SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

OTVORENO RAČUNARSTVO

Vježbe

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

Zagreb, 2016.

SADRŽAJ

Predgovor

Pojam otvorenog računarstva (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 i PHP, 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. Marija 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**, dr. sc. **Ivana Bosnić** i **Tomislav Sečen** dipl. ing.

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 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

[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/
[4] SlideShare, http://www.slideshare.net/

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 *WikiPres* 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] te **LibreOffice** 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 ususatvljivanjem pojedinačnih opisa vježbi u cjelinu.

[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. Svaka se 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 - aplikaciju Weba zasnovanu 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 i PHP.

Inačice vježbi

Zadaci vježbi su, u skladu s gradivom koje se obrađuje, konceptualno isti, no sama primjena, tj. tema Web sjedišta koje student izrađuje je drugačija. Student prije početka vježbi sâm treba odabrati temu, uz nekoliko ograničenja:

- Web sjedište će biti u obliku kataloga/baze podataka/zapisa, podaci će biti prikazani u strukturiranom obliku i tablično, a bit će omogućeno i pretraživanje
- tema treba biti takva da pojedini podaci/zapisi (koje će student sâm predložiti i zapisati u datoteku kao izvor podataka) imaju **javno dostupne stranice na *Facebooku*** (*Facebook Pages*), s ciljem dohvaćanja dodatnih podataka
- podaci/zapisi trebaju imati neku **geografsku lokaciju** (također dostupnu na istoj *Facebook Pages* stranici). Najbolje je da to bude puna adresa (s ulicom), no može se iskoristiti i samo naziv grada.
- podaci mogu biti osobe samo ako su „javno poznate“, tj. imaju javno dostupne stranice. Popis vlastitih prijatelja nije prikladna tema jer je dohvat privatnih informacija s *Facebooka* kompliciraniji nego pristup javnim podacima.

Općeniti primjeri takvih tema su:

- hoteli/parkovi/spomenici/crkve/znamenitosti nekog područja/grada/države
- mjesta na svijetu koja biste željeli obići
- katalog sportskih klubova/specijaliziranih prodavaonica
- katalog određene vrste javnih osoba
- (političara, sportaša, glumaca, glazbenika, umjetnika, *celebrityja*…)
- filmoteka (samo ako je dostupna neka lokacija filma – mjesto radnje, itd.)
- …

Nakon odabira teme, student treba definirati podatke koji će biti dostupni na Web sjedištu. Takva je definicija potrebna i korisna zbog lakše izrade Web obrasca (2. laboratorijska vježba) te XML datoteke s podacima (3. laboratorijska vježba). U Prilogu se nalazi primjer podataka za telefonski imenik i opis pojedinih stupaca iz primjera. Student treba za svoju temu napraviti sličnu tablicu po zadanom predlošku, koja će biti dostupna tijekom predaje vježbi za lakše snalaženje i veću dosljednost.

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.

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

Za vlastitu inačicu zadatka, potrebno je **izraditi sjedište Weba s dvije stranice** koje će predstavljati osnovno korisničko sučelje za sve predstojeće laboratorijske vježbe. Stranice će se razlikovati u temi, a 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 Sublime Text[10], Notepad2[11], Notepad++[12], jEdit[13] 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 tema i vrijede za sve zadatke, a uključuju sljedeće:

- treba koristiti naredbu obrade **DOCTYPE** i deklaraciju tipa dokumenta **HTML 4.01 Strict**[14] **-** daljnje su upute napisane za ovu normu

Napomena: Moguće je koristiti i specifikaciju **HTML5**, ali je u tom slučaju potrebno smisleno koristiti semantičke oznake za elemente stranice, a ne služiti se „labavijom“ specifikacijom kao „izlikom“ za pisanje kôda s manje ograničenja. Moguće je da sustav ORVVIS neće podržavati validaciju ove specifikacije, pa ju treba provjeriti posebnim validatorom (W3C).

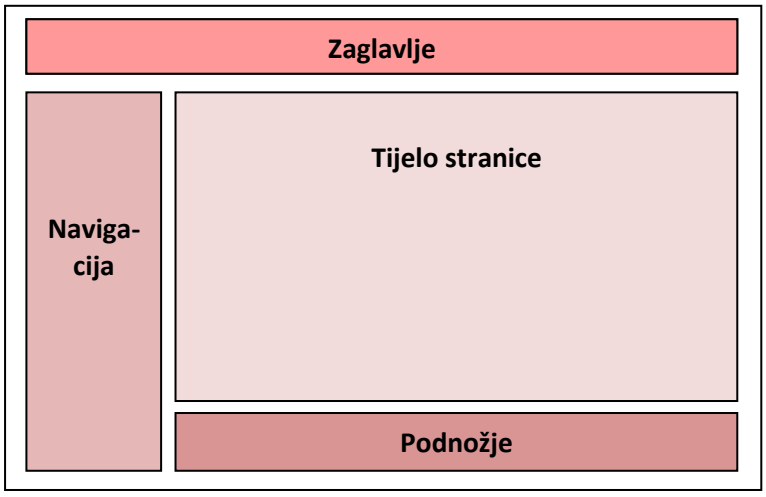
- stranice Weba trebaju biti napisane u normi za zapis tekstualnih dokumenata (kodnoj stranici) **UTF-8**
  - prilikom snimanja paziti da dokument bude **pohranjen** u odgovarajućem obliku zapisa - oblik UTF-8 bez *byte-order-mark*, BOM[15]
  - u dokumentu oznakom postaviti **pretpostavljenu kodnu stranicu** 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**

