



LAIMA RIČKUTĖ

Tinklalapių kūrimas, dizainas ir valdymas

Mokomoji priemonė



Projektas „Socialinių mokslų kolegijos vykdomų studijų programų internacionalizacija kuriant atvirą aukštąją mokyklą užsienio šalių studentams“ (projekto Nr. VP1-2.2-ŠMM-07-K-02-074) finansuojamas pagal 2007–2013 m. Žmogiškųjų išteklių plėtros veiksmų programos 2 prioritetą „Mokymasis visą gyvenimą“

LAIMA RIČKUTĖ

TINKLALAPIŲ KŪRIMAS, DIZAINAS IR VALDYMAS

Mokomoji priemonė

VšĮ Socialinių mokslų kolegija
Klaipėda
2013

Laima Ričkutė

TINKLALAPIŲ KŪRIMAS, DIZAINAS IR VALDYMAS
Mokomoji priemonė

Recenzavo: Darius Šoparas, direktorius, UAB „Creative Partner“.

Rekomenduota leisti:

VšĮ Socialinių mokslų kolegijos Akademinės Tarybos 2013 08 28 d. protokolas Nr. 4.

Kalbą redagavo UAB „Fortuna Publica“

Maketavo Aida Zybartė

Viršelio dailininkė Aida Zybartė

Projektas „**Socialinių mokslų kolegijos vykdomų studijų programų internacionalizacija kuriant atvirą aukštąją mokyklą užsienio šalių studentams**“ (projekto Nr. VP1-2.2-ŠMM-07-K-02-074) finansuojamas pagal 2007–2013 m. Žmogiškųjų išteklių plėtros veiksmų programos 2 prioritetą „Mokymasis visą gyvenimą“

Laima Ričkutė

**TINKLALAPIŲ KŪRIMAS,
DIZAINAS
IR VALDYMAS**

Mokomoji priemonė

UAB „Creative partner“

Laimos Ričkutės mokomosios priemonės "WEB KŪRIMAS, DIZAINAS IR VALDYMAS"

RECENZIJA

2013.08.15, Klaipėda

Laimos Ričkutės mokomojoje priemonėje "WEB KŪRIMAS, DIZAINAS IR VALDYMAS" išsamiai apžvelgiama šiuo metu labiausiai augančios technologijų erdvės – interneto svetainių kūrimo priemonių sąrašas, siekiant nesudėtingomis priemonėmis sukurti sprendimus, veikiančius šioje erdvyje.

Norėtūsi pagirti autorę, jog temos dėstymas nuoseklus, pradedant atviraisiais formatais, pasauliniais standartais bei interneto istorija, tai būtina žinoti norint suprasti, kaip vystėsi technologijos. Palaipsniui tolimesnėje knygos medžiagoje labai praktiškai padedama planuoti web svetainės kūrimo procesą, mokoma apie skirtingus svetainių tipus ir kokias praktines užduotis turi išspręsti kūrėjas, kad galėtų tinkamai užbaigti užduotis. Labai svarbu, kad techninės žinios derinamos su rinkodaros patarimais, tokiais kaip internetinis marketingas ar optimizavimas paieškos sistemos, leidžiančiais sukurti naudingą ar rinkoje vertę turintį sprendimą. Didelis mokomosios medžiagos privalumas yra tas, jog knygoje aprašomos pačios naujausios technologijos, tokios kaip HTML5, JavaScript bei PHP kalbos. Be to, aptariamas ir visų technologijų panaudojimas naudojant turinio valdymo sistemas, kurios leidžia sukurti kompleksinius sprendimus paprasčiau ir greičiau valdant duomenis dinamiškai. Susipažinęs su puikiai pateiktu skyriumi "Web dizaino projektas" studentas turės pakankamai žinių, kad galėtų atliktį web svetainės kūrimo darbus nuo pradžių iki pabaigos.

Kiekvienos temos pabaigoje pateikiami klausimai diskusijai, žinių patikrinimui arba tolimesniam temos gilinimuisi bei terminų žodynėlis. Visa tai leidžia šia mokomąją medžiagą drąsiai rekomenduoti kaip puikų žinių šaltinį būsimiems interneto svetainių kūrėjams.

Įmonės pavadinimas
Veiklos adresasUAB „Creative partner“
H. Manto g. 7 / Mažvydo al. 2, KlaipėdaDirektorius
Mobilus telefonas
El. paštasDarius Šoparas
8 670 27 384
darius@cpartner.lt

TURINYS

IVADAS	11
1. INTERNETAS	13
1.1. Internetas ir interneto protokolai	13
1.2. Elektroninių dokumentų formatai	14
1.2.1. Atviri formatai	15
1.2.2. Uždari formatai	16
1.2.3. Atvirieji standartai internete	17
1.2.4. Svetainės atitiktis atviriesiems standartams	19
1.3. „Web 2.0“ saityno savybės	21
1.4. HTML istorija	23
1.5. Žiniatinklio atsiradimas	24
1.6. Interneto naršyklės	25
Klausimai ir užduotys	27
2. INTERNETO SVETAINIŲ KŪRIMAS	28
2.1. Interneto svetainių tipai	28
2.2. Interneto svetainių kūrimo principai	30
2.2.1. Tinklalapio planavimas	32
2.2.2. Tinklalapio ir jo dizaino kūrimas	35
2.2.3. Keletas patarimų prieš kuriant naują tinklalapį	37
2.2.4. Lankytojų tipai	43
2.2.5. Tinklalapio funkcionalumas	43
2.2.6. Tinklalapio talpinimas	44
2.3. Tinklalapio struktūra	45
2.3.1. Hierarchinė struktūra	48
2.3.2. Linijinis maketas	49
2.3.3. Tinklinė struktūra	50
2.4. Tinklalapių kūrimo priemonės	51
2.5. Tinklalapių etiketas, priežiūra ir autorystės teisės	52
2.6. Internetinis marketingas	52
Klausimai ir užduotys	56

3. HTML IR CSS STANDARTAI	57
3.1. HTML standartas	57
3.1.1. HTML5 standartas	58
3.1.2. HTML5 standarto privalumai	60
3.2. HTML redaktoriai	63
3.2.1. Ar verta mokėti HTML?	66
3.2.2. Dokumentų formato ir perdavimo standartai	66
3.3. HTML pagrindai	69
3.3.1. HTML elemento konstrukcija	70
3.3.2. Vaizdas ar turinys	71
3.3.3. Apipavidalinimas ir stilius	72
3.4. HTML dokumento struktūra	72
3.4.1. Paprasčiausias HTML dokumentas	72
3.4.2. Svarbiausi HTML dokumento elementai	74
3.4.3. Teksto formavimas	76
3.4.4. Nuorodos	80
3.4.5. Sąrašai ir apibrėžimai HTML dokumente	83
3.4.6. Spalvų kodai ir vardai	86
3.4.7. Papildomi simboliai HTML dokumente	86
3.4.8. Lentelės HTML dokumente	87
3.4.9. Grafiniai žemėlapiai	95
3.5. Dinaminis HTML	97
3.6. Dažnai pasitaikančios HTML klaidos	98
3.7. HTML5 taikymas	100
3.7.1. Naujos struktūrinės žymės	102
3.7.2. Vaizdo ir garso failai	106
3.8. Pakopiniai stiliai CSS	108
3.8.1. Tinklalapio elementų išdėstymas	110
3.8.2. CSS sintaksė	111
3.8.3. Populiariausi selektoriai	112
3.8.4. CSS įterpimas į HTML	115
3.8.5. CSS hierarchija	118
3.8.6. CSS stilių parametrai	122
3.9. Svetainių atitikimo standartams testavimas ir taisymas	133
Klausimai ir užduotys	137

4. TINKLALAPIŲ KŪRIMO PRIEMONĖS IR VALDYMAS	139
4.1. „JavaScript“ skriptų kalba	139
4.1.1. Pagrindiniai „JavaScript“ elementai	140
4.1.2. Standartiniai „JavaScript“ objektai	147
4.1.3. „JavaScript“ funkcijos	149
4.1.4. Masyvai	155
4.2. Dažniausiai naudojamos tinklalapių kūrimo priemonės	163
4.3. PHP programavimas	167
4.3.1. Žiniatinklio serveris „Apache“	169
4.3.2. Duomenų bazių valdymo sistema „MySQL“	170
4.3.3. „MySQL“ valdymo įrankis phpMyAdmin	171
4.3.4. Laisvasis įrankis PHP	172
4.3.5. PHP „MyAdmin“ duomenų bazių administravimo sistema	176
4.3.6. „MySQL“ duomenų bazių apdorojimo sistema	176
4.4. Optimizavimas paieškos sistemoms	177
4.4.1. Tinklalapių reitingavimas	177
4.4.2. Teisingų raktažodžių parinkimas	178
4.4.3. Raktiniai žodžiai „Google“ įrankių pagalba	179
4.4.4. HTML5 gali padėti SEO	181
4.5. Tinklalapio šablonai	185
4.6. Turinio valdymo sistemos	187
4.6.1. TVS struktūra ir naudojimas	189
4.6.2. Vartotojo sąsaja	191
4.6.3. TVS privalumai	192
4.6.4. Turinio valdymo sistema „Joomla“	194
Klausimai ir užduotys	201
5. TINKLALAPIŲ DIZAINAS	203
5.1. Tinklalapių dizaino projektas	203
5.2. Interneto svetainės dizaino formulė	204
5.3. Tinklalapio dizaino kūrimo procesas	207
5.4. Tinklalapio dizaino elementai	209
5.5. Tinklalapio dizaino spalvos	211
5.5.1. Spalvų teorija	211
5.5.2. Spalvų ratas	213
5.5.3. Spalvų tipai	216

5.5.4. Spalvų deriniai	217
5.5.5. Spalvų derinimo principai	219
5.6. Tipografijos svarba tinklalapiuose	222
5.6.1. Kontrastas	223
5.6.2. Hierarchija ir erdvė	223
5.7. Dizaino taisyklių naudojimas tinklalapių dizaine	224
5.7.1. Aukso proporcijos naudojimas	224
5.7.2. Trijų trečiųjų taisyklės taikymas	225
Klausimai ir užduotys	227
TERMINŲ ŽODYNĖLIS	228
LITERATŪRA IR ŠALTINIAI	231
PRIEDAI	233
1 priedas. Reikšmingos interneto istorijos datos	233
2 priedas. Specialūs simboliai	236

ĮVADAS

Kasdien internetas vis labiau skverbiasi į žmogaus gyvenimą, palengvindamas jo buitį ir darydamas jį patogesne. Internetas atlieka bibliotekos, parduotuvės, mokymo ar valdžios įstaigų vaidmenis. Pasitelkus internetą, galima nusipirkti ar parduoti įvairių daiktų, susirasti knygų, gauti reikalingas pažymas, išmokti užsienio kalbų ar rasti kitų naudingų patarimų net neišėjus iš namų. Internetas suteikia galimybę bendrauti ir dalintis informacija su draugais, esančiais įvairiausiose pasaulio šalyse.

Internetas – pasaulinis (tarptautinis, globalus) kompiuterių tinklas (angl. *Wide Area Network – WAN*), jungiantis virš 100.000 kompiuterių tinklų visame pasaulyje tam tikrais ryšio kanalais (optiniu kabeliu, palydoviniu ryšiu) ir bendraujančių pagal vieningas taisykles (protokolus). Tai milžiniška ir begalinė informacijos saugykla. Vieni interneto puslapiai išnyksta, todėl sukuriami ir internete talpinami nauji puslapiai.

Interneto svetainė (angl. *website, site*) – daugybė tarpusavyje hipernuorodomis susietų interneto puslapių (tinklalapių). Tinklalapių kūrimo terminas naudojamas bet kokiai veiklai susijusiai su tinklalapio, svetainės kūrimu žiniatinklui ar intranetui nusakyti. Į šį terminą gali įeiti e. komercijos verslo kūrimas, tinklalapio dizainas, tinklalapio turinio kūrimas, specializuotas programavimas ar serverio paruošimas ir priežiūra. Tinklalapių profesionalams tinklalapių kūrimas gali siekti nuo paprasto statinio puslapio su paprasta tekstine informacija sukūrimo iki pačių sudėtingiausių internetinių sistemų.

Didelėms įmonėms ir organizacijoms gali prireikti ištisų tinklalapių kūrimo komandų, kurias sudarytų šimtai žmonių. Mažoms įmonėms gali pakakti vien puslapio administratoriaus, papildomas dizainerio pareigas einančio darbuotojo ar informacinių sistemų techniko. Visais atvejais, kuriant bet kokio dydžio svetainės projektus ar tiesiog prižiūrint įmonės tinklalapį, reikalingos tam tikros tinklalapių kūrimo ir valdymo žinios bei įgūdžiai.

Šioje metodinėje priemonėje pateikiamos svarbiausios žinios apie internetą, interneto protokolus, elektroninių dokumentų formatus, aprašomi interneto svetainių tipai, kūrimo principai, priemonės ir valdymas, nagrinėjamas svetainių dizaino kūrimo procesas ir principai, dizaino spalvų parinkimas.

1. INTERNETAS

1.1. Internetas ir interneto protokolai

Interneto pradžia siejama su 1960-aisiais metais, kai JAV mokslo tyrimų institutai, siekdami keistis mokslinių tyrimų rezultatais, pradėjo jungti kompiuterius tarpusavyje. Kompiuterių tinklo funkcionavimą bei jo galimybių išplėtimą finansavo ir rėmė JAV gynybos ministerija. 1970 m. buvo sukurtas keitimosi kompiuterių duomenimis tinklo projektas, kuris pradžioje jungė tik karines organizacijas, vėliau į jį pradėjo jungtis ir mokymo įstaigos. Dar vėliau į tinklą pradėjo jungtis ne tik JAV, bet ir įvairios organizacijos iš viso pasaulio. Normaliam tinklo veikimui, 1973 m. buvo sukurtos naujos duomenų perdavimo taisyklės, kurios leido neriboti naudotojų skaičiaus ir užtikrinti sklandų duomenų perdavimą. 1973 m. interneto tinklas tapo pasauliniu tinklu.

Lietuvoje interneto atsiradimo pradžia siejama su Lietuvos mokslo ir studijų kompiuterių tinklo (LITNET) veiklos pradžia, kai 1991 m. buvo sukurtas palydovinio ryšio kanalas tarp Oslo (Norvegija) universiteto bei Matematikos ir informatikos instituto Vilniuje.

Šiandien internetas – sudėtinga mažų ir didelių kompiuterių tinklų struktūra. Jis sudarytas iš tarpusavyje sujungtų daugelio visuotinių ir vietinių kompiuterių tinklų su milijonais serverių, kurie kaupia ir teikia įvairią informaciją bei paslaugas.

Nors pradžioje internetas nekėlė didelio pasitikėjimo, buvo kaltinamas nesaugumu, mėgėjiškumu ir kitais trūkumais, jis buvo plečiamas eksponentine progresija, nes buvo laisvas, visus priimantis ir nieko nediskriminuojantis. Tai buvo naujas verslo modelis, pasaulinis tinklas, neturintis savininko, nereguliuojantis kainų, neparduodantis licencijų, nereikalaujantis specialios įrangos, egzistuojantis pagal bendruomenės kuriamas taisykles, laisvas (nemokamas) tinklas, apie kurį sukasi didžiausi pasaulio pinigai.

Internetas (tarptautinis tinklas, angl. *inter* + *net*) – pasaulinė kompiuterių tinklų sistema, jungianti visuotinius ir vietinius kompiuterių tinklus. Kompiuterių tinklai jungiami įvairiomis ryšio linijomis ir maršrutizatoriais, kurie iš vieno tinklo gautus duomenų paketus nukreipia į kitą tinklą. Ryšiui tarp kom-

piuterių naudojamas TCP/IP (angl. *Transmission Control Protocol/Internet Protocol*) protokolas, kuris užtikrina interneto veikimą.

Interneto protokolas – taisyklės, kurios pateikia duomenų mainų tarp dviejų kompiuterinių sistemų internete būdą. Kiekviena kompiuterių sąveika tinkle turi būti aprašoma protokoliais. Kompiuteriams susijungiant vieni protokolais sudaro sąlygas veikti kitiems protokolams. Tokiai sudėtingai sistemai aprašyti sukurtas OSI modelis (angl. *Open Systems Interconnection Reference Model*).

Interneto standartus kuria nemažai organizacijų, bet aktyviausiai tuo užsiima IETF (angl. *The Internet Engineering Task Force*). Tai didelė ir atvira tarptautinė bendruomenė, kurios nariai yra tinklų kūrėjai, operatoriai, įrangos gamintojai ir tyrėjai.

1.2. Elektroninių dokumentų formatai

Elektroniniai dokumentai yra kompiuterinės bylos (dvejetainės informacijos srautas). Kompiuterinės bylos (el. dokumento) formatas – tai būdas (taisyklių rinkinys) skirtas apibrėžti, kokia informacija ir kaip yra pateikiama kompiuterinėje byloje. Formatas taip pat yra standartas. Tarptautinių žodžių žodyne rašoma, kad standartas – tai kompetentingų organizacijų priimtas ir patvirtintas normatyvinis dokumentas, nustatantis produkcijos, technologinių procesų, metodų, sąvokų, simbolių arba kitų objektų privalomas normas, taisykles ir reikalavimus jiems, siekiant optimalios tvarkos apibrėžtoje situacijoje.

Standartas – tai bendras susitarimas dėl kažko, nustatant normas, taisykles ir reikalavimus.

Pagal tai, kas ir kaip gali susitarti ir kaip tie susitarimai bus skelbiami bei pagal jų veikimo principus, standartai gali būti skirstomi į atvirosius (laisvus) ir uždarus (kurie dažniausiai priklauso vienam gamintojui arba gamintojų grupei). Atvirosius (laisvus) dokumento formatus (standartus) bet kas gali laisvai naudoti, matyti jų aprašymus (specifikacijas) bei reikšti pasiūlymus dėl jų tobulinimo.

Pagrindiniai atvirųjų formatų principai:

- *Prieinamumas*. Jie yra laisvai prieinami visiems (perskaitymo ir įgyvendinimo prasme).

- *Padidina galutinio vartotojo pasirinkimo laisvę.* Atvirieji standartai sukuria sąžiningą, konkuruojančią rinką standarto realizavimams. Jie „nepriša“ vartotojo prie konkretaus tiekėjo ar tiekėjų grupės.
- *Nemokami.* Atvirąjį standartą galima laisvai realizuoti visiems, už tai negali būti imamas mokestis (mokestis gali būti imamas tik už atitikimo standartui patikrinimą – sertifikavimą).
- *Jokios diskriminacijos.* Atvirieji standartai bei organizacijos, kurios juos administruoja negali teikti pirmenybės vienam realizuotojui prieš kitą dėl jokių kitų priežasčių išskyrus techninius pateiktos realizacijos atitikimus standartui. Sertifikavimo organizacijos turi sudaryti galimybę patikrinti atitikimą standartų ir pigių bei nemokamų įgyvendinimų (pvz., pasaulinio žiniatinklio konsorciumas leidžia bet kam nemokamai patikrinti (X)HTML ar CSS dokumentų atitiktį standartams).

1.2.1. Atviri formatai

JPEG formatas

JPEG formatas – atviras elektroninių paveikslėlių formatas, standartizuojamas „Joint Photographic Experts Group“ (www.jpeg.org). Kadangi tai yra atviras formatas, bet kas gali nemokamai sužinoti formato specifikacijas ir pagaminti produktą, kuris bus 100 % suderinamas su JPG formatu. Todėl dauguma programinės (interneto naršyklės, el. nuotraukų apdorojimo programos) ir aparatūrinės (pvz., fotoaparatai, mobiliųjų telefonų) įrangos gamintojų naudoja JPEG formatą nuotraukoms ar kitiems elektroniniams paveikslėliams. Vartotojai gali laisvai rinktis, kurį įrankį nori naudoti darbui su JPEG paveikslėliais, nes įvairių gamintojų programinė ir aparatūrinė įranga yra visiškai suderinama su šiuo formatu. Taip pat yra daug laisvos programinės įrangos (GIMP, „Mozilla“, „OpenOffice.org“ ir kt.), kuri visiškai palaiko šį formatą.

HTML/XHTML formatas

HTML (angl. *HyperText Markup Language*) yra pasaulinio žiniatinklio konsorciumo (www.w3.org) standartizuojamas atviras formatas. Bet kas gali

nemokamai sužinoti HTML specifikacijos, patikrinti konkretaus HTML dokumento atitiktį standartui <http://validator.w3.org> puslapyje. XHTML yra pavadintas „išplečiamas HTML“ – tai naujos kartos HTML formatas.

HTML formato privalumai – labai didelis paplitimas tarp įvairios programinės įrangos gamintojų, nepriklausomumas nuo jokios konkrečios kompanijos ar uždaros kompanijų grupės (standartizuojamas www.w3.org konsorciui), taip pat geras HTML formato palaikymas laisvoje programinėje įrangoje („Mozilla“, „Firefox“, „NVU“, „OpenOffice.org“ ir kt.).

„OpenDocument“ formatas

„OpenDocument“ (angl. *Open Document Format for Office Applications*) – tarptautinės organizacijos OASIS (www.oasis-open.org) standartizuojamas biuro dokumentų formatas, sukurtas XML formato pagrindu. Universaliausias iš visų (atvirų ir uždarų) biuro dokumentų formatų, turi daugiausia galimybių – tinka visų tipų (teksto, grafikos, elektroninių lentelių, prezentacijų, grafikų, matematinių formulių, pildomų formų, duomenų bazių) biuro dokumentų saugojimui, turi skaitmeninio parašo palaikymą ir pan. Palaikomas ir naudojamas smulkių ir stambių programinės įrangos gamintojų produktuose – SUN, IBM, „Novell“, „Oracle“, KDE ir t.t. Pripažintas Europos Komisijos, Europos Sąjungos institucijų bei kai kurių ES šalių narių. Planuojama, kad 2005 metų pabaigoje „OpenDocument“ formatas taps ISO (angl. *International Standards Organization*) standartu.

PDF (angl. *Portable Document Format*) – „Adobe“ kompanijai priklausantis atvirasis formatas labiausiai tinka praktiškai bet kokio tipo neredaguojamiems dokumentams publikuoti. Gana plačiai paplitęs, yra pakankamai laisvos programinės įrangos (pvz., „OpenOffice.org“, „PDFCreator“) dokumentų konvertavimui į PDF formatą.

1.2.2. Uždari formatai

Uždaras formatas (standartas) – dažniausiai vieno gamintojo arba uždaros gamintojų grupės naudojamas standartas, kurio aprašymas (specifikacija) nėra paskelbtas viešai ir kuriuo negalima laisvai pasinaudoti. Uždari formatai neleidžia keistis informacija (pvz., elektroniniais dokumentais) tarp skirtingų

gamintojų ir nesuteikia galimybės vartotojui laisvai rinktis. Uždari formatai kenkia konkurencijai, o tai paveikia aukštesnius programinės įrangos naudojimo kaštus.

Naudojant vien uždarus dokumentų formatus informacijos mainams, kyla tokios problemos: nepriklausomi programinės įrangos kūrėjai negali sukurti programinės įrangos, šie dokumentų formatai yra „Microsoft“ (JAV kompanijos) nuosavybė, skatinamas ir propaguojamas vieno gamintojo produktų naudojimas, neužtikrinamas laisvas priėjimas prie viešosios informacijos neturint „Microsoft“ produktų – piliečiai bei įmonės yra priversti pirkti arba nelegaliai naudoti šiuos produktus.

Praktinė atvirųjų formatų nauda:

- dokumentai „OpenDocument“, HTML ar PDF formatais dažniausiai užima gerokai mažiau vietos nei „MS Office“ formatai, todėl internetu galima greičiau atsisiųsti ar nusiųsti;
- atvirieji formatai yra saugesni ir atsparesni virusams, pvz., konvertavus virusu užkrėstus „MS Office“ dokumentus į „OpenDocument“, HTML ar PDF formatą, virusas tampa neveiksmingas – „OpenDocument“ bei PDF formatai turi integruotą skaitmeninio parašo galimybę.

1.2.3. Atvirieji standartai internete

Ekonominis atvirųjų standartų naudojimo efektas yra toks, kad jie sukuria sąžiningą, konkurencingą aplinką, skatina konkurenciją tarp tiekėjų ir tokiu būdu sudaro geresnes sąlygas naudotojui.

Atvirieji standartai, aprašantys interneto technologijas, lėmė interneto populiarumą ir leido šioms technologijoms nugalėti konkuruojančias nuosavas technologijas. Tik šių technologijų atvirumas leido atsirasti visuotinai techninių, programinių priemonių bei paslaugų rinkai. Iki tol egzistavę informaciniai kompiuterių tinklai arba perėjo prie interneto technologijų, arba išnyko.

Atvirųjų standartų dominavimas leidžia naudotojui laisvai rinktis elektroninio pašto serverius, elektroninio pašto programas, žiniatinklio serverius, naršykles ir beveik nesirūpinti suderinamumo klausimais.

HTML ir HTTP technologijos atsirado ankstyvaisiais 1990-aisiais metais neformalių specifikacijų pavidalu. Šios specifikacijos buvo standartizuotos tuo tikslu susikūrusios organizacijos W3C, kuri 1995 metais išleido HTML 2.0 standartą. Standarte buvo pabrėžiama HTML, kaip teksto struktūros žymėjimo kalba, o paties dokumento išvaizdos aprašymo savybės buvo gana ribotos.

„Netscape“ ir „Microsoft“ kompanijos, siekdamos diferencijuoti savo produktus, į juos ėmė dėti su standartais nesusijusių savybių. Dažniausiai tos savybės buvo skirtos vizualių interneto puslapių savybių žymėjimui. Dėl šios priežasties, vienai naršyklei sukurtas puslapis buvo visai kitaip rodomas kitų naršyklių. Siekiant išspręsti atsiradusius nesuderinamumus buvo išleista HTML 3.2 versija, įteisinusi daugelį tuo metu praktiškai naudotų vizualaus formavimo priemonių. HTML 4.01 ir XHTML 1.1 standartai pabrėžia dokumento struktūros žymėjimą, bet leidžia kurti išvaizdžius puslapius HTML dokumentą papildant CSS formato stiliaus schema.

