** - http://www.w3schools.com/xml
- Zvon XML Tutorial - http://zvon.org/xxl/XMLTutorial/General/book.html
- Tizag XML Tutorial - http://www.tizag.com/xmlTutorial/index.php
- **W3Schools XSLT Tutorial** - http://www.w3schools.com/xsl
- Norman Walsh – XSL Tutorial - http://nwalsh.com/docs/tutorials/xsl
- Zvon XSLT Tutorial - http://zvon.org/xxl/XSLTutorial/Books/Book1/index.html

3. laboratorijska vježba: Dinamičke stranica Weba i pretraga strukturiranih podataka

Primarni cilj vježbe „Dinamičke stranice Weba i pretraga strukturiranih podataka“ je upoznavanje sa stvaranjem dinamičkih stranica Weba uporabom široko prihvaćenog jezika PHP. Sekundarni cilj je rukovanje strukturiranim podacima za prikaz na stranici Weba, zbog čega je potrebno upoznati se s objektnim modelom dokumenta (DOM) te programskim sučeljem za rad s DOM stablom u jeziku PHP. Za zadovoljenje tog cilja potrebno je usvojiti osnovne radnje s XML dokumentom u obliku DOM, odnosno radnje s podacima korištenjem sučelja DOM, te konverziju podataka iz oblika DOM u oblik XHTML koji se koristi za prikaz u pregledniku. Osnovni postupci pronalaženja strukturiranih podataka upoznaju se kroz metodu pretraživanja podataka u DOM stablu.

Priprema za vježbu

U cilju pripreme za vježbu potrebno je **proučiti osnovne dijelove jezika PHP** s naglaskom na:

- osnove sintakse jezika PHP
- uključivanje koda u jeziku PHP u dokument - oznake <?php … ?>
- osnovne konstrukte jezika PHP – podaci, operatori, upravljačke strukture, iteratori …
- rad s objektima i razredima
- ugrađene funkcije i programsko sučelje jezika PHP – rad s nizovima znakova, poljima …
- metode preuzimanje podataka iz obrazaca
- generiranje izlaza
- izradu vlastitih funkcija

Dodatno, potrebno je **proučiti objektni model dokumenta (DOM)** s naglaskom na:

- strukturu objektnog modela dokumenta (DOM)
- inicijalizaciju DOM stabla iz postojećeg XML dokumenta
- dohvat elemenata iz stabla
- pristup sadržaju elemenata i atributa

Za potrebe adresiranja podataka unutar DOM strukture, potrebno je **nadopuniti znanje o jeziku XPath** iz područja XSL transformacija:

- inicijalizaciju procesora koji izvodi XPath upite
- filtriranje elemenata korištenjem XPath upita
- pristupanje elementima i atributima
- postavljanje uvjeta za pretragu (predikata)
- postavljanje složenih uvjeta korištenjem logičkih operatora

Zadatak za vježbu

Potrebno je **izraditi datoteku u jeziku PHP** koja će služiti za pretraživanje strukturiranog zapisa podataka u XML dokumentu. Rad sa XML dokumentom izvodi se koristeći **programsko sučelje objektnog modela dokumenta (DOM)**. U vježbi se koristi

obrazac za unos parametara za pretraživanje, testni **XML dokument** i **tablični prikaz rezultata pretraživanja u XHTML obliku**, izrađeni u prethodnim vježbama.

Pretraga podataka u XML dokumentu moguće je na dva načina:

- korištenjem **upita u jeziku XPath**, odnosno tzv. **XPath upita** (preporuča se) ili
- **ručnim provjeravanjem** na način da se prolaskom kroz cijelu strukturu podataka provjerava udovoljava li neki podatak uvjetima pretrage.

Napomena: Preporuča se korištenje XPath upita.

Parametre za pretraživanje koji će se koristiti za oblikovanje upita, odnosno provjeru smije li se neki podatak prikazati, treba preuzeti iz obrasca (izvedenog u 1. laboratorijskoj vježbi).

Napomena: Prilikom izrade rješenja vježbe potrebno je podržati pretragu podataka na hrvatskom jeziku, odnosno upite koji sadrže sve hrvatske grafeme, tj. dijakritičke znakove.

Radi lakšeg snalaženja, **funkcije** za **oblikovanje upita, provjeru uvjeta nad elementom** i sve druge pomoćne funkcije potrebno je definirati u **posebnoj datoteci u jeziku PHP**, koja se onda uključuje u glavnu datoteku za pretraživanje.

Pretragu treba omogućiti **barem po svim elementima i atributima do druge razine** u opisu inačica (uključujući drugu razinu). Kod pretraživanja tekstualnih podataka treba uzeti u obzir sve nizove znakova koje sadrže uvjet pretrage, a ne samo one koji su potpuno jednaki uvjetu. To znači da je potrebno pretraživati podniz u nizu. Pretraga bez postavljenih uvjeta treba prikazati sve podatke iz strukture.

Napomena: Pretraživanje podniza u nizu znači da je svaki postavljeni uvjet pretrage zapravo podniz koji se treba pretražiti u svim podacima. Tako se uneseni uvjet zapravo može promatrati kao da je okružen višeznačnicima (*wildcards*), odnosno zamjenskim znakovima % ili * kod zadavanja uvjeta pretrage. Primjer takvog zadavanja uvjeta u sintaksi jezika SQL ili regularnim izrazima je korištenje zapisa: `%podniz%` ili `*podniz*`. Kod pretrage je moguće izvesti i napredno korištenje zamjenskih znakova tj. višeznačnika unutar teksta uvjeta, ali se zbog kompleksnosti izvedbe ne preporuča.

Predaja vježbe

Vježba se predaje prikazivanjem (demonstracijom) dinamičke stranice Weba u pregledniku. Dinamičke stranice Weba izvode se na računalu s poslužiteljem stranica Weba i poslužiteljskim programom/dodatkom za PHP, što može biti na fakultetskom poslužitelju `pinus.cc.fer.hr`, na kojem su instalirani poslužitelj Apache i podrška za PHP, ili na vlastitom prijenosnom računalu. Predaja vježbe podrazumijeva:

- prikaz dokumenta s testnim podacima u XML obliku
- prikaz obrasca za pretragu podataka u dinamičkoj stranici
- pretragu podataka prema proizvoljnim uvjetima uz prikaz rezultata pretrage
- provjeru ispravnosti pretrage korištenjem različitih kombinacija testnih uvjeta
- eventualne naknadne promjene u načinu pretrage i prikazu rezultata

Datoteke koje treba predati u datoteci ZIP oblika i prikazati su sljedećih naziva:

- `pretraga.php`
- `funkcije.php`

U ovoj su vježbi potrebne i druge datoteke iz 1. i 2. laboratorijske vježbe, npr. `podaci.xml`. Na poslužitelj Moodle treba postaviti ZIP arhivu sa svim datotekama potrebnim za izvođenje rješenja vježbe. Datoteke `pretraga.php` i `funkcije.php` moraju se nalaziti u glavnoj mapi ZIP arhive, dok ostale datoteke mogu biti u proizvoljnim mapama.

Priprema okruženja

Preporučeni način izvođenja rješenja vježbe je korištenjem fakultetskog poslužitelja `pinus.cc.fer.hr` i aplikacijske podrške koja ja već instalirana na to računalo. Na fakultetski poslužitelj instalirani su poslužitelj stranica Weba Apache i podrška za PHP za poslužitelj Apache, no nisu pokrenuti do termina u kojem se očekuje početak izrade i testiranja vježbe. Taj termin uobičajeno započinje nakon održanog pripremnog predavanja za laboratorijsku vježbu.

Napomena: Trenutno su na fakultetski poslužitelj `pinus.cc.fer.hr` instalirani poslužitelj Apache inačice 2.2.8 i podrška za PHP inačice 5.2.5. Ako se ukaže potreba za instalacijom novijih inačica, o tome ćete biti obaviješteni kroz vijest na stranici kolegija. No, ova konfiguracija je dovoljna za uspješno izvođenje rješenja vježbe.

Umjesto na fakultetskom poslužitelju, vježba se može predati i na vlastitom prijenosnom računalu. Za alternativnu predaju vježbe na vlastitom prijenosnom računalu potrebno je instalirati poslužitelj stranica Weba i podršku za PHP. Preporučamo korištenje poslužitelja stranica Weba Apache i podršku za PHP za poslužitelj Apache koji su besplatni i slobodni za korištenje.

Većina distribucija operativnog sustava Linux dolazi s paketima za instalaciju poslužitelja Apache s podrškom za PHP. Alternativno, podrška za PHP se može instalirati i u kombinaciji s drugim poslužiteljima, kao što je npr. poslužitelj Microsoft IIS. Instalacija poslužitelja Apache s podrškom za PHP za osobno računalo dolazi i u paketu XAMPP koji je dostupan za različite verzije operativnog sustava Windows i distribucije operativnog sustava Linux. Instalacija paketa XAMPP nalazi se na adresi: http://www.apachefriends.org.

Napomena: Instalacijske opcije podrške za PHP su detaljno objašnjene na adresi: http://hr.php.net/install. Instalaciju na vlastito prijenosno računalo izvode sami studenti, ali isključivo na vlastitu inicijativu i odgovornost, bez podrške od strane nastavnika. No, nastavnici će potencijalno, ako se ukaže potreba i ako budu u mogućnosti, pomoći oko instalacije i podešavanja odgovorima na pitanja i savjetima na forumu stranice kolegija.

Korištenje fakultetskog poslužitelja `pinus.cc.fer.hr`*:*

Fakultetski poslužitelj `pinus.cc.fer.hr` (skraćeno `pinus`) je računalo s operativnim sustavom Unix. Na računalo `pinus` se prijavljujete **korištenjem korisničkog imena i lozinke** koju koristite i za sva druga domenska računala u laboratorijima. Za prijavu putem terminala možete koristiti terminal PuTTY sa SSH vezom.

Napomena: Za probleme s prijavom na računalo `pinus`, kao što je zaboravljena lozinka, obratite se CIP-u FER-a.

Nakon prijave, u osnovnom korisničkom direktoriju (kazalu, mapi) naziva `~korisnicko_ime` (naziv direktorija za korisničko ime `ab12345` bi bio `~ab12345`) treba stvoriti direktorij naziva `public_html`, unutar kojeg se stavljaju sve datoteke stranica Weba. Direktorij se može stvoriti naredbom `mkdir`:

```
$> mkdir public_html
```

Prava pristupa na osnovnom korisničkom direktoriju, direktoriju `public_html` i svim ostalim direktorijima i datotekama koji trebaju biti dostupni procesu poslužitelja Weba trebaju biti:

- Nad direktorijima – čitanje i izvođenje (pristup direktorijima) za sve te pisanje samo za korisnika:
  - dozvola 755 ili za korisnika: rwx, za grupu: r-x, za ostale: r-x
- Nad datotekama – čitanje za sve te pisanje samo za korisnika:
  - dozvola 644 ili za korisnika: rw-, za grupu: r--, za ostale: r--

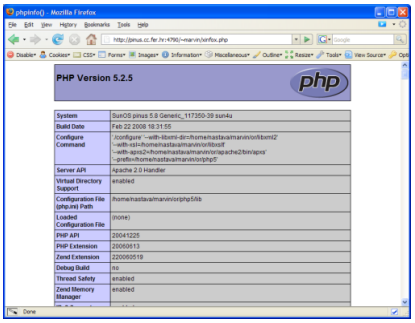
Dozvole se mogu postaviti sljedećim naredbama:

```
$> chmod 755 ~
$> chmod 755 public_html
$> chmod 644 public_html/index.html
```

Pristup datotekama najlakše je ostvariti programom koji podržava tzv. Secure Copy (SCP), kao što je WinSCP, dohvatljiv na adresi http://winscp.net. WinSCP možete podesiti tako da koristi vanjski uređivač datoteka, primjerice isti koji ste koristili u prethodnim vježbama, te tako možete jednostavnije uređivati datoteke pohranjene na poslužitelju.