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

    - **poveznicu** izvedenu nad naslovnom slikom koja treba voditi na početnu stranicu sjedišta Weba
  - **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 stranicu predmeta OR (http://www.fer.unizg.hr/predmet/or)
    - 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:
  - podnožje može biti ostvareno ili kao na slici, ili kroz cijelu širinu



- 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 elemenata 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

Ako postoji opravdan razlog da validacija ne prolazi, potrebno je objasniti zašto.

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 tablica s barem tri stupca 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. Primjeri takvih sjedišta su http://www.pixabay.com, http://www.sxc.hu i http://www.rgbstock.com (provjerite licenciju pojedine slike!).

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 temi vježbe.**

Napomena: Prilikom izrade ove vježbe potrebno je definirati strukturu podataka, što je **objašnjeno u Prilogu na kraju ovog dokumenta.**

U ovoj vježbi su za definiranje polja obrasca **bitni sljedeći stupci** iz definicije strukture podataka za vlastitu temu vježbe (objašnjenja u Prilogu):

- **REDNI BROJ**
- **NAZIV**
- **VRIJEDNOSTI**
- **PRIMJER: SLOBODAN UPIS VRIJEDNOSTI ili SKUP DOZVOLJENIH VRIJEDNOSTI**
- **ELEMENT U OBRASCU na stranici za pretraživanje**

Napomena: Iz uputa u Prilogu vidljivo je 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 Tip broja telefona kao KRUŽIĆ ZA ODABIR.

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

- **barem 5 polja za unos teksta** parametara za pretraživanje
- **barem 1 grupu 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.

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“).

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: U ZIP datoteku uključite i datoteku s definicijom podataka (vidjeti predložak i objašnjenja u Prilogu).

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
- **Forms -** http://www.w3.org/TR/REC-html40/interact/forms.html
- **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
- **The W3C Markup Validation Service** - http://validator.w3.org
- The W3C CSS Validation Service - http://jigsaw.w3.org/css-validator

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

podatakanalazi se u tablici koju ste sami pripremili za 1. laboratorijsku vježbu. 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 stvoriti novi dokument u obliku HTML te u njega dodati podatke iz XML dokumenta. Otvaranjem u pregledniku podaci će se prikazati u obliku **tablice** unutar stranice Weba definirane dizajnom iz prethodne vježbe.

Pri izradi gramatike i testnog dokumenta koristimo se opisom strukture podataka koja ima 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.

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 alatom po izboru.

U drugom dijelu vježbe potrebno je definirati **transformaciju u jeziku XSL** za izradu stranice u HTML obliku, dohvat podataka iz XML dokumenta i popunjavanje redaka **tablice** unutar novoizrađene stranice. 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 učita 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 dokumenta
- 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
- 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 (*namespace*) 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
- 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/comp/r/tut-XSLT_1.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 moguća 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 druge razine i atributima treće razine** (razine su definirane u tablici strukture podataka). 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 predaje vježbe je na vlastitom prijenosnom računalu. Za to je na vlastitom prijenosnom računalu potrebno 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. Osim pojedinačno, instalacija poslužitelja Apache s podrškom za PHP za osobno računalo dolazi i u paketima dostupnima za različite verzije operativnog sustava Windows i distribucije sustava Linux:

- **XAMPP** - http://www.apachefriends.org
- **WampServer** - http://www.wampserver.com
- 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.

Napomena: Preporuča se **instalacija vlastitog okruženja**, bilo na svom prijenosnom računalu, bilo kao prijenosnu aplikaciju na USB-memoriju za rad na računalu u učionici. **Fakultetski poslužitelj** *pinus.cc.fer.hr* **će možda biti (zauvijek) ugašen tijekom semestra.** ☹

Napomena: Ako instalirate poslužitelj i PHP pojedinačno, podrška za PHP detaljno je objašnjena na adresi: http://www.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`*:*

Drugi način izvođenja rješenja vježbe je korištenjem fakultetskog poslužitelja `pinus.cc.fer.hr` i aplikacijske podrške koja je 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 izvođenje vježbe.