Dažniausiai internetinių svetainių svarbiausios funkcijos yra informacijos sklaida ir paslaugų teikimas, todėl viena svarbiausių jų savybių yra sąveika (angl. *interoperability*). Labai svarbu, kad šios svetainės būtų prieinamos kuo platesniam naudotojų ratui: gyventojams, įmonėms, taip pat ir keičiantis informacija su kitomis institucijomis. Atvirųjų standartų laikymasis užtikrina, kad svetainė bus pasiekiamą žmonėms su negalia, besinaudojantiems tekstinėmis naršyklėmis su kalbos generatoriais ar Brailio terminalais. Be to, atvirųjų standartų laikymasis leidžia pasiekti, kad svetainėje esanti informacija būtų lengvai gaunama per šiuolaikinius mobiliuosius telefonus ir kitokius mobilius įrenginius, indeksuojama interneto paieškos sistemų, prieinama automatizuotoms priemonėms, naudojančioms svetainėje pateiktą informaciją.

Technologijų kaita internete yra labai greita, todėl atvirųjų standartų laikymasis suteikia lygias teises visiems naudotojams, nes nesusieja su konkrečiu gamintoju. Be to, standartų besilaikanti svetainė veiks su ateityje pasirodysiančiomis naršyklėmis, o šiuolaikinių naršyklių nestandartines savybes naudojančias svetaines ateityje gali tekti perdaryti.

1.2.4. Svetainės atitiktis atviriesiems standartams

Bendros sąvokos

HTML (angl. *HyperText Markup Language*) – hiperteksto dokumentų aprašymo kalba. HTML yra vienas iš svarbiausių žiniatinklio atvirųjų standartų.

XHTML (angl. *The Extensible HyperText Markup Language*) tai HTML kalbos variantas, pagrįstas atitikimu XML kalbos reikalavimų. XHTML dokumentus galima apdoroti daugybe egzistuojančių įrankių, skirtų darbiui su duomenimis XML formatu. Iš XHTML pašalintos nereikalingos HTML dalys, įdiegtas papildomas funkcionalumas (tobulesnis formų veikimas).

XML (angl. *Extensible Markup Language*) – išplečiama dokumentų aprašymo kalba. XML – tai formatas, leidžiantis struktūrizuoti tekstiniu pavaldalu aprašyti duomenis.

CSS (angl. *Cascading Style Sheets*) – pakopinių stilių schemos. CSS leidžia tiksliai aprašyti HTML dokumentų išvaizdą bei išdėstymą.

Svetainė, tinklalapis – vienai temai skirti, turintys bendrą temą HTML dokumentai.

WWW (angl. *World Wide Web*) – žiniatinklis, hiperteksto dokumentų sistema internete.

W3C (angl. *World Wide Web Consortium*) – tarptautinė organizacija, kurianti „WWW standartus16“.

HTML standartų istorija

HTML 1.0 neturi oficialios specifikacijos. Kuomet jis buvo sukurtas, buvo keletas neformalių HTML standartų. 1993 metais buvo pradėtas vieno bendro standarto kūrimas. Sukurta kalba buvo pavadinta HTML 2.0, norint atskirti ją nuo neoficialių „standartų“.

HTML 3.0 buvo pasiūlytas kaip standartas 1995 m. kovo mėnesį įkurtos W3C organizacijos. HTML 3.0 siūlė daugybę naujovių, pavyzdžiui, lentelės, tekstas, automatiškai apgaubiantis objektus, sudėtingų matematinių objektų atvaizdavimas. Tačiau HTML 3.0 buvo per sudėtingas, jo nepalaikė to meto naršyklės. HTML 3.1 niekada nebuvo oficialiai pasiūlytas kaip standartas. Standartu tapo kompromisinis HTML 3.2, atsisakęs daug naujovių ir apėmęs daugelį naršyklėms „Netscape“ ir „Mosaic“ specifinių elementų. HTML 3.0

siūlytas matematinių simbolių palaikymas buvo integruotas atskirame standarte „MathML“ (<http://www.w3.org/Math/>).

Į HTML 4.0 taip pat buvo įtraukta naršyklėms specifinių elementų, tačiau buvo nutarta išgryninti HTML kaip standartą ir dauguma šių elementų tapo nerekomenduojamais. Specialieji elementai aprašo įvairius objektus, pavyzdžiui, mygtukus bei sąrašus.

Nuo HTML specifikacijų kūrimo pradžios buvo išleisti šie HTML kalbos standartai:

- HTML 2.0 (<http://www.ietf.org/rfc/rfc1866.txt>) – (RFC 1866) 1995-09-22
- HTML 3.2 (<http://www.w3.org/TR/REC-html32>) – 1997-01-14
- HTML 4.0 (<http://www.w3.org/TR/REC-html40-971218/>) – 1997-12-18
- HTML 4.01 (<http://www.w3.org/TR/html401>) – 1999-12-24

Organizacija „World Wide Web Consortium“ (W3C) publikavo pilnas naujos HTML5 versijos specifikacijas. Šios kalbos kūrimas vykdomas nuo 2004 metų. Publikuotos ir galutinės „Canvas2D“ specifikacijos. Šis instrumentas skirtas žiniatinklio puslapiams, taip pat yra atsakingas už atsitiktinės grafikos rodymą.

Šiuo metu daugelis interneto naršyklių jau palaiko HTML5 standartą. Tačiau šis palaikymas remiasi anksčiau publikuotais naujos HTML versijos juodraštiniais variantais, o atskirose aplikacijose tos versijos skiriasi.

Galutinių HTML5 specifikacijų pasirodymas reiškia, jog internetinių aplikacijų kūrėjai dabar gali kurti produktus, kurie visiškai bus suderinami su minėtu standartu. Savo ruožtu, interneto svetainių dizaineriai galės kurti puslapius, kuriuos sėkmingai atvaizduoja visos naršyklės, be jokių specialių konkrečiai platformai būdingų kodavimo metodų. Po visų testavimų ir diskusijų galutinės HTML5 specifikacijas planuojama priimti 2014 metais.

Stilius ir turinys

Dabartinės HTML versijos siūlo atsisakyti prezentacinių elementų naudojimo. HTML dokumente turėtų būti žymima tik struktūra, o išvaizda nurodoma naudojant CSS. HTML 4.01 Strict ir XHTML specifikacijose panaikintos ``, ``, `<i>` žymės.

Visą prezentaciją palikus CSS, o HTML naudojant tik struktūrai aprašyti, svetainė atitiks standartus be papildomų pastangų, nes nebūs naudojama neleistinų žymių bei atributų.

1.3. „Web 2.0“ saityno savybės

„Web 2.0“ – antrosios kartos saitynas – antrosios kartos internetas pasižymi interaktyvumu. Kiekvienas naudotojas „Web 2.0“ eroje gali būti ne vien turinio, kurį jis randa internete pasyviu priėmėju kaip anksčiau, bet ir jo pateikėju. Turinį bendradarbiaudami kuria visi, jis skirtas ir skaitymui, ir rašymui.

Antrosios kartos saitynui būdingos specifinės technologinės priemonės (plg. Dagienė, 2008) ir jų gausa:

- tinklaraštis;
- komentavimas;
- žymės;
- prisijungimas;
- vikis;
- paieška;
- jungimasis į grupes;
- nuorodų dalijimasis;
- folksonomijos;
- hibridai (angl. *mashup*);
- sklaidos kanalai;
- „Ajax“ metodas;
- reitingavimas;
- tiesioginiai pokalbiai;
- transliavimas ir kt.

Folksonomija – pasaulinio tinklo svetainių klasifikavimo, kategorizavimo būdas, besiremiantis naudotojų bendruomenės spontaniškai naudojamais reikšminiais žodžiais, tinklalapių turiniui apibūdinti. Tradiciniai metaduomenys paprastai išdėstyti hierarchine tvarka, pasižymi aiškia struktūra, kurią iš anksto nustato turinio pateikėjai. Folksonominė klasifikacijos sistema neturi jokios

tvarkos, joje metaduomenis sudaro laisvai pasirinkti žodžiai – žymės (angl. *tags*), kuriuos siūlo ir pateikia naudotojai.

Hibridas – interneto tinklalapis ar taikomoji programa, į kuriuos „Ajax“ priemonėmis integruoti vienas kitą papildantys elementai iš dviejų ar daugiau šaltinių („Web 2.0“ programų, servisų, paslaugų). Taip sukuriama naujos paslaugos. Terminas hibridas (angl. *mashup*) – skaitmeninių duomenų mišinys.

RSS (angl. *Really Simple Syndication*) technologija leidžia iš įvairių interneto šaltinių surinkti ir susisteminti naujausius pasirodžiusius įrašus. RSS leidžia sekti svetainių naujienas be kasdieninio apsilankymo jose. Tam tereikia pasirinkti norimą RSS srautų peržiūros programą ir užsiprenumeruoti RSS srautus.

„Ajax“ metodas – interaktyvių saityno taikomųjų programų kūrimo metodas, kurį naudojant tinklalapių turinys atnaujinamas nedelsiant (neįkeliant iš naujo viso tinklalapio, kaip yra įprasta apdorojant HTTP užklausas) naudotojui įvedus duomenis ar atlikus kurį nors veiksmą.

Dalijimasis nuorodomis. Esminis antrosios kartos saityno principas – viešas dalinimasis savo surasta ar (ir) parengta informacija. Svarbu rūšiuoti internete esantį turinį, nes laikui bėgant turinio kiekis vis didėja, o tas turinys yra nevienodos kokybės. Nuorodų dalijimasis padeda rūšiuoti interneto turinį. Socialiniuose nuorodų kataloguose esanti informacija yra kokybiška, nes įkeltos nuorodos yra vertinamos daugelio žmonių. To negalima pasakyti apie socialiniuose tinkluose platinamą informaciją, ji gali būti reklaminio pobūdžio.

Jei socialiniame tinkle kuriamas tinklaraštis bus pramoginio pobūdžio, tai socialinių nuorodų katalogo kuriamas tinklaraštis bus informacinio pobūdžio. Jei profesionalus tam tikros srities žinovas gali priskirti vienokią žymę tam tikram objektui, tai mėgėjas priskirs kitokią, ir tada sėkmingas informacijos radimas priklausys nuo paieškos realizacijos.

Antrosios kartos saityno technologinės priemonės yra tarpusavyje persipynusios, pavadinimai neatskleidžia tikrojo jų tikslo. Technologines priemones taip pat galima susisteminti sukuriant jų ontologiją (sąvokų visumos ir ryšių tarp jų specifikuojamas išreikštu pavidalu) (plg. Maskeliūnas, 2001).

Ontologijos, iš esmės, yra tam tikros srities terminų ir jų sąryšio žodynas. Yra keletas ontologijų kūrimo metodų, kurie siūlo nustatyti svarbiausius

ontologijos panaudojimo aspektus, pasidaryti visų terminų sąrašą. Jie gali kartotis, persipinti, tuo parodydami sąryšius tarp jų.

Nėra vieno teisingo ontologijos sudarymo būdo. Ontologijos paprastai kuriamos iteraciniu būdu, t. y. pirmiausia sudaroma grubi klasių struktūra, kuri vėliau peržiūrima, tikslinama ir pildoma (plg. Tankelevičienė, 2008).

Tos pačios technologinės priemonės gali turėti skirtingą naudotojų požiūrį panaudojant jas skirtingose „Web 2.0“ sistemose. Atliktas pagrindinių priemonių sisteminimas gali padėti nustatyti antrosios kartos saityno technologinių priemonių paskirtį. Sukurta išanalizuotų sistemų technologinių priemonių ontologija gali padėti naudotojui pasirinkti tą „Web 2.0“ sistemą iš tam tikros kategorijos, kuri turi vartotojo poreikius atitinkančias technologines priemones, kadangi antros kartos saityno sistemų įvairovė ir gausa yra pakankamai didelė, kad patenkintų naudotojus.

1.4. HTML istorija

Šiuolaikinių technologijų ir informacijos pasaulyje neįsivaizduojama nei dienos be interneto ir be spalvingų žiniatinklio puslapių. Nors interneto istorija prasidėjo dar Šaltojo karo įkarštyje, bet žiniatinklio paslauga pasaulinių tinklų tinkle atsirado tik 1990 metais, kai Ženevoje esančio Europos branduolinės fizikos tyrimų centre CERN (<http://www.cern.ch>) dirbęs britų informatikas Timas Bernersas-Lee kartu su keliais kolegomis nusprendė pasinaudoti interneto duomenų perdavimo tinklais tekstinių ir iliustruotų dokumentų peržiūrai internetiniu režimu. Tuo tikslu buvo pradėtas kurti naujas tokių duomenų perdavimo protokolas HTTP (angl. *HyperText Transfer Protocol*) bei tekstinių dokumentų aprašymo kalba HTML. Svarbiausia jos naujovė buvo nuorodos (angl. *links*) į kitus žiniatinklio puslapius, nepriklausomai nuo jų buvimo vietos. Ši idėja nulėmė neapsakomą žiniatinklio populiarumą ir patiko daugeliui programuotojų, kurie ėmėsi kurti priemones HTML dokumentų peržiūrai.

Ypač gerai tai sekėsi daryti JAV NCSA (angl. *National Center for Supercomputing Applications*, <http://www.ncsa.com>) organizacijos darbuotojui Marciui Andreessenui, kurio sukurtą programą „Mosaic“ ypač pamėgo negausūs tų laikų internautai. Kiek vėliau jis įkūrė kompaniją „Netscape“ ir „Mosaic“ pagrindu sukūrė „Netscape Navigator“ tinklo naršyklę. Timo Bernerso-Lee

vadovaujama HTML kalbos kūrimo grupė 1994 metais atsiskyrė nuo CERN ir įkūrė nekomercinę organizaciją „W3-Consortium“ (<http://www.w3.org>), kurios pagrindinis tikslas yra HTML bei kitų „Interneto“ standartų kūrimas, tvirtinimas bei priežiūra.

HTML – tai vienas iš SGLM (angl. *Structured Generalized Markup Language*) kalbos variantų. Pastarasis dokumentų struktūros aprašymo būdas buvo sukurtas dar 1980-1984 metais ir patvirtintas ISO 8779 standartu. SGLM kalba vartojama siekiant standartizuoti didelių tarptautinių organizacijų raštvedybą ir tarpusavio susirašinėjimą.

HTML yra kompiuterinė hiperteksto žymėjimo kalba, kuri naudojama pateikti turinį internete. Interneto standartu tapusi HTML – tai ne programavimo kalba ir ne griežtas dokumento formatas. HTML visų pirma aprašo loginę žiniatinklio puslapių struktūrą: dokumentų bei juos sudarančių skyrių ir skirsnių antraštes, pastraipas, iliustracijas, lenteles, nuorodas į kitus dokumentus ar kitokius duomenis ir pan.

HTML buvo sumanyta kaip grynai loginės struktūros aprašymo kalba. Tačiau greitai paaiškėjo, jog žiniatinklio puslapių kūrėjams bei skaitytojams to nepakankama. Dėl to HTML be loginių struktūrų gali aprašyti ir fizinės dokumento savybes kaip vartojamo šrifto parametrus, lentelių, iliustracijų bei kitų elementų dydžius ir pan.

Skirtingai nuo tekstų redaktoriaus „Microsoft Word“ vartojamo dokumentų formato DOC ar puslapių aprašymo kalbos „PostScript“, HTML niekuomet neapibrėžia vienareikšmiškai, kaip žiniatinklio puslapis turi atrodyti kompiuterio ekrane. Tai priklauso nuo pasirinktos ekrano raiškos, tinklo naršyklės lango dydžių, teksto vaizdavimui pasirinkto šrifto dydžio bei daugybės kitų parametrų ir pasirinktos naudojamos interneto naršyklės. Pastarasis faktas labiausiai nepatinka daugeliui žiniatinklio dizainerių, kadangi reikia nuolat tikrinti, kaip puslapį parodo viena ar kita tinklo naršyklė bei ieškoti kompromiso.

1.5. Žiniatinklio atsiradimas

Žiniatinklis (angl. *World Wide Web*) yra prieinamų ir tarpusavyje susietų HTML kalba aprašytų dokumentų visuma. Tam, kad taptų prieinami, jie turi

būti pateikti tam skirtuose serveriuose. Be to, jie gali būti papildomi kitų formatais ar grafinėmis iliustracijomis.

Perduodant dokumentą iš vieno kompiuterio į kitą nebuvo garantijos, kad gavėjas galės juo tinkamai pasinaudoti. Sudėtingesnio dokumento perdavimas nebuvo įmanomas tuo atveju, jei siuntėjas ir gavėjas nesinaudojo ta pačia programine įranga. Net ir naudojantis identiška programine įranga perduodant dokumentą galėjo kilti problemų dėl skirtingos vaizduoklių raiškos ar kitų skirtumų.

Atsirado būtinybė rasti būdą, kaip perduoti dokumento esmę tarp skirtingų sistemų. Užtikrinti technologinį neutralumą, tai pasinaudoti dokumentu nepriklausomai nuo naudojamo kompiuterio ar operacinės sistemos. Pradinis tikslas buvo dokumento turinio, o ne jo išvaizdos perdavimas.

1.6. Interneto naršyklės

Naršyklė (angl. *browser*) yra programa, skirta atvaizduoti internetinius puslapius (tinklalapius) žiniatinklyje, vidiniuose įmonės tinkluose ar savo kompiuteryje. Interneto puslapių peržiūra, dažnai naudojantis juos siejančiomis hipertekstinėmis nuorodomis, vadinama naršymu. Be HTML pagrindu sukurtų dokumentų, naršyklės paprastai gali atvaizduoti ir kitokio tipo dokumentus.

Naršyklės daugiausia naudojamos asmeniniuose kompiuteriuose. Prieigą prie žiniatinklio per naršykles turi ir delniniai kompiuteriai bei išmanieji telefonai. Mobiliesiems prietaisams skirta pirmoji naršyklė buvo „PocketWeb“ 1994 m. „TecO“ sukurta „Apple Newton“ delniniam kompiuteriui. Tarp šiuolaikinių mobiliųjų prietaisų naršyklių galima paminėti „Opera Mini“, „IE-mobile“, „Fennec“, „Minimo“, „Safari“ ir „Skyfire“.

Didėjant interneto bei multimedijos reikšmei, naršyklė tapo pagrindine įprasto asmeninio kompiuterio taikomąja programa. Šiuolaikinėse naršyklėse galima atvaizduoti kompiuterinę grafiką, leisti muziką ir radiją, rodyti filmus (tam kai kuriais atvejais naudojami specialūs programiniai įskiepiai).

Interneto puslapių atidarymo greitis

Interneto puslapių atidarymo greitis priklauso nuo kelių dalykų:

1. Interneto spartos.

Tinklalapiuose būna ne tik tekstas, bet ir iliustracijos ar „Flash“ animacija. Visą šią informaciją reikia parsiusiti iš interneto.

2. Kompiuterio spartos.

Parsiusčius iš interneto duomenis naršyklė privalo apdoroti. Ji iš skirtingų elementų sukuria tinklalapio vaizdą. Kuo kompiuteris greitesnis, tuo naršyklė su savo darbu susidoroja greičiau.

3. Interneto naršyklės spartos.

Įdiegus kelias naršyklės viename kompiuteryje, galima pastebėti, kad jos tas pačias interneto svetaines atidaro skirtingu greičiu. Taip yra todėl, kad jos skirtinga sparta atlieka sudėtingus matematinius skaičiavimus. Jų daugiausiai būna įvairiose internetinėse programose, el. pašto ir filmukų svetainėse. Kaip tik tokių tinklalapių įsikrovimo greitis labiausiai priklauso nuo naršyklės skaičiavimo talentų.

Papildomos naršyklių galimybės

Kiekvienas vartotojas internete veikia skirtingus dalykus: vieni mėgsta žiūrėti filmukus, kiti daugiausiai laiko praleidžia bendraudami su draugais socialiniuose tinkluose ar rašydami tinklaraščius, tretį renka informaciją savo darbui. Nė viena naršyklė negali turėti visų mums reikalingų funkcijų, nes tuomet programa bus labai didelė ir stabdys kompiuterio darbą. Todėl labai svarbu, ar programą galėsime pritaikyti savo poreikiams ir pomėgiams. Tai lengviausia padaryti, diegiant įvairius priedus.

Klausimai ir užduotys

1. Kada ir kaip atsirado internetas?
2. Kokie protokolai naudojami internete?
3. Kokie elektroninių duomenų formatai naudojami internete?
4. Išvardinkite ir apibūdinkite atvirųjų ir uždarytųjų duomenų formatų savybes.
5. Kokios „Web 2.0“ saityno savybės?
6. Kokios specifinės technologinės priemonės būdingos antrosios kartos saitynui?
7. Kaip atsirado HTML?
8. Kaip atsirado žiniatinklis?
9. Kokia interneto naršyklių paskirtis?
10. Nuo ko priklauso interneto puslapių atidarymo greitis?

2. INTERNETO SVETAINIŲ KŪRIMAS

2.1. Interneto svetainių tipai

Internetu vyrauja keletas interneto svetainių (tinklapių) tipų. Pagrindiniai interneto svetainių tipai:

- komercinės svetainės (e. parduotuvės);
- reprezentacinės svetainės;
- bendruomenių svetainės;
- pramogų svetainės;
- naujienų portalai;
- interneto dienoraščiai.

Komercinių svetainių (elektroninių parduotuvių) tikslas yra parduoti produktus arba paslaugas internetu. Šios interneto svetainės suteikia galimybę klientams užsisakyti prekes ar paslaugas neišeinant iš namų. Tai gali būti ir dalis įprasto verslo, kai turima ir įprasta parduotuvė, tačiau paslaugas teikiant ir internetu. Gerai suprojektuota elektroninė parduotuvė padeda sutaupyti verslui laiko bei pinigų. Detali lankytojų veiksmų statistika suteikia vertingos informacijos planuojant pardavimus.

Reprezentacinėse svetainėse pateikiama informacija apie žmogų ar įmonę. Šio tipo interneto puslapiai yra retai atnaujinami, reikalauja mažai priežiūros. Dažniausiai reprezentacinėse svetainėse pateikiama statinė, nekintanti informacija apie įmonę.

Bendruomenių svetainių pagrindinis tikslas yra keisti informacija. Svetainė, kurioje žmonės, turintys panašių interesų komunikuoja vieni su kitais, dažniausiai forumuose. Šiam svetainių tipui galima priskirti šiandien vis labiau populiarėjančias socialinių tinklų svetaines. Įmonės puslapis socialinių tinklų svetainėje gali padėti pritraukti naujų klientų.

Pramogų svetainėse galima rasti žaidimus, filmus bei muziką. „Web 2.0“ atvėrė daug naujų galimybių interneto pramogų portalams. Šiandien pra-

mogų portaluose galima žaisti žaidimus naudojantis interneto naršykle, žiūrėti filmus, klausyti muzikos.

Naujienu portaluose pateikiamos naujausių įvykių apžvalgos, straipsniai, lankytojai turi galimybę komentuoti. Šios svetainės palaikymui reikia nemažai žmoniškųjų resursų. Naujienu portalui palaikyti reikia tiek pat pastangų kaip ir leidžiant žurnalą ar laikraštį.

Interneto dienoraščiai (tinklaraščiai) dažniausiai prižiūrimi vieno arba keleto žmonių. Galimybė pateikti savo dienoraštį internete, sparčiai populiarėja tarp įmonių, kurios stengiasi neatsilikti nuo naujovių bei nori atrasti naujus reklamų kanalus. Reguliariai talpinant įrašus, pritraukiant lankytojų auditoriją, galima sukurti savo internetinio dienoraščio skaitytojų bendruomenę, kuri padeda populiarinti parduodamus produktus ar paslaugas.

Asmeniniame tinklalapyje (tinklaraštyje) įprastai talpinama asmeninė informacija apie tinklalapio savininką ir jo veiklą, nuotraukų galerija arba asmeniniai straipsniai. Profesionalai savo tinklalapiuose talpina savo darbus – aplanką, kuris naudingas tam, kad žmonės susidarytų nuomonę apie tinklalapio autorių ir jo veiklą.

Tinklalapiai gali būti skirstomi ne tik pagal paskirtį, bet ir pagal kūrimo metodus. Taip atsiranda skirstymas į statinius ir dinامينius tinklalapius. Pagrindinis skirtumas tarp jų yra tas, kad dinaminuose tinklalapiuose esanti informacija gali būti pritaikyta vartotojui pagal tam tikrus kriterijus.

Žiniatinklio objektų klasifikacija:

- Tinklalapiai. Vienas puslapis su bendra informacija ir koordinatėmis.
- Tinklalapiai:
 - Puslapis apie įmonę su bendra informacija ir koordinatėmis susieiti loginėmis sąsajomis (nuorodomis).
 - Papildomas programinis palaikymas.
 - Animacija.
- Portalai:
 - Puslapiai skirtingomis temomis ir nuorodomis į kitus tinklalapius.
 - Papildomas programinis palaikymas.

- Animacija.
- Galimybė pasiekti organizacijos informacinius resursus.
- Komercinės paskirties portalai:
 - Parduotuvės.
 - Aukcionai.
 - Užsakymų stalai.
 - Sandėliai.
 - Katalogai.
- Paieškos sistemos:
 - Globalios.
 - Regioninės.
 - Specializuotos.

Interneto portalas – dažniausiai apibrėžiamas kaip svetainė, pasižyminti didele informacijos bei papildomų paslaugų gausa, kuri gali tarnauti kaip išėities taškas į kitus interneto resursus.

Portalai paprastai pateikia kombinuotą, pritaikomą prieigą prie informacijos, duomenų ir funkcionalumo bei pasižymi dideliu informacijos ir funkcionalumo integravimo lygmeniu.

Portalai pagal savo tipą skirstomi į dvi kategorijas: vertikalieji portalai, besispecializuojantys vienoje srityje ir apimantys daugelį sričių – horizontalūs portalai.

2.2. Interneto svetainių kūrimo principai

Kuriant interneto svetainę reikia nuspręsti:

- kokius klientų poreikius tenkins svetainė;
- kaip ateityje bus galima praplėsti svetainę;
- ar lengva bus prirėikus perkurti svetainės struktūrą.

Tinklalapių kūrimas susideda iš kelių pagrindinių etapų:

- planavimas;
- turinio sukūrimas;
- apipavidalinimas;

- įgyvendinimas;
- reklama;
- tobulinimas.

Reikalavimai konstruojant tinklalapį:

- išvaizda;
- įkeltis;
- skiriamoji geba;
- lengvas keičiamumas;
- turinys.