Pristup stranicama Weba na poslužitelju `pinus` u vlastitom sjedištu određenom direktorijem `public_html` moguć je protokolom HTTP na vratima (portu) 4790, kako je prikazano: http://pinus.cc.fer.hr:4790/~korisnicko_ime/. Za korisnika ab12345 ta adresa bi glasila: http://pinus.cc.fer.hr:4790/~ab12345/.

Napomena: Poslužitelj Weba na računalu `pinus` aktivan je samo za potrebe laboratorijskih vježbi, pa nema smisla isprobavati rješenja u terminima kad je isključen. Nastavnici će vas obavijestiti o točnom terminu dostupnosti poslužitelja putem vijesti na stranici kolegija. U slučaju problema, kao što je greška 403 (*Forbidden*), prvo je potrebno provjeriti dozvole čitanja i izvođenja (pristupa) na osnovnom direktoriju, direktoriju `public_html` i na svim datotekama unutar `public_html` direktorija.



Izrada PHP skripte

Za provjeru radi li PHP ispravno, može se u direktoriju `public_html` stvoriti jednostavna PHP skriptu `info.php` sljedećeg sadržaja:

```
<?php phpinfo(); ?>
```

Ako se preglednikom pristupi stranici koja pokreće PHP skriptu dobiva se informacija o okruženju PHP. Adresa stranice bi trebala biti oblika http://pinus.cc.fer.hr:4790/~korisnicko_ime/info.php a rezultat bi trebao biti kao na slici.

U rješenju se PHP skripta za pretraživanje treba pozivati iz obrasca za postavljanje uvjeta pretrage, što se definira atributom `action` u početnoj oznaci `form` obrasca:

```
<form action="pretraga.php" ...>
```

Podaci uneseni u obrazac i poslani na poslužitelj, u PHP skripti su dostupni u globalnim varijablama `$_GET` ili `$_POST`, u ovisnosti o tome koja metoda se koristi za prijenos podataka iz obrasca, odnosno globalnoj varijabli `$_REQUEST`, neovisno o metodi prijenosa.

Primjer: Ako u obrascu postoji tekstualno polje naziva „ime“, tekstualni sadržaj upisan u to polje bit će dohvatljiv u PHP skripti na sljedeći način: `$_REQUEST['ime']`

Za provjeru postoji li neka varijabla (podatak) i ima li postavljenu vrijednost može se koristiti jedna od sljedećih naredbi:

```
isset( $varijabla );
empty( $varijabla );
```

Primjer: Provjera postojanja sadržaja globalne varijable „ime“ izvela bi se ovako:

```
if( !empty( $_REQUEST['ime'])
```

Prilikom izvođenja PHP skripte ne prijavljuju se sve greške, već se prijava grešaka uključuje naredbom `error_reporting` kako slijedi:

```
error_reporting( E_ALL );
```

Uključivanje vanjske datoteke, kao što je primjerice datoteka s opisom funkcija, izvodi se naredbom `include` ili `include_once` na sljedeći način:

```
include( 'funkcije.php' );
include_once( 'funkcije.php' );
```

Za ispis vrijednosti varijable mogu se koristiti naredbe `print_r` i `var_dump`:

```
print_r( $_REQUEST );
var_dump( $_REQUEST );
```

Polja s višestrukim odabirom vrijednosti

Za polja u obrascu koja mogu rezultirati višestrukim vrijednostima, kao što je polje izbornika za višestruki odabir (*multiple select*) izvedenom oznakom `<SELECT multiple="multiple" …>` ili niz kvadratića za izbor (*checkbox*) izvedeno oznakama `<INPUT type="checkbox" …>`, podaci se za prijenos kodiraju u obliku **niza parova** `ime=vrijednost`.

Ovo ćemo najbolje objasniti primjerom za kvadratiće za izbor veličine (S, M, L)

```
<input type="checkbox" name="velicina" value="S" />
<input type="checkbox" name="velicina" value="M" />
<input type="checkbox" name="velicina" value="L" />
```

Odabirom više mogućnosti iz liste, npr. S i M, podaci će se prenijeti u obliku:

```
velicina=S&velicina=M
```

Prilikom obrade zahtjeva stvorit će se unos `$_REQUEST['velicina']` tipa `string`, koji će se prvo postaviti na vrijednost "S", a odmah zatim na vrijednost "M". Izvorni upit nije izgubljen i ručnom obradom upita se može doći do željenih podataka, ali time se gube prednosti obrade u PHP skripti.

Stoga se polju u obrascu `velicina` treba dati naziv koje sadrži oznake polja `[]`:

```
<input type="checkbox" name="velicina[]" value="S" />
<input type="checkbox" name="velicina[]" value="M" />
<input type="checkbox" name="velicina[]" value="L" />
```

Podaci se tada prenose u obliku:

```
velicina[]=S&velicina[]=M
```

Prilikom obrade zahtjeva, stvara se unos `$_REQUEST['velicina']` tipa polja (`array()`) koje se popunjava vrijednostima odabranim u obrascu:

```
velicina[0] = "S";
velicina[1] = "M";
```

Rezultat je polje veličine 2 s dva podatka:

```
array(2) { [0]=> string(1) "S" [1]=> string(1) "M" }
```

Podacima se pristupa na isti način kao i skalarnim parametrima upita:

```
print_r( $_REQUEST['velicina'] );
```

Napomena: U obradi parametara može se služiti petljom `foreach`, funkcijom `implode` itd.

Objektni model dokument i upit XPath

Inicijalizacija procesora za objektni model dokumenta DOM koji učitava XML dokument u obliku DOM stabla može se izvesti ovako:

```
$dom = new DOMDocument();
$dom->load( 'podaci.xml' );
```

Podelementima nekog čvora pristupa se na sljedeći način:

```
foreach( $dom->childNodes as $element ) {
   ...
}
```

Pristupanje podelementima nekog čvora prema oznaci može se izvesti kao u sljedećem primjeru za podelemente s oznakom `element`:

```
foreach( $dom->getElementsByTagName( 'element' ) as $element ) {
   ...
}
```

Atributima se pristupa na sličan način, predstavljen u sljedećem primjeru:

```
$element->getAttribute( 'atribut' );
```

Inicijalizacija XPath procesora i postavljanje upita može se izvesti ovako:

```
$xp = new DOMXPath( $dom );
$rezultat = $xp->query( "/korijen/element" );
```

Funkcija za stvaranje XPath upita treba provjeriti koji su sve parametri za pretraživanje zadani i formirati XPath upit na osnovu njih. Zbog izrade složenih upita preporučamo dodavanje elemenata upita u polje:

```
if( !empty( $parametri['boja'] ))
   $upit[] = "boja='" . $parametri['boja'] . "'";
```

Jednom formirano polje pretvara u oblik tipa `string` funkcijom `implode` koja spaja članove polja u niz znakova:

```
$xpath_upit = implode( " and ", $upit );
```

Rezultati upita pohranjeni su u objekt `DOMNodeList`, a pojedinim elementima može se pristupiti petljom `foreach, kako slijedi`:

```
foreach( $rezultat as $cvor ) {
   ...
}
```

Postojeću XSL datoteku s opisom transformacije XML dokumenta možete iskoristiti kao predložak za generiranje rezultata PHP skripte. Tako se iteriranje po podelementima nekog

elementa može izvesti petljom naredbom `foreach`, gdje se XSL oznaka `<xsl:for-each>` zamjenjuje naredbom `foreach`:

```
<xsl:for-each select="element"> ... </xsl:for-each>

foreach( $lista as $element ){
   ...
}
```

Dohvaćanje sadržaja trenutnog elementa iz strukture izvodi se korištenjem svojstva `nodeValue`, a naziva korištenjem svojstva `nodeName`:

```
<xsl:value-of select="." />

$element->nodeValue;
$element->nodeName;
```

Dohvaćanje podelementa moguće je prema imenu oznake ili identifikatoru (ID-u), pa se tako, npr. za podelement *boja* koji se pojavljuje samo jednom, može koristiti metoda `item(0)` koja dohvaća prvi element:

```
$element->getElementsByTagName( 'boja' )->item(0)->nodeValue;
```

Kao i u drugim jezicima, unutar iteratora (`foreach`), nije dozvoljeno mijenjanje sadržaja kolekcije po kojoj se iterira, jer su tada mogući neočekivani rezultati. Stoga, ako se iterira po listi čvorova, nije dozvoljeno istovremeno i brisati ih iz liste:

```
foreach($roditelj->getElementsByTagName( 'boja' ) as $cvor) {
   $roditelj->removeChild( $cvor );
}
```

Ako je iznimno potrebno brisanje, može se upotrijebiti posebno polje u koje se pohranjuju svi elementi za brisanje i naknadno se obrišu u posebnoj petlji:

```
$lista = array();
foreach($roditelj->getElementsByTagName( 'boja' ) as $cvor)
   if( treba_brisati( $cvor ))
      $lista[] = $cvor;              // dodaj u listu za brisanje
foreach($lista as $brisi)
   $roditelj->removeChild( $brisi ); // obrisi
```

Ispitno gradivo vježbe

Ispitno gradivo uključuje sve navedeno u pripremi za vježbu te detaljno razumijevanje izrađenog rješenja i snalaženja u prepravcima istog.

Primjeri pitanja:

- Što je to PHP, što DOM, a što XPath?
- Kako se PHP kod uključuje u HTML dokument?
- Koje još programske jezike za aplikacije Weba koji se izvršavaju na strani poslužitelja poznajete?

- Što se događa kada korisnik pošalje zahtjev za stranicu koja ima sufiks ".php"?
- Kako se razdvajaju dijelovi HTML i PHP koda u istoj datoteci?
- Koje sve postavke moraju biti prisutne da bi korisnik mogao postaviti PHP skriptu u svoj direktorij `public_html`?
- Kako preglednik šalje podatke iz obrasca na poslužitelj i kako se oni prihvaćaju?
- Koje sve načini prijenosa parametara između preglednika i PHP skripte poznajete i koje su razlike u mehanizmima koji se koriste?
- Koji se protokol koristi za prijenos parametara između obrasca u pregledniku i PHP skripte te kako se kodiraju podaci za prijenos?
- Na što treba paziti prilikom imenovanja elemenata obrasca?
- Kako se struktura XML dokumenta odražava u modelu DOM?
- Kako se pristupa pojedinim elementima u modelu DOM?
- Koja je sličnost XSL oznaka za filtriranje i DOM-a kako je korišten u PHP-u?
- Usporedite XPath i jezike za rad s podacima (npr. SQL)? Kako bi se kod u PHP-u prilagodio korištenju SQL-a?
- Koju bi strategiju pisanja ovog primjera odabrali, kada bi ovi podaci bili u nekoj od baza podataka, umjesto u XML datoteci?

Poveznice i literatura

- **W3Schools XPath Tutorial** - http://www.w3schools.com/Xpath
- Službene stranice PHP-a - http://www.php.net
- **Dokumentacija za PHP -** http://www.php.net/manual/en/language.variables.php
- **Dokumentacija za PHP – polja** - http://www.php.net/manual/en/ref.array.php
- **Dokumentacija za PHP – String -** http://www.php.net/manual/en/ref.strings.php
- **Dokumentacija za PHP – DOM** - http://www.php.net/manual/en/ref.dom.php
- XAMPP - http://www.apachefriends.org

4. laboratorijska vježba: Stvaranje strukture podataka iz vanjskog izvora

Cilj vježbe „Stvaranje strukture podataka iz vanjskog izvora“ je učitavanje podataka iz vanjskog izvora kao što je tekstualna datoteka (ili baza podataka), formiranje objektnog modela podataka u radnoj memoriji te zapisivanje tih podataka u strukturiranom obliku u XML datoteku. U sklopu vježbe upoznaje se s programiranjem u programskom jeziku Java i tako se neizravno savladavaju i osnove objektno orijentiranog dizajna aplikacija. Naglasak se stavlja na uporabu podataka iz vanjskog izvora podatka u smislu učitavanja podataka iz tekstualne datoteke, uporaba kolekcija objekata te posljedično stvaranje podatkovne strukture i njena pohrana u obliku XML.