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:

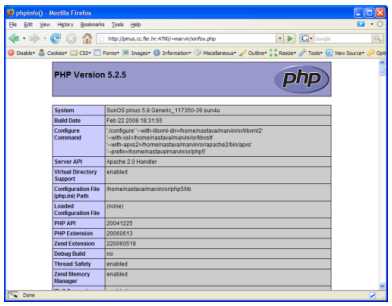
```
$> chmod 755 ~
$> chmod 755 public_html
$> chmod 644 public_html/index.html
```

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 – varijable - 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

4. laboratorijska vježba: Vanjski izvori podataka - pristup s poslužitelja

Cilj vježbe „Vanjski izvori podataka – pristup s poslužitelja“ je povezivanje vlastitih podataka sa strukturiranim podacima dobivenim iz vanjskih izvora podataka, kojima će se pristupati na strani poslužitelja, pomoću REST API-ja. Dobiveni podaci, u formatima JSON i XML, parsirat će se i pripremiti za objedinjavanje s postojećim podacima te prikazati unutar HTML-kôda. Kao primjer izvora podataka odabrani su *Facebook Graph API* za pristup podacima o javnim stranicama (*Facebook Pages*) te *OpenStreetMap* usluga za geokodiranje – pridruživanje geografskih koordinata zadanoj adresi/gradu.

Priprema za vježbu

U cilju pripreme za vježbu potrebno je **proučiti:**

- oblik zapisa **JSON**
- funkcije programskog jezika **PHP vezane za**
  - slanje HTTP-upita metodom GET – funkcija *file_get_contents* (i slično) http://www.php.net/file_get_contents
  - pretvorbu iz zapisa JSON u polje podataka http://www.php.net/manual/en/function.json-decode.php
  - čitanje podataka iz zapisa XML pomoću *SimpleXML* metoda http://www.php.net/manual/en/simplexml.examples-basic.php
- pravne uvjete korištenja, oblik HTTP-upita, parametara i odgovora za REST sučelja API-je vanjskih izvora podataka korištenih u ovoj vježbi
  - Facebook Graph API Quickstart https://developers.facebook.com/docs/graph-api/quickstart/ https://developers.facebook.com/docs/graph-api/using-graph-api/
  - Facebook Graph Explorer https://developers.facebook.com/tools/explorer
  - Nominatim - usluga za geokodiranje: http://wiki.openstreetmap.org/wiki/Nominatim

Napomena: Ako niste primjereno ili potpuno riješili zadatak iz 3. laboratorijske vježbe, sada ga trebate doraditi s obzirom da se većinom isti programski kod koristi za dohvat podataka XML datoteke i izradu izlaznog HTML koda.

Zadatak za vježbu

Potrebno je **nadopuniti PHP kôd** iz prethodne vježbe, kako bi se aplikaciji dodale nove funcionalnosti. Za svaki podatak iz postojeće XML datoteke potrebno je iz vanjskih izvora pribaviti dodatne podatke, te ih prikazati u tablici s rezultatima pretrage iz prethodne vježbe. Potrebni podaci koje treba dohvatiti su:

- podaci s javne *Facebook* stranice (*Facebook Pages*)
  - URL javne slike
  - geografska lokacija pojedinog podatka – ulica, grad, država (NE koristiti geografske koordinate, ako postoje)
  - URL službene stranice izvan Facebooka, ako postoji

- podaci o geografskim koordinatama (**geografska dužina i širina**) dobiveni uporabom usluge *Nominatim*, kojoj se kao parametar predaje geografska lokacija pronađena na Facebooku (adresa, grad, država)

Dio primljenih podataka potrebno je na sljedeći način prikazati u HTML tablici ostvarenoj u prethodnoj vježbi:

- javnu fotografiju potrebno je prikazati u jednom od stupaca tablice dodavanjem koda za umetanje URL-a slike
- geografske koordinate također trebaju biti vidljive u tablici (kao i adresa dohvaćena iz XML-a)
- URL vanjske službene stranice treba samo dohvatiti, ne i prikazati

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 i pretagu podataka
- slanje izdvojenih HTTP-zahtjeva uslugama *Facebook Graph* i *Nominatim*
- opcionalne promjene u programskom kodu

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

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

Datoteke su već napravljene u prošloj vježbi te ih je potrebno samo nadopuniti. 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.

Izrada Facebook aplikacije i oznake za pristup (Access Token)

Svrha uporabe *Facebook Graph* API-ja je dohvatiti dodatne podatke o već postojećim zapisima koji se nalaze u prethodno napravljenoj XML datoteci.