Reikalavimų dokumentavimas:

- tinklalapio arba kliento pavadinimas;
- kas parengė;
- data;
- kliento kontaktai;
- versija;
- vykdomoji santrauka;
- prielaidos;
- priklausomybės;
- tikslai;
- veikimo elementai;
- išsamūs reikalavimai (išvaizda, funkcijos, žemėlapis);
- siūlomi sprendimai;
- tinklalapio ateitis;
- pasirašymo sritis.

Klausimai prieš kuriant tinklalapio kompoziciją:

- auditorija;
- tinklalapio tikslas;
- emocijos;
- plečiamumas;
- naršyklės ir skiriamoji geba;

- giliausios informacijos lygis;
- pirmojo puslapio informacija;
- teksto ir grafikos pavyzdžiai;
- tinklalapio atvaizdai ir spalvos;
- tinklalapio slinkimo vertikaliai svarba;
- funkcijos, priemonės;
- įkelties dydis;
- bendrovės iššūkis;
- galutinis terminas.

Spalvos:

- šiltos: raudona, geltona, oranžinė, rožinė;
- šaltos: mėlyna, žalia, violetinė;
- neutralios: balta, juoda, pilka, ruda.

Tipografiniai standartai:

- „Serif“ tipo šriftai: „Times New Roman“, „Times“, „Georgia“, „Garamond“, „Baskerville“.
- „Sans serif“ tipo šriftai: „Helvetica“, „Arial“, „Verdana“, „Futura“, „Gill sans“.

Praktiškumo principas:

- 10-20 sekundžių pradžios puslapiui;
- našumas: perdavimo sparta, skiriamoji geba, naršyklės suderinamumas, tinklalapio architektūra, išplanavimas, naršymo sistema.

2.2.1. Tinklalapio planavimas

Pradedant kurti tinklalapį, būtina jį kruopščiai suplanuoti. Keletas klausimų, į kuriuos būtina rasti atsakymą, gali padėti tinkamai suplanuoti tinklalapį:

- Kokią informaciją norėtumėte patalpinti savo puslapyje?

- Kodėl reikalingas toks tinklalapis?
- Kokio tipo tinklalapis bus kuriamas?
- Kokio stiliaus tinklalapis bus kuriamas (linksmas, pokštau-
jantis, netikėtas, originalus, standartinis, rimtas, solidus ir t.t.)?
- Kas tinklalapyje siūloma?
- Kaip tinklalapis turi atrodyti?
- Kokios programos būtų tinkamos darbui atlikti?
- Ar reikės rinkti informaciją apie tinklalapio lankytojus?
- Kokia bus tinklalapio bylų hierarchinė struktūra? Jeigu bylos
bus sugrupuotos pagal kažkokių kriterijus, tai po to bus daug
lengviau perkelti savo puslapio atskirus dokumentus ar dalis
kitur bei juos keisti.
- Kokie interneto šaltiniai ir nuorodos bus naudojami tinkla-
lapyje? Taip galima supaprastinti tinklalapio struktūrą, suma-
žinti informacijos dubliavimo tikimybę ir padaryti savo pus-
lapį „lengvesniu“.
- Kokia bus žiniatinklio puslapio atskirų dokumentų hierar-
chija? Kaip bus organizuojama navigacija tarp tinklalapio do-
kumentų, kurie jų bus pagrindiniai – nuorodų dokumentai, o
kurie tik informacijos nešėjai? Kokiu būdu sukurti paprastą
navigaciją tarp skirtingų dokumentų ir jų dalių?

Tinklalapio projekto paruošimas susideda iš pagrindinių dokumentų skaičiaus parengimo, dokumentų tarpusavio sąsajos, nuorodų iš vienos vietos į kitą sudarymo, puslapių apipavidalinimo, pateikimo formos sukūrimo, optima-
lios informacijos antraštiniam puslapiui parinkimo.

Pagrindiniai planavimo principai:

- minimizuotas dokumentų persiuntimo laikas;
- lengvai pasiekiamą informaciją;
- alternatyvūs informacijos pateikimo būdai;
- minimalus garso kiekis;
- spalvos;
- gausios grafikos skaidymas;

- didelės apimties grafikos sumažintos kopijos su nuoroda į originalą.

Būtina minimizuoti tinklalapio dokumentų persiuntimo laiką – ne visada puiki ir gausi grafika patraukia, jei po 10 minučių tinklalapio lankytojas pamatys tik trečdalį puslapio, tai vargu ar jis liks sužavėtas. Todėl ypatingai pradinius tinklalapio dokumentus geriausia daryti kuo paprastesnius, bet paliekančius išpūdį.

Informacija puslapyje turi būti lengvai pasiekiami – patartina svarbią informaciją talpinti ne giliau nei antrame žiniatinklio puslapio lygyje. Vartotojas gali laukti dviejų dokumentų pasirodymo ekrane, bet vargu ar jis norės trečio bandymo, nebent jei ši informacija jam yra labai reikalinga.

Patartina naudoti alternatyvius informacijos pateikimo būdus – ne visos naršyklės supranta ir teisingai interpretuoja specifines ar naujas HTML kalbos komandas, todėl patartina šalia grafinių nuorodų naudoti tekstines, šalia vaizdo bylų – grafines, šalia sudėtingos struktūros HTML dokumentų – paprastesnius. Tokiu būdu galima padidinti vartotojų skaičių, kuriems bus pasiekiamas ir suprantamas puslapis.

Naudojamo garso kiekio puslapiuose minimizavimas taip pat svarbus, nes garsinės bylos yra didelės ir reikalauja didesnio greičio perdavimo linijų, todėl gali smarkiai stabdyti puslapio siuntimą internetu vartotojui. Be to, ne visi mėgsta ar turi galimybę klausytis muzikos dirbdami su naršyklėmis. Todėl geriausia suteikti galimybę vartotojui pačiam rinktis – reikalingas garsas ar ne.

Spalvotas ar margas fonas gali visiškai užgožti tekstą dokumente, o spalvotas tekstas gali „paskęsti“ fone, todėl svarbu atsargiai ir atidžiai naudoti spalvas, neperkrauti. Visur reikalinga grožio, išiūrėjimo bei patogumo pusiausvyra.

Puslapyje naudojamą grafiką vertėtų išskaidyti į atskirus dokumentus – kuo daugiau grafikos yra dokumente, tuo lėčiau jis atsiranda vartotojo ekrane ir tuo greičiau senka vartotojo kantrybė. Todėl patartina pateikti gausią grafiką išskaidžius ją į prasmingai susijusias dalis. Reikėtų surasti kompromisą tarp naudojamos grafikos kiekio ir dydžio bei tikėtinų vartotojų prisijungimo greičio bei noro pamatyti tai, kas siūloma tinklalapyje. Didelės apimties grafika puslapyje galima pakeisti sumažintomis kopijomis, kartu pridėdant nuorodą į originalą tiems, kurie norės, susiras ir pažiūrės.

Vartotojui naudingos funkcijos:

- Tinklalapių struktūra turi būti paprasta, nesudėtinga.
- Vartotojas turi intuityviai žinoti, kur reikia ieškoti reikiamos informacijos.
- „Protingosios“ priemonės (pvz.: atsimena vartotojo veiksmus, kitą kartą pasiūlys tik dominančias prekes).
- Galima panaudoti įvairias programines priemones, pvz. prekės apžiūrėjimą.
- Bendravimo priemonės. Vartotojai gali aptarti paslaugas, naujas prekes ir pan.
- Dažnai užduodami klausimai.

2.2.2. Tinklalapio ir jo dizaino kūrimas

Viena iš dažniausiai pasitaikančių ir svarbių tinklalapio kūrimo klaidų yra skubus ir labai gerai neapgalvotas dizaino kūrimas ir kodavimas. Tai lemia poreikių aiškinimąsi jau kūrimo eigoje, tuomet tenka kelis kartus viską daryti iš naujo, kai kurių funkcijų atsisakyti. Dėl šios priežasties, tinklalapių kūrimo darbai užsitiesia labai ilgai, o kai kurie tinklalapiai taip ir neišvysta dienos šviesos.

Todėl labai svarbu pirmiausiai suprasti idėją, užduoti teisingus klausimus ir padaryti viską, kad pasiruošimo etape viskas vyktų sklandžiai, dar iki pirmos kodo eilutės parašymo. Svarbu nustatyti tinklalapio būsimą auditoriją, kūrėjų komandą, uždarbui būdus, atsarginius planus ir interneto svetainės projekto idėjos svarbiausius niuansus.

Paprasti tinklalapiai neturi nereikalingos informacijos, todėl tinklapis turi mažiau puslapių ir skyrių, o dizainas mažiau apkraunamas, navigacija tampa lengvesnė.

Naudojant tik vieną meniu reikia užtikrinti, kad navigacija būtų susieta visame tinklalapyje. Reikėtų naudoti subnavigaciją siekiant užtikrinti patogų informacijos pasiekiamumą tinklalapyje.

Paprastas dizainas įprastai siejasi su nedidelės apimties failais, kurie greičiau įkeliami. Taip pat naudojant paprastą dizainą mažėja pakopinių failų, „JavaScript“ ar kitų papildomų elementų skaičius. Paprastas dizainas gerai ir

tuo atveju, kai siekiama išryškinti tekstą. Kuo mažiau dekoratyvių elementų, tuo labiau išryškėja informacija ir tampa lengviau ją skenuoti.

Vieno tyrimo rezultatai parodė, kad 79 % lankytojų tekstą skenuoja, o tik likę 16 % skaito kiekvieną žodį. Su tokiais lankytojais reikia dirbti, o ne kovoti. Centruotas tekstas jiems atrodo priimtinausias ir tokiu atveju tinklalapis laikomas labiau pritaikytas vartotojams. Tokius tinklalapius žymiai lengviau suprojektuoti, kai turima vos kelis puslapius, kuriuose naudojamas tas pats maketas ir nereikia sukti galvos dėl daugybės puslapių ir kurti jiems atskirų maketų. Taip pat lengviau surasti klaidas. Pavyzdžiui, turint 30 eilučių kodą yra žymiai lengviau rasti klaidą, nei turint 300 eilučių kodą.

Sukurtą tinklalapio dizainą pirmiausiai reikia sukarpyti dalimis, tam, kad būtų galima jį pritaikyti tolesniems darbams. Dizaino karpymas gana ilgas procesas, kadangi norint tai atlikti neužtenka tiesiog iškarpyti dizainą ir tikėtis, kad viskas veiks. Norint teisingai atvaizduoti dizainą kodo pagalba, reikia jį skaidyti dalimis: vieni sluoksniai tam tikru metu turi būti paslepiami ir paliekami tik reikalingieji, siekiant sukurti norimą išvaizdą. Karpymui galima naudoti „Photoshop“ įrankį „Slice Tool“. Karpyti dizainus galima ir be šio įrankio, tačiau tai užimtų daugiau laiko. Naudojant „Slice Tool“ įrankį žymimos tam tikros tinklalapio vietos, kurias ir norima iškirpti, programoje jos vadinamos „Slices“ arba tiesiog dalys.

Netinkamai sukarpius arba bandant netinkamai išsaugoti sukarpytą dizainą, pati „Photoshop“ programa laisvas, nesukarpytas dalis interpretuoja taip pat kaip ir sukarpytas, taigi išsaugojus gaunamos ne tik reikalingos dalys, bet ir daugybė niekam netinkamų šiukšlių, kurios tik apsunkina darbą. Norint, kad taip neatsitiktų, reikia gerai išanalizuoti išsaugojimo meniu, kuriame vyksta netgi svarbesni dalykai nei pats dizaino karpymas. Galimi įvairūs išsaugojimo variantai, paprasčiausias jų – visų dalių išsaugojimas, kuris gali tik pridaryti problemų, taip pat tik vartotojo sukarpytų dalių išsaugojimas, kuris būtų tinkamesnis pasirinkimas. Taip pat ne ką mažiau svarbu pasirinkti tinkamą failų formatą.

Kita svarbi dalis karpant dizainą – failų kokybė. Kuo didesnė failų kokybė, tuo daugiau laiko jiems reikės, kol jie bus įkelti į tinklalapį atsidariusioje naršyklėje. Nors šiais laikais, kai interneto greitis yra žaibiškas, tai ne tokia didelė problema, tačiau vis tiek reikia į tai atsižvelgti.

Sukarpius tinklalapio dizainą, reikia jį paversti tinklalapiu, t. y. reikia rašyti HTML kodą, kuris yra pagrindinė kiekvieno tinklalapio dalis. Kodą reikia surašyti tvarkingai, nes be tvarkingo HTML kodo nebūtų įmanoma pereiti prie kitos dalies – CSS stilių rašymo.

- Gero tinklalapio dizaino savybės:
- geras dizainas nulemia verslo sėkmingumą;
- neperkrautas dizainas (grafikos elementai ir geras turinys);
- kitos kalbos;
- lengvai surandama reikiama informacija;
- orientavimasis į įvairias vartotojų grupes;
- prekės pagal įvairius kriterijus;
- multimedija.

Nors šiai dienai technologijos stipriai patobulėjo, tačiau vis dar gali būti, kad ne visi vartotojai turi galimybę peržiūrėti vaizdo įrašą ar paklausti muzikos. Napatartina per daug apkrauti internetinės svetainės, reikia branginti vartotojo laiką, kad kitą kartą jis sugrįžtų.

Svetainė privalo būti gerai ištestuota. Svarbu įvertinti vartotojo norus, verslo plėtrą ir naujų technologijų atsiradimą. Kai kurios kompanijos gali ištestuoti svetainės įvairiais kriterijais:

- lankomumas;
- sandorių įvykdymas;
- vartotojai;
- parsisiuntimo laikas.

2.2.3. Keletas patarimų prieš kuriant naują tinklalapį

Siekiant, kad kuriamas tinklalapis būtų įdomus ne tik jos kūrėjui, bet sulauktų sėkmės ir būtų lankomas, svarbu labai gerai apgalvoti, kam jis bus kuriamas, kokią problemą padės išspręsti ir pan. Žemiau pateikiama keletas patarimų, kuriuos reiktų gerai apgalvoti ir įvertinti prieš kuriant naują interneto svetainės projektą.

Prieš pradėdant projektą svarbu išgryninti idėją:

- nereikia kurti dar vieno tokios pačios paskirties tinklalapio;

- tinklalapiai skirti žmonėms;
- aiškiai apibrėžti tinklalapio auditoriją;
- nustatyti, kokią problemą sprendžia būsima sistema;
- pasiūlyti lankytojams kažką unikalaus;
- kurti tai, kam jaučiama aistra;
- nuodugniai ištyrinėti rinką ir potencialius konkurentus;
- nustatyti, lokali ar globali tai sistema;
- aiškiai žinoti, iš ko bus uždėbama ar gaunama naudos;
- geri projektai nereikalauja investicijų;
- paieškoti rinkoje potencialių partnerių;
- nesitikėti staigių rezultatų.

Nereikia kurti dar vieno tokios pačios paskirties tinklalapio

Pradedant kurti interneto svetainės projektą, pirmiausia reikia gerai apgalvoti, ar ta idėja unikali, ar artima esamo tinklalapio kopijavimui? Ar tikrai ir kam reikalinga kito tinklalapio kopija?

Dažnai atsakymai į šiuos klausimus būna nekonkretūs. Paklausus, kuo naujoji svetainė bus geresnė, išaiškėja, jog tai dizaino pakeitimai, kitokios spalvos ir patogumas lankytojui, o funkcionalumo atžvilgiu būtų tik kopija. Geriau tokiu atveju mąstyti originaliau, nes tų pačių funkcijų kopijavimas retai būna sėkmingas.

Norint sukurti „kažką panašaus“ į kitą tinklalapį, patartina išanalizuoti jį ir susirašyti, kuo norima išsiskirti ir ar tikrai potencialiems lankytojams tai bus taip svarbu, kad jie ateitų, o ne liktų savo mėgstamame tinklalapyje.

Tinklalapiai skirti žmonėms

Svarbu nepamiršti, kad tinklalapiai skirti žmonėms, o ne mašinoms, todėl labai svarbu kuriant tinklalapius juos kiek įmanoma pritaikyti žmonių patogumui. Dažnai interneto svetainės projektai yra kuriami siekiant pamatyti konkrečius skaičiukus rezultatų suvestinėje. Tikslai gali būti įvairūs: sulaukti 10 000 apsilankymų per parą, surinkti 1 000 RSS prenumeratorių, 5 000 registruotų vartotojų ir pan.

Visa tai būtų gerai, jeigu šie skaičiai neapakintų kūrėjų, kurie gali pamiršti svarbiausią mintį: kad kiekvienas iš tų tūkstančių lankytojų yra žmogus

su savo jausmais, apsilankymo tikslais ir įspūdžiais. Todėl norint pasiekti šiuos tūkstantinius skaičius, visgi reiktų pradėti nuo vieno individualaus žmogaus, kuriam ir būtų skirtas šis tinklalapis, t. y. jo poreikių patenkinimui. Tuomet jis papasakotų savo draugams, draugų draugams ir t.t. Tik tada galima bus džiaugtis tikraisiais apvaliais skaičiukais lankytojų statistikos protokoluose.

Kūrėjai dažnai laiko savo tikslu būtinai patekti į pirmąjį „Google“ paieškos rezultatų puslapį arba net užkariauti pirmąją vietą tarp panašios temos tinklalapių pagal kažkokį raktažodį. Tikslas yra geras ir teisingas, taip sulaukiama daugiau lankytojų, tačiau priemonės, kaip siekiama gerų pozicijų „Google“, dažnai yra sunkiai suprantamos.

Terminas SEO (angl. *Search Engine Optimization*) jau nusako, kad vyksta kažkas ne visai teisingai – jeigu tinklalapiai skirti žmonių poreikiams patenkinti, tai kodėl reikia kažką specialiai optimizuoti savo HTML kode, kad įtiktų paieškos robotams? Galbūt šie tikslai gali derėti, bet dažnai vadovai tiesiog apie tai pamiršta – daugiau laiko skiria optimizavimui paieškos mašinoms, negu žmonių pasitenkinimui po apsilankymo. Šią problemą spręsti ėmėsi ir pati „Google“ kompanija, kuri su „Google Panda“ algoritmu ir dar kitais paieškos sistemos tobulinimais, bando neleisti pakilti į pirmus rezultatų puslapius tiems tinklalapiams, kurių tikslas yra įtikti „Google“ robotams.

Galvojant apie idėją, kuri verta milijono lankytojų, reikia galvoti šiek tiek lokaliau ir pamąstyti apie vieną, arba kelis žmones. Pamąstyti, kas gali būti naudinga jiems, o ne robotams.

Aiškliai apibrėžti tinklalapio auditoriją

Gali nutikti taip, kad į tinklalapį užeis labai įvairaus amžiaus ir socialinio statuso žmonės, bet prieš pradėdant reikia nuspręsti, kam skiriamas dėmesys – į kokią auditoriją, į ką taikomasi su savo projekto teikiamomis paslaugomis?

Kas naudosis svetainės funkcionalumu:

- Vyrai ar moterys?
- Vaikai ar suaugę?
- Verslininkai ar neturtingi studentai?
- Programuotojai ar namų šeimininkės?

Panašių tikslinamųjų klausimų galima sugalvoti daug, į juos būtina atsakyti jau tuomet, kai gryninama ir detalizuojama idėja.

Reikia labai aiškiai suprasti, į kokius socialinio tinklo naudotojus nukreipta tinklalapio strategija. Kiekviena demografinė grupė su savimi neša tam tikrą scenarijų ir žaidimo taisykles, pagal kurias privaloma žaisti siekiant sėkmingo tinklalapio.

Nustatyti, kokią problemą sprendžia būsima sistema

Kas būtų, jeigu kuriamos sistemos nebūtų – ar tai trukdytų žmonėms gyventi? Kodėl žmonės turės užėiti į šį tinklalapį? Kokią gyvenimišką funkciją tai jiems padės atlikti? Ar tikrai tam yra poreikis? Keletas globalių pavyzdžių paaiškina problemos sprendimo sėkmę.

„Facebook“ ir kiti socialiniai tinklai patenkina bendravimo su nutolusiais draugais poreikį, taip pat patogūs informacijos pasidalinimui ir nuotraukų peržiūrai. Jeigu jų nebūtų, tai žmonės vis dar daugiau naudotųsi telefonais, žymų (angl. *bookmarks*) tarnybomis, o nuotraukas talpintų kur nors į „Picasa“ ir siųstų daug informacijos el. paštu.

DELFI ir kiti naujienų portalai patenkina žingeidžių vartotojų poreikį ir leidžia žmonėms greitai sužinoti naujausią informaciją. Jeigu tokių portalų nebūtų, būtų daug populiariesni laikraščiai ir žurnalai. Bet tie, kurie naujienas skaito internete, dažniausiai laikraščius perka jau labai nenoriai.

„Google“ sprendžia informacijos paieškos problemą, nors senais laikais bibliotekos būdavo perpildytos.

Lokalioje sistemoje reikia pagalvoti, kaip žmones „užkabinti“ taip, kad jie pajautų palengvėjimą dėl kažkokio savo aiškaus poreikio patenkinimo, kad nebenorėtų atsisakyti ir išeiti iš svetainės. Tada tai bus tikra sėkmė.

Konkretus e. verslo idėjos pavyzdys: lokalus tinklalapis, skirtas vieno namo butų savininkų informavimui, kur galima pasižiūrėti, kaip ir kiek mokėti už komunalines paslaugas, galima būtų spręsti bendrijos valdymo klausimus, informuoti apie remontus ir t.t.

Jeigu galvojant apie būsimą sistemą, neįmanoma įsivaizduoti, kad žmonės į tinklalapį tiesiogiai kreiptųsi dėl savo kažkokios problemos sprendimo ar poreikio patenkinimo. Tokiu atveju reikės juos įtikinti užėiti arba atvesti lankytojus per paieškos sistemas, o tai jau daug sudėtingiau. Tad kartais geriau laiku sustoti ir atsisakyti idėjos, kurios poreikis žmonėms nėra viena-reikšmiškai aiškus.

Prieš kuriant tinklalapį galima pagalvoti apie asmeninę patirtį, apie dažniausiai lankomas svetaines, kodėl pasirenkamos būtent tos svetainės, kokias problemas sprendžia tie tinklalapiai?

Pasiūlyti lankytojams kažką unikalaus

Nors internete yra milijonai, o gal net ir milijardai tinklalapių, iš jų tikrai galima išskirti nemažai ypatingų tinklalapių. Tai nebūtinai turi būti visiškai originali idėja, kurios niekas kitas nėra pakartojęs – tokių pasaulyje beveik nėra. Bet galima į paprastą, kartais net banalią idėją įberti žiupsnelį originalumo ir vaizduotės, ir bendras vaizdas gali atrodyti visai kitaip.

Pavyzdžiui, pasaulyje yra labai daug elektroninio pašto paslaugas siūlančių įmonių. „Gmail“ paštas pasižymėjo dėl dviejų unikalių „Gmail“ savybių: viena yra praktiškai neribota dėžutės talpa, o kitas dalykas – labai kokybiška apsauga nuo laiškų-šiukšlių arba potencialių virusų prisegtukuose. Šios dvi savybės suteikia žmonėms pasitikėjimą „Gmail“ sistema, o tai yra svarbiausia.

Todėl prieš kuriant naują svetainę reikia užduoti klausimą sau: kuo ji gali būti išskirtinė iš kitų? Kokios unikalios savybės privers lankytojus ne tik pasilikti, bet ir sugrįžti dar kartą būtent į šią svetainę, o ne į konkurentų tinklalapius?

Šis punktas yra labai svarbus marketingo atžvilgiu. Pasakojant apie kuriamą arba jau sukurtą tinklalapį potencialiems partneriams ar klientams, reikia mokėti labai glaustai, per kokią minutę, išdėstyti, kuo šis projektas yra unikalus. Kodėl užsakovai reklamą turi talpinti šioje svetainėje, o ne konkurentų? Kuo projekto vizija skiriasi nuo kitų tos rinkos tinklalapių? Neturint aiškaus atsakymo į šiuos klausimus, tinklalapis bus prilygintas tokiems pat neišsiskiriantiems tinklapiams.

Tokios išskirtinės savybės pasirinkimas gali priklausyti nuo būsimos projekto auditorijos. Galima turėti projektą, skirtą labai konkrečiai žmonių grupei: konkrečios profesijos darbininkams, kažkokio miesto bendruomenei arba šešiolikmečių grupei. Pagal tai ir gali natūraliai gimti ta unikali sistemos savybė.

Patarimai realaus kūrimo procesui:

- Kurti ne tinklalapį, o prekinį ženklą.
- Nepamiršti išmaniųjų telefonų, planšečių ir programėlių.

- Pasirinkti tinkamą programavimo kalbą ir darbo įrankius.
- Naudoti jau sukurtus įrankius.
- Nebijoti turinio valdymo sistemų.
- Pradinė projekto architektūra – labai svarbi.
- Duomenų bazė – projekto vertingiausia dalis.
- Geras tinklalapio dizainas gali daug nulemti.
- Programinis kodas turi būti „universalus“.
- Testavimas – ar tikrai tai turi daryti lankytojai?
- Patogumas lankytojams – kelias į sėkmę.
- Nepervertinti savo lankytojų geranoriškumo.
- Taisyklinga kalba labai svarbi.
- Panaudoti mygtukai ir nuorodos turi būti veiksmingi.
- Nepamiršti apie klaidų puslapius.
- Pagrindinis puslapis nebūtinai dažniausiai lankomas.
- Nepersistengti su reklama ir reklamjuostėmis.
- SEO – svarbu, bet ne svarbiausia.
- Atsarginė laikmena, atsarginė laikmena ir dar kartą atsarginė laikmena.
- Sužadinti susidomėjimą projektu dar prieš jo pradžią.

Patarimai pradėjus projektą:

- Pirminę versiją parodyti draugams.
- Paruošti prezentaciją ir oficialų aprašymą.
- Telefono numeris suteikia stiprų solidumo įspūdį.
- Išmokti pasakoti apie projektą.
- Socialiniai tinklai – nemokamas reklamos būdas.
- „Google Analytics“ ir „Webmaster Tools“ – būtini įrankiai.
- Nesukti skaičių „aklai“, domėtis tendencijomis.
- Projekto palaikymas svarbesnis už paleidimą.
- Dalintis pasiekimais su savo vartotojais.
- Nesitikėti didelių pinigų per pirmus metus.
- Nesistengti dirbti per daug, o dirbti tai, ką reikia.