Priprema za vježbu

U cilju pripreme za vježbu potrebno je **proučiti osnovne dijelove programskog jezika Java** s naglaskom na:

- pisanje i prevođenje koda
  - u naredbenom retku ili
  - korištenjem razvojnog okruženja npr. Eclipse, NetBeans ...
- osnove programskog jezika Java – tipovi podataka, operatori, izrazi, razredi, objekti, paketi …
- osnovne ulazno/izlazne operacije s datotekama

Zadatak za vježbu

Potrebno je izraditi samostalnu (*standalone*) **aplikaciju** u programskom jeziku Java koja će **učitavati podatke iz tekstualne datoteke**, formirati **model podataka** u radnoj memoriji i zatim te iste podatke **zapisivati u XML datoteku**.

Oblik ulazne datoteke

Svaki pojedini redak datoteke (ako nije prazan ili ne započinje znakom #) označava zapis jednog podatkovnog objekta tj. entiteta, kao što je npr. student u popisu studenata. Retci koji započinju znakom # su komentari. Primjer oblika ulazne tekstualne datoteke nalazi se u pokaznoj inačici rješenja laboratorijske vježbe u datoteci `data.txt`.

Pojedini podaci objektu, odnosno entiteta odvojeni su znakom za odvajanje (*delimiter*), u ovom slučaju znakom | (uspravna crta).

Ugniježđeni podaci prikazuju se u obliku liste kao što su npr. popis predmeta studenta. Oni su također od ostalih podataka odvojeni istim znakom za odvajanje |, dok su pojedini elementi unutar liste odvojeni znakom *;* (točka zarez).

Dodatno, ako su elementi liste složeni podaci koji se sastoje od više atomičnih podataka, tada su pojedini atomični podaci odvojeni znakovima za odvajanje *,* (zarez).

Iz navedenog se može zaključiti da imamo tri znaka za odvajanje:

- | (uspravnu crtu)
- ; (točka zarez)

- , (zarez)

Za bolje razumijevanje, kao primjer oblika zapisa navodimo jedan zapis o pametnoj kartici:

```
8745884582322454342 | 0036123456 | Vlatko | Pokos | SXAA | 1 | 110930
 | 1,korisnik,01,0000 ; 2,e-indeks,01,0000 ; 3,menza,01,0001 |
```

Očigledno, zapis sadrži osam podataka odvojenih znakovima | (uspravna crta). Zadnji podatak je složeno polje, tj. lista koja se sastoji od tri elementa. Elementi liste odvojeni su znakom ; (točka zarez). Svaki element liste je složeni podatak koji se sastoji od četiri atomična podatka odvojenih znakom , (zarez).

Izrada ulazne tekstualne datoteke

Kao prvi dio zadatka potrebno je za zadanu inačicu izraditi ulaznu tekstualnu datoteku prema opisanom obliku zapisa podataka. Ulazna datoteka mora sadržavati minimalno 15 podataka u svakom zapisu, odnosno o svakom entitetu ovisno o inačici koja se rješava, te mora sadržavati podatke o minimalno 10 zapisa, odnosno entiteta (npr. 10 studenata u popisu studenata s po 15 pripadnih podataka).

Izrada podatkovne strukture razreda

Nakon što je izrađena ulazna datoteka, potrebno je kreirati strukturu razreda, odnosno **definirati razrede** za sljedeće elemente:

- **korijenski element** koji sadržava sve osnovne elemente dokumenta (npr. razred `CardList`, koji sadrži zapise o pametnim karticama)
- **osnovni element** (npr. razred `Card`)
- **svaki element brojnosti N** u opisu inačica određen stupcem „Brojnost“ (to su oni elementi iz inačice XML zapisa koji se mogu pojaviti više puta, npr. razred `Service`)

Za sve definirane razrede potrebno je **izraditi tzv. *getter/setter* metode**, odnosno metode `setX()` i `getX()`, gdje su `X` jednostavna svojstva entiteta kojeg reprezentiraju (npr. za `Card` to su ID, JMBAG, ime korisnika, prezime korisnika, lista usluga itd.).

Za složena svojstva (npr. listu usluga) potrebno je implementirati tri metode: `addX()`, `removeX()` i `Xs()` – npr. razred `Card` definira metode `void addService(Service)`, `void removeService(Service)` i `Iterator<Service> services()`. Definirani razredi moraju se nalaziti u paketu (*package*) `model`.

Izrada factory *razreda*

Kada su stvoreni razredi i njihove metode, potrebno je definirati tzv. ***factory* razrede** koji posjeduju logiku za parsiranje tekstualnog zapisa i stvaranje objekta određenog razreda, te za pohranu objekta u obliku XML zapisa. Navedeni razredi se moraju zvati isto kao i razredi podatkovnog modela čije objekte znaju stvarati i zapisivati. Razredi trebaju biti smješteni u paket `model.factory`. Ti razredi trebaju implementirati dvije statičke metode:

- metodu `fromText()` koja vraća objekt razreda iz tekstualnog zapisa

- metodu `toXML()` koja zapisuje stanja objekta u obliku XML formatu

Na primjer, u pokaznoj inačici zadatka za razred `model.Card` postoji razred istog naziva `model.factory.Card` koji ima dvije statičke metode, `fromText()` koja vraća objekt razreda `model.Card` iz tekstualnog zapisa i `toXML()` koja zapisuje stanja objekta u obliku XML.

Za parsiranje podataka iz ulazne tekstualne datoteke treba koristiti razred `java.util.StringTokenizer`. Kod stvaranja XML zapisa treba paziti kako se pojedini atribut objekta preslikava u XML, kao element ili kao atribut, a što je dano u opisu inačica.

Izrada razreda Konverter

Na kraju treba izraditi razred `Konverter` koji će sadržavati samo metodu `main()`. U metodi `main()` treba otvoriti dvije datoteke čija imena su zadana kao parametri komandne linije. Nazivi datoteka su proizvoljni, a predlažemo `in.txt` i `out.xml`.

Objedinjavanje koda u aplikaciju

Na osnovu podataka iz ulazne tekstualne datoteke, skupina razreda iz paketa `model.factory` treba stvoriti ekvivalentnu strukturu podataka u memoriji koristeći razrede iz paketa `model`. Nakon toga, koristeći skupinu razreda iz paketa `model.factory`, treba stvoriti XML datoteku koja reprezentira prethodno spomenutu strukturu podataka.

Napomena: Detalje o mogućem rješenju svih dijelova zadatka potražite u pokaznoj inačici vježbe u repozitoriju na stranici kolegija.

Predaja vježbe

Vježba se predaje demonstracijom na fakultetskom poslužitelju `pinus.cc.fer.hr`, na lokalnom računalu ili na vlastitom prijenosnom računalu. Predaja vježbe podrazumijeva:

- predočavanje datoteka s izvornim kodom uz objašnjenja
- pokretanje aplikacije sa zadanim izvornim podacima
- provjeru valjanosti generirane XML datoteke gramatikom (iz 2. laboratorijske vježbe)
- opcionalne promjene u programskom kodu

Datoteke koje treba predati u datoteci ZIP oblika i prikazati su nužno:

- `Konverter.java`
- `<imeOsnovnogElementa>.java`
- `<imePotomkaOsnovnogElementa>.java` (jedna ili više, ovisno o inačici)
- `<imeElementaBrojnostiN>.java` (jedna ili više, ovisno o inačici)

U ovoj će vježbi biti potrebne i druge datoteke (npr. gramatika iz 2. laboratorijske vježbe). Na poslužitelj Moodle treba postaviti ZIP arhivu sa svim datotekama potrebnima za izvođenje i provjeru, pri čemu se treba pridržavati pravila postavljanja datoteka u pakete.

Ispitno gradivo vježbe

Ispitno gradivo uključuje sve navedeno u pripremi za vježbu, te detaljno razumijevanje izrađenog rješenja i snalaženja u prepravcima istog.

Primjeri pitanja:

- Skicirajte dijagram razreda svoje inačice.
- Validirajte izlaznu XML datoteku.
- Koja metoda se poziva prilikom instanciranja objekta, ako razred nema definiran konstruktor?
- Koje metode su dostupne za skup objekata koji ste koristili?
- Kako iterirate kroz objekte u vašem skupu?

Poveznice i literatura

- **Java osnovna web stranica –** http://java.sun.com
- **J2SE API i dokumentacija** - http://java.sun.com/javase/reference/api.jsp
- **Java Tutorial** - http://java.sun.com/docs/books/tutorial/
- **Eclipse** – http://www.eclipse.org

5. laboratorijska vježba: Stranica Weba s udaljenim izvorom podataka

Cilj vježbe „Stranica Weba s udaljenim izvorom podataka“ je izrada dinamičke stranice Weba koja dohvaća strukturirane podatke u obliku XML isporučene od strane udaljenog poslužiteljskog programa. Poslužiteljski program izvedene je iz gotove aplikacije iz prethodne laboratorijske vježbe, njenom transformacijom u Servlet.

Priprema za vježbu

U cilju pripreme za vježbu potrebno je **proučiti osnove izrade i funkcioniranja Servleta u programskom jeziku Java** s naglaskom na:

- osnove izrade Java Servleta - preporučena literatura:
  - Java Servlet Programming (O'Reilly) – poglavlja 1 do 5 http://www.unix.com.ua/orelly/java-ent/servlet/index.htm
  - The Java EE 5 Tutorial - Java Servlet Technology, Chapter 4 – dijelovi koji se odnose na vježbu (stvaranje, inicijalizacija, odgovor) http://download.oracle.com/javaee/5/tutorial/doc/bnafd.html
- konfiguriranje datoteke `web.xml` za Servlete
- pokretanje Servleta na poslužitelju

Zadatak za vježbu

Potrebno je izraditi **Servlet** u programskom jeziku Java koji **isporučuje podatke u obliku XML**. Java Servlet treba izraditi na osnovu gotove aplikacije iz 4. laboratorijske vježbe koju je potrebno preraditi u skladu sa specifikacijom Servleta kako bi odgovarala na zahtjeve metoda protokola HTTP.

Osim toga, potrebno je koristeći rješenje 3. laboratorijske vježbe **doraditi PHP skriptu** za pretraživanje strukturiranog zapisa podataka u XML obliku koje se dobiva od Java Servleta (a ne iz XML datoteke).

Java Servlet treba na poziv **metodom GET zahtjeva protokola HTTP dohvatiti podatke** iz tekstualne datoteke**, transformirati ih u oblik XML** (što je pripremljeno u 4. laboratorijskoj vježbi), te ih u XML obliku **proslijediti PHP skripti** (bez bilo kakvog dodatnog filtriranja/sortiranja podataka).

Napomena: Ako niste primjereno ili potpuno riješili zadatak iz 4. laboratorijske vježbe, sada ga trebate doraditi s obzirom da se većinom isti programski kod koristi za dohvat podataka iz tekstualne datoteke i prosljeđivanje PHP skripti u ovoj vježbi.

S obzirom da Servlet treba odgovarati na zahtjev metodom GET protokola HTTP, potrebno je **izraditi metodu `doGet()`.**

Napomena: Za izradu metode `doGet()` možete iskoristiti programski kod iz prije implementiranog razreda `Konverter`, odnosno njegove metode `main()`.

U metodi `doGet()` treba (jednako kao i prije) otvoriti ulaznu datoteku s podacima. **Naziv datoteke s podacima** zadan je **kao parametar konfiguracije Servleta** i zapisan u `web.xml` datoteci, koji se učitava u metodi `init()` Servleta (a ne više kao parametar komandne linije kako je bilo u 4. laboratorijskoj vježbi).