Iako su nama potrebni podaci o javnim stranicama javno dostupni, nakon promjena *Facebook Graph* API-ja od kraja 2015. godine, nije im više moguće pristupiti programski bez uporabe oznake za pristup (*Access Token*). Takve oznake mogu biti na razini **korisnika** *Facebooka* (*User Access Token*), na razini **vlastite aplikacije** (*App Access Token*) ili na razini **Facebook stranice** (*Page Access Token*). Za potrebe ove vježbe najprikladniji je odabir izrade vlastite aplikacije. Za to je potrebno:

1. Otvoriti korisnički račun na *Facebook Developers* stranici: https://developers.facebook.com (možete koristiti vlastiti *Facebook* korisnički račun, ili otvoriti privremeni)

Napomena: Kako bi se spriječile zlouporabe pri izradi zloćudnih aplikacija, *Facebook* zahtijeva verifikaciju korisničkog računa slanjem kôda na broj mobitela. Ako ne želite svoj broj mobitela povezati s *Facebookom*, pokušajte nabaviti neku privremenu SIM karticu i verificirati se preko nje. Jedan telefonski broj može se povezati samo s jednim korisničkim računom.

2. Izraditi novu aplikaciju (tip: *Website*)
3. Na upravljačkoj ploči aplikacije (*Dashboard*) pronaći *App ID* i *App Secret*
4. Prilikom slanja zahtjeva API-ju za dodatnim podacima, u HTTP-zahtjev uvijek nadodati parametar:
   `access_token=app_id|app_secret` (odvojiti ih uspravnom crtom |)

Napomena: Tajni kôd *app_secret* ne smije biti javno dostupan, te se koristi isključivo za izradu poslužiteljskih aplikacija. Gore navedeni način je „prečica“ umjesto dobivanja prave oznake za pristup, pročitajte više o tome na:
https://developers.facebook.com/docs/facebook-login/access-tokens

Uporaba Facebook Graph API-ja

Facebook Graph predstavlja podatke u obliku grafa, s vrhovima i bridovima. Za izradu vježbe nije potrebno dublje ulaziti u strukturu, već treba istražiti kako kreirati HTTP-zahtjeve koji će dohvatiti potrebne podatke te kako željene podatke pročitati iz odgovora. Postoji nekoliko načina pristupa, generiranjem čistih HTTP-zahtjeva (REST), uporabom biblioteka za *PHP*, *JavaScript*, *iOS* te *Android*, no ovdje ćemo koristiti HTTP (REST).

Svaka je javna stranica dostupna preko naziva stranice (ako postoji) ili preko *Facebook ID-a*. Ovaj podatak za svaki zapis iz XML datoteke treba ručno pronaći i zapisati ga u XML datoteku prije izrade ove vježbe. Naziv je moguće pronaći u URL-u odabrane *Facebook* stranice, a ID iz URL-a za sliku profila (i još nekoliko načina, ovise o trenutnoj izvedbi javnih *Facebook* stranica). Tako npr. stranica „Park Zrinjevac“ ima ID 179490168764441.

Pregled dostupnih podataka i testiranje upita, tj. HTTP-zahtjeva moguće je na stranici *Facebook Graph Explorer* (https://developers.facebook.com/tools/explorer). HTTP zahtjev tvori se od naziva poslužitelja, ID-a ili naziva stranice, dodatnih čvorova (nisu obavezni za vježbu) te dodatnih polja (npr. informacije o javnoj fotografiji). Polja se dohvaćaju navodeći ih nakon parametra `fields`, a više polja moguće je istovremeno dohvatiti dodavanjem zareza između naziva polja u zahtjevu. Dio polja dostupan je pregledom osnovnog čvora stranice, a dio treba dohvatiti zasebno. Tako je na primjer podatke o javnoj fotografiji Parka Zrinjevac moguće dohvatiti preko URL-a:

https://graph.facebook.com/179490168764441?fields=picture

Popis i objašnjenje polja moguće je dohvatiti dodajući parametar `metadata=1`.

Ova usluga nema mogućnost odabira formata vraćenih podataka, te **podatke vraća u formatu JSON**. Takve je podatke u PHP-u jednostavno pretvoriti u polje uporabom metode `json_decode`.

Uporaba usluge Nominatim

Web sjedište *OpenStreetMap* nudi otvorene društveno-generirane karte uz mnoštvo dodataka, poput *usluge Nominatim* za generiranje geografske lokacije (širina i dužina) iz određene adrese. Usluga je dostupna slanjem HTTP-zahtjeva na poslužitelj:
http://nominatim.openstreetmap.org/search

Parametrom `format` moguće je definirati izlazni format: HTML, JSON ili XML. Za ovu vježbu potrebno je **rezultate primiti kao XML**.

Parametrom `q` zadaje se upit, koji može biti zapisan i na drugi način (dodatno strukturiran, vidjeti upute). Upit je potrebno enkodirati u URL-zapis metodom `url_encode`.