2.2.4. Lankytojų tipai

Kuriant tinklalapio išvaizdą reikia atsižvelgti į būsimus lankytojus ir netgi jų aplinką. Lankytojus galima suskirstyti į tris tipus. Pirmojo tipo lankytojai į tinklalapius patenka naudodami įvairius paieškos variklius, tokius kaip „Google“ arba „Yahoo“. Tačiau jie ilgai tinklalapyje neužsibūna ir radę tai ko ieškojo palieka tinklalapį.

Antras lankytojų tipas – tai tokie lankytojai, kurie jau yra buvę tinklalapyje arba girdėjo apie jį ir patenka į jį tiesiogiai. Šie lankytojai, dažnai pirmą kartą lankydami svetainėje, nori tik produkto arba paslaugos informacijos, siekdami apgalvoti savo išlaidas. Tokie lankytojai įprastai lankosi tinklalapyje dar kartą lygindami duomenis, todėl jiems labai svarbi navigacija. Tai, ko gero, vienintelis dizaino elementas, kuris darys jiems įtaką.

Trečiajam tipui priskiriami lankytojai, kurie suinteresuoti atnešti pelną būtent tuo apsilankymu tinklalapyje. Daugeliu atvejų šie lankytojai tinklalapyje lankosi nebe pirmą kartą ir patenka į jį ne per paieškos variklį, bet tiesiogiai.

Kuo paslauga brangesnė, tuo labiau tikėtina, kad lankytojai ja nepasinaudos iš karto, o prieš tai bent kelis kartus apsilankys tinklalapyje. Taip pat dažnai lankytojai pradeda kaip naršantieji ir kyla per pakopas iki tol, kol išsiją paslaugą ar prekę, ir tada ciklas prasideda iš naujo.

Skirtumą gali sudaryti tokie veiksniai: lankytojų lytis, amžius, tautybė bei politinė padėtis valstybėje.

2.2.5. Tinklalapio funkcionalumas

Kiekvienam tinklalapiui reikalingas ne tik gražiai atrodantis dizainas, bet ir funkcionalumas. Lankytojas, patekęs į tinklalapį ir negalintis jame rasti to, ko norėjo, nekreips dėmesio į taip, kaip vizualiai jis atrodo ir jį paliks. Nesvarbu, kaip gražiai tinklalapis atrodo ar kaip gerai sudarytas „Java“ skriptas, padailinantis efektus, jei negalima rasti, ko tikimasi – tai blogas tinklalapis.

Lyginant tinklalapio išvaizdą ir funkcionalumą gali būti, kad kur kas svarbiau yra ne išvaizda, o tai, kaip jis veikia. Blogai parinkti šriftai, neišvaizdžios nuorodos, keistos spalvos. Jei tinklalapyje radote ko ieškojote, tai buvo geras tinklalapis.

Tam, kad būtų įgyvendintas siekis suteikti tinklalapiui gerą funkcionalumą, reikia atsižvelgti į keletą kriterijų:

- rišlumas;
- navigacija;
- informacija.

Siekiant rišlumo, reikia paprastumo, kuo mažiau papildomos medžiagos ir aiškaus informacijos išdėstymo. Atitinkami elementai turi būti išdėstyti taip, kad lankytojiui nekiltų klausimo, kur jis yra.

Lankytojai įprastai nemėgsta vargti su nestandartine tinklalapio navigacija ir galvoti, kaip pasiekti norimą informaciją. Lankytojas nesivargins aiškintis, kaip veikia tinklalapio navigacija ir vietoj to mieliau paieškos sistemoje įves kito tinklalapio adresą. Todėl tinklalapio navigacija turi būti intuityvi. Ji turi būti tokia, kad lankytojiui būtų lengva atsakyti į klausimus: „Kur aš esu?“, „Kur aš buvau?“ ir „Kur aš galiu patekti toliau?“. Nuorodos turi būti taip pat ir raktažodžiai, nes svarbu, kad būtų aišku, kur paspaudus nuorodą pateks lankytojas (plg. Nielsen, 2000).

Geras tinklalapio funkcionalumas daro įtaką ne tik lankytojų požiūriui į tinklalapį, bet ir paieškos variklių optimizavimo galimybėms. Jei tinklalapis tinkamai neveikia arba kuris nors jo puslapis visiškai neveikia, gaunama klaida 404. Tai reiškia, kad paieškos robotas aptikęs tokį puslapį gali visiškai nepriimti naujo tinklalapio arba pašalinti iš paieškos sistemos jau esamą. Jei tinklalapis yra internete, bet jo negalima rasti paieškos sistemoms, reiškia – jis negali niekaip teigiamai daryti įtakos įmonės pelnui. Tai turi įtakos ne tik naujų lankytojų požiūriui, bet ir jau esamų klientų nuomonei. Auditorijai nėra nieko blogiau nei turint išsaugojus tinklalapio žymę ir bandant per ją patekti į tinklalapį, suprasti, kad jo ten nebėra.

Nors funkcionalumas svarbiau už išvaizdą, tačiau į rinką atėjus HTML 5 ir CSS3 technologijoms, blogai atrodantis tinklalapis taip pat nepritrauks norimo lankytojų skaičiaus.

2.2.6. Tinklalapio talpinimas

Katalogus bei bylas į virtualų serverį perkelti galima naudojantis FTP prieiga. FTP serverio adresas yra ir virtualaus serverio adresas, pvz.:

www.mano.domenas, ftp.mano.domenas ar jei jungiamasi per naršyklę, ftp://www.mano.domenas. Prisijungimui vietoje adreso taip pat galima naudoti serverio, kuriame yra virtualus serveris, IP adresą, kurį galima sužinoti. Jis dažniausiai pateikiamas el. laiške, pranešančiame apie paslaugos aktyvavimą. Prisijungimui prie FTP naudojamas standartinis 21 *portas*. Norint perkelti bylas per FTP prieigą, yra būtinas FTP klientas (programa), kurio pagalba galėtumėte prisijungti prie FTP serverio.

Keletas nemokamų FTP klientų:

1. coreFTP Lite–<http://www.coreftp.com/>;
2. FileZilla–<http://filezilla-project.org/>;
3. GoFTP–<http://www.goftp.com/>;
4. FireFTP–<http://fireftp.mozdev.org/> (skirtas Firefox naršyklei).

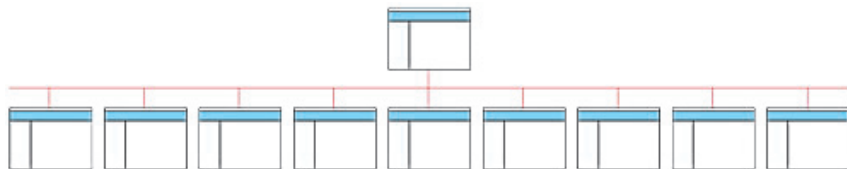
Tam, kad bylos (failai) būtų pasiekiami tinklalapio lankytojams, juos reikia kelti į virtualiame serveryje esantį „public_html“ katalogą. Nėra draudimo failus kelti ir į žemesnį nei „public_html“ katalogą, tačiau šių duomenų svetainės lankytojai neturės galimybės „pasiekti“ (tai naudinga Perl/PHP kalbomis parašytas programas (skriptus) naudojantiems vartotojams, kuriems būtina užtikrinti tam tikrose bylose esančios informacijos saugumą).

Taip pat reikia atkreipti dėmesį į tai, kad norint, jog virtualus serveris veiktų tinkamai, būtina išsaugoti tipinę jo struktūrą, netrinti ir nekeisti šių katalogų: *.cpaddons*, *.cpanel*, *htpasswd*, *.trash*, *access-logs*, *etc*, *mail*, *tmp*. Taip pat ir šių bylų: *.bashrc*, *.bash_logout*, *.bash_profile*, *.contactemail*, *.lang*, *.lastlogin*, *.zshrc* turinio ar leidimų (angl. *permissions*).

2.3. Tinklalapio struktūra

Sudarius tinklalapio dalis ir norint suprojektuoti kuo geresnę navigaciją, patartina imtis testavimo. Testuoti reikėtų ir komandoje ir pasitelkus realius vartotojus. Gera navigacija priklauso nuo ryšio tarp pagrindinių puslapių ir tarp informacijos (plg. Lynch, 2012).

Blogai, kaip puslapiai yra išdėstyti nuosekliai, tačiau blogai ir naudoti per daug sluoksnių, nes taip sunku rasti norimą informaciją. Taip pat nereikia versti lankytojų keliauti per nuorodas, ieškant tokios informacijos, kurią naudojant tinkamai sutvarkytą išdėstymą galima būtų pasiekti nedelsiant.

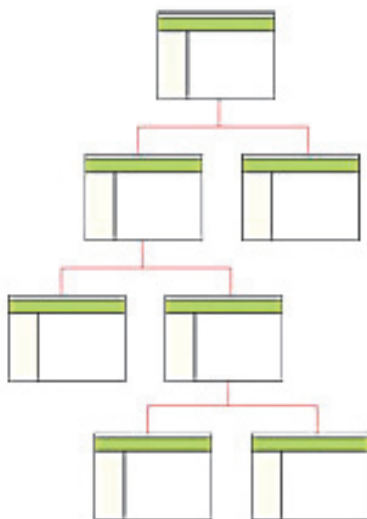


1 pav. **Per mažai lygių turinti struktūra**

Šaltinis: <http://webstyleguide.com/wsg3/3-information-architecture/3-site-structure.html>

Tinklalapiui nuolatos augant jo struktūra tampa dar sudėtingesnė. Tokiu atveju labai praverčia išsiaiškinti lankytojų nuomonę, nes tik taip galima rasti navigacijos vietas, kurias reikia atnaujinti arba tvarkyti.

Kartais tiesiog neišvengiamai reikia suprojektuoti paieškos funkciją puslapyje. Tai būtina didelės apimties tinklalapiuose, kuriuose be paieškos galimybės nebūtų įmanoma arba būtų labai sunku surasti informaciją.


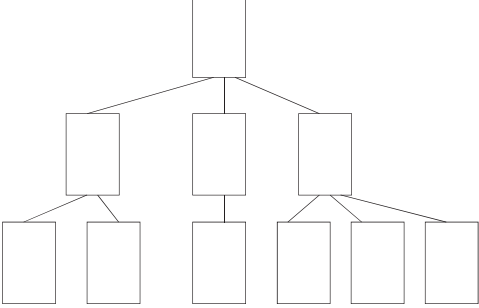
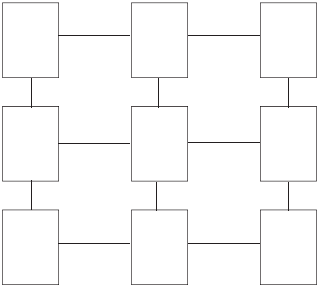
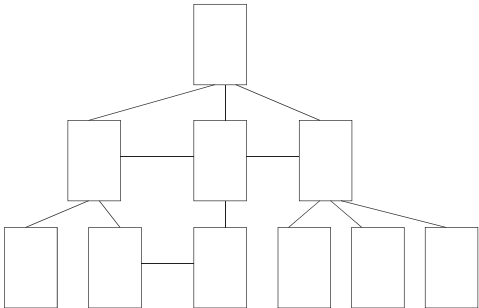


2 pav. **Per daug lygių turinti struktūra**

Šaltinis: <http://webstyleguide.com/wsg3/3-information-architecture/3-site-structure.html>

Kuriant tinklalapį gali būti naudojamos įvairios tinklalapio struktūros: linijinė, hierarchinė, tinklinė ar mišri (1 lentelė).

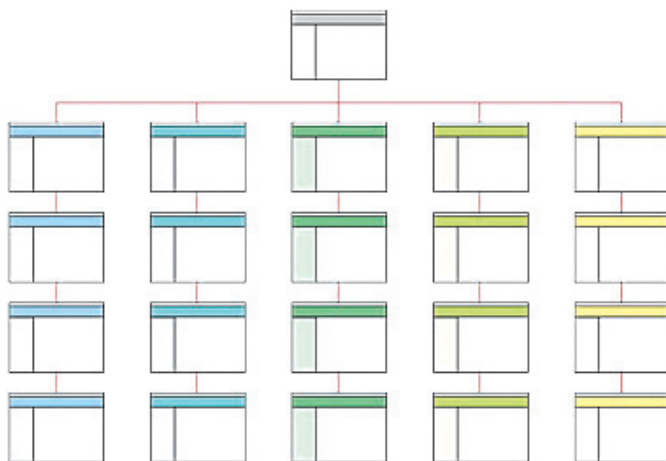
1 lentelė. Tinklapių struktūros

Struktūra	Pavyzdys
Linijinė	
Hierarchinė	
Tinklinė	
Mišri	

2.3.1. Hierarchinė struktūra

Vienas iš lengviausių ir logiškiausių būdų susisteminti savo interneto dokumentus yra naudoti hierarchinę arba meniu struktūrą (3 pav.). Su tokio tipo diagramomis yra susidūrę daugelis vartotojų, todėl jiems nebus sunku logiškai susigaudyti, kaip tinklalapyje yra išdėliota informacija.

Hierarchija labai tinka tinklalapiams. Dauguma interneto pagalbos sistemų yra hierarchinio tipo. Darbas pradedamas su pagrindinių temų sąrašu arba meniu, pasirenkant vieną iš poskyrių, kuris veda į diskusiją apie ieškomą temą. Žinoma, kiekvienas tinklalapis turi atskirą kiekį poskyrių ir temų, tačiau dauguma jų vadovaujasi šia paprasta struktūra.



3 pav. **Hierarchinė struktūra**

Šaltinis: <http://webstyleguide.com/wsg3/3-information-architecture/3-site-structure.html>

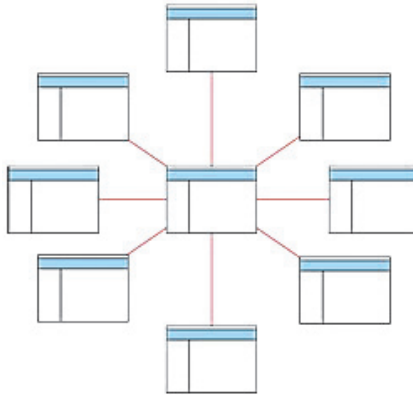
Hierarchinėje sistemoje lankytojai gali lengvai orientuotis. Jų pasirinkimas yra arba judėti link bendros, arba link konkrečios informacijos. Jei būtų sukurta nuoroda į vieną iš skyrių, lankytojai turėtų galimybę apsilankyti tame puslapyje, kuriame jie jau buvo ir galimai susidūrė su jiems rūpima informacija. Šioje sistemoje pagrindinis puslapis suteikia pačią elementariausią informaciją ir nuorodas, kurių pagalba galima rasti ieškomą detalią informaciją.

Kiekvienas lygis turi nuoseklią sąsają (aukštyn, žemyn, atgal į indeksą). Hierarchinė sistema yra sudaryta taip, kad būtų beveik neįmanoma pasiklysti. Tai ypač veiksminga, jei vartotojui nurodoma, kur kas yra. Be to, jei organizuojant kiekvieną skyrį išvengiama temų dubliavimosi, hierarchinis tipas gali padėti lengvai rasti tam tikras duomenų dalis.

Kuriant hierarchinę sistemą reikia vengti per didelio pasirinkimų sudarymo. Reiktų stengtis nesukurti gilesnio nei trijų lygių meniu, nes kitaip lankytojai gali pasimesti ir jiems tiesiog neberūpės, dėl ko jie lankosi tinklapyje.

Paprasčiausias hierarchinės struktūros pavyzdys yra žvaigždės struktūra (4 pav.), kurioje visi puslapiai išdėlioti, o tai reiškia susiję su pagrindiniu puslapiu.

Nors tokia struktūra patogiai susieja pradinį puslapį su kitomis pagrindinėmis tinklalapio dalimis, tačiau reikia sudaryti galimybę norintiems greitai patekti iš vienos dalies į kitą globaliomis nuorodomis. Įprastai šią funkciją atlieka tinklalapio viršuje esantis meniu.



4 pav. Žvaigždės tipo struktūra

Šaltinis: <http://webstyleguide.com/wsg3/3-information-architecture/3-site-structure.html>

2.3.2. Linijinis maketas

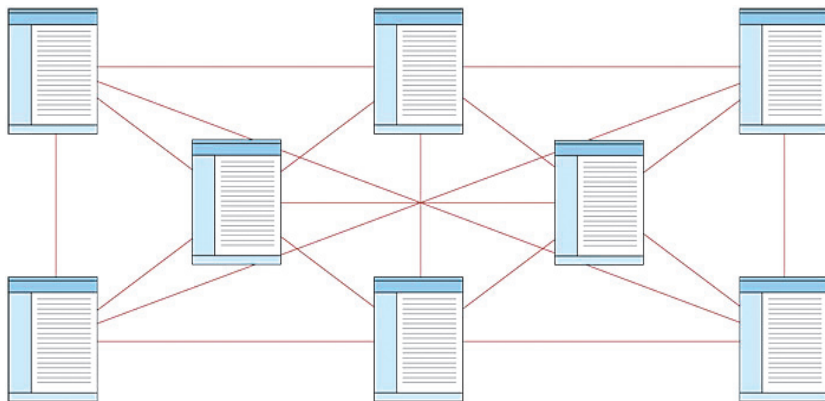
Kitas galimas būdas rūšiuoti puslapius – linijinis, kurio išsidėstymas panašus į atspausdinto dokumento, kai pradedant antraštiniu visi puslapiai eina

iš eilės. Griežtoje linijinėje struktūroje nuorodos juda tik pirmyn ir atgal. Linijinėje struktūroje lengva susigaudyti, nes yra labai nedaug naršymo galimybių, tačiau taip pat dėl to nukenčia galimybė įdomiai pateikti duomenis, todėl vartotojas nesijaučia laisvas tokioje sistemoje. Ši sistema tinka, kai pateikiama informacija taip pat yra linijinės struktūros arba kai norima, kad lankytojas naršydamas nieko nepraleistų.

Taip pat populiarus informacijos pateikimo internete būdas yra sujungti linijinę sistemą su hierarchine. Puslapiai turi nuorodas į kitą puslapį ir atgal tiems lankytojams, kurie naršo linijine struktūra per internetines pamokas ir hierarchinio tipo nuorodas puslapio meniu. Ši kombinacija veikia gerai tol, kol lankytojams užtenka nurodymų, kur ir ko ieškoti.

2.3.3. Tinklinė struktūra

Taip pat gali būti naudojama tinklinė struktūra (5 pav.). Naudojant tokią struktūrą, lankytojui suteikiama galimybė nevaržomai keliauti per puslapius iš bet kur į bet kur, tačiau tokia struktūra neretai sukelia sąmyšį. Toks išdėstymas tinka tik nedideliams tinklalapiams arba tinklalapiams, skirtiems patyrisiai auditorijai.



5 pav. Tinklo struktūra

Šaltinis: <http://webstyleguide.com/wsg3/3-information-architecture/3-site-structure.html>

2.4. Tinklalapių kūrimo priemonės

Norint profesionaliai kurti žiniatinklio puslapius, būtina žinoti kaip realiai atrodo tai, ką rašai. Dabartiniai WYSIWYG (angl. *What You See Is What You Get*) redaktoriai realizuoja tik dalį visų HTML kalbos galimybių ir tam naudoja standartinį, tačiau ne visada efektingą kodavimo algoritmą. Tada tenka pačiam „apeiti“ šią papildomą sieną, skiriančią nuo realios informacijos ir bandyti koreguoti tiesiogiai bylas, kuriose saugomas konkretus puslapis.

Dauguma žiniatinklio informacijos kūrimo programų, skirtų tokios informacijos patalpimui į internetą, atskiria vartotoją nuo realaus (fizinio) kuriamos informacijos vaizdo ir iš karto rodo informaciją tokiame pavidale, kuriame ji matysis jau patalpinta į interneto žiniatinklio serverį. Tokia technologija vadinama WYSIWYG (angl. *What You See Is What You Get*). Tokiu būdu, kuriant tinklalapį internetui, visiškai nereikia žinoti kaip visa kuriamo tinklalapio informacija yra perduodama iš žiniatinklio serverio visiems besikreipiantiems vartotojams ir koku būdu ji užrašoma į bylą ar bylas.

Visa žiniatinklio puslapiuose esanti informacija susideda iš vieno ar daugelio HTML dokumentų – ASCII koduotų tekstinių bylų, naudojančių HTML (angl. *Hypertext Markup Language*) kalbą.

Tinklalapių kūrimo būdai:

- tekstinis,
- vizualinis (WYSIWYG – angl. *What You See Is What You Get*),
- konverteriai,
- TVS (turinio valdymo sistemos).

Turinio valdymo sistema (TVS) yra patogi naudoti, skirta tiek pradedančiam, tiek profesionalui, funkcionali ir lanksti, turi galimybę integruoti programinius paketus, universali ir saugi.

2.5. Tinklalapių etiketas, priežiūra ir autorystės teisės

Sukūrus tinklalapį svarbu jo nepamiršti, nuolat atnaujinti informaciją, informuoti vartotoją apie puslapio naujienas, problemas ir taisymus, apie informacijos svarbumą ir platinimą.

Kai keičiasi informacija, kuri paskelbta puslapyje ar keičiasi nuorodos, būtina koreguoti puslapį, nauja ir teisinga informacija ypač vertinama internete.

Puslapyje galima įdėti nuorodas į naujausių naršyklių versijų puslapius, ypač jei naudojamos naujos tų programų teikiamos galimybės. Tada vartotojas galės greitai „patobulinti“ savo programą, jei informacija puslapyje jį sudomins, bet bus nepasiekama dėl naršyklės senumo.

Svarbu informuoti vartotoją apie puslapio naujienas, problemas ir taisymus – tai padės lengviau išspręsti išskylančias problemas arba greičiau susiorientuoti puslapyje.

Jeigu tenka perkelti puslapį į kitą serverį ar vietą, patartina bent jau kelis mėnesius puslapį palikti senoje vietoje, trumpai informuoti vartotojus apie situaciją ir automatišką perkėlimą į naująją vietą.

Geriausia, kad žiniatinklio puslapio pagrindinės bylos vardas sutaptų su žiniatinklio serverio, vartojamu pagal nutylėjimą, vardu. Tai palengvins puslapio radimą, nes nereikės spėti puslapio vardo, o tereikės žinoti jo adresą.

Pagrindinė taisyklė internete yra tokia: visa, kas yra pateikta internete gali būti kopijuojama be apribojimų. Todėl jei informacija puslapyje yra svarbi ar dėl kažkokios kitos priežasties negali būti laisva platinama, būtina apie tai perspėti vartotojus.

2.6. Internetinis marketingas

Tinklalapis gali išplėsti galimybes pasiekti tą vartotoją, kurį gali būti sudėtinga pasiekti per tradicines reklamos priemones. Ieškodami informacijos žmonės nebevaro laikraščių ar žurnalų, o jungiasi prie interneto. Elektronine prekyba galima pardavinėti prekes ar paslaugas net jei tuo metu nedirbama –

naktį ar savaitgalį. Sukuriama galimybė prekiauti ne tik Lietuvos, bet ir užsienio rinkose be didelių papildomų investicijų.

Vienos didžiausių kompanijos patiriamų išlaidų – rinkodara. Stendų bei reklamų gaminimas, jų spausdinimas ir pristatymas. Interneto svetainė – greitesnis ir efektyvesnis būdas pateikti reklamą, lengviau atnaujinti informaciją. Pirkti vietą reklamai labai brangu, nesvarbu, ar naujienų laikraštyje, ar stende, ar televizijoje bei radijuje. Tuo tarpu tinklapyje galima talpinti visą reikiamą informaciją be didelių papildomų išlaidų.

Internetas leidžia smulkų verslą pateikti geriau nei didelę konkurentų kompaniją. Jei klientas neranda tinklalapio internete – jis nueina pas konkurentus. Kai vis daugiau įmonių skverbiasi į internetą, tos, kurios to nepadaro – praranda pranašumą prieš konkurentus, todėl tikėtina, jog pralaimės ir kovą dėl savo rinkos dalies.

Internetinio marketingo tikslas – rasti tikslinį klientą ir efektyviai bei pelningai patenkinti jo reikmes. Taip pat užtikrinti, kad šis procesas kuo dažniau pasikartotų. Internetas ir svetainė turi padėti uždirbti pinigus bei automatizuoti dalį kasdienių užduočių. Visa tai nesunkiai pasiekama, jei bus teisingai naudojamos reikiamos technologijos.

Sėkmingas internetinis marketingas tai ne tik reklamjuostės, kurios kartais generuoja lankomumą, bet dažnai neatneša jokios naudos (lankytojų negalima paversti klientais, nes tai nėra tiksliniai lankytojai).

Tinklalapiai turi būti kuriami visą laiką galvojant apie vartotojus. Ne visi vartotojai turi pačius naujausius kompiuterius, gerą priėjimą prie interneto, taip pat naudoja skirtingas interneto naršyklės, todėl negalima versti vartotojo pernelyg ilgai laukti. Šiam tikslui įgyvendinti reikia tinkamai parinkti adresą, pateikti naujausias žinias, pateikti tikslias nuorodas, galima įtraukti reklamas, prekinius ženklus, sukurti organizaciją atspindinčią atmosferą, talpinti klientų atsiliepimus ir asmenines istorijas, aktyvinti diskusijas, akcijas, žaidimus.

Sėkmingam projektui įgyvendinti būtina gera marketingo strategija, svarbu siekti kuo daugiau pritraukti naujų lankytojų ir išlaikyti senus, todėl reikia saugoti vartotojų duomenis, analizuoti ir sekti apsilankymus. Iš šių duomenų galima spręsti apie reklamos efektyvumą. Tačiau daugelį vartotojų atgraso prašymas įvesti slaptą informaciją (pvz.: mokėjimo kortelės numerį), todėl svarbu vartotojui nurodyti, kaip ir kur bus panaudojami duomenys.

Veiksmai, būtini sėkmingam internetiniam marketingui:

- Suformuluoti konkretų tikslą.
- Nustatyti savo tikslinę auditoriją.
- Domėtis lankytojų nuomone.
- Realiai įvertinti savo poreikius, resursus ir galimybes.
- Pradėti nuo nesudėtingų tinklalapių.
- Pasirinkti konsultantus, vėliau plėstis.
- Reklamuoti savo projektą.