Na osnovu učitanih podataka iz tekstualne datoteke, metoda `doGet()` (isto kao i metoda `main()` u 4. laboratorijskoj vježbi) treba stvoriti ekvivalentnu strukturu podataka u memoriji koristeći razrede iz paketa `model.factory`, a nakon toga koristeći istu skupinu razreda iz paketa `model.factory` stvoriti XML datoteku koja reprezentira prethodno spomenutu strukturu podataka, uključujući korijenski element.

Stvorenu XML strukturu podataka, odnosno dokument, potrebno je vratiti PHP skripti koja je poslala zahtjev Servletu.

Predaja domaće zadaće i predaja vježbe

Vježbu je moguće predati na fakultetskom poslužitelju `pinus.cc.fer.hr` korištenjem aplikacijskog poslužitelja Tomcat ili alternativno na vlastitom prijenosnom računalu. Aplikacijski poslužitelj Tomcat je poslužitelj otvorenog koda za tehnologije Servlet i JavaServer Pages instaliran na računalo `pinus`, a inicijalno ga pokreće nastavnik te ga po potrebi ponovno pokreće (*restart*).

Ova vježba ima domaću zadaću koju je potrebno izraditi do određenog roka koji je zadan u vijesti na stranici kolegija. Predaja domaće zadaće podrazumijeva:

- stvaranje odgovarajuće strukture direktorija na poslužitelju
- stvaranje konfiguracijske datoteke `web.xml` i postavljanje u odgovarajući direktorij na poslužitelju
- opcionalno stvaranje testne datoteke `test.jsp` i postavljanje u odgovarajući direktorij na poslužitelju

Napomena: Predaja domaće zadaće nije obvezna ako se vježba predaje na vlastitom prijenosnom računalu. Instalaciju poslužitelja za tehnologije Servlet i JavaServer Pages na vlastito prijenosno računalo izvode sami studenti, ali isključivo na vlastitu inicijativu i odgovornost, bez podrške od strane nastavnika. No, nastavnici će potencijalno, ako se ukaže potreba i ako budu u mogućnosti, pomoći oko instalacije i podešavanja odgovorima na pitanja i savjetima na forumu stranice kolegija.

Nakon što su predane sve domaće zadaće (osim onih studenata koji predaju na vlastitom računalu), asistent će ponovno pokrenuti (*restart*) poslužitelj kako bi se osvježile sve konfiguracije (objašnjeno kasnije).

Predaja vježbe podrazumijeva:

- predočavanje datoteka s izvornim kodom uz objašnjenja
- pokretanje Servleta ručnim pokretanjem dohvata iz preglednika metodom GET protokola HTTP
- pokretanje PHP skripte koja koristi XML dokument dobiven od Servleta

Napomena: Aplikacijski poslužitelj Tomcat na računalu `pinus` aktivan je samo za potrebe laboratorijskih vježbi, pa nema smisla isprobavati rješenja u terminima kad je isključen. Nastavnici će vas obavijestiti o točnom terminu dostupnosti poslužitelja putem vijesti na stranici kolegija. U slučaju problema, kao što je greška 403 (Forbidden), prvo je potrebno provjeriti dozvole čitanja i izvršavanja na direktorijima i datotekama.

Datoteke koje treba predati u datoteci ZIP oblika su:

- datoteke iz direktorija `web-app` uključivo Servlet, konfiguracijske datoteke, itd. (kopija direktorija `web-app` s poslužitelja), a ovdje treba biti i direktorij `src` s izvornim kodom,
- datoteke iz direktorija `public_html` uključujući PHP skriptu i sve prateće datoteke (kopija direktorija `public_html` s poslužitelja).

Konfiguracija aplikacijskog poslužitelja

Da bi rješenje vježbe proradilo potrebno je konfigurirati aplikacijski poslužitelj Tomcat za korištenje odgovarajućeg Servleta. Za to je potrebno u osobnom direktoriju **na poslužitelju** `pinus` **kreirati** sljedeću strukturu poddirektorija:

```
web-app
   /WEB-INF
      /classes     ← ovdje idu .class datoteke (sa strukturom paketa)
      /lib         ← ovdje idu .jar datoteke
      /src         ← ovdje idu .java datoteke s izvornim kodom
```

Direktorij `web-app` sadrži direktori `WEB-INF` koji sadrži još nekoliko poddirektorija. Direktorij `WEB-INF` sadrži konfiguraciju aplikacije, a u poddirektoriju `classes` trebaju se nalaziti `.class` datoteke potrebne za rad aplikacije. Ako ste pakirali aplikaciju u arhivu JAR, kreirajte i poddirektorij `lib` unutar `WEB-INF` direktorija i tamo stavite `.jar` datoteku.

Napomena: U slučaju problema, možete otpakirati `.jar` datoteku i razrede postaviti u direktorij `classes`.

Za **test** da li ste dobro izradili strukturu direktorija i može li poslužitelj Tomcat pristupiti datotekama, u direktoriju `web-app` možete postaviti datoteku naziva `test.jsp` sljedećeg sadržaja, kao probu ispravnosti rada poslužitelja:

```
<HTML><BODY>
Vrijeme na posluzitelju: <%= new java.util.Date() %>
</BODY></HTML>
```

Da bi Servlet bio dostupan, potrebno je **u direktorij** `WEB-INF` **postaviti konfiguracijsku datoteku** `web.xml` koja konfigurira aplikaciju Weba (u ovom slučaju Servlet) u aplikacijskom poslužitelju. Ta datoteka treba sadržavati opis Servleta, vezu na odgovarajući razred i mapiranje Servleta na određeni URL.

Sadržaj navedene datoteke `web.xml` bi trebao biti sličan sljedećem (podcrtane dijelove nadopunite proizvoljnim nazivima datoteke, razreda Servleta, samog naziva Servleta, parametra i mapiranja):

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE web-app PUBLIC
"-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
"http://java.sun.com/j2ee/dtds/web-app_2_3.dtd">
<web-app>
   <servlet>
      <servlet-name>NazivMojegServleta</servlet-name>
      <servlet-class>hr.fer.rasip.or.MojServlet</servlet-class>
         <init-param>
            <param-name>InputFile</param-name>
            <param-value>in.txt</param-value>
         </init-param>
   </servlet>
   <servlet-mapping>
      <servlet-name>NazivMojegServleta</servlet-name>
      <url-pattern>/servlet</url-pattern>
   </servlet-mapping>
</web-app>
```

Nakon postavljanja konfiguracije u datoteci `web.xml`, potrebno je ponovno pokrenuti (*restart*) aplikacijski poslužitelj

Napomena: **Ponovno pokretanje** aplikacijskog poslužitelja ne izvodite sami, već će se to obavljati **barem nekoliko puta dnevno od strane nastavnika**. Sve probleme možete prijaviti na forum stranice predmeta.

Jednom podešena (konfigurirana) aplikacija Weba u obliku Servleta može se osvježavati. Ako je potrebno promijeniti sadržaj razreda mogu se stavljati nove inačice istih razreda u direktorij `classes` bez ponovnog pokretanja poslužitelja. No, ako je promijenjena konfiguracija, odnosno datoteka `web.xml`, ili ako su promijenjeni nazivi razreda potrebno je **ponovno pokretanje** aplikacijskog poslužitelja (od strane nastavnika).

Osvježavanje aplikacije Weba (Servleta)

Ako nije promijenjena konfiguracija ili nazivi razreda, studenti mogu samostalno osvježiti aplikaciju, čemu **služi stranica za osvježavanje** aplikacije Weba, koja se nalazi na adresi http://www.fer.hr/predmet/or/upravljanje_web_aplikacijom. Za osvježavanje aplikacije potrebno je prijaviti se na sjedište Weba FER-a, nakon čega bi sustav trebao prikazati korisnički račun na poslužitelju `pinus`.

Napomena: Ako oznaka računa nije ispravna, možete se javiti nastavnicima putem foruma.

Nakon promjene stanja aplikacije gumbom **List** možete u popisu podešenih aplikacija provjeriti stanje svoje aplikacije. Veza *Poveznica na aplikaciju* vodi do direktorija s aplikacijom na aplikacijskom poslužitelju Tomcat.

Napomena: Ako dobijete grešku poslužitelja (kôd greške 5xx), aplikacija nije primjereno podešena unutar poslužitelja Tomcat.

Ako je ispravno izrađena testna datoteku `test.jsp` prema gornjem primjeru, pristupom URL http://pinus.cc.fer.hr:8080/~korisnicko_ime/test.jsp, treba se dobiti stranica s ispisom trenutnog vremena poslužitelja.

```
Vrijeme na posluzitelju: Mon May 25 10:31:14 MEST 2009
```

Servletu se pristupa preko adrese http://pinus.cc.fer.hr:8080/~korisnicko_ime/servlet, odnosno u skladu s vrijednosti parametra `<url-pattern>` u konfiguracijskoj datoteci `web.xml`.

Napomena: Zbog hirova mrežnih administratora, u nekim mrežama blokiran je pristup vratima (portu) 8080. U tom slučaju, pokušajte s vratima (portom) 12632.

Iako to ovdje nije nužno, aplikacija naravno može sadržavati i više Servleta, pa se tada u datoteci `web.xml` navode više puta elementi `<servlet>` i `<servlet-mapping>` za svaki Servlet unutar elementa `<web-app>`, kao u sljedećem primjeru:

```
...
<web-app>
   <servlet>
      <servlet-name>ImeMojegServleta</servlet-name>
      <servlet-class>hr.fer.rasip.or.RazredMogServleta</servlet-class>
   </servlet>
   <servlet>
      <servlet-name>ImeDrugogServleta</servlet-name>
      <servlet-class>hr.fer.rasip.or.RazredTogServleta</servlet-class>
   </servlet>
   <servlet-mapping>
      <servlet-name>ImeMojegServleta</servlet-name>
      <url-pattern>/mojservlet</url-pattern>
   </servlet-mapping>
   <servlet-mapping>
      <servlet-name>ImeDrugogServleta</servlet-name>
      <url-pattern>/drugiservlet</url-pattern>
   </servlet-mapping>
</web-app>
```

Izrada razreda Servleta

Razred Servleta treba naslijediti razred `HttpServlet` kao u sljedećem primjeru:

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;

public class MyServlet extends HttpServlet {
 …
```

Servlet ne treba prihvaćati nikakve parametre zahtjeva, niti ih preuzimati iz dobivenog zahtjeva (`HttpServletRequest`), ali treba učitati parametar naziva ulazne datoteke s podacima iz konfiguracije Servleta u datoteci `web.xml`, slično kao u primjeru:

```
String filename = getServletConfig().getInitParameter("InputFile");
```

Servlet treba u metodi `init()` definirati naziv datoteke iz koje se učitavaju podaci (a koja je učitana kao parametar iz konfiguracije Servleta u datoteci `web.xml`), slično kao u primjeru:

```
String inputFilename;
public void init(ServletConfig config) throws ServletException {
   super.init(config);
   inputFilename = ...     //definicija naziva ulazne datoteke
}
```

Servlet nije samostalna aplikacija te datoteku ne može otvoriti na isti način kao u aplikaciji, jer put do datoteke nije izravno poznat. U ovom slučaju datoteku treba dohvatiti kao resurs, a putanje resursa Servleta relativne su u odnosu na kontekst Servleta, odnosno direktorij `web-app`. Sljedeći primjer pokazuje otvaranje resursa kao toka podataka. Put do resursa relativan je u odnosu na kontekst aplikacije Weba.

```
getServletConfig().getServletContext().getResourceAsStream(filename);
```

Servlet vraća podatke u obliku XML dokumenta, a povratni tip MIME za oblik XML se postavlja pomoću metode `setContentType` unutar metode `doGet()` kao u primjeru:

```
public void doGet(HttpServletRequest request,
 HttpServletResponse response) throws ServletException, IOException {
...
   response.setContentType("text/xml; charset=UTF-8");
...
```

Također je potrebno dohvatiti objekt `PrintWriter` nad odgovorom `HttpServletResponse` kako bi se mogli „ispisivati“ podaci, a na kraju je potrebno zatvoriti tok podataka, kao u primjeru:

```
PrintWriter out = response.getWriter();
...
// primjer korištenja objekta PrintWriter za ispis u Servletu
out.println("<?xml version=\"1.0\" encoding=\"utf-8\"
 standalone=\"no\"?>");
...
out.close();
```

Napomena: Budući da nasljeđuje razred `Writer`, navedeni objekt `PrintWriter` se može koristiti prilikom poziva `factory` razreda iz 4. laboratorijske vježbe.