Tako je upit lokacije Parka Zrinjevac, „Nikole Šubića Zrinskog, Zagreb, Croatia“:
http://nominatim.openstreetmap.org/search?q=Trg%20Nikole%20%C5%A0ubi%C4%87a%20Zrinskog,%20Zagreb,%20Croatia&format=xml

Odgovor može sadržavati više rezultata. Za ovu vježbu možete odabrati prvi rezultat, a po želji možete provjerom dodatnih parametara usporediti tip odgovora.

Rezultat uporabe usluge *Nominatim* u ovoj vježbi mora biti XML. Za rad s XML-om možete koristiti znanje iz prethodnih vježbi (DOM, XPath), no u jednostavnim slučajevima kao što je ovaj, često se koristi *SimpleXMLElement* razred, koji XML dokument pretvara u objekt s lako dostupnim XML elementima i atributima.

Napomena: Obavezno pročitajte uvjete uporabe korištenih API-ja i poštujte ih!

Ispitno gradivo vježbe

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

Primjeri pitanja:

- Objasnite parametre HTTP-zahtjeva upućenih uslugama.
- Objasnite primljene odgovore.
- Skicirajte komunikaciju prilikom obavljanja pretrage.
- Objasnite način umetanja podataka u HTML-kôd.

Poveznice i literatura

- **PHP manual:** http://docs.php.net/manual/en/
- **Learn REST: A Tutorial:** http://rest.elkstein.org/
- **Learn REST: A RESTful Tutorial:** http://www.restapitutorial.com/

5. laboratorijska vježba: Interaktivnost i dinamičnost elemenata stranice Weba, vanjski izvori podataka - pristup s klijenta

Cilj vježbe „Interaktivnost i dinamičnost elemenata stranice Weba, vanjski izvori podataka - pristup s klijenta“ 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. Također, u vježbi je potrebno povezati podatke dobivene iz vanjskih izvora podataka s dodatnim uslugama na klijentskoj strani – *mashup*. Za izradu dinamičkih elemenata stranice i *mashup* 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
- osnove pristupanja i promjene elemenata HTML dokumenta korištenjem objektnog modela dokumenta (DOM)
  - W3Schools JavaScript HTML DOM tutorial http://www.w3schools.com/js/js_htmldom.asp
  - 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
  -

Također, potrebno je proučiti osnove rada JavaScript Web API-ja za prikaz i upravljanje geografskim mapama te uvjete korištenja:

- Leaflet.js
  - Overview - http://leafletjs.com/
  - Quick start guide - http://leafletjs.com/examples/quick-start.html
  - Leaflet FAQ - https://github.com/Leaflet/Leaflet/blob/master/FAQ.md
- Uključivanje otvorenih skupova pločica (*tiles*) za prikaz karte
  - Primjeri skupova pločica – http://leaflet-extras.github.io/leaflet-providers/preview/

- Imenovanje pločica – http://wiki.openstreetmap.org/wiki/Slippy_map_tilenames (samo osnove)
- Uvjeti korištenja *OpenStreetMap* pločica – http://wiki.openstreetmap.org/wiki/Tile_usage_policy

Zadatak za vježbu

Postojeću stranicu rezultata pretrage potrebno je nadopuniti sljedećim mogućnostima:

- **promjena boje retka** prelaskom pokazivača miša preko pojedinog retka tablice s rezultatima
- **odjeljkom za prikaz detaljnih podataka** o pojedinom zapisu rezultata pretrage
- **prikazom geografske lokacije na karti**, ostvarenoj pomoću JavaScript biblioteke otvorenog kôda i javno dostupnih pločica za prikaz karte.

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.

Promjena boje retka

S obzirom da je stranica rezultata pretrage izvedena PHP skriptom `pretraga.php`, treba ju izmijeniti tako da se prelaskom pokazivača preko pojedinog retka, a time i pojedinog zapisa, promijeni boja toga retka. Za to je potrebno **registrirati događaj prelaska miša** `onmouseover` korištenjem obrade događaja (*event handler)* za svaki pojedini element. Na detektirani događaj se poziva JavaScript funkcija (definirana u vanjskoj datoteci sa sufiksom .js), na primjer na sljedeći način:

```
<tr onmouseover="promijeniBoju(this)”>
   <td> ...  </td>
   <td> ...  </td>
   ...
</tr>
```

Funkciju `promijeniBoju(moj_red_tablice)` potrebno je ostvariti u JavaScriptu, i promijeniti boju elementa svojstvom `style`:

```
moj_red_tablice.style.backgroundColor = "#336699"
```

Promjena područja stranice za prikaz detaljnih podataka

U drugom dijelu vježbe potrebno je **detaljne podatke** o odabranom osnovnom elementu **prikazati** na stranici. To znači da se postojeća stranica, koju izrađuje PHP skripta `pretraga.php` iz 3. laboratorijske vježbe, treba preurediti:

- izmijeniti tablicu za prikaz rezultata (dodati stupac „Akcija“, po potrebi smanjiti broj stupaca s podacima)
- dodati novo područje za prikaz detaljnih podataka
- omogućiti aktiviranje akcije prikaza detaljnih podataka u tom području.