Veiksmai, kurie lemia nesėkmingą internetinį marketingą:

- Bendravimo su klientais nebuvimas.
- Perteikiant informaciją apsiribojama internetine svetaine.
- Netinkama grafika ir apipavidalinimas.
- Neatnaujinama pateikiama informacija.
- Nuolat nesekama techninė serverio eksploatacija.

Internetinio marketingo sėkmei pasiekti svarbus įvairių technologijų naudojimas, tokių kaip elektroninis paštas, paieškos sistemos, reklamjuostės ir kitos priemonės.

Elektroninio pašto naudojimas yra pigus ir greitas. Galima paklausti apie produktą, pasiskųsti ir t.t. Į tokius laiškus būtina reaguoti. Galima el. paštu leisti vartotojams užsisakyti naujienas, sukurti el. pašto grupę saviems vartotojams, prie siunčiamo el. laiško pridėti trumpą žinutę apie kompaniją.

Paieškos sistemos būna mokamos ir nemokamos. Jos tikrina pagal raktinius žodžius (META kintamieji), puslapio turinį, taip pat atlieka kombinuotą paiešką. Paieškos sistema ieško, ar puslapis atitinka tematiką, rezultatus išveda pagal užregistravimo laiką, duomenų aktualumą, dažniausiai peržiūri tik pirmuosius puslapius.

Reklamjuostės – būtinas pasiūlymas atlikti kažkokį veiksma, turi žadėti kokią nors materialinę naudą, nurodyti, kodėl vartotojas turi kreiptis į puslapį. Tai gali būti konkursų ir išpardavimų skelbimas. Reklamjuostės yra trumpai naudojamos, todėl reikia analizuoti rezultatus.

Papildomos tinklalapių priemonės:

- Spaudos konferencijos.
- Internetinės viktorinos (paprasti klausimai, dažnai besikeičiantys klausimai, atliekantys tiriamąją funkciją, apklausiamąją funkciją, prizai).
- Diskusijų klubai/Forumai (interesai, prekės, prekių savybės, pomėgiai, išsami oficiali nuomonė, diskusija pagal skaitytojo pareikštą nuomonę).
- Vartotojų klubas (galimybė suinteresuoti ir išlaikyti vartotoją, priemonė tapti komandos nariu, komandinės nuolaidos).
- Nuolaidų akcijos (užsakant prekes, gaunant informaciją, atliekant kitus veiksmus, šventinės akcijos).
- Gaunamų rezultatų analizė ir nuolatinis skatinimo programos palaikymas.

Klausimai ir užduotys

1. Išvardinkite ir apibūdinkite pagrindinius tinklalapių tipus.
2. Kokie yra interneto svetainių kūrimo principai?
3. Ką reikėtų padaryti prieš pradėdant interneto svetainės kūrimo projektą?
4. Kokie yra interneto svetainių lankytojų tipai?
5. Kuo svarbus tinklalapio funkcionalumas?
6. Išvardinkite ir apibūdinkite tinklalapio struktūras?
7. Kokios yra tinklalapių kūrimo priemonės?
8. Kokie yra tinklalapių kūrimo būdai?
9. Kaip reikia prižiūrėti tinklalapį?
10. Ar galima be apribojimų kopijuoti informaciją iš interneto svetainių?
11. Koks internetinio marketingo tikslas?
12. Kokie veiksmai būtini internetinio marketingo sėkmei?
13. Kokie veiksmai atneša internetinio marketingo nesėkmę?
14. Kokiomis priemonėmis kuriamas internetinis marketingas?

3. HTML IR CSS STANDARTAI

3.1. HTML standartas

HTML (angl. *Hyper Text Markup Language*) – tai kompiuterinė žymėjimo kalba, naudojama pateikti turinį internete. HTML visų pirma aprašo loginę žiniatinklio puslapių struktūrą: dokumentų bei juos sudarančių skyrių ir skirsnių antraštes, pastraipas, iliustracijas, lenteles, nuorodas į kitus dokumentus ar kitokius duomenis.

HTML versija žymima skaičiumi, nurodant versiją ir subversiją, pavyzdžiui, HTML versija 4.01. Po šios HTML versijos HTML pagrindu buvo plėtojamas XHTML standartas, kuris dar labiau atskyrė struktūros logiką nuo vaizdavimo, bei remiasi ne SGML, o XML žymėjimo kalba.

2009 m. vasario 12 d. žiniatinklio architektų taryba patvirtino HTML5 juodrašį. HTML5 keičia ir HTML4, ir XHTML1, bet išlieka suderinama su jais. Naujasis standartas smulkiai ir iki galo tiksliai aprašo žymėjimo gaires, nepalikdamas vietos interpretacijai – naršyklės turės vienodai atvaizduoti tinklalapius, vienodai elgtis net ir su HTML rašto klaidomis.

HTML aprašomoji kalba yra visų interneto svetainių pagrindas. Iš pat pradžių HTML buvo statinio teksto aprašomoji kalba, tačiau palaipsniui HTML standartas bei kompiuterių tinklai tobulėjo ir plėtėsi. Todėl laikui bėgant buvo nuspręsta atskirti vaizdavimo aprašus bei duomenų struktūras nuo puslapio struktūros. Taip atsirado puslapių stiliaus aprašai CSS bei XML struktūros. Plintant daugiaterpėms technologijoms norint sukurti interaktyvią svetainę HTML nepakanka, todėl pradėta pildyti visokiausiais papildomais įskiepiais – nuo „JavaScript“ pusprogramių iki dvejetainių „Flash“, „Silverlight“ ir kitų įterptinių objektų.

Tokie dalykai kaip „jQuery“ įterpiniai, formatavimo technikos, keičiasi labai greitai ir jau yra priimta, kad kai kurie dalykai naudojami dabar, bus nevertingi ateityje, tačiau tai yra dalis interneto industrijos. Dažniausiai šioje srityje ieškant nekintančių dalykų atsižvelgiama į paprasčiausią kodą, kuris sąlyginai ilgą laiką būna nepakitęs. Tačiau pasirodžius HTML5 to neliko, iš dalies pakito visa iki tol buvusi HTML sintaksė.

Oficialųjį HTML standarto aprašymą galima rasti „W3Consortium“ puslapyje: <http://www.w3.org/MarkUp/>.

3.1.1. HTML5 standartas

HTML5 standartas – nauja internetinių portalų galimybių era. HTML5 pradėtas kurti nuo 2006 m. Pirmoji HTML5 standarto palaikymą naršyklėje pradėjo įdiegti „Apple“ kompanija, taip „skelbdama karą“ „skelbdama karą“ „Adobe flash“ technologijai.

HTML5 versijoje keitėsi DOCTYPE eilutė, kuri buvo labai supaprastinta. Naudojant HTML5 užtenka tiesiog įrašyti `<!DOCTYPE html>`. Tai palengvino duomenų įrašymą, todėl nebereikia ieškoti internete, kaip tiksliai įrašyti visus duomenis, užtenka paprasčiausiai nurodyti HTML. Keitėsi ir kiti elementai. W3C atsižvelgė į žmonių norus, todėl dabar yra tokie esminiai elementai kaip `<header>`, `<nav>`, `<section>` ir kiti, kurie labai palengvina darbą.

„Header“ elementas savyje talpina įžanginę informaciją į skiltį arba puslapį. Į tai įeina viskas nuo įprastų dokumento antraščių iki duomenų lentelių. „Nav“ elementas skirtas pagrindinėms nuorodoms į tinklalapio dalis ar į atskirus puslapius talpinti. Ne visos nuorodos turi būti šiame elemente. Pakanka į jį įterpti pagrindinį tinklalapio meniu. „Section“ elementas veikia panašiai kaip anksčiau tai darydavo `<div>` – atskiria pasirinktas puslapio dalis. „Article“ skirtas tokioms puslapio dalims kaip tinklaraščio ar forumo įrašas, komentarai ar kitas nepriklausomas turinio elementas. „Footer“ naudojamas žymėti ne tik puslapio pabaigą, bet ir kiekvieną puslapio dalį.

Iš dalies visi šie nauji elementai pakeičia DIV. Jie gali būti naudojami ne vieną, bet keletą kartų, kaip klasės elementai ar kiti, kuriuos galima naudoti pakartotinai. Šie elementai yra naudingi, nes jie labai gerai apibrėžia puslapio struktūrą ir labai palengvina darbą, tačiau reikėtų nepersistengti juos taikant.

Nors HTML 4.01, XHTML 1.0, HTML5 yra labai panašūs, yra smulkių detalių, kurias praleidus kodas būtų sugadintas. Nagrinėjant HTML5 gali kilti klausimas, kam reikia naudoti naują žymėjimą, jei senasis veikia taip pat puikiai. Vadinasi, HTML5 turi daug privalumų, t. y. turi įvairiausių naujų funkcijų, kodo trumpinimo technikų ir kitų naudingų dalykų.

Keletas privalumų – tai daugybė API ir galimybės, kurias jis suteikia ateities internetinėms aplikacijoms ir atjungties (angl. *offline*) galimybės. Didžiausia naujovė, kurią siūlo HTML5 tai jungties galimybės. Dabar galima naudotis tokiais programomis, kaip „Outlook“ ir „Gmail“, ir pasiekti savo senus duomenis be interneto prieigos. HTML5 suteikia tokią galimybę tiesiog naršyklėje.

Daugelyje svetainių vis dar paplitęs „Flash“ įskiepis. „Microsoft“ pateikė savą „Flash“ alternatyvą – „Silverlight“. Tačiau tai yra papildomi įskiepiai, kuriuos vartotojui tenka diegti ir atnaujinti pačiam. Neretai šie priedai tampa nestabilaus naršyklės darbo priežastimi, o autoriui nesuskubus kuriai nors naršyklei sukurti reikiamos priedo versijos – rimtu galvos skausmu tų naršyklių versijų vartotojams. Kuriant naująjį HTML5 standartą, siekta atsikratyti uždarojo kodo trečiųjų šalių įskiepių ir sudaryti galimybę didžiąją dalį interaktyvaus turinio generuoti vien tik su HTML.

HTML5 yra numatyti tokie nauji elementai, kaip <footer> ar <nav>, <section>, <datagrid>, <audio>, <video>, <canvas> ir pan. Pasitelkus <video> elementą į svetainę galima įkelti internetu transliuojamą vaizdo įrašą, nesinaudojant jokiais papildomais priedais, kaip „Flash“ ar „SilverLight“. Vartotojui reikalingi tik vaizdo įrašo *kodekai*, kurie dekoduoja naršyklės atsisiunčiamą duomenų srautą. Tam, kad turinys būtų pasiekiamas kuo didesniai lankytojų srautui, siūloma visiems naudoti atvirąjį vaizdo duomenų formatą. Atvirojo kodo alternatyvos ne tik nenusileidžia, bet ir yra pranašesnės esant mažesnei raiškai.

HTML5 elementas <canvas> leidžia naršyklės pagalba generuoti dvimatės grafikos objektus. Ši <canvas> elementą sugalvojo „Apple“ kompanija, o vėliau jį į savo naršyklės įtraukė „Mozilla“, „Opera“, „Chrome“ kūrėjai ir tik po to buvo nuspręsta jį įtraukti, kaip naudingą funkciją į HTML5 standartą. Pasitelkus <canvas> HTML leidžia aprašyti (nupiešti) norimą dvimatį objektą, nurodant linijas, spalvas, taškus, transformacijas ir pan. Tai leidžia tam tikrais atvejais išsiversti be paveikslėlių, o įvairių grafikų atvaizdavimą aprašyti skaičių lentelėmis ir funkcijomis bei patikėti naršyklės vaizdo generavimo posistemii. Tai gali būti paranku automatiškai generuojamam svetainių turiniui, kuriame reikia pateikti būtent tokio pobūdžio informaciją. Taip sutaupoma daug serverio išteklių, nes užuot generavus reikiamus grafikų paveikslėlius, galima paprasčiausiai išrinkti duomenis ir patikėti šį darbą interneto naršyklei. Naudo-

jantis šiomis priemonėmis galima kurti netgi žaidimus ar kitokius animuotus svetainės elementus.

Didžiąją dalį HTML5 naujovių gali išbandyti naudojantys „Safari 4“ naršyklę, taip pat „Opera 10“, „Firefox 3.5“ , „Chrome 3.0“ ir naujesnes šių naršyklių versijas, nors kai kurios jų jau yra ir senesnėse naršyklių versijose.

3.1.2. HTML5 standarto privalumai

Naršyklių kūrėjai turi daug idėjų, kaip galima patobulinti ir išplėsti naršyklių bei svetainių funkcionalumą, visa tai bandoma sudėti į HTML5 standartą, tačiau susitarimai užtrunka. Naujos HTML paantraštės ir naujos „JavaScript“ funkcijos kai kuriose naršyklėse jau egzistuoja, tačiau tai nėra pilnas standarto palaikymas.

HTML5 turėtų pakeisti daug interneto puslapių kūrimo ir panaudojimo aspektų, jis neišstums „flash“ (pvz., žaidimai Miniclip.com parodo, kiek daug funkcijų HTML5 turėtų būti įdiegta, kad visiškai galėtų pakeisti „flash“ technologiją), tačiau HTML5 pakeis daugumą standartinių interneto svetainių.

Pasikeitimai, įsisavinus HTML5 standartą, t. y. HTML5:

- sumažins įskiepių (angl. *plug-ins*) svarbą;
- leis naudoti interaktyvesnę grafiką;
- leis naudoti daugiau vietos diske;
- turės duomenų formato identifikavimo priemones;
- galės nustatyti kliento kompiuterio buvimo vietą;
- supaprastins žiniatinklio vaizdo įrašų panaudojimą;
- numato priemones tarpusavyje bendraujantiems „Widgets“ realizuoti;
- pagerins saugumo užtikrinimą;
- supaprastins svetainių kūrimo procesą.

HTML5 sumažins įskiepių (angl. *plug-ins*) svarbą

Naršyklės įskiepių panaudojimas paskatino kūrybiškumą ir privertė svetainių kūrėjus eksperimentuoti. „Sun“, „Adobe“, „RealAudio“, „Microsoft“ ir kitų kūrėjų sukurtų įskiepių dėka puslapiuose atsirado garsai, judantys paveiks-

lėliai ir kiti „triukai“. Naujos HTML5 galimybės leis pakeisti kai kuriuos Flash panaudojimo aspektus, „JavaFX“ yra geras dalykas, tačiau kam mokintis kitą programavimo sintaksę kai tuo tarpu HTML5 „Canvas“ objektas ir „JavaScript“ turi tokias pat galimybes. Vaizdo žymė sinchronizuoja vaizdą ir garsą pati, todėl nebereikalinga „Real ecosystem“. Paprastesniems darbams atlikti įskiepai jau nebebus reikalingi, tačiau visi įskiepai iš naršyklių neturėtų dingti, nes sudėtingesniems sumanymams realizuoti (pvz., 3D žaidimai) HTML5 galimybių gali ir nepakakti.

HTML5 leis naudoti interaktyvesnę grafiką

Dabartiniai puslapiai atvaizduoja grafiką užkraudami ją iš JPG, GIF ar PNG failų. HTML5 turi priemonės grafikos atvaizdavimui „on the fly“, tam naudojamas „Canvas“ objektas. Jau atsirado nemažai grafikos apdorojimui skirtų bibliotekų, jos visos gali puslapio grafiką padaryti interaktyvesnę.

Atsiradus sudėtingesnėms grafikos apdorojimo galimybėms svetainių lankytojai, besinaudojantys senais lėtais kompiuteriais, gali susidurti su didelio kompiuterio apkrovimo problema. Dabar „Flash“ įskiepi galima atjungti siekiant išvengti kompiuterio apkrovimo atvaizduojant daug „flash'o“ turinčią svetainę. Tačiau HTML5 tokios galimybės nebus.

HTML5 leis naudoti daugiau vietos diske

Interneto svetainių programuotojai visada galėjo saugoti sąlyginai didelį informacijos kiekį „sausainiukuose“ (angl. *cookies*, IE 300 *cookies* iki 4096 baitų), tačiau sudėtingesniems darbams šis kiekis gali būti per mažas. HTML5 programuotojams leis naudoti daugiau vietos diske, šią atmintį jie galės naudoti savo nuožiūra, tai leis persiusti ir vykdyti aplikacijas, kurios bus kaip paprastos programos, žaidimų kūrėjai galės informaciją saugoti lokaliai kompiuteryje, taupydami laiką šios informacijos persiuntimui vėl ir vėl.

Ši puslapių informacija saugoma giliai sisteminiuose kataloguose ir šių duomenų atsarginių kopijų darymas bei perkėlimas į kitą sistemą gali būti ne pats paprasčiausias žingsnis.

HTML5 turės duomenų formato identifikavimo priemones

Dabar HTML informacijoje kol kas nėra numatytos priemonės identifikuoti pateiktos informacijos prasmę. HTML5 turi taip vadinamą „mikrofor-

matą“ – priemonės duomenims HTML kode paženklinti, kad vėliau būtų paprasčiau tuos duomenis analizuoti. Pavyzdžiui, jeigu būtų sutartas vienas duomenų „mikroformatas“ datai atvaizduoti, tada analizuojant duomenis nebereiktų naudoti specifinių duomenų formato analizės priemonių, tai supaprastintų kalendorių, planavimo priemonių integralumą.

HTML5 galės nustatyti kliento kompiuterio buvimo vietą

Į žiniatinklio serverį kol kas siunčiamas tik kliento IP adresas – sąlyginai anonimiški skaičiai, nenusakantys realios kliento buvimo vietos. HTML5 standarte numatyta, kad „JavaScript“ pagalba bus galima užklausti naršyklės nustatyti realias koordinates. Su staliniais kompiuteriais tai paprastai nereikalingas funkcionalumas, nes koordinatų nustatymui reikalinga tam tikra techninė įranga (GPS, „Wi-Fi“), tačiau tai puikiai galima pritaikyti išmaniesiems telefonams.

HTML5 supaprastins vaizdo panaudojimą žiniatinklyje

HTML5 vaizdo žymėjimas leis svetainių kūrėjams paprasčiau panaudoti vaizdo informaciją, paprasčiau ją integruoti su visais kitais svetainės duomenimis. Vaizdo informaciją bus galima valdyti ir PHP bei „jQuery“ pagalba ne tik per „Flash“, „Silverlight“ ar „JavaFX“. HTML5 standarte vaizdo ir garso „grojimas“ nepririštas prie konkrečių *kodekų*, o tai reiškia, kad jei reikiamo *kodeko* kliento kompiuteryje nebus – ši informacija nebus korektiškai atvaizduojama.

HTML5 numato priemonės tarpusavyje bendraujantiems „Widgets“ realizuoti

„IFrame“ technikos priemonėmis realizuoti „Widgets“ ilgą laiką buvo naudojami informacijai iš kitų svetainių atvaizduoti, tačiau jų panaudojimo galimybės buvo apribotos saugumo sumetimais laikant kiekvieną „Widget“ atskiroje „dėžutėje“. HTML5 siūlo standartinį mechanizmą „Widgets“ panaudojimui, be to, „Widgets“ vienas su kitu galės „susikalbėti“ siųsdami vienas kitam pranešimus koordinuodami savo darbą. Reklamų talpintojai turėtų susidomėti galimybe koordinuoti po puslapį išmėtytų reklamjuosčių elgseną, o kūrėjai ras ir kitų šios technikos pritaikymo galimybių.

HTML5 pagerins saugumo užtikrinimą

Kiekvienas naršyklės įskiepis – tai atskira skirtingų kūrėjų programa su skirtingais standartais ir skirtingais saugumo modeliais. Kai kurie įskiepiei yra „saugesni“ už kitus. Gali kilti daugybė klausimų, ar saugumo skylės naršyklėse atsiranda dėl pačios naršyklės, ar dėl kažkurio įskiepio, tuomet reikia sekti tiek naršyklės, tiek įskiepių atnaujinimus. HTML5 integruotos tam tikrų įskiepių funkcijos leidžia atsisakyti šių įskiepių, todėl sumažinama ir saugumo spragų tikimybė.

HTML5 supaprastins svetainių kūrimo procesą

HTML5 palaiko vieną kalbą („JavaScript“), vieną duomenų modelį (XML ir DOM) ir vieną turinio išdėstymo taisyklių rinkinį (CSS). Sukurti ką nors gražaus – vis dar išlieka didžiulis iššūkis, tačiau tai padaryti bus paprasčiau.

3.2. HTML redaktoriai

Skirtingai nuo tekstų redaktoriaus „Microsoft Word“ vartojamo dokumentų formato DOC ar puslapių aprašymo kalbos „PostScript“, HTML niekuomet neapibrėžia vienareikšmiškai, kaip žiniatinklio puslapis turi atrodyti kompiuterio ekrane. Tai priklauso nuo pasirinktos ekrano raiškos, tinklo naršyklės lango dydžių, teksto vaizdavimui pasirinkto šrifto dydžio bei daugybės kitų parametrų, taip pat ir nuo to, kokia interneto naršyklė naudojama žiniatinklio puslapių žiūrėjimui: „Internet Explorer“, „Opera“, „Chrome“, „Safari“ ar dar kitokia. Pastarasis faktas labiausiai nepatinka daugeliui interneto svetainių dizainerių, kadangi reikia nuolat tikrinti, kaip puslapį parodo viena ar kita tinklo naršyklė bei ieškoti kompromiso.

Pirmieji žiniatinklio puslapiai buvo rašomi paprasčiausiais teksto redaktoriais, bet greitai atsirado ir specializuotų programų – HTML redaktorių. Programinės įrangos rinka siūlo milžinišką HTML redaktorių pasirinkimą. Dabar egzistuoja trijų tipų HTML redaktoriai:

1. *Text* – tai tekstiniai redaktoriai, kurie specialiai patobulinti, pritaikant juos HTML dokumentų kūrimui.

2. *Toolbar* – tai specializuoti HTML dokumentų redaktoriai, kurių meniu, kodų ir kitos sistemos optimaliai pritaikytos HTML dokumentų kūrimui. Dažnai turi papildomas galimybes – interaktyvų ir kompleksinį HTML dokumentų kūrimą, tikrinimą, administravimą ir pan.
3. *WYSIWYG* – redaktorių šeima, kuri suteikia galimybes kurti ir redaguoti tiesiogiai HTML vaizdą ir visiškai ar beveik nereikalauja HTML kalbos žinojimo, nes dokumento konvertavimą į HTML kalbą atlieka (sėkmingai ar ne) pati programa.

Pastarieji redaktoriai vadinami WYSIWYG (angl. *What You See Is What You Get*) – tai, ką tu matai yra tai, ką tu gausi) programomis, kadangi ir redagavimo metu puslapis būna maksimaliai panašus į tą, kurį tinklo naršyklės ekrane matys tinklalapio lankytojai. Šis trumpinys turėtų skambėti WYSIAWYG (angl. *What You See Is Approximately What You Get*), nes HTML apibrėžia žiniatinklio puslapio išvaizdą tik gana apytiksliai.

Tekstiniai redaktoriai pažįsta daugelį standartinių HTML komandų (angl. *tags*) ir automatiškai įterpia jas nurodytoje HTML teksto vietoje. Rezultatą galima pamatyti išsikvietus įprastą tinklo naršyklę („Explorer“, „Chrome“, „Opera“, „Safari“ ar kitą). Šiuo atveju iš pat pradžių reikia žinoti HTML kalbą ir turėti pakankamai lanki vaizduotę. Kitaip gali tekti daug laiko prarasti junginėjančiam iš HTML redaktoriaus į peržiūros programą tam, kad pamatytume, kaip atrodys kuriamas puslapis, po to vėl atgal, kad ką nors pataisytume. Pirmieji kūrybiniai bandymai su tekstiniais redaktoriais pareikalauja iš pradedančiųjų gana daug laiko. Bet toks būdas leidžia dirbti lanksčiau, nes tekstiniai HTML redaktoriai tik padeda, ne nurodinėja, o tai yra didelis privalumas, ypač labiau patyrusiems interneto svetainių kūrėjams. Pradedančiųjų autorių puslapiai, parašyti naudojant tekstinius redaktorius, dažniausiai būna gana paprasti ir kuklūs, be ypatingų įmantrybių, bet užtat vienodai gerai matomi visomis žiniatinklio peržiūros priemonėmis.

Naudojantis vizualinėmis (WYSI(A)WYG) tinklalapių kūrimo programomis atkrenta būtinybė žinoti HTML kalbos struktūrą ir komandas, galima greičiau gauti norimą rezultatą ir mikliai sukurti gana sudėtingus tinklalapius, bet dizaineris lieka labiau priirištas prie programos, kuri kartais pernelyg daug prirašo, kartais kažką savavališkai surikiuoja ar ištaiso. Teksto redaktoriumi „Microsoft Word“ taip pat galima kurti HTML puslapius, tačiau pabandžius

pagaminti juo ką nors sudėtingesnio ir pasižiūrėjus, kaip tai atrodo tinklo naršyklės lange, galbūt praeitų noras išvis ką nors kurti. Ypač pradedančiųjų autorių vizualinėmis priemonėmis sukurti puslapiai dažnokai išsidarko pasirinkus ne tą naršyklės ekrano dydį, kurį vartojo kūrėjas arba vietoje lietuviškų raidžių juose matyti tik kažkokie neaiškūs simboliai. Tačiau daugeliu atvejų WYSIWYG redaktoriai įgalina pasiekti pakankamai gerų rezultatų per trumpiausią laiką.

Naujosios tinklalapių kūrimo programos stengiasi išvengti minėtų trūkumų: WYSIWYG redaktoriai siekia suteikti autoriams vis daugiau lankstumo ir kūrybinės laisvės, o tekstiniai redaktoriai įgyja daugiau darbą palengvinančių ir automatizuojančių funkcijų. Kai kurių HTML redaktorių, kaip antai „HoT-MetaL“, jau nebegalima vienareikšmiškai priskirti nei prie vienos, nei prie kitos kategorijos, nes jie turi ir vizualinių, ir tekstinių HTML rašymo priemonių.