Razredi kao što su `HttpServlet`, `ServletException` i slični, definirani su u programskom sučelju Servleta (Servlet API). Za prevođenje programskog koda potrebno je

dohvatiti biblioteka `servlet-api.jar` koje se nalazi u repozitoriju datoteka na stranici predmeta.

Dohvat podataka u obliku XML metodom GET iz PHP skripte

Za dohvat podataka iz nekog izvora pomoću metode GET u PHP skripti možete koristiti metodu `file_get_contents`, na sljedeći način:

```
$a = file_get_contents(
   "http://pinus.cc.fer.hr:8080/~korisnicko_ime/servlet");
```

Za inicijalizaciju objektnog modela dokumenta DOM i učitavanje XML dokumenta kao DOM stabla prihvatom podataka iz znakovnog niza možete, umjesto dosadašnje metode `load` koja je čitala datoteku s diska, koristiti metodu `loadXML`, slične primjene:

```
$dom = new DOMDocument();
$dom->loadXML( $a );
```

Ispitno gradivo vježbe

Ispitno gradivo uključuje sve navedeno u pripremi za vježbu, te detaljno razumijevanje izrađenog rješenja i snalaženje u prepravcima istog.

Primjeri pitanja:

- Objasnite detaljno izradu razreda Servleta
- Objasnite konfiguraciju u datoteci web.xml
- Objasnite životni ciklus Servleta
- Objasnite preinake u PHP skripti
- Objasnite kako se čitaju parametri zahtjeva
- Objasnite izlazni proces Servleta (PrintWriter)

Poveznice i literatura

- **The Java EE 5 Tutorial - Java Servlet Technology** (Chapter 4) - http://java.sun.com/javaee/5/docs/tutorial/doc/bnafd.html
- Java Servlet Programming – http://www.unix.com.ua/orelly/java-ent/servlet/index.htm
- A Hello, World Servlet – http://www.caucho.com/resin-3.0/servlet/tutorial/helloworld/index.xtp
- Servlet Essentials - http://www.novocode.com/doc/servlet-essentials
- Servlet-Tutorial - http://www.apl.jhu.edu/~hall/java/Servlet-Tutorial
- Servlet Basics - http://courses.coreservlets.com/Course-Materials/pdf/csajsp2/02-Servlet-Basics.pdf

6. laboratorijska vježba: Interaktivnost i dinamičnost elemenata stranice Weba

Cilj vježbe „Interaktivnost i dinamičnost elemenata stranice Weba“ je izrada dinamičke stranice Weba s elementima za interakciju s korisnikom koji su promjenjivi bez osvježavanja stranice. Za izradu interaktivnih elemenata koristi se tehnika AJAX (*Asynchronous JavaScript and XML*) te objekt XMLHttpRequest za preuzimanje dodatnih podataka s poslužitelja. Za izradu dinamičnih elemenata stranice koriste se javno dostupne biblioteke.

Priprema za vježbu

U cilju pripreme za vježbu treba **proučiti osnove jezika JavaScript** s naglaskom na:

- osnove jezika JavaScript, rad s varijablama, funkcije i događaji
- uključivanje vanjske datoteke u jeziku JavaScript u HTML dokument
  - W3Schools JavaScript Tutorial - http://www.w3schools.com/js

Dodatno, potrebno je **proučiti osnove jezika tehnika DHTML**, kao i **promjene dokumenta HTML korištenjem modela DOM** s naglaskom na:

- osnove koncepta DHTML
- objektni model HTML dokumenta (DOM)
- obradu događaja (*event handler)*,
- promjenu stilova elemenata
  - W3Schools DHTML tutorial - http://www.w3schools.com/dhtml
- osnove pristupanja i promjene elemenata HTML dokumenta korištenjem objektnog modela dokumenta (DOM)
  - Gecko DOM Reference: Introduction https://developer.mozilla.org/en/Gecko_DOM_Reference/Introduction
- osnovne metode nad objektnim modelom dokumenta
- svojstva objektnog modela dokumenta

Osim toga, potrebno je **proučiti tehnike AJAX**:

- korištenje objekta XMLHttpRequest i tehnike AJAX
  - XUL AJAX Tutorial - http://www.xul.fr/en-xml-ajax.html
  - Dynamic HTML and XML: The XMLHttpRequest Object http://developer.apple.com/internet/webcontent/xmlhttpreq.html

Zadatak za vježbu

Na postojećoj stranici rezultata pretrage potrebno je **prikazati iskočni okvir s dodatnim informacijama** o zapisu. Dodatno, potrebno je nadopuniti postojeću dinamičku stranicu **odjeljkom za prikaz detaljnih podataka** o pojedinom zapisu rezultata pretrage.

Prikaz iskočnog okvira s dodatnim informacijama

U prvom dijelu vježbe, potrebno je na stranici rezultata pretrage **prelaskom pokazivača** (*cursor*) **preko pojedinog retka** tablice **prikazati iskočni okvir** (*tooltip, "balončić", "oblačić"*) **s dodatnim informacijama o osnovnom elementu** čiji su podaci prikazani u tom

retku. Ovo se treba izvesti korištenjem neke od vanjskih biblioteka JavaScript funkcija, koje služe kao pomoć u bržoj izradi atraktivnih dinamičkih elemenata aplikacije Weba.

Napomena: Primjer iskočnog okvira vidljiv je prelaskom miša preko datuma objave bilo koje obavijesti na Webu FER-a.

S obzirom da je stranica rezultata pretrage izvedena PHP skriptom `pretraga.php`, potrebno je izmjeniti je tako da se prelaskom pokazivača (miša) preko pojedinog retka, a time i pojedinog zapisa, u iskočnom okviru prikaže nekoliko dodatnih informacija o tom zapisu. Te dodatne podatke PHP skripta već dobiva u obliku XML od Servleta (u 5. laboratorijskoj vježbi). U skripti treba promijeniti HTML kôd oznake retka tako da se na događaj prijelaza pokazivača preko područja retka poziva funkcija za prikaz iskočnog okvira uz predaju potrebnih argumenata.

Napomena: Podaci za prikaz u iskočnom okviru se nakon generiranja stranice već trebaju nalaziti pohranjeni unutar koda stranice, no skriveni od korisnika. Podaci se ne zahtijevaju dodatno od poslužitelja, već samo treba omogućiti njihov prikaz uporabom JavaScript funkcija.

U iskočnom okviru je **potrebno prikazati najmanje 2, a najviše 4 dodatna podatka** koja treba proizvoljno odabrati, s time da je potrebno svaki podatak staviti u novi redak okvira ili odvojiti nekim separatorom. Predlaže se prikaz u obliku "naziv: vrijednost" (npr. "Telefon: 01 5555555").

Za prikaz iskočnog okvira potrebno je **registrirati događaj prelaska miša** `onmouseover` korištenjem obradu događaja (*event handler)* za svaki pojedini element. Na događaj se poziva JavaScript funkcija (definirana u vanjskoj datoteci sa sufiksom .js), na sljedeći način:

```
<tr onmouseover="prikaziTooltip('Dodatni podaci ', 'green', true)”>
   <td> ...  </td>
   <td> ...  </td>
   ...
</tr>
```

Funkcionalnost iskočnog okvira nije potrebno samostalno izraditi, već se preporuča korištenje postojećih JavaScript biblioteka, koje se mogu slobodno koristiti. Korištenje provjerenih biblioteka drastično smanjuje vrijeme izrade aplikacije, a često osigurava veću razinu robusnosti i kompatibilnosti s različitim preglednicima.

Predlažemo korištenje sljedećih biblioteka iskočnih okvira u jeziku JavaScript:

- overLIB – Erik Bosrup - http://www.bosrup.com/web/overlib
- Yahoo UI Library: Tooltip - http://developer.yahoo.com/yui/container/tooltip
- DHTML JavaScript Tooltips – W. Zorn - http://www.sf.net/projects/wztip

Osim predloženih biblioteka, može se koristiti bilo koju drugu biblioteku sličnih funkcionalnosti. Pri predaji vježbe nije potrebno znati unutarnje funkcioniranje biblioteke, ali je potrebno znati objasniti povezivanje biblioteke, prilagodbu i prikazivanje iskočnog okvira.

Napomena: Prilikom preuzimanja biblioteke iskočnih okvira potrebno je obratiti pažnju na autorska prava i licencije.

Na stranicama navedenih biblioteka postoje kraći uvodi u korištenje biblioteka koje je potrebno proučiti. Kod većine biblioteka, nakon preuzimanja, obično se radi o uključivanju datoteke sufiksa `.js` s JavaScript kodom u stranicu te pozivu funkcije za prikaz iskočnog okvira. Funkciji se najčešće prosljeđuju argumenti kao što su tekst, boja teksta i pozadine, parametri trajanja i opcija vidljivosti i slično.

Iskočne okvire je potrebno prilagoditi dizajnu vaše stranice Weba. Prilagodbu možete obaviti **izravno** putem parametara biblioteka ili **pomoću CSS-a**. Iskočnom okviru je često moguće predati i sadržaj u obliku HTML koda. Tada iskočni okvir ne prikazuje samo tekst, već sadržaj koji sadrži oznake oblikovanja i stilove nadređene stranice definirane u datoteci CSS. Sadržaj u obliku HTML je moguće predati izravno **kao niz znakova** (paziti na jednostruke i dvostruke navodnike) ili kao **JavaScript varijablu** (polje).

Promjena područja stranice i prikaz detaljnih podataka

U drugom dijelu vježbe potrebno je **dohvatiti detaljne podatke** o odabranom osnovnom elementu i **prikazati ih** na stranici. To znači da se postojeća stranica, koju izrađuje PHP skripta `pretraga.php` iz 3. laboratorijske vježbe, treba preurediti tako da se izmijeni tablica za prikaz rezultata, doda novo područje za prikaz detaljnih podataka i omogući aktiviranje akcije prikaza detaljnih podataka u tom području. Za postizanje bolje uporabivosti stranice koristit će se objekt XMLHttpRequest i tehnika AJAX (*Asynchronous JavaScript and XML*) kako bi se izbjeglo nepotrebno osvježavanje stranice. Objekt XMLHttpRequest omogućuje asinkronu komunikaciju s poslužiteljem, slanje zahtjeva i dobivanje odgovora od poslužitelja bez osvježavanja cijele stranice. Stranica se uporabom jezika JavaScript i objekta XMLHttpRequest nadopunjava dinamički promjenjivim elementima.

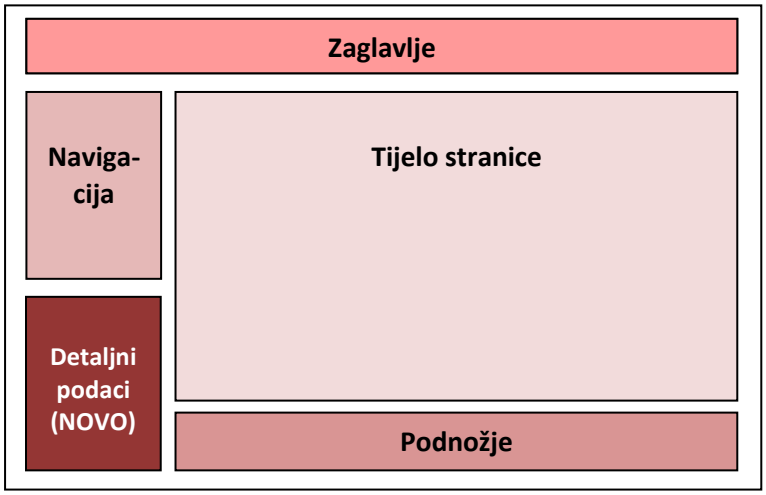
Napomena: Svi novi ili promijenjeni elementi stranice trebaju se uklapati u izgled, dizajn i strukturu postojeće stranice, a za njihovo oblikovanje potrebno je, kao i dosad, koristiti datoteku u jeziku CSS.