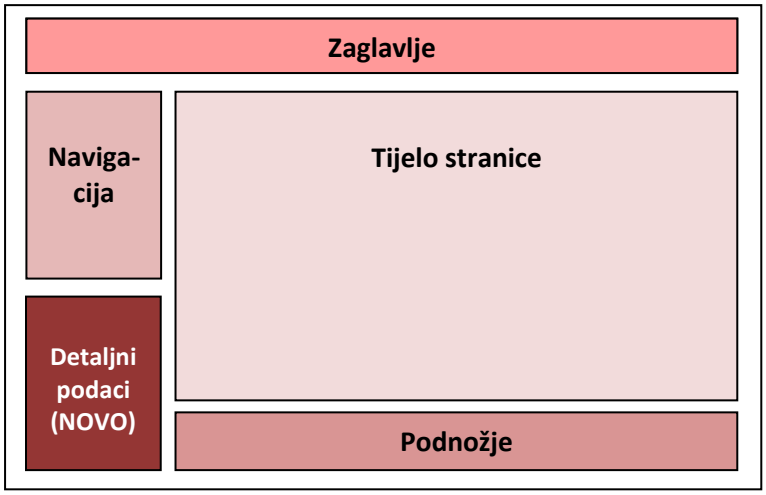
Za postizanje bolje uporabivosti stranice koristit će se objekt XMLHttpRequest i tehnika AJAX (*Asynchronous JavaScript and XML*) kako bi se izbjeglo nepotrebno osvježavanje stranice. Objekt XMLHttpRequest omogućuje asinkronu komunikaciju s poslužiteljem, slanje zahtjeva i dobivanje odgovora od poslužitelja 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 **puni** 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 novoj PHP skripti `detalji.php` `metodom GET` koja prima parametar `id` kao identifikator elementa čije detalje treba prikazati. Skripta dohvaća sve podatke iz XML datoteke 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. Ovaj zahtjev šalje se kroz objekt *XMLHttpRequest* (detaljno objašnjen u nastavku), zbog čega neće doći do ponovnog učitavanja stranice.

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. Radi jednostavnosti, ova skripta **neće preuzimati podatke** s *Facebook Pages.*

U PHP skripti `pretraga.php` potrebno je za svaki pojedini redak rezultata, dohvatom događaja `onclick` omogućiti poziv JavaScript funkcije koja će pripremiti zahtjev s argumentom `id` i poslati ga PHP skripti `detalji.php`.

U funkciji koja se poziva pri promjeni stanja XMLHTTPRequest 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 prikazane podatke. 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.

PHP skripta `detalji.php` **treba vratiti odgovor u obliku segmenta koda u jeziku HTML (ne cjelovite stranice!)**, 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 ...).

Izlaz iz skripte `detalji.php` je u obliku segmenta HTML koda koji sadrži druge detaljne podatke 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. Treba obratiti pažnju da se ne poremeti izgled cjelokupne stranice.

Objekt XMLHttpRequest

Objekt XMLHttpRequest služi za asinkronu komunikaciju s poslužiteljem i dohvat manje količine podataka slanjem zahtjeva metodom GET ili POST te primanjem odgovora. Tako se podaci mogu primiti 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.

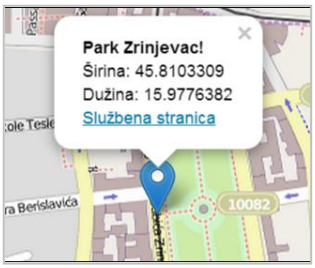
Svojstvo `onreadystatechange` služi za povezivanje funkcije koja se poziva pri promjeni stanja zahtjeva. U sinkronom načinu odaziv stranice je blokiran do dobivanja odgovora, dok se u asinkronom načinu nakon slanja zahtjeva izvođenje funkcije i rad sa stranicom nastavlja, a naknadno se pri promjeni stanja zahtjeva, odnosno prispijeća odgovora, poziva 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 jednostavnu komunikaciju s poslužiteljima izvan domene iz koje je stranica isporučena (*cross-domain scripting)*. Načine komunikacije izvan domene nećemo obrađivati na vježbama.

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);
      }
   }
}
```

Prikaz geografske lokacije na karti



Odabirom gumba „Detalji“ ili sličnog, uz zahtjev za dohvaćanje detalja o svakom zapisu, izvodi se i dio koda za **prikaz geografske lokacije na karti**. Dio stranice s kartom na početku nije vidljiv, a prikazuje se nakon **odabira gumba/poveznice za prikaz detalja**. Na karti je, nakon učitavanja, potrebno prikazati marker s traženom geografskom lokacijom, a u *oblačiću* je potrebno zapisati naziv zapisa (npr. „Park Zrinjevac“), njegove koordinate (geografska širina i dužina) te vezu na službenu stranicu, ako postoji.

Biblioteka *leaflet.js* služi za prikaz karte i upravljanje podacima. Kao što je opisano u tutorialu, na stranicu je potrebno uključiti CSS datoteku te samu biblioteku.