Dar viena internetinių puslapių kūrimo priemonių grupė – tai programos, konvertuojančios įvairiais kitais būdais sukurtus duomenis į HTML formatą. Vieni iš tokių programų yra kompanijos „Microsoft“ nemokamai platinami „Interneto asistentai“, bene visoms „Microsoft Office“ programoms („Word“ ar „Excel“). Panašių instrumentų galima rasti ir daugiau, bet absoliuti dauguma jų dirba greitai ir nešvariai (angl. *Quick and dirty*) stiliumi ir naudojami tik norint publikuoti internete daug informacijos, paruoštos įprastomis teksto redagavimo programomis ar skaičiuoklėmis. Šiuo atveju, vargu ar galima bus apsieiti be tolimesnio HTML teksto pataisymo. Gudresni HTML redaktoriai ir patys moka paimti duomenis iš kitomis programomis sukurtų failų.

Viena iš kai kurių WYSIWYG redaktorių savybių (blogybių) yra nesaikingas teksto parametrų aprašymas, kai nurodoma vartoti konkrečius šriftus, pav., „TimesLT“. „TimesLT“ ar bet kokią kitą šriftą savo kompiuteryje gali turėti toli gražu ne kiekvienas puslapio lankytojas, juo labiau jeigu jis vartoja ne „Windows“, o tarkim „Linux“ operacinę sistemą.

Prieš publikuojant puslapius būtina pasitikrinti, kaip juos atvaizduoja įvairios tinklo naršyklės – „Internet Explorer“, „Chrome“, „Safari“, „Opera“, „Mozilla Firefox“ bei skirtingos jų versijos.

3.2.1. Ar verta mokėti HTML?

Vartojant vizualinius (WYSIWYG) tinklalapių redaktorių, HTML kalbą mokėti nėra būtina. Naujausios programos įgalina kurti gana gerus tinklalapius visiškai nieko nenučiuojant apie HTML. Bet daugelis profesionalių interneto svetainių dizainerių iki šiol rašo HTML kodą patys ir naudoja tekstinius redaktorių, kadangi dažnai tik toks puslapių kūrimo būdas leidžia iki galo išreikšti savo, kaip kūrėjo, mintį.

Vizualinių HTML redaktorių (pvz.: „Microsoft FrontPage“ ir kt.) sukurtame HTML tekste dažnai būna pernelyg daug puslapio išvaizdą apibrėžiančių komandų. Dėl šios priežasties, nepatyrusio ir patiklaus internauto sukurtas puslapis netenka savo universalumo ir tinkamai matosi tik vartojant konkrečią tinklo naršyklę ar ekrano raišką, turint įdiegtus tam tikrus šriftus ir t.t. Dar blogiau, kai tokiais programomis tekstas rašomas taip, kaip pradedančios sekretorės tai daro teksto redaktoriumi „Word“, vietoj to, kad naudotų <CENTER> ar <P ALLIGN= "Center"> komandas pastūmimas atliekamas į dešinę keliais ar keliasdešimčia tarpo klavišo paspaudimų ir pan. Štai dėl ko, bent jau teorinės HTML žinios tinklalapių kūrimui yra būtinos.

Visuomet bus tokių dalykų, kurių kompiuteris pats padaryti nesugebės arba padarys taip, kaip norės jis, o ne žmogus. WYSIWYG redaktoriaus vartojimas neatleidžia nuo būtinybės bent jau elementariai nusimanyti apie HTML ir esant būtinybei mokėti ištaisyti tai, ką programa padarė savaip. Beveik visos vizualinio puslapių kūrimo sistemos turi integruotus HTML teksto redaktorių tam, kad palengvintų šį darbą. Vadinasi, bent jau HTML kalbos pagrindus turėtų žinoti kiekvienas tinklalapių kūrėjas.

3.2.2. Dokumentų formato ir perdavimo standartai

Žiniatinklio protokolai

Žiniatinklis (pasaulinis voratinklis, angl. *World Wide Web* – WWW) – tarpusavyje susietų HTML dokumentų ir juos papildančių kitokių failų, prieinamų internete http protokolu, visuma.

HTML (angl. *Hypertext Markup Language*) – teksto su nuorodomis žymių kalba, kurios paskirtis aprašyti dokumento struktūrą.

HTTP (angl. *Hyper Text Transfer Protocol*) – hiperteksto persiuntimo protokolas – taisyklių ir susitarimų rinkinys, skirtas hipertekstinių dokumentų persiuntimo būdai apibrėžti, kai informacija siunčiama iš tarnybinės stoties į vartotojo kompiuterį. Kai vartotojas savo interneto naršyklėje nurodo adresą ar paspaudžia interneto puslapyje nuorodą, jo kompiuteris susijungia su tinklo tarnybine stotimi ir siunčia į ją pranešimą. Tada iš tarnybinės stoties siunčiami failai, reikalingi interneto svetainės puslapiui vaizduoti.

Pagrindinis šio protokolo tikslas buvo suteikti galimybę skelbti ir peržiūrėti HTML puslapius. HTTP protokolas pagrįstas užklausų/atsakymų apsikeitimu tarp kliento ir serverio. Tarp paplitusių protokolų yra FTP, NNTP, SMTP ir kt. HTTP protokolas yra naudojamas dažniausiai.

Šiuo metu žiniatinklio paslaugos netelpa tik į HTTP protokolo rėmus, todėl sukurta daug patobulintų protokolų, skirtų įvairioms kliento reikmėms tenkinti. Vienas iš populiariausių papildomų servisų yra HTTPS (Secure HTTP) – saugus HTTP protokolas, kurį UNIX sistemoje valdo SSL (angl. *Secure Socket Layer*). HTTPS – šifruota HTTP protokolo atmaina.

SSL (angl. *Secure Socket Layer*) – tai rinkos standartu tapusi technologija, kuri garantuoja saugų duomenų perdavimą internetu. Tai kriptografinis protokolas skirtas informacijos, sklindančios internete apsaugojimui šifruojant.

SSL protokolo veikimą supaprastintai galima pavaizduoti:

1. Vartotojas ateina į apsaugotą svetainę.
2. Serveris patvirtina svetainės tapatybę pasirašydamas sertifikatą ir nusiųsdamas jį klientui. Naršyklė panaudoja sertifikato viešąjį raktą (viešasis raktas gaunamas kartu su sertifikatu) serverio sertifikato savininko tapatybei patikrinti.
3. Naršyklė patikrina, ar sertifikatas buvo išduotas žinomo sertifikavimo paslaugų teikėjo. Jeigu sertifikatas yra išduotas nežinomo paslaugų teikėjo, naršyklė apie tai informuoja vartotoją.
4. Vartotojas pats gali patikrinti ar sertifikavimo paslaugų teikėjas išdavė sertifikatą tikrai tai svetainei, į kurią atėjo vartotojas.
5. Serveris reikalauja vartotojo skaitmeninio sertifikato, kad patikrinti jo tapatybę.
6. Vartotojas pasirenka, kurį iš jo turimų sertifikatų (žinoma, jeigu jis turi daugiau nei vieną sertifikatą) parodys serveriui.

7. Serveris sukuria ir užkoduoja seanso raktą bei saugiai nusiunčia jį internetu, ir tokiu būdu sukuriamas saugus virtualus kanalas tarp vartotojo naršyklės ir žiniatinklio serverio.

HTTPS protokolas paremtas SSL technologija yra puiki priemonė saugiai keistis duomenimis internete. HTTPS protokolas tinka naudoti:

- e. versle;
- elektroninėje bankininkystėje;
- visur, kur reikalingas saugus duomenų perdavimas.

Nuorodos

Labai svarbus HTML komponentas yra nuorodos. Ne veltui jos paminėtos ir kalbos pavadinime (ang. *hypertext*). Buvo svarbu sukurti bendrą adresų sintaksę, kuri leistų nurodyti tikslią vietą konkrečiame dokumente.

Bendra nuorodos struktūra (adresas rašomas be tarpų):

```
protokolas: serveris / kelias / failas ?  
parametrai # vieta
```

Pavyzdžiui:

```
http://www.pvz.lt/studentai/tpm.php?id=10#2_sk
```

Svarbu tai, kad bet kuris elementas gali būti praleistas. Jeigu nėra nurodyto protokolo, naršyklė pasirenka įprastą HTTP protokolą. Jeigu nėra nurodyto serverio, naršyklė ieškos dokumento tame pačiame serveryje, kuriame buvo nuoroda. Taip gaunamos santykinės nuorodos.

Kelias nenurodomas, jei norima atversti serverio šakniniame aplanke esantį dokumentą. Jei nenurodytas nei serveris, nei kelias, tada failo ieškoma tame pačiame aplanke, iš kurio gauta nuoroda. Nenurodžius failo, serveris pateikia nustatytą standartinį aplanko failą. Parametrų gali ir nebūti, o nenurodžius konkrečios vietos dokumentas rodomas nuo pradžios.

Nuorodose į kitus resursus tame pačiame serveryje visada įmanomas absoliučių ir santykinų nuorodų pasirinkimas.

Pavyzdžiui, absoliutus adresas:

```
http://www.pvz.lt/abc/def.html#pirmas
```

Tas pats resursas rodant iš kito aplanko tame serveryje:

```
/abc/def.html#pirmas
```

Tas pats resursas rodant iš kito failo tame pačiame aplanke:

```
def.html#pirmas
```

Vieta dokumente rodant iš to paties dokumento:

```
#pirmas
```

3.3. HTML pagrindai

HTML (angl. *HyperText Markup Language*) – tai tinklalapių aprašymo, hiperteksto žymėjimo kalba, kuria „kalba“ pasaulinio tinklo žiniatinklio serveriai ir kurią „supranta“ tinklo naršyklės. Iš esmės, HTML dokumentas yra paprastas tekstinis failas, kuriame yra tekstas ir nieko daugiau. Taigi, sukurti interneto puslapį galima su bet kurio teksto redaktoriaus pagalba.

Pagrindinis HTML kalbos vienetas yra elementas. Kaip ir XML, HTML elementas turi vardą ir gali turėti bet kokį skaičių atributų. Elemento viduje gali būti tekstas bei kiti elementai. Tiek tekstas, tiek ir dukteriniai elementai paprastai gali kartotis ir sekti bet kuria tvarka.

Elemento atributai turi vardą ir reikšmę. Jei galimi atributai nenurodomi, paprastai galioja sutartos nutylėjimo taisyklės. Pavyzdžiui, HTML fragmente `ne mėlyna ` yra du elementai. Elementas „font“ paverčia rodomo teksto spalvą į mėlyną, o šio elemento viduje esantis antrasis elementas „b“ paverčia šriftą paryškintu. Pirmasis elementas turi vieną atributą „color“, kurio reikšmė yra „blue“. Antrasis elementas atributų neturi. HTML kode atributų reikšmes rekomenduojama apgaubti kabutėmis, nors HTML 4.01 standartas to nereikalauja.

3.3.1. HTML elemento konstrukcija

HTML kalba sukurtas dokumentas – tai tekstas su specialiais žymenimis tarp ženklų „<“ ir „>“. Taip tekstas padalinamas į tokio formato elementus:

```
<komanda> turinys </komanda>
```

Turinys gali būti tekstas bei kiti elementai, komanda gali turėti parametrus:

```
<komanda param="reikšmė"> turinys </komanda>
```

HTML dokumentus sudarantys žymėjimo elementai (angl. *tags* – žymės) gali būti išskirstomi į šias grupes:

- Struktūriniai elementai – nusako teksto paskirtį, pavyzdžiui:

```
<h1>Pavadinimas</h1>
```

naršyklei nurodo, kad tai yra pirmo lygio antraštė (angl. *header*). Standartinė išvaizda gali būti pakeista naudojant stilius (CSS).
- Prezentaciniai elementai – nusako teksto išvaizdą, neatsižvelgiant į teksto paskirtį, pavyzdžiui:

```
<b>tekstas</b>
```

atvaizduos tekstą paryškintai (angl. *bold*), tačiau siūloma prezentacinių elementų atsisakyti ir naudoti CSS.
- Hipertekstiniai elementai (nuorodos) – sukuria nuorodas tarp dokumentų ar jų dalių, pavyzdžiui:

```
<a href="http://www.w3c.org">W3C</a>
```

sukurs tekstą „W3C“, kurį spragtelėjus naršyklė bus nukreipiama į www.w3c.org tinklalapį.
- Kiti elementai, pavyzdžiui:

```

```

įkels paveikslėlį "paveikslas.gif"

3.3.2. Vaizdas ar turinys

Nuo pat pradžių HTML kalba buvo sukurta kaip priemonė turiniui perteikti nepriklausomai nuo naudojamos įrangos. Autorius sužymi dokumento struktūrą, o naršyklė ją atvaizduoja taip, kad dokumento skaitytojui būtų kuo patogiau. Autorius neturėtų rūpintis išvaizda, nes jis nežino, kokios yra dokumentą atverčiančios naršyklės galimybės.

Tačiau HTML dokumentus ėmus kurti plačiajai visuomenei paaiškėjo, kad daugelis autorių labiau rūpinasi vaizdu negu turiniu. Naršyklių kūrėjai siekė papildyti HTML kalbą kuo daugiau išvaizdą aprašančių elementų. Autoriai pamėgo klaidingai naudoti turinio elementus dokumentui pagražinti. Pavyzdžiui, `<h1>` elemente pateikiamas visai ne pavadinimas, bet sakinio dalis, kurią norima pabrėžti ar išskirti. HTML kūrėjai pritarė „švariam“, be dizaino priemaišų HTML, o naujai atsiradusi ir vis stiprėjanti nekvalifikuotų naudotojų grupė reikalavo galimybių gražinti dokumentą, kištis į jo atvaizdavimą.

Kalba papildyta ne tik išvaizdai skirtais elementais `` („bold“ – pušiau juodis šriftas), `<i>` („italic“ – kursyvas – pasvirasis šriftas), bet ir kitiems elementams suteikiami išvaizdai nurodyti skirti parametrai, pavyzdžiui, `<body background=“red“>`.

Nevienareikšmiška buvo ir bendruomenės pozicija dėl HTML taisyklių griežtumo. Kompiuteriams tampant vis galingesniems, naršyklės lengvai ištaiso HTML netikslumus. Pavyzdžiui, toks kodas, nors ir yra netikslus, buvo pripažintas teisingu:

```
<p>Kur lygūs laukai,<p> Snaudžia tamsūs miškai
```

Nors nepažymėta pirmosios pastraipos pabaiga, bet akivaizdu, kad jei prasidėjo antroji pastraipa, tai pirmoji pasibaigė.

Naršyklėms nebuvo sudėtinga priimti dokumentą nepriklausomai nuo to, ar didžiosios, ar mažosios raidės naudojamos HTML komandoms. Todėl komandas kiekvienas rašė kaip kam patinka. Todėl paplito trys užrašymo sistemos: `<body>`, `<BODY>` ir `<Body>`. Sintaksės reikalavimai leido rašyti ir `<BoDy>` ar dar 12 šios komandos variantų.

Dėl sintaksės tikslumo nors ir nebuvo tokių didelių ginčų, kaip esmės ir formos kovoje, bet vieni autoriai siūlė praleisti nuspėjamus HTML kompo-

mentos, taip sutrumpinti dokumento apimtį, sutaupyti vietą serveryje bei krovimosi laiką, kiti siūlė laikytis griežtos HTML sintaksės.

3.3.3. Apipavidalinimas ir stilius

Į interneto svetainę derėtų žiūrėti kaip į knygą ar žurnalą. Pirmasis tinklapio puslapis – tai lyg leidinio viršelis, kurio tikslas yra pritraukti potencialaus skaitytojo dėmesį, sudominti ir priversti pasilikti čia ilgiau. Pirmas puslapis dažniausiai būna spalvingesnis už kitus, bet pernelyg neperkrautas informacija. Čia galima rasti nuorodas į kitus svetainės puslapius bei jų grupes, kartais – svarbiausias naujienas.

Likusiems puslapiams derėtų suteikti vieningą išvaizdą. Savo stiliumi jie gali skirtis nuo „viršelio“, bet neturėtų pernelyg skirtis vienas nuo kito. Tuo tikslu reikėtų naudoti vienodas piktogramas bei kitus grafinius elementus.

Būdamas bet kurioje tinklapio vietoje, lankytojas turėtų lengvai rasti kelią į pagrindinį puslapį. Negalima paslėpti ir savo elektroninio pašto adresų bei telefono numerių.

3.4. HTML dokumento struktūra

3.4.1. Paprasčiausias HTML dokumentas

Kai interneto puslapis atidaromas naršyklėje, ji „peržiūri“ HTML kodą, randa specialius simbolius, vadinamus žymėmis (angl. *tag*) ir naudoja jas paveikslėlio įterpimui, teksto dydžio, spalvos ar kitų parametru keitimui, nuorodos sukūrimui į kitus interneto puslapius ar el. pašta ir t.t.

HTML dokumentą sudaro keturios pagrindinės dalys:

- HTML – visi HTML dokumentai prasideda nuoroda, kuri informuoja naršyklę, kad dokumentas yra parašytas naudojant HTML kalbos elementus.

- HEAD – vadinamoji HTML „antraštė“, nurodanti įvairią techninę informaciją, pavyzdžiui dokumento autorių, programą, su kuria dokumentas buvo kurtas, naudojamą kodų lentelę ir pan.
- TITLE – dokumento pavadinimas, kuris rodomas lango antrašėje ir naudojamas automatiškai kuriant nuorodą į dokumentą.
- BODY – pagrindinė HTML dokumento dalis, susidedanti iš HTML komandų ir teksto.

Visas HTML kodas pateikiamas elemente `<html>`, kuris susideda iš dviejų dalių. Pradžioje techninė informacija pateikiama elemente `<head>`, pasakui dokumento turinys pateikiamas elemente `<body>`. HTML kalbą sudaro komandos, kurios nurodo peržiūros programai, kaip reikia vaizduoti tekstą ir kitus objektus. HTML komanda susideda iš vardo ir argumentų, „apskliaustų“ ženklais '`<`' ir '`>`', pavyzdžiui `<p align="center">`. dažniausiai komandos turi ir savo pabaigos žymę, rodančią, kur baigiasi komandos veikimas ir susidedančią iš ženklo '`/`' ir komandos vardo „apskliaustų“ ženklais '`<`' ir '`>`', pavyzdžiui `</p>`. Komanda nurodo kaip turi atrodyti tekstas, esantis tarp komandos ir jos pabaigos žymės. Pats elementariausias HTML dokumento pavyzdys:

```

<html>
  <head>
    <title>Pirmasis HTML pavyzdys</title>
  </head>
  <body>
    <h1>Pavadinimas</h1>
    <p>Pirmoji pastraipa.</p>
    <p>Antroji pastraipa.</p>
  </body>
</html>

```

(1)

Visos keturios dokumento dalys yra nurodomos HTML komandomis. Jų vardai atitinka dalių angliškus pavadinimus. Kiekviena jų turi pabaigos žymę. Tai pagrindinės HTML kalbos komandos. Pirmoji jų – `<HTML>` yra privaloma. Visi HTML dokumentai turi prasidėti komanda `<html>` ir baigtis žyme `</html>`.

Analogiškai kitas dokumento dalis atitinka komandos:

- *Head* – `<head>` ir pabaigos žymė `</head>`.
- *Title* – `<title>` čia rašomas pavadinimas ir `</title>`.
- *Body* – `<body>` čia talpinama visa informacija ir kitos HTML komandos ir `</body>`.

Be anksčiau paminėtų elementų, buvo panaudoti:

- `<p>` – pastraipos elementas.
- `<h1>` – skirtas kurti aukščiausio lygio pavadinimams dokumento tekste (angl. *heading 1*). Smulkesnių skyrių pavadinimai žymimi `<h2>`, jų poskyriai – `<h3>` ir taip toliau.

Šis HTML kodas atitinka visų HTML versijų reikalavimus.

3.4.2. Svarbiausi HTML dokumento elementai

`<BODY> ... </BODY>`

Informacija talpinama tarp komandos `<BODY>` ir jos pabaigos žymės. Komanda `<BODY> ... </BODY>` turi daugybę atributų ir skirta ne tik išskirti HTML dokumento kūną, kaip atskirą jo struktūros elementą, bet ir nurodyti pagrindines dokumento charakteristikas. Trumpas atributų aprašymas:

- `BGCOLOR="spalvos kodas|vardas"`
– nurodo, kokios spalvos bus viso HTML dokumento fonas.
- `TEXT="spalvos kodas|vardas"`
– nurodo, kokios spalvos bus tekstas HTML dokumente (jei spalva nebus nurodyta kitaip).
- `LINK="spalvos kodas|vardas"`
– nurodo, kokios spalvos bus „naujos“ nuorodos: nuorodos, kurios dar nebuvo aplankytos vartotojo (neužfiksuotos naršyklės kaip lankytos).
- `BACKGROUND="bylos_vardas|nuoroda"`
– nurodo, kokia grafinė byla bus naudojama kaip fonas HTML dokumentui. Paprastai tai nedidelis grafinis fragmentas, kartojamas tiek kartų, kol užpildo visą HTML dokumentą.

Tekstą galima rašyti naudojant patį įvairiausių teksto formatavimą – tarpus, tabuliaciją, perkėlimą į kitą eilutę ar centravimą, tačiau tai ignoruos naršyklė – ji dėmesį kreips tik į HTML komandas.

```

<html>
  <head>
    <title>Pirmasis HTML pavyzdys</title>
  </head>
  <body>
    Pavadinimas
    Pirmoji pastraipa.
    Antroji pastraipa.      Pabaiga.
  </body>
</html>

```

(2)

Pateiktame pavyzdyje pavadinimas yra tarpais nukeltas į centrą, pirmoji ir antroji pastraipos pateiktos naujoje eilutėje, o pabaiga nukelta į dešinę. Tačiau būtent taip pateikto teksto dokumento peržiūros programa nevaizduos. Visas tekstas bus vienoje eilutėje ir tarpo tarp dviejų paskutinių sakinių irgi nebus – visi teksto formatavimo ženklai ignoruojami.

Pavadinimai

Tekstas dažniausiai skirstomas į dalis ir kiekviena jų turi savo atskirą pavadinimą. Tai frazė, atskirta nuo kito teksto ir galbūt rašoma didesnėmis ir ryškesnėmis raidėmis nei visas kitas tekstas. Pavadinimams yra išskirta atskira HTML komandų grupė „Headings“.

Komandos susideda iš raidės „h“ ir skaičiaus nuo 1 iki 6. Didžiausias raidės nurodo komanda <h1>, o mažiausias – <h6>. Komandos pabaigos žymė yra </h>, čia „i“ – skaičius nuo vieno iki šešių.

```

<H1>Didelis tekstas</H1>

  <H2>Truputį mažesnis te-
    kstas</H2>
  <H3>Dar truputį mažesnis</H3>
    <H4>Mažas tekstas</H4>
      <H5>Visai visai mažas tekstas</H5>
        <H6>Mažesnio pavadinimo vargu ar bereikia...</H6>

```

(3)

Pastraipos

HTML dokumente esantį tekstą naršyklė pati automatiškai dalija į eilutes pagal peržiūros lango plotį ir ignoruoja visus formatavimo simbolius. Teksto skirstymui į atskiras pastraipas naudojama komanda `<p>` (angl. *Paragraph*). Kiekviena nauja pastraipa turėtų būti pradedama ja. Pastraipos pabaigos žymė yra `</p>`, bet ji nėra būtina. Antro kodo pavyzdį galima pataisyti taip:

```

<html>
  <head>
    <title>Pirmasis HTML pavyzdys</title>
  </head>
  <body>
    <h1>Pavadinimas</h1>
    <p>Pirmoji pastraipa.</p>
    <p>Antroji pastraipa. Pabaiga. </p>
  </body>
</html>

```

(4)

Dabar tekstas, esantis po komandos `<p>` prasidės naujoje eilutėje. Tokiu būdu galima patogiai suskirstyti ilgą tekstą į dalis, tuo palengvinant jo skaitymą.

3.4.3. Teksto formavimas

Visos teksto formavimo komandos turi pradžios ir pabaigos žymes. Komanda lemia tik teksto, esančio tarp šių dviejų žymių, vaizdą. Tekste, esančiame tarp vienos komandos pradžios ir pabaigos žymių, taip pat galima naudoti kitas komandas.

Pagrindiniai teksto formavimo būdai

Pagrindiniai trys teksto stiliai (0) yra jau seniai įprasti paryškintas (angl. **Bold**), kursyvas (angl. *Italic*) ir pabrauktas (angl. Underline).

2 lentelė. Pagrindiniai teksto formavimo stiliai

Komanda	Pabaigos žymė	Pavyzdys
		Šis tekstas ryškesnis ir geriau matomas
<i>	</i>	<i>Kursyvu parašytas tekstas</i>
<u>	</u>	<u>Pabrauktas tekstas, paprastai kas nors svarbaus</u>

Šrifto pakeitimas

Skirtingoms teksto dalims galima naudoti skirtingus šriftus ir spalvas. Šrifto ir spalvos keitimui naudojama komanda , kurios pabaigos žymė yra . Pagrindiniai šios komandos atributai pateikti 3 lentelėje.

Vienoje komandoje galima rašyti ir keletą atributų, pavyzdžiui, komandos rezultatas bus toks:

Penkto dydžio mėlynas tekstas, parašytas „TimesLT“ šriftu

3 lentelė. Pagrindiniai komandos „font“ atributai

HTML komanda	Trumpas aprašymas	Pavyzdžiai
 tekstas 	Tekstas bus vaizduojamas tokiu šriftu, koks nurodytas, jei toks šriftas yra vartotojo kompiuteryje	MS LineDraw, Modern
 tekstas 	Tekstas vaizduojamas spalva, nurodyta šešiolyktainiu kodu (rrggbb). Spalvą galima nurodyti ir raktiniu žodžiu.	geltona, žalia, raudona
 tekstas 	Nurodo, kokio dydžio turi būti raidės. Leistini skaičiai nuo 1 iki 7. 1 atitinka 9 taškų dydį, o 7 – 36-ių. Dydis gali būti nurodomas ir reliatyviai: -3, -2, -1, +1, +2, +3 - nurodo kiek turi pasikeisti raidžių dydis, lyginant su naudojamomis.	trys, penki, septyni

Pastaba. Reikia nepamiršti, kad laukiamo efekto nebus, jeigu nurodyto šrifto nebus vartotojo kompiuteryje. Todėl patariama šia galimybe nepiktinaudžiauti.