Stranicu rezultata pretrage treba preurediti tako da tablica sadrži **najmanje 3, a najviše 5 stupaca s podacima** koji se prikazuju. U tablici treba **prikazati samo osnovne informacije** (npr. za telefonski imenik to bi bili ime, prezime i prvi broj mobitela) tako da prikaz bude pregledan, a ako je tablica dosad imala više stupaca potrebno ih je obrisati.

Nadalje, s desne strane u tablici treba dodati **novi stupac** naslova **"Akcija"** koji će sadržavati neki proizvoljni **aktivni element**, koji će aktivirati **zahtjev za detaljnim podacima**. Za aktivni element može se odabrati poveznica s tekstom "Detalji" ili „Više o …“, ili neki prikladni grafički element poput sličice ili gumba.

Stranicu treba dodatno preraditi tako da se ispod dijela za navigaciju postavi **područje za detaljne podatke**, koje se **odabirom aktivnog elementa** u bilo kojem retku **ispunjava**

podacima dobivenim s poslužitelja. Područje za detaljne podatke treba definirati kao i ostala područja korištenjem elementa `<div>` i postaviti ispod područja za navigaciju.



Napomena: Kako bi se području za detaljne podatke moglo pristupiti iz funkcija jezika JavaScript, predlažemo da mu se kao atribut pridijeli jedinstveni identifikator.

Prikaz detaljnih podataka

Odabirom aktivnog elementa (klikom miša) **treba poslati zahtjev za detaljnim podacima** poslužitelju. Zahtjev se šalje PHP skripti `detalji.php` koja dohvaća sve podatke u XML obliku od Servleta na isti način kao i stranica prikaza rezultata pretrage. No, razlika je u tome što ova PHP skripta treba vratiti samo detaljne podatke vezane upravo za taj jedan osnovni element. PHP skripta **treba vratiti odgovor u obliku segmenta koda u jeziku HTML**, kojeg korištenjem jezika JavaScript i objektnog modela HTML dokumenta (DOM) treba prikazati u području za detaljne podatke na stranici rezultata pretrage. Za promjenu svojstava ili izgleda objekata tog područja potrebno je koristiti funkcije jezika JavaScript. S obzirom da zahtjev za detaljnim podacima treba biti prilagođen jednom zapisu tj. osnovnom elementu koji je prikazan u tom retku i za koji tražimo detaljne podatke, potrebno je u zahtjevu kao dodatni argument zadati **jedinstveni identifikator elementa**, već naveden u opisu inačica (npr. ID, OIB, ISBN ...).

Objekt XMLHttpRequest služi za komunikaciju s poslužiteljem i dohvat manje količine podataka slanjem zahtjeva metodom GET ili POST zahtjeva te primanjem odgovora. Najveća prednost ove tehnike je što se dio stranice može promijeniti bez osvježavanja cijele stranice. Primjer rada s objektom je sljedeći:

```
var req;                          // deklarirana globalna varijabla
function loadXMLDoc(url) {
    if (window.XMLHttpRequest) {          // FF, Safari, Opera, IE7+
        req = new XMLHttpRequest();       // stvaranje novog objekta
    } else if (window.ActiveXObject) {    // IE 6+
        req = new ActiveXObject("Microsoft.XMLHTTP");  //ActiveX
    }
    if (req) {                            // uspješno stvoren objekt
        req.onreadystatechange = doSomething;
```

```
            req.open("GET", url, true);    // metoda, URL, asinkroni način
            req.send(null); //slanje (null za GET, podaci za POST)
        }
    }
```

Napomena: Različiti preglednici na različite načine instanciraju objekt XMLHttpRequest, no nakon instanciranja, funkcije su jednake.

Svojstvo `onreadystatechange` služi za povezivanje funkcije koja se poziva pri promjeni stanja zahtjeva. U sinkronom načinu odaziv stranice blokira do dobivanja odgovora, dok se u asinkronom načinu nakon slanja zahtjeva izvođenje funkcije i rad sa stranicom nastavlja, a naknadno se pri promjeni stanja zahtjeva, odnosno kod odgovora, zove povezana funkcija. Kod metode GET parametri se šalju u URL-u (npr. … `detalji.php?id=523&show=simple`), dok se kod metode POST parametri šalju kao argument funkcije `send` (npr. `req.send("id=523&show=simple");`).

Napomena: Iz sigurnosnih razloga objekt XMLHttpRequest ne dozvoljava komunikaciju s poslužiteljima izvan domene s koje je stranica isporučena (*cross-domain scripting)*.

Funkcija se poziva pri svakoj promjeni stanja, što uključuje i slučaj nedostupnosti poslužitelja, pa je potrebno provjeriti status i kôd odgovora (`readyState` i `status`). Sadržaj odgovora je moguće dohvatiti kao niz znakova u svojstvu `responseText`, ili u obliku XML u značajki `responseXML`. Primjer funkcije koja će se pozvati pri promjeni stanja zahtjeva:

```
function doSomething () {
   if (req.readyState == 4) {       // primitak odgovora
      if (req.status == 200) {      // kôd statusa odgovora = 200 OK
         // kod uspješnog odgovora
      } else {                      // kôd statusa nije OK
         alert("Nije primljen 200 OK, nego:\n" + req.statusText);
      }
   }
}
```

JavaScript funkcije i DOM

Jezik JavaScript omogućuje upravljanje objektnim modelom dokumenta (DOM). Kako bi pročitali sadržaj elementa ili ga promijenili potrebno je prvo pronaći i dohvatiti element. Navedene su neke funkcije i svojstva koja se mogu koristiti:

- `document.getElementById (id)` – dohvaća jedan element po identifikatoru
- `document.getElementsByTagName (name)` – dohvaća sve elemente određenog naziva
- `element.innerHTML` – dohvaća (ili mijenja) sadržaj elementa
- `element.setAttribute (name, value)` – postavlja vrijednost atributa
- `element.getAttribute (name)` – dohvaća vrijednost atributa
- `element.style` – dohvaća (ili mijenja) vrijednost stila CSS
  - `element.style.textAlign = 'center'` ili neko drugo poravnavanje
  - `element.style.color = 'red'` ili neke druge boje
  - `element.style.visibility = 'hidden'` ili `'visible'`

- `element.style.display = 'none'` ili `'block'`

Napomena: Za rad s jezikom JavaScript i objektom XMLHttpRequest mogu se koristiti različiti alati dostupni za različite preglednike, kao što je JavaScript debugger, te dodaci za praćenje prometa protokola HTTP. Tako npr. za preglednik Firefox postoji pregled grešaka (Tools -> Error Console), a moguće je i koristiti dodatke DOM Inspector i Firebug.

Za rješavanje ovog dijela vježbe potrebno je napisati **datoteku u jeziku PHP** `detalji.php` koja metodom GET prima parametar `id` kao identifikator elementa čije detalje treba prikazati.

Napomena: Datoteka `detalji.php` je zapravo umanjenja verzija datoteke `pretraga.php`, u kojoj treba iz dohvaćene XML strukturu podataka pronaći podatke traženog elementa prema identifikatoru.

Izlaz iz skripte `detalji.php` je u obliku segmenta HTML koda koji sadrži druge detaljne podatke, odnosno sve informacije o pronađenom zapisu. Izlaz iz PHP skripte može sadržavati različite podatke i oznake jezika HTML uključivo oznake sakrivene tablice, oznake , oznake , razrede za poziv CSS stila i slično. U području za detaljne podatke potrebno je prikazati što je više moguće informacija, a poželjno sve podatke o nekom zapisu, koji su dobiveni iz XML datoteke. Pri tome treba obratiti pažnju da se ne poremeti izgled cjelokupne stranice.

U PHP skripti koja je poslala zahtjev za detaljnim podacima potrebno je za svaki pojedini aktivni element dohvatom događaja `onclick` pozvati funkciju koja će pripremiti zahtjev s argumentom `id` i poslati ga PHP skripti `detalji.php`.

U funkciji koja se poziva pri promjeni stanja zahtjeva treba preuzeti dobiveni HTML s detaljnim podacima u svojstvu responseText i prikazati u području za detaljne podatke.

Nakon svakog novog zahtjeva za detaljnim podacima o novom elementu, u području za detaljne podatke je potrebno zamijeniti prikazan podatke.

Dodatno, za vrijeme čekanja na odgovor od poslužitelja, pored aktivnog elementa treba prikazati tekst "Tražim ..." ili prikazati statičnu ili animiranu sličicu koja prikazuje status čekanja (npr. http://en.wikipedia.org/wiki/Image:Spinning_wheel_throbber.gif). Po uspješnom primitku odgovora treba ukloniti tekst ili sličicu.

Napomena: Ako je vraćanje odgovora prebrzo za prikaz sličice, možete se za potrebe demonstracije usporiti korištenjem funkcije `sleep(seconds)` u jeziku PHP.

Predaja vježbe

Rješenje vježbe se na poslužitelj predaju u obliku ZIP arhive koja sadrži sve datoteke potrebne za ispravan rad rješenja. Datoteke trebaju biti složene u strukturi direktorija (`public_html, web-app` ...), a trebaju sadržavati i uporabljenu biblioteku iskočnog okvira.

Nazivi datoteka izrađenih u vježbi trebaju biti:

- `detalji.php` – PHP skripta koja rukuje objektom XMLHttpRequest i koja dohvaća detaljne podatke o zapisu
- `detalji.js` – datoteka s JavaScript kodom koja sadrži programsku logiku (registracija rukovatelja događaja može biti zapisana izravno u PHP skripti)

Ispitno gradivo vježbe

Ispitno gradivo uključuje sve navedeno u pripremi za vježbu, te detaljno razumijevanje izrađenog rješenja i snalaženje u prepravcima istog.

Primjeri pitanja:

- Što je to DHTML?
- Objasnite način rada s objektom XMLHttpRequesta
- Objasnite prednosti korištenja tehnike AJAX
- Objasnite promjenu svojstava elemenata korištenjem jezika JavaScript
- Objasnite način dohvaćanja pojedinog elementa u objektnom modelu HTML dokumenta, odnosno DOM stablu.
- Dinamički promijenite boju teksta nekog elementa

Poveznice i literatura

- W3Schools JavaScript and DOM Reference - http://www.w3schools.com/jsref
- XML.com: Very Dynamic Web Interfaces http://www.xml.com/pub/a/2005/02/09/xml-http-request.html
- The DOM and JavaScript - http://developer.mozilla.org/en/docs/The_DOM_and_JavaScript
- Using the W3C DOM Level 1 Core - http://developer.mozilla.org/en/docs/Using_the_W3C_DOM_Level_1_Core

Licencija

Prilog - inačice laboratorijskih vježbi

U ovom poglavlju prikazani su opisi pojedinih inačica laboratorijskih vježbi s objašnjenjima. Iako su zadaci vježbe načelno isti, vježbe postoje u osam inačica, koja opisuju različita područja primjene rješenja zadatka. U svakoj vježbi koristi se dio informacija iz dokumenta o inačicama, pa za objašnjenje pojedinog stupca treba pogledati uputa za odgovarajuću laboratorijsku vježbu. Neki stupci imaju značenje tek u kasnijim vježbama.