```
<link rel="stylesheet"
       href="http://cdn.leafletjs.com/leaflet-0.7.7/leaflet.css" />
<script src="http://cdn.leafletjs.com/leaflet-0.7.7/leaflet.js">
</script>
```

Nakon definiranja područja (`<div>`) gdje će karta biti prikazana, mjesto koje će biti prikazano na karti postavlja se naredbom poput ove:

```
var map = L.map(<div_id>).setView([<širina>, <dužina>],
          razina_povećanja);
```

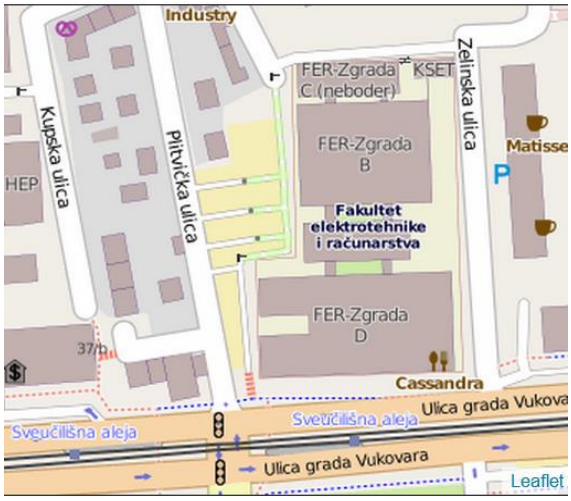
Pločice za prikaz karte nisu dio ove biblioteke, no postoje mnogi izvori pločica, s raznim uvjetima korištenja, besplatnih, komercijalnih, itd, koji imaju jednak način imenovanja i pristupa. Na stranici http://leaflet-extras.github.io/leaflet-providers/preview/ dostupan je primjer velikog broja otvorenih izvora pločica, s JavaScript kodom pripremljenim za uključivanje.

Primjer uključivanja *OpenStreetMap* pločica je:

```
var OpenStreetMap_Mapnik =
L.tileLayer('http://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', {
      attribution: '&copy; <a
href="http://www.openstreetmap.org/copyright">OpenStreetMap</a>'
      });
```

Napomena: Prilikom preuzimanja biblioteke te pločica za prikaz karte, potrebno je obratiti pažnju na autorska prava i licencije, te na stranicu uključiti propisano imenovanje.

Primjer različitih oblika pločica za lokaciju FER-a:



`OpenStreetMap.Mapnik`

`Stamen.Watercolor`

Na kraju je na željenu lokaciju na karti potrebno dodati marker te uz njega vezati oblačić s podacima:

```
var marker = L.marker([<širina>, <dužina>).addTo(map);
marker.bindPopup("Ovdje će doći naziv, koordinate i
                  URL!").openPopup();
```

S obzirom da će koordinate biti prikazane na taj način, možete ih ukloniti iz tablice. Zahtjev za detaljima nekog drugog zapisa treba osvježiti prikaz markera i oblačića.

Predaja vježbe

Rješenje vježbe se na poslužitelj predaju u obliku ZIP arhive koja sadrži sve datoteke potrebne za ispravan rad rješenja.

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.
- Objasnite način povezivanje događaja s JavaScript kodom
- Dinamički promijenite boju teksta nekog elementa
- Objasnite način rada s bibliotekom za prikaz karte

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
- OpenStreetMap Wiki - http://wiki.openstreetmap.org/

6. laboratorijska vježba: primjena API-ja po izboru

Cilj ove vježbe je samostalan rad s postojećim API-jima, prikladnim i primjenjivim za određenu svrhu. Rad uključuje odabir API-ja i prikladnog primjera, kojim se aplikacija razvijena za laboratorijsku vježbu proširuje uz dodatnu korist za krajnjeg korisnika, proučavanje dokumentacije i licenciranja te implementaciju jednostavnog primjera. **Ova vježba je neobavezna, a predaje se zajedno s 5. vježbom.**

Priprema za vježbu

U cilju pripreme za vježbu potrebno je:

- promisliti koji bi dodatni izvor podataka ili usluga bili prikladni za povećavanje korisnosti aplikacije izrađene u laboratorijskoj vježbi
- pronaći odgovarajući dostupan API
  - (jedna od) lista API-ja za inspiraciju: *ProgrammableWeb* http://www.programmableweb.com/apis/directory
- proučiti dokumentaciju za osnovno korištenje API-ja i uvjete korištenja

Zadatak za vježbu

Postojeću aplikaciju nadopuniti novom funkcionalnošću, ili novim podacima iz vanjskog izvora po vlastitom izboru, koristeći prikladan API, koji može biti bilo **na strani poslužitelja**, bilo **na strani klijenta**. Obratiti pažnju na uvjete korištenja, korisnost funkcionalnosti te uklapanje u dizajn stranice.

Predaja vježbe

Rješenje vježbe se na poslužitelj predaju u obliku ZIP arhive koja sadrži sve datoteke potrebne za ispravan rad rješenja.

Rješenje se predaje u istoj arhivi kao i 5. laboratorijska vježba.

Ispitno gradivo vježbe

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

Primjeri pitanja:

- Koju svrhu obavlja API?
- Koja je dodana vrijednost aplikacije uz korišteni API?
- Kako se API povezuje s aplikacijom?
- Koje još mogućnosti API posjeduje, a da nisu korištene u ovoj aplikaciji?
- Koji su uvjeti korištenja?

Licencija

Prilog – definiranje strukture podataka za izradu vježbe

U ovom prilogu prikazan je način definiranja strukture podataka za izradu vježbu. Iako su zadaci vježbe načelno isti, svaki student odabir svoju temu – područje primjene rješenja zadatka. Svaki student treba, ovisno o odabranoj temi vježbi, sam izraditi svoju tablicu po uzoru na primjer telefonskog imenika, po dostupnom predlošku.

Upute za izradu strukture

- redaka u tablici strukture (podataka o pojedinom zapisu) treba biti **barem 15**
- mora postojati jedinstveni identifikator pojedinog zapisa
- mora postojati *Facebook* ID/name – identifikator zapisa na Facebooku (trebate ga sami ručno pronaći)
- u tablici moraju postojati 4 hijerarhijske razine strukturiranih podataka
- moraju postojati barem 2 elementa brojnosti N
- moraju postojati barem 2 atributa
- barem 2 elementa/atributa ne moraju nužno postojati u određenom zapisu
- barem 3 atributa moraju biti vrijednosti SKUP
- barem jedan će biti predstavljen kao KVADRATIĆ ZA IZBOR
- barem jedan će biti predstavljen kao KRUŽIĆ ZA ODABIR
- barem jedan će biti predstavljen kao IZBORNIK ZA VIŠESTRUKI ODABIR

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 UPIS VRIJEDNOSTI ili SKUP DOZVOLJENIH VRIJEDNOSTI – označava primjer tekstualnog podatka ili 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

Primjer podataka

- U primjeru se nalazi dio podataka za telefonski imenik zapisanih u XML-u, nastao iz definicije strukture koja se nalazi na sljedećoj stranici.

```
...
<imenik>
    <osoba>
        ...
    </osoba>
    <osoba oib="12345678901" kategorija="prijatelj">
        <ime>Marko</ime>
        <prezime>Ferković - Enter</prezime>
        <telefon tip="fiksni">
            <broj pozivni="01">6129999</broj>
        </telefon>
        <telefon tip="mobilni">
            <broj pozivni="098">6129999</broj>
        </telefon>
        <adresa>
            <ulica>Unska</ulica>
            <kbroj>3</kbroj>
            <mjesto pbroj="10000">Zagreb</mjesto>
            <drzava>Hrvatska</drzava>
        </adresa>
        <email>marko.ferkovic_enter@fer.hr</email>
        <email>marko.ferkovic_enter@gmail.com</email>
    </osoba>
    <osoba>
        ...
    </osoba>
</imenik>
```

Primjer definicije strukture podataka: 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**							

Otvoreno računarstvo 2015/16. – struktura podataka

Ime i prezime studenta: _____ Tema vježbe: _____

REDNI BROJ	NAZIV	HIJERARHIJSKA RAZINA (1-4)	BROJNOST (1 ILI N)	ELEMENT ili ATRIBUT	OBAVEZNOST POSTOJANJA (DA ili NE)	VRIJEDNOSTI (NEMA, DA ili SKUP)	PRIMJER: SLOBODAN UPIS VRIJEDNOSTI ili SKUP DOZVOLJENIH VRIJEDNOSTI	ELEMENT U OBRASCU na stranici za pretraživanje (POLJE ZA UNOS, KVADRATIĆ ZA IZBOR – checbox, KVADRATIĆ ZA ODABIR – radio button, IZBORNIK ZA VIŠESTRUKI ODABIR – multiple select)
1	*osoba/klub/park/grad…*	1	N	E	DA	NEMA	…	
2								
3								
4								
5								
6								
7								
8								
9								
10								
11								
12								
13								
14								
15								
16								
17								
18								
19								
20								

Način ispunjavanja tablice objašnjen je u Uputama za vježbe.