Galima nurodyti ir visam dokumento tekstui skirtą šriftą, išskyrus ribojamą komanda ``. Tai daroma komanda `<basefont>` be pabaigos žymės, kurios atributai tokie pat, kaip ir komandos ``.

Keli fiziniai teksto formatai

Tai teksto formatavimo komandos, kurios nepriklauso nuo peržiūros programų (joms priklauso ir minėtos ``, `<i>` ir `<u>` komandos):

- `<big>` tekstas `</big>` – tekstas taps **truputį** didesnis už aplinkinį.
- `<small>` tekstas `</small>` – tekstas taps truputį mažesnis už aplinkinį.
- `<strike>` tekstas `</strike>` – tekstas bus ~~perbrauktas~~.
- `_{` tekstas `}` – tekstas bus ^{nuleistas ir sumažintas} už aplinkinį.
- `^{` tekstas `}` – tekstas bus ^{pakeltas ir sumažintas} už aplinkinį.
- `<blink>` tekstas `</blink>` – tekstas turėtų „mirkseti“, priešingu atveju naršyklė „nesupranta“ šios komandos.

Teksto grupavimo komandos

Tai komandos, skirtos prasminių teksto dalių išskyrimui į grupes. Šios komandos galioja visam tekstui nuo komandos pradžios iki jos pabaigos žymės:

- `
` perkelia tekstą į naują eilutę.
Skirtingai nei komanda `<p>` ji nepalieka vertikalaus tarpo tarp teksto iki komandos ir po komandos.
- `<nobr>` ... `</nobr>`
Visas tekstas, esantis tarp šios komandos pradžios ir pabaigos žymės privalo būti vienoje eilutėje ir negali būti perskeltas į kelias. Analogišką rezultatą galima pasiekti naudojant specialų simbolį `' '`, reiškiantį vietą, kurioje negalimas perkėlimas ar eilutės dalijimas, bet yra tarpas.
- `<wbr>` priešinga komandai `<nobr>`.
Ji „sufleruoja“ naršyklei, kuriuo žodžiu ji gali perkelti tekstą į kitą eilutę, jei to prireiktų. Naudojama tekste, esančiame tarp

komandos `<nobr>` ir jos pabaigos žymės, norint fiksuoti leistiną perkėlimo vietą.

- `<center> ... </center>`
Tekstas, esantis tarp komandos ir jos pabaigos žymės, bus centruojamas.
- `<div> ... </div>`
Komanda, skirta dalies teksto orientacijai keisti, t. y. lygiuoti jį pagal kairįjį/dešinįjį kraštą arba centruoti. Tam yra naudojamas atributas `align="left|center|right"`. Ši atributą turi ir kitos teksto grupavimo komandos: `<p>`, `<h1>`, ...`<h6>`.
Pavyzdžiui: `<div align=right> Šis tekstas lygiuojamas pagal dešinįjį kraštą </div>`.
- `<blockquote> ... </blockquote>`
Komanda skirta citatoms išskirti, tekstas truputį atitraukiamas nuo puslapio kraštų ir paprastai pradedamas ir baigiamas nauja pastraipa.
- `<address> ... </address>`
Adresui, autoriui ar kitai tokio tipo informacijai rašyti skirta komanda. Visa informacija pradedama iš naujos eilutės ir baigiama taip pat nauja eilute. Tekstas viduje rašomas kursyvu.
- `<comment> ... </comment>` arba `<!-- ... -->`
Skirta teksto „paslėpimui“ – tekstas tarp komandos ir jos pabaigos žymės nebus matomas naršyklės vartotojui. Naudojama norint paslėpti komentarus ar nereikalingas šiuo momentu, bet gal dar praversiančias dokumento dalis.
- `<hr>` komanda galima linija atskirti vieną teksto dalį nuo kitos.

Naudojant atributus galimi patys įvairiausi komandos `<hr>` variantai:

- `<hr size=skaičius>` – nurodo, kokio pločio turi būti linija.
Pavyzdžiui:



- `<hr width=skaičius|procentai>` – nurodo kokio ilgio bus linija: absoliučiu dydžiu, t. y. kiek taškų arba santykinu dydžiu – kiek procentų puslapio užims.
Pavyzdžiai (10, 50 ir 100 %):



- `<hr align=left|right|center>` – nurodo koks bus lygiavimas: prie kairiojo krašto, centruota linija ar prie dešiniojo krašto. Tai naudinga tik tada, kai linija nėra viso puslapio pločio.
Pavyzdžiui:



- `<hr noshade>` – tai stora „neįspausta“ linija be šešėlių:
- `<hr color=vardas#rrggbb>` – galima nurodyti kokios spalvos bus linija naudojant spalvos vardą arba šešioliktainį kodą.
Pavyzdžiui:



3.4.4. Nuorodos

Nuorodoms (angl. *anchor*) formuoti naudojama komanda `<a>` su pabaigos žyme ``. Pavyzdžiui:

```
<a href ="2psl.html">Mano antrasis puslapis</a>
```

Čia „2psl.html“ – dokumento adresas (matomas tik naršyklei), o „Mano antrasis puslapis“ – nuorodos tekstas dokumente (matomas tik vartotojui).

Paprastai nuorodos (angl. *hyperlinks*) išskiriamos iš teksto spalva ir pabraukimu, todėl yra nesunkiai atpažįstamos.

Šiame pavyzdyje nuoroda yra į dokumentą, kuris yra tame pačiame kompiuteryje ir tame pačiame kataloge, kaip dokumentas, kuriame ši nuoroda rašoma. Tačiau taip būna ne visada. Dažnai dokumentas būna kitame kataloge ar net kompiuteryje, todėl naudojami keli nuorodų tipai:

- *Reliatyvi nuoroda* – nuoroda į dokumentą, esantį kitame kataloge, pradedant katalogu, kuriame yra dokumentas.

Pavyzdžiui:

```
<a href = "../Puslapiai/2psl.html">Mano antrasis  
puslapis</a>.
```

- *Absoliuti nuoroda* – nuoroda į dokumentą, esantį kitame kataloge, pradedant šakniniu katalogu.

Pavyzdžiui:

```
<a href ="/vartotojai/Med/Html/Puslapiai/2psl.htm  
1"> Mano antrasis puslapis</a>.
```

- *Bendroji nuoroda* – nuoroda į dokumentą, dažniausiai (bet ne būtinai) esantį kitame kompiuteryje, naudojant universalų informacijos šaltinio adresą (URL – angl. *Uniform Recource Locator*).

Pavyzdžiui:

```
<a href="http://www.dar.lt/Med/Puslapiai/2psl.htm  
1">Mano antrasis puslapis</a>.
```

Universalaus šaltinio adresas (URL) susideda iš trijų dalių: serverio tipo, serverio, dokumento ir dokumento vietos nurodymo. Pirmoji dalis naudojama todėl, kad šiuolaikinės peržiūros programos dažnai leidžia naudotis ne tik žiniatinklio, bet ir kitų serverių resursais.

Pagrindiniai tipai:

- `` – nuoroda į HTML dokumentą, kuris yra žiniatinklio serveryje.
- `` – nuoroda į FTP serverį; dažnai naudojama kartu su bylos nurodymu. Paprastai naršyklė naudoja anoniminį FTP serverį.
- `` – nuoroda į „Gopher“ serverį.

- `` – naudojama laiškui siųsti. Naršyklė kreipiasi į ją aptarnaujančią pašto programą ir nurodytu adresu yra siunčiamas laiškas.
- `` – nuoroda į naujienų grupę (ang. *newsgroup*). Naudoti rizikinga, nes neaišku, ar turi galimybę konkretus vartotojas ir jo naudojamas naujienų serveris prisijungti prie nurodytos grupės.
- `` – nuoroda į „Telnet“ serverį. Paprastai būna aktyvuojama programa, kuri naudojama „Telnet“ seansams.
- `` – nuoroda į WAIS (ang. *Wide Area Information Servers*) indeksuotų duomenų bazių serverį.

Nuorodos galimos ne tik tarp dokumentų, bet ir vieno dokumento ribose. Tam naudojamas papildomas atributas `name="vardas"`. Toje dokumento vietoje, į kurią bus kreipiamasi, reikia įterpti nuorodą su šiuo atributu, tarp komandos pradžios ir pabaigos žymių galimas, bet nebūtinai tekstas.

Taigi, rašoma komanda:

```
<a name="vardas"> galimas tekstas arba jo nėra  
</a>.
```

Toje dokumento vietoje, iš kurios kreipiamasi, rašoma nuoroda į reikiamą vietą tokiu būdu: dokumento pavadinimas nerašomas, o vietos vardas nurodomas toks, koks yra apibrėžtas atributu `name` tik su ženklų '#' priekyje:

```
<a href="#vardas">nuoroda į kitą to paties  
dokumento vietą</a>.
```

Žinoma, tokį konkrečios kažkokio HTML dokumento kreipimosi vietos nurodymą galima naudoti ir kreipiantis iš kito dokumento, tereikia po dokumento vardo parašyti ženklą '#' ir vietos pavadinimą.

Pavyzdžiui:

```
<a href="basic.html#A">Pirmasis komandos <A>  
aprašymas</a>.
```

3.4.5. Sąrašai ir apibrėžimai HTML dokumente

** ... nenumeruojamas sąrašas**

Bendras šios komandos formatas yra:

```
<ul>
  <li> tekstas
  <li> tekstas
  ...
</ul>
```

(5)

Sąrašo pabaigos žymę rašyti būtina. Sąrašo elementams atskirti naudojama komanda , kuria prasideda kiekvienas naujas sąrašo elementas. Komandoje arba naudojant atributą type="disc|square|circle" galima pakeisti sąrašo elemento simbolį. Jei atributas naudojamas komandoje , tai jis galioja visiems sąrašo elementams, jei su komanda , tai jis galioja tik elementams, esantiems po tos komandos. Kitas sąrašas taip pat gali būti sąrašo elementas. Jei atributas „type“ nenurodomas, tai elemento simbolis priklauso nuo sąrašo gylio. Elemento simboliai gali būti trijų tipų:

- – skritulio tipas. Komanda <li type=disc>.
- – kvadrato tipas. Komanda <li type=square>.
- – apskritimo tipas. Komanda <li type=circle>.

Sąrašo elementai yra truputį atitraukti nuo kairiojo puslapio krašto ir juose galima naudoti kitas HTML komandas.

** ... numeruojamas sąrašas**

Komandos formatas analogiškas komandai :

```
<ol>
  <li> 1 numerio tekstas
  <li> 2 numerio tekstas
  ...
</ol>
```

(6)

Komanda turi papildomą atributą start="numeris", kuriuo nurodomas numeracijos pradinis numeris. Komandoje tai pat galima naudoti atributą

value="numeris", skirtą numeracijos pakeitimui, pradedant šiuo elementu. Tada naujoji numeracija galios visiems elementams, esantiems po komandos.

Komandos atributu type="kodas" galima nurodyti, kokie simboliai bus naudojami numeracijai. Galimi keli numeracijos tipai pateikti 4 lentelėje.

4 lentelė. Numeracijos tipai

Kodas	Stilius	
1	arabiški numeriai	1, 2, 3, ...
a	mažosios raidės	a, b, c, ...
A	didžiosios raidės	A, B, C, ...
i	maži romėniški skaičiai	i, ii, iii, ...
I	dideli romėniški skaičiai	I, II, III, ...

Kaip ir nenumerojamoose sąrašuose, numerojamų sąrašų elementai gali būti kiti sąrašai.

Pavyzdžiui:

- Sąrašai gali būti nenumerojami. Pvz.:
 - Apskritimo tipo numerojamas sąrašas.
 - Apskritimo tipo numerojamas sąrašas.
 - Kvadrato tipo numerojamas sąrašas.
 - Kvadrato tipo numerojamas sąrašas.
- O taip pat gali būti numerojami:
 - x. Romėniško stiliaus (maži skaičiai) sąrašas, pradėtas ketvirtu numeriu. Komanda <ol start=4 type="i">.
 - Pakeičiame numeraciją: pradedame numeruoti nuo dešimties – <li value=10>
 - Ir taip toliau.

<dl> ... </dl> apibrėžimų sąrašas

Apibrėžimų sąrašas skirtas sąvokoms kartu su jų apibrėžimais vardinti. Apibrėžimų sąrašą sudaro elementai, patys susidedantys iš dviejų dalių: sąvokos pavadinimo ir sąvokos apibrėžimo. Pilnas <dl> (angl. *Definition List*) komandos formatas yra toks:

```
<dl>
<dt> (Definition term) Apibrėžiamos sąvokos
pavadinimas.
<dd> (Term Definition) Sąvokos apibrėžimas.
... (toliau kartojamos "poros" <dt> ir <dd>)
</dl>
```

(7)

Po komandos `<dt>` galima rašyti tik tekstą, be grupavimo komandų, o po komandos `<dd>` – bet koki tekstą su bet kokiomis komandomis teksto viduje. Kiekviena sąvoka pradedama iš naujos eilutės, o jos apibrėžimas sudaro atskirą, kiek atitrauktą nuo vertikalių puslapio kraštų tekstą, taipogi prasidedantį iš naujos eilutės.

Pavyzdžiui:

Informatika

Mokslas apie informaciją, jos perdavimą, kaupimą, saugojimą, apdorojimą.

















Informacija

Tai žinios, perduodamos vienu asmenų kitiems žodžiu arba žiniasklaidos priemonėmis: per spaudą, radiją, televiziją, kiną, kompiuterių tinklus.

3.4.6. Spalvų kodai ir vardai

HTML kalboje naudojamas trijų bazinių spalvų – raudonos, žalios ir mėlynos (angl. RGB – *Red-Green-Blue*) kodas.

5 lentelė. Spalvų vardai ir RGB kodai

	black = "#000000"		green = "#008000"
	silver = "#c0c0c0"		lime = "#00ff00"
	gray = "#808080"		olive = "#808000"
	white = "#ffffff"		yellow = "#ffff00"
	maroon = "#800000"		navy = "#000080"
	red = "#ff0000"		blue = "#0000ff"
	purple = "#800080"		teal = "#008080"
	fuchsia = "#ff00ff"		aqua = "#00ffff"

Kodas yra šešių pozicijų šešiolyktainis skaičius, kurio pirmi du skaitmenys atitinka raudonos, antri – žalios, treči – mėlynos spalvos intensyvumą. Kodas "#000000" atitinka juodą spalvą, o kodas "#ffffff" – baltą. Visų kitų spalvų kodai yra tarp šių dviejų kodų. Dauguma spalvų turi ne tik kodus, bet ir savo pavadinimus.

3.4.7. Papildomi simboliai HTML dokumente

HTML dokumente kartais naudojami simboliai, kurių nėra standartinėje klaviatūroje (tokie kaip ©, ® ir pan.), taip pat simboliai, kurie gali būti interpretuojami kaip HTML komandų elementai (pavyzdžiui < ar >), nacionaliniai kalbų simboliai ir pan.

Tokiu atveju yra naudojami vadinamieji specialūs simboliai (angl. *Special characters*) arba papildomi simboliai (angl. *Extended characters*). Tai keleto raidžių kodai, būtinai prasidedantys ženklų '&' ir besibaigiantys ženklų ';'.

Simboliai, jų kodai ir aprašymai pateikti 2 priede.

3.4.8. Lentelės HTML dokumente

Lentelėms naudojama komanda `<table>`, kuri būtinai turi būti užbaigta pabaigos žyme `</table>`. Lentelę sudaro eilutės ir langeliai. Tiek eilutėms, tiek langeliams formuoti yra atskiros komandos, eilutėms – komanda `<tr>` (angl. *Table Row*), su privaloma pabaigos žyme `</tr>`, o langeliams – `<td>` (angl. *Table Data*) ir atitinkamai `</td>`. Komanda `<caption>` su pabaigos žyme `</caption>` yra skirta lentelės pavadinimui, o komanda `<th>` (angl. *Table Header*) su pabaigos žyme `</th>` skirta atskirų stulpelių pavadinimams. Taigi, bendra `<table>` komandos forma yra:

```

<table>
<caption> Pavadinimas </caption>
<tr> <!-- Stulpelių pavadinimai -->
<th> Pirmas stulpelis </th>
<th> Antras stulpelis </th>
</tr>
<tr> <!-- Pirmą eilutę -->
<td> Duomenys </td> <td> Duomenys </td>
</tr>
...
</table>

```

(8)

Gaunamas rezultatas:

Pavadinimas	
Pirmas stulpelis	Antras stulpelis
Duomenys	Duomenys

Komandos `<table>` atributai

- `align="left|center|right"`
Šis atributas leidžia „orientuoti“ lentelę puslapyje į kairę, centre, į dešinę.
- `width="taškai|procentai"`
Nurodo, kokio pločio turi būti lentelė. Galimi du nurodymo būdai: absoliutus ir reliatyvus. Pirmu atveju nurodoma, kiek tiksliai taškų

turi užimti lentelė (width=250), antru – kiek procentų puslapio pločio turi užimti lentelė (width="50%" – pusė puslapio).

- border="taškai"
Nurodo, kokio pločio (kiek taškų) turi būti lentelę ribojančios kraštinės. Jei atributas nurodomas be konkrečių skaičių ar nenurodomas, tai lentelės kraštinės bus tokio pločio, kokie yra naršyklėje pagal nutylėjimą (dažniausiai 1 taškas), o jei bus nurodyta border=0, tai kraštinės iš viso nebus.
- cellspacing="taškai"
Kiekvienas lentelės langelis HTML dokumente gali būti atskirtas nuo kitų langelių, o šis atributas ir nustato, kokio pločio bus skiriamoji langelių erdvė.
- cellpadding="taškai"
Nurodo, koks turi būti atstumas tarp informacijos langelio viduje ir langelio kraštinės.
- bgcolor="spalvos_kodas"
Šiuo atributu galima pasirinkti ir lentelės fono spalvą.
- background="nuoroda į grafinę bylą"
Lentelės fonui galima pasirinkti grafinį vaizdą.

Sudarius tokią lentelę:

```
<table cellspacing=10 border=4 cellpadding=5
align="right" width="70%" bgcolor="#c0c0c0">
<tr> <!-- Stulpelių pavadinimai -->
<th> Pirmas stulpelis </th> <th> Antras
stulpelis </th>
</tr>
<tr> <!-- Pirmą eilutę -->
<td> Duomenys </td> <td> Duomenys </td>
</tr>
...
</table>
```

(9)

Gaunamas rezultatas:

Pirmas stulpelis	Antras stulpelis
Duomenys	Duomenys

`<tr> ... </tr>`

Paskirtis – suskirstyti lentelę į eilutes. Kiekviena lentelės eilutė HTML dokumente turi prasidėti komanda `<tr>` ir baigtis jos pabaigos žyme `</tr>`.

`<tr>` atributai:

- `align="left|center|right"`
Nurodo, kokia bus horizontalioji eilutės langelių turinio orientacija – link kairiojo/dešinio krašto ar langelio centre.
- `valign="top|middle|bottom|baseline"`
Nurodo, kokia bus vertikaliąji šios eilutės langelių turinio orientacija – informacija bus viršuje, centre, apačioje ar visų eilutės langelių turinys „lygiuos“ savo apačia vienu lygiu.
- `bgcolor="spalvos_kodas"`
Nurodo, kokia bus šios eilutės langelių fono spalva.

Sudarius tokią lentelę:

```

<table align="right" border=1>
<tr bgcolor="#c0c0c0">
<th> Pirmas stulpelis </th> <th> Antras
stulpelis </th>
</tr>
<tr bgcolor="#ffffff" align="center">
<td> Duomenys </td> <td> Duomenys </td>
<tr valign="middle"><td> Duomenys <br> Antra
eilutė</td> <td> Duomenys </td>
</tr>
...
</table>

```

(10)

Gaunamas rezultatas:

Pirmas stulpelis	Antras stulpelis
Duomenys	Duomenys
Duomenys Antra eilutė	Duomenys

`<th> ... </th> ir <td> ... </td>`

Komandos `<th>` ir `<td>` yra labai panašios, tiksliau komanda `<th>` yra atskiras komandos `<td>` atvejis, kai tekstas langelyje yra paryškintas (angl. *Bold*) ir jo turinys pagal nutylėjimą centruotas tiek horizontaliai, tiek vertikalčiai.

Komandos `<th>` ir `<td>` galimybės:

- `align="left|center|right"`
Analogiškas aukščiau minėtiems kitų komandų atributams, bet galioja vieno langelio ribose.
- `valign="top|middle|bottom|baseline"`
Analogiškas komandos `<tr>` atributui, bet galioja vieno langelio ribose.
- `width="taškai|procentai"`
Nurodo langelio plotį taškais arba procentais, lyginant su visos lentelės pločiu.
Pastaba. Pakeitus vieno langelio plotį, keisis atitinkamai viso stulpelio langelių plotis.
- `height="taškai|procentai"`
Nurodo langelio aukštį. Vieno langelio aukščio koregavimas lems ir visos eilutės aukštį.
- „nowrap“
Reikalavimas neskaidyti teksto langeliuose į dalis taip, kad lentelė tilptų į peržiūros langą. Analogišką rezultatą galima pasiekti visą langelio turinį „apskliaudžiant“ komanda `<nobr>`.
- `colspan="skaičius"`
Šis atributas naudojamas nurodyti, kiek stulpelių turi aprėpti konkretus langelis.

- `rowspan="skaičius"`
Analogiška „`colspan`“, bet eilučių atžvilgiu nurodo kiek eilučių ap-
rėps konkretus langelis.
- `bgcolor="spalvos_kodas"`
Nurodo, kokia bus šio langelio fono spalva.

Sudarius tokią lentelę:

```

<table border=1 align="right" width="70%">
  <tr bgcolor="#c0c0c0"><th colspan=2> Pirmas
  stulpelis </th>
  <th rowspan=2> Antras stulpelis </th></tr>
  <tr bgcolor="#c0c0c0"><th bgcolor="#808080">
  Pirma stulpelio dalis </th>
  <th> Antra stulpelio dalis</th></tr>
  <tr bgcolor="#ffffff"
  align="center"><td>Daug daug daug daug
  duomenų </td>
  <td nowrap> Daug daug daug daug duomenų
  </td>
  <td valign="bottom">Kiti duomenys</td></tr>
  <tr valign="middle"> <td align="right"
  rowspan=2>Duomenys <br>Antra eilutė</td>
  <td rowspan=2> Duomenys </td><td>Pirma
  eilutė</td></tr>
  <tr><td align="right">Antra eilutė</td></tr>
</table>

```

(11)

Gaunamas rezultatas:

Pirmas stulpelis		Antras stulpelis
Pirma stulpelio dalis	Antra stulpelio dalis	
Daug daug daug duomenų	Daug daug daug duomenų	Kiti duomenys
Duomenys Antra eilutė	Duomenys	Pirma eilutė
		Antra eilutė

<caption> ... </caption>

Tai komanda, specialiai skirta lentelės pavadinimui. Pavadinimas gali būti lentelės viršuje (angl. *top*) arba apačioje (angl. *bottom*). Komandos bendra forma yra:

`<caption align="top|bottom"> pavadinimas </caption>`

Čia atributas „align“ turi įprastinę prasmę ir reikšmes.

6 lentelė. Lentelės formavimo schema

TABLE			
Align Width Border Cellspacing Cellpadding Bgcollor Background	CAPTION	TR	
	Align	Align Valign Bgcolor	TD arba TH
			Align Valign Width Height Nowarp Colspan Rowspan Bgcolor

Pastaba. Jei pavyzdys aprašyme neatitinka to kas parašyta, tai dar nereiškia, kad jis neteisingas, gal tiesiog naudojama naršyklė „nesupranta“ kai

kurių naudojamų komandų ar jų atributų. Be to, čia paminėtos tik dalis komandų ir atributų, o skirtingos naršyklės dažnai „supranta“ skirtingas komandas.

Grafiniai vaizdai HTML dokumente

Grafiniams vaizdams įterpti naudojama komanda `` (angl. *image*), kuri naudojama be pabaigos žymės. Grafinis vaizdas turi būti pateikiamas GIF, JPEG ar PNG formato bylose. Bendra komandos išraiška yra tokia:

```

```

Pagrindinis ir būtinas `` komandos atributas yra „src“ (angl. *source*). Tai nuoroda, kokia byla bus rodoma kaip grafinis vaizdas dokumente ir jos buvimo vieta. Pavyzdys:

```

```

Kiti grafinių vaizdų atributai:

- `alt="tekstas"`
Naudojama kaip alternatyva vaizdui, jei dėl kažkokių priežasčių neatvaizduojamas grafinis vaizdas (nerandama byla, tekstinė peržiūros programa ar uždraustas grafinių vaizdų naudojimas). Tada vietoje grafinio vaizdo bus rašomas nurodytas tekstas. Naudoti nebūtina, bet patartina.
- `align="left |right |top |texttop |middle |absmiddle |baseline |bottom |absbottom"`
Nurodo, kokia bus grafinio vaizdo padėtis kitų HTML dokumento objektų atžvilgiu:
 - `left |right` – grafinis vaizdas orientuojamas į kairį ar dešinę HTML dokumento kraštą, o tekstas (ar kiti objek-

tai), rašomi po komandos „img“ „apgaubia“ vaizdą iš dešinės ar kairės pusės. Taigi, vaizdas niekaip neišskiriamas ir „įjungiamas“ į bendrą vaizdą.

- *top* – grafinio vaizdo pradžios linija lygiuojama su prieš tai esančios teksto eilutės viršumi. Vienoje eilutėje su vaizdu tebus viena teksto eilutė. Kitos bus žemiau vaizdo pabaigos.
- *middle* – grafinio vaizdo vidurio linija lygiuojama su teksto bazine linija.
- *bottom* | *baseline* – grafinio vaizdo pabaigos linija lygiuojama su teksto bazine linija.
- *texttop* – veikimo apibrėžimas analogiškas „top“, tačiau ne visada veikia taip pat.
- *absmiddle* | *absbottom* – analogiškas veikimas kaip ir „middle“ ar „bottom“, tik lygiuojama su atitinkamai teksto vidurio ir pabaigos linijomis.

- `border="taškai"`

Nurodo, kokio storio bus rėmelis aplink grafinį vaizdą. Dažnai jis daromas lygus nuliui, nes tada vaizdas neišsiskiria iš HTML dokumento. Kitos reikšmės dažnai naudojamos su komanda `<a>`, kurioje vietoje ar šalia teksto naudojamas grafinis vaizdas.