Inačice su označene slovom od A do H kako slijedi:

- Inačica A: DVD-teka
- Inačica B: knjižnica
- Inačica C: sustav dokumenata
- Inačica D: telefonski imenik
- Inačica E: evidencija djelatnika
- Inačica F: prodaja CD-a
- Inačica G: popis literature
- Inačica H: evidencija računalne opreme

Objašnjenje stupaca u opisu inačica je sljedeće:

- REDNI BROJ – označava redni broj podatka u strukturi
- NAZIV – označava naziv podatka u strukturi
- HIJERARHIJSKA RAZINA – označava razinu hijerarhije podatka u odnosu na korijenski element podataka (atributi se smatraju podrazinom elementa)
- BROJNOST – označava može li se element pojaviti samo jednom (1) ili više puta (N) unutar istog nadređenog elementa
- ELEMENT ili ATRIBUT – označava prikaz podatka kao elementa ili atributa
- OBAVEZNOST POSTOJANJA – označava obaveznost (nužnost) postojanja (barem jednog) podatka (elementa ili atributa)
- VRIJEDNOSTI – označava sadrži li element podatak
  - DA – sadrži (standardno upisanu) tekstualnu vrijednost
  - SKUP – sadrži podatak isključivo iz skupa dozvoljenih vrijednosti
  - NEMA – ne sadrži podatak, već se podaci opcionalno nalaze u podelementima i atributima
- PRIMJER: SLOBODAN UPISA VRIJEDNOSTI ili SKUP DOZVOLJENIH VRIJEDNOSTI – označava primjer tekstualnog podatka ili skup dozvoljenih vrijednosti podatka
  - Primjer tekstualnog podatka – npr. „tekst“
  - Skup dozvoljenih vrijednosti – npr. za ocjenu je 1, 2, 3, 4, 5
- ELEMENT U OBRASCU na stranici za pretraživanje – označava kako će se izvesti unos podatka po kojem će se pretraživati, odnosno pomoću kojeg elementa za pretraživanje će se ostvariti unos
  - POLJE ZA UNOS – unosno polje za tekstualni podatak
  - KVADRATIĆ ZA IZBOR (*checkbox*) – polje za odabir jedne ili više vrijednosti iz skupa dozvoljenih vrijednosti ili oznaka za postojanje/nepostojanje podatka
  - KRUŽIĆ ZA ODABIR (*radio*) – polje za odabir samo jedne vrijednosti iz skupa dozvoljenih vrijednosti
  - IZBORNIK ZA VIŠESTRUKI ODABIR (*multiple select*) – polje za odabir više (ili samo jedne) vrijednosti iz skupa dozvoljenih vrijednosti

Inačica A: DVD-teka

REDNI BROJ	NAZIV	HIJERARHIJSKA RAZINA	BROJNOST	ELEMENT ili ATRIBUT	OBAVEZNOST POSTOJANJA	VRIJEDNOSTI	PRIMJER: SLOBODAN UPIS VRIJEDNOSTI ili SKUP DOZVOLJENIH VRIJEDNOSTI	ELEMENT U OBRASCU na stranici za pretraživanje
1	**DVD**	1	N	E	DA	NEMA		
2	**EAN**	2	1	A	DA	DA	**13 znamenaka (EAN-13)**	POLJE ZA UNOS
3	**Naslov**	2	1	E	DA	DA	npr. Tko pjeva zlo ne misli	POLJE ZA UNOS
4	**Režiser**	2	N	E	DA	NEMA		
5	**Ime**	3	1	E	DA	DA	npr. Krešo	POLJE ZA UNOS
6	**Prezime**	3	1	E	DA	DA	npr. Golik	POLJE ZA UNOS
7	**Glumac**	2	N	E	DA	NEMA		
8	**Ime**	3	1	E	DA	DA	npr. Mirjana	POLJE ZA UNOS
9	**Prezime**	3	1	E	DA	DA	npr. Bohanec	POLJE ZA UNOS
10	**Jezik zvuka**	2	1	A	DA	**SKUP**	**Hrvatski, Engleski, Njemački, Slovenski**	KVADRATIĆ ZA IZBOR (checkbox) 4 jezika
11	**Omjer slike**	2	1	A	DA	**SKUP**	**4:3, 16:9, 2.35:1**	KRUŽIĆ ZA ODABIR (radio) jednog omjera
12	**Regija**	2	1	A	DA	**SKUP**	**0, 1, 2, 3, 4, 5, 6, 7, 8, 9, ALL**	IZBORNIK ZA VIŠESTRUKI ODABIR (multiple select) regije
13	**Posebni dodaci**	2	N	E	NE	DA	npr. komentar redatelja, isječci sa snimanja	POLJE ZA UNOS KVADRATIĆ ZA IZBOR (checkbox) postojanja posebnih dodataka
14	**Trajanje**	2	1	E	DA	DA	npr: 02:45:38	POLJE ZA UNOS
15	**Ocjena**	2	1	A	DA	**SKUP**	**1, 2, 3, 4, 5**	KVADRATIĆ ZA IZBOR (checkbox) 5 ocjena
16	**Cijena posudbe**	2	1	E	DA	DA	npr. 12,00	POLJE ZA UNOS
17	**PROIZVOLJNI ODABIR**							
18	**PROIZVOLJNI ODABIR**							
19	**PROIZVOLJNI ODABIR**							
20	**PROIZVOLJNI ODABIR**							

Inačica B: knjižnica

REDNI BROJ	NAZIV	HIJERARHIJSKA RAZINA	BROJNOST	ELEMENT ili ATRIBUT	OBAVEZNOST POSTOJANJA	VRIJEDNOSTI	PRIMJER: SLOBODAN UPIS VRIJEDNOSTI ili SKUP DOZVOLJENIH VRIJEDNOSTI	ELEMENT U OBRASCU na stranici za pretraživanje
1	**Knjiga**	1	N	E	DA	NEMA		
2	**ISBN**	2	1	A	DA	DA	**13 znamenaka (ISBN)**	POLJE ZA UNOS
3	**Naslov**	2	1	E	DA	DA	npr. Thinking in Java	POLJE ZA UNOS
4	**Podnaslov**	2	1	E	NE	DA	npr. Learning Java programming language	POLJE ZA UNOS
5	**Autor**	2	N	E	DA	NEMA		
6	**Ime**	3	1	E	DA	DA	npr. Bruce	POLJE ZA UNOS
7	**Prezime**	3	1	E	DA	DA	npr. Eckel	POLJE ZA UNOS
8	**Jezik**	2	1	A	DA	**SKUP**	**Hrvatski, Engleski, Njemački, Slovenski**	KVADRATIĆ ZA IZBOR (checkbox) 4 jezika
9	**Broj stranica**	2	1	E	DA	DA	npr. 1150	POLJE ZA UNOS
10	**Izdavač**	2	N	E	DA	NEMA		KVADRATIĆ ZA IZBOR (checkbox) postojanja izdavača
11	**Naziv**	3	1	E	DA	DA	npr. Prentice-Hall	POLJE ZA UNOS
12	**Izdanje**	2	1	A	DA	**SKUP**	**1, 2, 3, 4, 5, 6, 7, 8, 9, 10**	IZBORNIK ZA VIŠESTRUKI ODABIR (multiple select) izdanja
13	**Datum izdanja**	2	1	E	NE	DA	npr. 12.12.2005.	POLJE ZA UNOS
14	**Uvez**	2	1	A	DA	**SKUP**	**Meki, Tvrdi**	KRUŽIĆ ZA ODABIR (radio) jednog uveza
15	**Ocjena**	2	1	A	DA	**SKUP**	**1, 2, 3, 4, 5**	KVADRATIĆ ZA IZBOR (checkbox) 5 ocjena
16	**Način distribucije**	2	1	E	NE	DA	npr. knjiga, elektronička knjiga, audio knjiga	POLJE ZA UNOS
17	**PROIZVOLJNI ODABIR**							
18	**PROIZVOLJNI ODABIR**							
19	**PROIZVOLJNI ODABIR**							
20	**PROIZVOLJNI ODABIR**							

Inačica C: sustav dokumenata

REDNI BROJ	NAZIV	HIJERARHIJSKA RAZINA	BROJNOST	ELEMENT ili ATRIBUT	OBAVEZNOST POSTOJANJA	VRIJEDNOSTI	PRIMJER: SLOBODAN UPIS VRIJEDNOSTI ili SKUP DOZVOLJENIH VRIJEDNOSTI	ELEMENT U OBRASCU na stranici za pretraživanje
1	**Dokument**	1	N	E	NE	NEMA		
2	**Identifikator dokumenta**	2	1	A	DA	DA	**znakovi "DOC-" i 6 znamenaka**	POLJE ZA UNOS
3	**Naslov**	2	1	E	DA	DA	npr. OR - Uvod	POLJE ZA UNOS
4	**Naziv datoteke**	3	1	A	DA	DA	npr. OR_1_Uvod.pdf	POLJE ZA UNOS
5	**Opis dokumenta**	2	1	E	NE	DA	npr. 1. predavanje iz Otvorenog računarstva	POLJE ZA UNOS
6	**Tip dokumenta**	2	1	A	DA	**SKUP**	**ODT, DOC, PDF, GIF, JPG, TIF**	KVADRATIĆ ZA IZBOR (checkbox) tipa dokumenta
7	**Autor dokumenta**	2	N	E	DA	NEMA		
8	**Ime**	3	1	E	DA	DA	npr. Mario	POLJE ZA UNOS
9	**Prezime**	3	1	E	DA	DA	npr. Žagar	POLJE ZA UNOS
10	**Datum izrade**	3	1	E	DA	DA	npr. 12.12.2007.	POLJE ZA UNOS
11	**Veličina**	2	1	E	DA	DA	npr. 124 kB	POLJE ZA UNOS
12	**Datum zadnje promjene**	2	1	E	DA	DA	npr. 15.12.2007.	POLJE ZA UNOS
13	**Objava**	2	1	A	DA	**SKUP**	**Naslovnica, Predmet, Zavod**	IZBORNIK ZA VIŠESTRUKI ODABIR (multiple select) korisnika
14	**Pravo pristupa**	2	1	A	DA	**SKUP**	**Javni, Privatni, Samo nastavnici**	KRUŽIĆ ZA ODABIR (radio) prava pristupa
15	**Namjena**	2	1	A	DA	**SKUP**	**Predavanje, Lab.vježba, Rezultati, Ostalo**	KVADRATIĆ ZA IZBOR (checkbox) prava pristupa
16	**Vidljivost**	2	1	A	DA	**SKUP**	**Vidljivo, Skriveno**	KVADRATIĆ ZA IZBOR (checkbox) vidljivosti
17	**PROIZVOLJNI ODABIR**							
18	**PROIZVOLJNI ODABIR**							
19	**PROIZVOLJNI ODABIR**							
20	**PROIZVOLJNI ODABIR**							

Inačica D: telefonski imenik

REDNI BROJ	NAZIV	HIJERARHIJSKA RAZINA	BROJNOST	ELEMENT ili ATRIBUT	OBAVEZNOST POSTOJANJA	VRIJEDNOSTI	PRIMJER: SLOBODAN UPIS VRIJEDNOSTI ili SKUP DOZVOLJENIH VRIJEDNOSTI	ELEMENT U OBRASCU na stranici za pretraživanje
1	**Osoba**	1	N	E	DA	NEMA		
2	**OIB**	2	1	A	DA	DA	**11 znamenaka**	POLJE ZA UNOS
3	**Ime**	2	1	E	DA	DA	npr. Ivo	POLJE ZA UNOS
4	**Prezime**	2	1	E	DA	DA	npr. Ivić	POLJE ZA UNOS
5	**Kategorija**	2	1	A	DA	**SKUP**	**Prijatelj, Kolega, Obitelj, Bez kategorije**	KVADRATIĆ ZA IZBOR (checkbox) 4 kategorije
6	**Telefon**	2	N	E	DA	NEMA		
7	**Tip**	3	1	A	DA	**SKUP**	**Mobilni, Fiksni, Telefaks**	KRUŽIĆ ZA ODABIR (radio) jednog tipa
8	**Broj**	3	1	E	DA	DA	npr. 6129999	POLJE ZA UNOS
9	**Pozivni broj mreže**	4	1	A	DA	**SKUP**	**099, 098, 095, 092, 091, 01**	IZBORNIK ZA VIŠESTRUKI ODABIR (multiple select) poziv. broja
10	**Adresa**	2	1	E	NE	NEMA		KVADRATIĆ ZA IZBOR (checkbox) postojanja adrese
11	**Ulica**	3	1	E	DA	DA	npr. Ilica	POLJE ZA UNOS
12	**Kućni broj**	3	1	E	DA	DA	npr. 111	POLJE ZA UNOS
13	**Mjesto**	3	1	E	DA	DA	npr. Zagreb	POLJE ZA UNOS
14	**Poštanski broj**	4	1	A	DA	DA	npr. 10000	POLJE ZA UNOS
15	**Država**	3	1	E	DA	DA	npr. Hrvatska	POLJE ZA UNOS
16	**Mail adresa**	2	N	E	NE	DA	npr. ivo.ivic@fer.hr	POLJE ZA UNOS KVADRATIĆ ZA IZBOR (checkbox) postojanja mail adrese
17	**PROIZVOLJNI ODABIR**							
18	**PROIZVOLJNI ODABIR**							
19	**PROIZVOLJNI ODABIR**							
20	**PROIZVOLJNI ODABIR**							

Inačica E: evidencija djelatnika

REDNI BROJ	NAZIV	HIJERARHIJSKA RAZINA	BROJNOST	ELEMENT ili ATRIBUT	OBAVEZNOST POSTOJANJA	VRIJEDNOSTI	PRIMJER: SLOBODAN UPIS VRIJEDNOSTI ili SKUP DOZVOLJENIH VRIJEDNOSTI	ELEMENT U OBRASCU na stranici za pretraživanje
1	**Djelatnik**	1	N	E	DA	NEMA		
2	**Identifikator djelatnika**	2	1	A	DA	DA	**2 slova i 5 znamenki, slova inicijali**	POLJE ZA UNOS
3	**Ime**	2	1	E	DA	DA	npr. Ivo	POLJE ZA UNOS
4	**Prezime**	2	1	E	DA	DA	npr. Horvat	POLJE ZA UNOS
5	**Radno mjesto**	2	1	A	DA	**SKUP**	**Pripravnik, Djelatnik, Šef, Direktor**	KVADRATIĆ ZA IZBOR (checkbox) 4 kategorije
6	**Datum početka rada**	2	1	E	DA	DA	npr. 12.12.2004.	POLJE ZA UNOS
7	**Odjel**	2	1	A	DA	**SKUP**	**Nabava, Prodaja, Proizvodnja, Centrala**	IZBORNIK ZA VIŠESTRUKI ODABIR (multiple select) odjela
8	**Koeficijent plaće**	2	1	A	DA	**SKUP**	**1.0, 1.2, 1.3, 1.5, 1.7, 2.0**	KRUŽIĆ ZA ODABIR (radio) jednog koeficijenta
9	**Bonus**	2	1	A	DA	**SKUP**	**10%, 20%, 30%, 40%, 50%**	KVADRATIĆ ZA IZBOR (checkbox) 5 bonusa
10	**Adresa prebivališta**	2	1	E	NE	NEMA		KVADRATIĆ ZA IZBOR (checkbox) postojanja adrese
11	**Ulica**	3	1	E	DA	DA	npr. Ilica	POLJE ZA UNOS
12	**Kućni broj**	3	1	E	DA	DA	npr. 111	POLJE ZA UNOS
13	**Mjesto**	3	1	E	DA	DA	npr. Zagreb	POLJE ZA UNOS
14	**Poštanski broj**	4	1	A	DA	DA	npr. 10000	POLJE ZA UNOS
15	**Broj dana godišnjeg**	2	1	E	DA	DA	npr. 27	POLJE ZA UNOS
16	**Titula**	2	N	E	NE	DA	npr. dipl.ing.	POLJE ZA UNOS
17	**PROIZVOLJNI ODABIR**							
18	**PROIZVOLJNI ODABIR**							
19	**PROIZVOLJNI ODABIR**							
20	**PROIZVOLJNI ODABIR**							

Inačica F: prodaja CD-a

REDNI BROJ	NAZIV	HIJERARHIJSKA RAZINA	BROJNOST	ELEMENT ili ATRIBUT	OBAVEZNOST POSTOJANJA	VRIJEDNOSTI	PRIMJER: SLOBODAN UPIS VRIJEDNOSTI ili SKUP DOZVOLJENIH VRIJEDNOSTI	ELEMENT U OBRASCU na stranici za pretraživanje
1	**CD**	1	N	E	DA	NEMA		
2	**EAN**	2	1	A	DA	DA	**13 znamenaka (EAN-13)**	POLJE ZA UNOS
3	**Naslov**	2	1	E	DA	DA	npr. 07	POLJE ZA UNOS
4	**Autor**	2	N	E	DA	NEMA		
5	**Ime**	3	1	E	NE	DA	npr. Nina	POLJE ZA UNOS
6	**Prezime ili naziv grupe**	3	1	E	DA	DA	npr. Badrić	POLJE ZA UNOS
7	**Kategorija**	2	1	A	DA	**SKUP**	**Klasika, Pop/Rock, Domaće, Ostalo**	KVADRATIĆ ZA IZBOR (checkbox) 4 kategorije
8	**Regija izdanja**	2	1	A	DA	**SKUP**	**Europa, S. Amerika, J. Amerika, Azija, Australija, Sve**	IZBORNIK ZA VIŠESTRUKI ODABIR (multiple select) regije
9	**Izdavač**	2	1	E	DA	NEMA		
10	**Naziv**	3	1	E	DA	DA	npr. Aquarius Records	POLJE ZA UNOS
11	**Oblik zapisa**	2	1	A	DA	**SKUP**	**CD, SACD**	KRUŽIĆ ZA ODABIR (radio) jednog oblika
12	**Pjesma**	2	N	E	DA	NEMA		
13	**Naziv pjesme**	3	1	E	DA	DA	npr. Imati pa nemati	POLJE ZA UNOS
14	**Trajanje**	3	1	E	DA	DA	npr: 06:15	POLJE ZA UNOS
15	**Ocjena**	2	1	A	DA	**SKUP**	**1, 2, 3, 4, 5**	KVADRATIĆ ZA IZBOR (checkbox) 5 ocjena
16	**Cijena**	2	1	E	NE	DA	npr. 120,00	POLJE ZA UNOS KVADRATIĆ ZA IZBOR (checkbox) postojanja cijene
17	**PROIZVOLJNI ODABIR**							
18	**PROIZVOLJNI ODABIR**							
19	**PROIZVOLJNI ODABIR**							
20	**PROIZVOLJNI ODABIR**							

Inačica G: popis literature

REDNI BROJ	NAZIV	HIJERARHIJSKA RAZINA	BROJNOST	ELEMENT ili ATRIBUT	OBAVEZNOST POSTOJANJA	VRIJEDNOSTI	PRIMJER: SLOBODAN UPIS VRIJEDNOSTI ili SKUP DOZVOLJENIH VRIJEDNOSTI	ELEMENT U OBRASCU na stranici za pretraživanje
1	**Referenca**	1	N	E	DA	NEMA		
2	**Identifikator reference**	2	1	A	DA	DA	**3 početna slova prezimena prvog autora, "-" i 5 znamenki**	POLJE ZA UNOS
3	**Naslov rada**	2	1	E	DA	DA	npr. Atlas	POLJE ZA UNOS
4	**Autor**	2	N	E	DA	NEMA		
5	**Ime**	3	1	E	DA	DA	npr. Danko	POLJE ZA UNOS
6	**Prezime**	3	1	E	DA	DA	npr. Basch	POLJE ZA UNOS
7	**Kategorija**	2	1	A	DA	**SKUP**	**Knjiga, Rad u časopisu, Sudjel. na skupu, Doktorat, Diplomski rad, Zavr. rad, Ostalo**	IZBORNIK ZA VIŠESTRUKI ODABIR (multiple select) kategorije
8	**Područje**	2	1	A	DA	**SKUP**	**Računarstvo, Elektrotehnika, Ostalo**	KVADRATIĆ ZA IZBOR (checkbox) 3 područja
9	**Izdavač**	2	N	E	NE	NEMA		
10	**Naziv**	3	1	E	DA	DA	npr. Antonić	POLJE ZA UNOS
11	**Mjesto**	3	1	E	DA	DA	npr. Zagreb	POLJE ZA UNOS
12	**Jezik**	2	1	A	DA	**SKUP**	**Hrvatski, Engleski, Njemački, Ostalo**	KVADRATIĆ ZA IZBOR (checkbox) 5 jezika
13	**Recenzija**	2	1	A	DA	**SKUP**	**Međunarodna, Domaća, Nema**	KRUŽIĆ ZA ODABIR (radio) jednog oblika
14	**Datum izdanja**	2	1	E	DA	DA	npr. 12.12.2004.	POLJE ZA UNOS
15	**Datum preuzimanja**	2	1	E	DA	DA	npr. 12.12.2005.	POLJE ZA UNOS
16	**URL**	2	1	E	NE	DA	npr. www.fer.hr/nesto	POLJE ZA UNOS KVADRATIĆ ZA IZBOR (checkbox) URL-a
17	**PROIZVOLJNI ODABIR**							
18	**PROIZVOLJNI ODABIR**							
19	**PROIZVOLJNI ODABIR**							
20	**PROIZVOLJNI ODABIR**							

Inačica H: evidencija računalne opreme

REDNI BROJ	NAZIV	HIJERARHIJSKA RAZINA	BROJNOST	ELEMENT ili ATRIBUT	OBAVEZNOST POSTOJANJA	VRIJEDNOSTI	PRIMJER: SLOBODAN UPIS VRIJEDNOSTI ili SKUP DOZVOLJENIH VRIJEDNOSTI	ELEMENT U OBRASCU na stranici za pretraživanje
1	**Računalo**	1	N	E	DA	NEMA		
2	**Identifikator računala**	2	1	A	DA	DA	**znakovi "RO-" i 4 znamenke**	POLJE ZA UNOS
3	**Naziv ili opis**	2	1	E	DA	DA	npr. Računalo u A101	POLJE ZA UNOS
4	**Procesor**	2	1	E	DA	NEMA		
5	**Naziv procesora**	3	1	E	DA	DA	npr, Intel Pentium Core Duo	POLJE ZA UNOS
6	**Frekvencija proces.**	3	1	E	DA	DA	npr. 2,2GHz	POLJE ZA UNOS
7	**Broj procesora**	2	1	A	DA	**SKUP**	**1, 2, 4, 8**	KRUŽIĆ ZA ODABIR (radio) broja procesora
8	**Tvrdi disk**	2	N	E	DA	NEMA		
9	**Proizvođač**	3	1	E	NE	DA	npr. Western Digital	POLJE ZA UNOS
10	**Kapacitet**	3	1	E	DA	DA	npr. 200GB	POLJE ZA UNOS
11	**Količina RAM-a**	2	1	E	DA	DA	npr. 1GB	POLJE ZA UNOS
12	**Namjena računala**	2	1	A	DA	**SKUP**	**Osobno, Poslužiteljsko, Namjensko, Ugradbeno**	KVADRATIĆ ZA IZBOR (checkbox) 4 namjene
13	**Datum kupnje**	2	1	E	NE	DA	npr. 12.12.2004.	POLJE ZA UNOS KVADRATIĆ ZA IZBOR (checkbox) postojanja datuma kupnje
14	**Inventarski broj**	2	1	A	DA	DA	npr. 12345678	POLJE ZA UNOS
15	**Lokacija**	2	1	A	DA	**SKUP**	**PCLAB1, PCLAB2, A101, A102, CIP**	KVADRATIĆ ZA IZBOR (checkbox) 5 lokacija
16	**Glavni OS**	2	1	A	DA	**SKUP**	**Linux, UNIX, Windows XP, Windows Vista**	IZBORNIK ZA VIŠESTRUKI ODABIR (multiple select) korisnika
17	**PROIZVOLJNI ODABIR**							
18	**PROIZVOLJNI ODABIR**							
19	**PROIZVOLJNI ODABIR**							
20	**PROIZVOLJNI ODABIR**							