Pavyzdžiai su nuorodomis:

```
<a href="images/pav.gif"> </a>.
```

```
<a href="images/pav.gif"> </a>.
```

- `width="taškai" height="taškai"`

Nurodo, kokio dydžio (taškais) bus atvaizduotas grafinis vaizdas HTML dokumente. Jei nenurodyta – sutampa su realiu vaizdo dydžiu. Jei abu atributai nurodyti, tai peržiūros programa iš anksto žino, kiek vietos užims vaizdas ir pagal tai gali iš karto teisingai išdėstyti tekstą aplink vaizdą. Priešingu atveju teksto išsidėstymas

bus koreguojamas po pilno grafinės bylos persiuntimo į vartotojo kompiuterį.

- `vspace="taškai" hspace="taškai"`
Nurodo, kiek tuščios vietos reikia palikti aplink grafinį vaizdą vertikalia ar (ir) horizontalia kryptimi.

Pavyzdžiai

```

Šis paveikslukas yra teksto <BR>
kairėje pusėje
```

(12)

```
<img src =" images/pav01.gif" align ="right">
Šis paveikslukas yra teksto <BR>
dešinėje pusėje
```

(13)

```
<img src =" images/pav01.gif" align ="center"
width=220 height=40>
Pakeistas paveiksluko dydis.
```

(14)

3.4.9. Grafiniai žemėlapiai

Viename grafiniame vaizde galima apjungti nuorodas į skirtingus HTML dokumentus ar jų dalis. Tai realizuojama išskaidant grafinį vaizdą į sritis ir toms sritims priskiriant skirtingas nuorodas. Išskaidymui naudojama schema, sukuriama komanda `<map>`, kuri būtinai turi būti baigiama jos pabaigos žyme `</map>`. Komanda turi vieną atributą – schemos vardą „name“. Tarp komandos pradžios ir pabaigos žymių schemai sudaryti naudojamas komandų `<area>` rinkinys: kiekviena tokia komanda nurodo vieną grafinio vaizdo sritį ir susieja ją su viena nuoroda.

Bendra komandos išvaizda:

```
<map name="žemėlapio pavadinimas">
```

(15)

```

<area href="nuoroda" [alt="vardas"] sha-
pe=srities_tipas coords="koordinatės">
...
<area href="nuoroda" [alt="vardas"] sha-
pe=srities_tipas coords="koordinatės">
</map>

```

Sudarant žemėlapi, grafinį vaizdą galima skaidyti į stačiakampio, skritulio arba poligono formas. Visas grafinis vaizdas interpretuojamas kaip koordinatė plokštuma, su koordinatė pradžia viršutiniame kairiajame vaizdo kampe, „x“ koordinatė yra horizontali, o „y“ – vertikali. Vaizdas matuojamas taškais arba procentais. Suskirstymui naudojami komandos „area“ atributai „shape“ ir „coords“.

Galimos reikšmės:

- shape=default – visas grafinis vaizdas bus viena sritis. Koordinatė nurodinėti nereikia.
- shape=rect coords="kairysis_x, viršutinis_y, dešinysis_x, apatinis_y" – iš vaizdo išskiriama stačiakampė sritis (angl. rect). Reikia nurodyti viršutinio kairiojo ir apatinio dešiniojo kampų koordinatas. Galima nurodinėti ir taškais, ir procentais: SHAPE=RECT COORDS="0, 0, 50%, 100%" – nurodoma kairė vaizdo pusė (50 % pločio ir 100 % ilgio).
- shape=circle coords="centro_x, centro_y, spindulys" – iš vaizdo išskiriama skritulio sritis (angl. circle). Reikia nurodyti centro koordinatas ir spindulio ilgį.
- shape=poly coords="pirmas_x, pirmas_y, antras_x, antras_y, ..." – išskiriama nestandartinė sritis, nurodomos ją ribojančios uždaro laužtinės viršūnių taškų koordinatės.

Kiekvienai tokiu būdu nurodytai sričiai yra priskiriama arba nepriskiriama nuoroda į dokumentą ar jo dalį. Nuorodos sintaksė visiškai tokia pati kaip ir komandos <a>. Jeigu sritis išskiriama kaip neturinti nuorodos („iš vaizdo iškerpama skylė“, kuri bus tiesiog grafinis vaizdas ir „neves“ niekur toliau), tam naudojamas atributas „nohref“ (vietoje „href“).

„Area“ komandos atributas „alt“ naudojamas informacijos apie pasirinktą vaizdo sritį išvedimui būsenos juostoje, lange arba po pelės ženklų. Taip pat gali būti naudojamas kaip alternatyvi nuoroda, peržiūros programose be grafikos arba klientui naudojant peržiūros režimą be grafikos.

Grafinio vaizdo suskaidymas – pagrindinis žingsnis žemėlapių sudaryme. Tačiau reikalingas ir grafinis vaizdas, kuriam ta schema bus taikoma ir kuriam kuriamas žemėlapis. Taigi, schema turi būti užrašoma į HTML dokumentą ir susiejama su grafiniu vaizdu naudojant „img“ komandoje atributą usemap="schemos_vieta". "schemos_vieta" yra adresas į reikalingą komandą „map“, susidedantis iš nuorodos į HTML dokumentą, kuriame ji randasi ir jos vardo, atskirto ženklu '#' (kaip ir nuoroda į HTML dokumento dalį).

Komanda, realizuojanti žemėlapi:

```

```

Atskiros žemėlapio dalys štrichuotos (tam kad lengviau matytųsi išskiriamos sritys, realiai jokių regimų linijų ar kitokių pasikeitimų grafiniame vaizde, kuris naudojamas kaip žemėlapis, nesimato). Likusioje vaizdo dalyje nuorodų nėra.

3.5. Dinaminis HTML

Dinaminis HTML (DHTML) yra technologija, apjungianti savyje HTML, stilių sąrašus ir skriptus bei leidžianti žiniatinklio puslapiams keistis ir realizuoti animaciją. DHTML remiasi dokumento objektiniu modeliu (DOM), kuris leidžia valdyti CSS taisykles ir HTML elementų savybes. DOM aprašo tinklalapio turinį, kaip objektų rinkinį. Kreiptis į objektus ir jais manipuluoti galima įvairiais būdais, naudojant „JAVA script“, „Jscript“, ar „VB script“ scenarijus, vykdomus tiek kliento, tiek serverinėje dalyje. Objektai tarpusavyje yra susiję, turi tam tikrą hierarchiją, t. y. objektas gali turėti objektą – „tėvą“ (kuriam ir priklauso) arba objektą „sūnų“ (kuris priklauso).

DHTML privalumai:

- DHTML leidžia elgtis su visu žiniatinklio puslapiu, kaip su programuojamu objektu. Tai sukuria galimybę tinklalapiams efektyviau reaguoti į įvedamus duomenis ir komandas, pateikiamas vartotojo.
- Papildomos multimedijos panaudojimo, maketų kūrimo galimybės: elementų spalvų ir matmenų keitimas.
- Serverio apkrovimo sumažinimas. Dažniausiai DHTML scenarijai paimami iš serverio ir perkeliama į vartotojo kompiuterį, kur ir vyksta scenarijaus paleidimas.
- Interaktyvumo lygio padidėjimas.
- Duomenų bazių palaikymas.
- Duomenų išvedimas (patalpinimas). Įvedus duomenis, juos galima atvaizduoti tame pačiame puslapyje be kreipimosi į serverį.
- Sąrašų stilių pagerinimas.
- Dinaminiai meniu, palengvinantys informacijos pasirinkimą.
- Kelių platformų palaikymas („Windows“, „Macintosh“, UNIX ir kt.).

Nors dinaminiai HTML suteikia gerokai daugiau galimybių, tačiau yra vienas labai didelis minusas. Nėra visiškai bendro standarto. Ką supranta viena naršyklė, visiškai nesupranta kita naršyklė, todėl tam pačiam veiksmui skirtingoms naršyklėms rašomi skirtingi scenarijai. Dar vienas minusas, jog neleidžia įrašyti į failą (išskyrus „cookies“ atveju).

3.6. Dažnai pasitaikančios HTML klaidos

HTML dokumentą sudaro dokumento tipas, antraštė ir turinys, o dokumento struktūros pavyzdys pateiktas 16 kode:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01  
Transitional//EN">  
<html>  
<head>
```

(16)

```

<title>Dokumento pavadinimas</title>
<meta http-equiv="Content-Type"
  content="text/html; charset=windows-1257">
</head>
<body>
  <!-- Dokumento turinys -->
</body>
</html>

```

Pirmoji žymė yra HTML dokumento tipas, kuris nurodo, kokias dokumento apdorojimo taisykles turėtų naudoti naršyklė. Viena dažniausiai pasitaikančių klaidų – nenurodytas dokumento tipas. Jei HTML dokumente jo trūksta – jį reikia įrašyti. Jei naudojate XHTML dokumento tipą, šis sakinyss atrodys taip: `<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN">`. Galiojančių dokumentų tipų sąrašą galima rasti adresu <http://www.w3.org/QA/2002/04/valid-dtd-list.html>. Kiekvienas HTML dokumentas prasideda žyme `<html>` ir baigiasi žyme `</html>`.

Antraštėje tarp žymių `<head></head>` nurodomas dokumento pavadinimas, koduotė, CSS stilius, aprašymas, raktiniai žodžiai paieškos sistemoms, „JavaScript“ scenarijai.

HTML specifikacija reikalauja nurodyti dokumento pavadinimą. Jis turi tiksliai nusakyti turinį, bet turi būti ne per ilgas. Dokumento pavadinimas rašomas taip: `<title> Dokumento pavadinimas </title>`. Šis pavadinimas bus matomas naršyklės lango antraštėje.

Antra labai dažnai pasitaikanti klaida – nenurodyta dokumento koduotė. Koduotę nurodyti būtina! Koduotė parodo, kokios kalbos simbolius naudosite ir koku būdu jie užkoduoti. Lietuvių kalbai naudojama „Windows-1257“ arba „ISO-8859-13“ koduotė. Norint viename dokumente naudoti kelių kalbų simbolius, naudojama universalioji „UTF-8“ koduotė.

Tam, kad tinklo naršyklė automatiškai persijungtų į lietuviško teksto vaizdavimui skirtą „Baltic“ kodų lentelę, HTML failo pradžioje po komandos `<HEAD>` reikia įterpti tokią eilutę:

```

<META Http-Equiv="Content-Type"
  Content="text/html; charset=windows-1257">

```

Tarp `<body>`/`</body>` žymių rašomas dokumento turinys.

Kitos klaidos:

1. `` žymėje praleistas `alt=""` atributas. Šiuolaikinio dizaino puslapiuose dažnai sunku apsieiti be paveikslėlių, tačiau kai kurios naršyklės yra tekstinės, todėl jų atvaizduoti negali. Dėl šios priežasties W3C standartai reikalauja `alt=""` atributo, kuriame įrašomas tekstas, rodomas vietoje paveikslėlio. Jei paveikslėlis nesvarbus, `alt=""` atributą galima palikti tuščią.
2. Lentelėse naudojamas atributas `height=""`, nors W3C standartuose toks lentelių atributas neaprašytas. Kaip alternatyvą galima naudoti CSS `height: ;` atributą.
3. HTML kalboje „&“ yra specialusis simbolis, naudojamas kitiems simboliams užrašyti naudojant jų simbolinius pavadinimus arba kodus. Pavyzdžiui, vienguba kabutė gali būti užrašyta kaip `"`, o raidė „A“ gali būti užrašoma kaip `A`. Labai dažnai pasitaikanti programuotojų klaida, kai PHP ar kituose scenarijuose generuojamos nuorodos, kuriose naudojamas nepakeistas „&“ simbolis. Jis turėtų būti keičiamas simbolių seka `&`.
4. Naudojant XHTML neretai pamirštama uždaryti pavienes žymes; teisinga paveiksluko žymė turėtų atrodyti taip: ``.
5. „Macromedia Flash“ sugeneruotas HTML kodas neatitinka W3C standartų, todėl naudojant „Flash“ savo tinklalapyje galima gauti daugybę klaidų. Siūloma atsakyti „Flash“ arba jį naudoti tik sugebant sutvarkyti „Flash“ sugeneruotą kodą.

3.7. HTML5 taikymas

Žymiai sumažėjo `<div>` gairių. Vietoj jų HTML5 siūlo specifines tik jam būdingas gaires, tokias kaip `<header>`, `<article>`, `<footer>`, `<title>` ir kitas. Nors iškart sunku pastebėti kuo tai skiriasi, tačiau rašant kodą, tokius elementus

naudoti yra kur kas paprasčiau, nes nereikia kreipti dėmesio ar tai „id“ elementai, kurie ankstesnėse versijose apibūdindavo specifinius elementus, kurie būdavo naudojami vieną kartą, ar tai „class“ elementai, kurie įprastai naudojami apibūdinti tam tikrų duomenų grupėms, kurioms reikia taikyti tuos pačius parametrus. Dauguma HTML5 gairių galima priskirti „class“ grupei, nes jų taikymo kiekis tam tikroms puslapio dalims nėra ribojamas. Pavyzdžiui `<div id=header>` HTML4 ir ankstesnėse versijose apibūdindavo tinklalapio antraštę, kaip vienintelį elementą, o `<header>` HTML5 galima naudoti kaip gairę skirtą tiek puslapio, tiek ir straipsnių antraštėms.

Pirmiausia HTML5 versijoje galima pastebėti supaprastintą „doctype“. Senesniame HTML dokumente paprastai reikėdavo nurodyti, kokį standartą naudosime tokiu būdu:

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">

```

(17)

Šiuo atveju nurodomas XHTML „Transitional“ 1.0 standartas, tačiau toks standarto apibrėžimas nėra itin patrauklus. HTML5 standartą apibrėžti yra kur kas paprasčiau ir patogiau.

Dabar pakanka nurodyti tik `<!doctype html>` ir naršyklė atpažins, kad naudojamas HTML5.

XHTML elementas IMG apibrėžtas tokia sintakse:

```



```

(18)

Pradėjus naudoti HTML5 nebereikia rūpintis žymės uždarymu (kai žymė yra „vienetinė“: br, hr):

```


<br>
<hr>

```

(19)

Siekiant suderinamumo su ankstesniais standartais, galima naudoti sintaksę kaip ir anksčiau (XHTML), tačiau patartina iš karto išmokti tinkamą sintaksę ir prie jos įprasti.

Pagrindiniai HTML5 nauji elementai:

- naujos struktūrinės žymės: `<section>`, `<header>`, `<footer>`, `<nav>`, `<article>`, `<aside>`
- vaizdo failai su `<video>`;
- garso failai su `<audio>`;
- `<canvas>`.
- nauji formos elementų tipai ir atributai.

3.7.1. Naujos struktūrinės žymės

Iki šiol `<div>` elementas buvo naudojamas dizaino struktūrai. Kadangi beveik kiekvienas tinklalapis turi viršutinę dalį (angl. *header*), apatinę dalį (angl. *footer*), atskiras turinio sekcijas arba blokus (angl. *sections*) bei blokus kažkokiems įrašams, todėl yra naudinga šias sritis apibrėžti kaip atskirus elementus.

`<section>`

Šis elementas apibrėžia turinio sritį, kuri gali talpinti bet kokius elementus. Joje paprastai gali būti `<header>` ir `<footer>` elementai. Taip pat šioje srityje gali būti kitos sritys, taip sukuriant gilesnę ir konkretesnę struktūrą. Verta paminėti, jog `<section>` elementas nėra skirtas kaip stilizavimo objektas. Kitaip sakant, jeigu žadate naudoti `<section>` elementą „apjuosti“ turiniui, kad jį būtų galima stilizuoti, tai darysite neteisingai. Tam yra skirtas mūsų gerai žinomas `<div>` elementas.

```
<section>  
  <h1>Antraštė</h1>  
  Turinys  
</section>
```

(20)

`<section>` turi antraštę (angl. *heading*) ir ji yra elemento viduje, o ne išorėje prieš pat jį. Todėl neįdėjus antraštės, sekcija yra neįvardinta (angl. *untitled section*).

<header>

Dauguma turinio sričių turi kokią nors įvadinę dalį – tai paprastai yra antraštė, turinys (angl. *table of contents*) ar net paieškos forma:

```
<header>
  <h1>Straipsniai</h1>
</header>
```

(21)

Svarbu atkreipti dėmesį, jog dabar yra dvi labai panašios žymės: <head> ir <header>, tačiau tai du atskiri objektai.

```
<article>
  <header>
    <h1>Straipsnio ar įrašo pavadinimas</h1>
  </header>
  <p>Straipsnio turinys</p>
</article>
```

(22)

<footer>

Šis elementas naudojamas baigiamajai informacijai talpinti: informacija apie įrašo autorių, susijusios nuorodos ir pan. Paprastai <footer> elementas naudojamas <article> viduje bei kaip atskiras objektas dokumento pabaigoje:

```
<article>
  Įrašo turinys.
  <footer>
    Baigiamoji įrašo informacija.
  </footer>
</article>
```

(23)

Pavyzdžiui, tinklalapio apačioje dažniausiai pateikiami pagrindiniai meniu punktai (pasikartojantys iš viršutinės dalies, kur yra tikroji navigacija):

```
<div id="footer">
  <ul>
    <li><a href="">Pagrindinis</a></li>
    <li><a href="">Straipsniai</a></li>
    <li><a href="">Naujienos</a></li>
```

(24)

```
</ul>
</div>
```

Tačiau naudojant HTML5 standartą korektiška naudoti `<footer>` elementą:

```
<footer>
  <ul>
    <li><a href="">Pagrindinis</a></li>
    <li><a href="">Straipsniai</a></li>
    <li><a href="">Naujienos</a></li>
  </ul>
</footer>
```

(25)

<article>

`<article>` elementas įvardijamas kaip atskira ir nepriklausoma turinio sritis, kuri gali būti vadinama tiesiog įrašu. Tačiau įrašas gali būti interpretuojamas labai įvairiai: įrašą gali atitikti naujienų įrašas, straipsnis, komentaras ir pan. Svarbu nepainioti `<section>` su `<article>`. Sekcijos naudojamos apibrėžti gana abstrakčioms puslapio struktūros sritims arba išskirti turinį į kažkokias logines grupes, tuo tarpu `<article>` yra konkretus įrašas arba informacijos blokas. Įrašas gali turėti tiek antraštinį elementą, tiek baigiamąjį. Pavyzdys: puslapis, kuriame yra straipsnis, o po jo seka komentarai. Tokios struktūros HTML 5 kodas būtų toks:

```
<article>
  <header>
    <h1>Straipsnis apie HTML 5</h1>
    <p><time pubdate datetime="2011-03-01">2011-03-01</time></p>
  </header>
  <p>Straipsnio turinys</p>
  <!-- ... -->
  <section>
    <h2>Komentarai</h2>
  </section>
  <article>
    <header>
```

(26)

```

        <h3>Autorius: Edmundas</h3>
        <p><!-- Data ir laikas--></p>
    </header>
    <p>Komentaras</p>
</article>

<article>
    <header>
        <h3>Autorius: Irmantas</h3>
        <p><!-- Data ir laikas --></p>
    </header>
    <p>Komentaras</p>
</article>
</section>
</article>

```

<aside>

<aside> buvo griežtai surištas su **<article>** ir jo paskirtis buvo įrašę įterpti susijusią turinio atžvilgiu informaciją. Pavyzdžiui, kurioje nors straipsnio dalyje gali būti įterpta citata arba termino paaiškinimas.

```

<article>
    <header>
        <h1>Straipsnio antraštė</h1>
    </header>
    <p>Straipsnio tekstas.</p>
    <aside>
        <q>Citata, susijusi su įrašu ar įrašo
pastraipa.</q>
    </aside>
    <p>Tolimesnis straipsnio tekstas.</p>
</article>

```

(27)

Dabar **<aside>** gali būti ne tik įrašo viduje kaip susijusi išskirta informacija, tačiau ir kaip atskira puslapio sritis, kuri nebūtinai glaudžiai susijusi su įrašu (papildoma navigacija ar net reklama).

<nav>

`<nav>` apibrėžia navigacijos sritį arba nuorodų rinkinį, kur nuorodos veda į kitus tinklalapio puslapius. Paprastai pagrindinė tinklalapio navigacija užrašoma atskirame `<div>` elemente ar sąrašė (arba abiejuose).

```
<div id="menu">
  <ul>
    <li><a href="">Straipsniai</a></li>
    <li><a href="">Naujienos</a></li>
  </ul>
</div>
```

(28)

```
<ul id="menu">
  <li><a href="">Straipsniai</a></li>
  <li><a href="">Naujienos</a></li>
</ul>
```

(29)

Tačiau su HTML5 tai galima nurodyti šiek tiek aiškiau ir konkrečiau:

```
<nav>
  <h1>Navigacija</h1>
  <ul>
    <li><a href="">Straipsniai</a></li>
    <li><a href="">Naujienos</a></li>
  </ul>
</nav>
```

(30)

3.7.2. Vaizdo ir garso failai

Vaizdo failai su `<video>`

Norint įdėti vaizdo įrašą puslapyje, vienu ar kitu atveju naudojama „Adobe Flash“ technologija, kuri kartais sukelia nepatogumų, ypatingai integruojant.

```
<object classid="clsid:d27cdb6e-ae6d-11cf-
96b8-444553540000" width="425"
height="344">
```

(31)

```

<param name="allowFullScreen"
  value="true" />
<param name="allowscriptaccess"
  value="always" />
<param name="src"
  value="http://www.youtube.com/v/oHg5SJYRHA0&hl=en&fs=1&" />
<param name="allowfullscreen"
  value="true" />
<embed type="application/x-shockwave-
  flash" width="425" height="344"
  src="http://www.youtube.com/v/oHg5SJYR
  HA0&hl=en&fs=1&"
  allowscriptaccess="always"
  allowfullscreen="true">
</embed>
</object>

```

Pakankamai daug vargo dėl vieno vaizdo failo įkėlimo. Tačiau HTML5 siūlo lengvą išeitį, kuri leidžia įterpti vaizdo failą taip pat paprastai kaip paveikslėlį naudojantis `<video>` elementu.

```

<video width="640" height="360"
  src="http://www.youtube.com/demo/google_main.
  mp4" controls autobuffer>
<p>Peržiūrėk vaizdo įrašą tinklalapyje arba
  jį <a href="video.mp4">parsisiųsk</a>.</p>
</video>

```

(32)

Kadangi vaizdo failai gali būti išsaugoti įvairiais formatais ir kodekais, gali kilti problemų su `<video>`, nes naršyklės palaiko skirtingus kodekus `<video>` elementui. Problema yra ta, kad dauguma jų yra uždari, apsaugoti patentų ir brangiai kainuoja. Atvirieji kodekai netenkina kokybės reikalavimų.

Garso failai su `<audio>`

Garso įrašus įkelti į tinklalapį taip pat labai paprasta:

```

<audio src="irasas.ogg" controls
  preload="auto" autobuffer></audio>

```

(33)

HTML5 siūlo daug naujovių, kurios palengvina nesudėtingų ir interaktyvių tinklalapių aplikacijų kūrimą.

3.8. Pakopiniai stiliai CSS

CSS (angl. *Cascading Style Sheets*) – tai kalba, skirta nusakyti kita struktūrine kalba aprašyto dokumento vaizdavimą. Dažniausiai CSS aprašomas HTML dokumentų pateikimas, tačiau ją galima taikyti ir įvairiems kitiems XML dokumentams. Hipertekstinės kalbos stiliai, kuriais aprašomi hipertekstinio dokumento formatai.

CSS kūrėjai yra W3C (angl. *World Wide Web Consortium*) organizacija, kuri kartu su didžiosiomis informacinių technologijų bendrovėmis kuria HTML ir kitus interneto standartus. Interneto naršyklių kūrėjai atsižvelgia ir pritaiko savo gaminius prie W3C rekomendacijų. Pirmasis CSS standartas pristatytas dar 1996 metais, antrasis – 1998 metais. Stiliai vadinami pakopiniais, nes tai atitinka HTML dokumentų apipavidalinimo principą. Naudojant daug stilių, jie veikia pagal svarbą, tarsi pakopomis. CSS – sąlyginai senas ir pripažintas būdas paprasčiau ir tiksliau išdėstyti tinklalapio turinį.

HTML kalba atsako į klausimą – ką atvaizduoti, tuo tarpu CSS – kaip atvaizduoti? Daugelis seniau naudojamų HTML žymių, kurios buvo skirtos stiliams nurodyti yra nebenaudojamos, nes jas pakeičia CSS stilių taisyklės.

Nors ankstesnėse HTML versijose buvo nemažai išvaizdą aprašančių elementų, bet rekomenduojama HTML kalba žymėti dokumento sandarą, o išvaizdą (teksto spalvas ir pan.) aprašyti atskirame CSS dokumente. Toks dokumentas užima mažiau vietos ir greičiau pasirodo vartotojo naršyklėje (įvairių puslapių išvaizdą aprašantis CSS dokumentas iš serverio atsisiunčiamas tik vienažart). Toks dokumentas lengviau bei kokybiškiau apdorojamas automatiškai, todėl tokius dokumentus geriau indeksuoja paieškos sistemos.

Nuo HTML versijos „HTML 4.01 Strict“ CSS panaudojimas išvaizdos aprašymui yra privalomas.

CSS privalumai:

- toks dokumentas užima mažiau vietos ir greičiau pasirodo vartotojo naršyklėje (įvairių puslapių išvaizdą aprašantis CSS dokumentas iš serverio atsisiunčiamas tik vienažart);

- toks dokumentas lengviau bei kokybiškiau apdorojamas automatiškai, todėl tokius dokumentus geriau indeksuoja paieškos sistemos;
- naudojant CSS lengviau iškart keisti visų puslapių išvaizdą;
- paprasčiau pasiekti, jog šiuos puslapius įvairios naršyklės rodytų vienodai;
- esant ribotam perdavimo kanalui galima siųsti tik patį HTML dokumentą bei nesiųsti jo CSS;
- galima siųsti tik patį HTML dokumentą, bei nesiųsti jo CSS, jei naršyklė nepajėgi jį atvaizduoti.

Panaudojus kelis skirtingus CSS dokumentus, tą patį HTML dokumentą galima gerai pritaikyti skirtingoms naršyklėms, pvz., vienaip atvaizduoti kompiuterio ekrane, o kitaip – mobiliajame telefone.

Į stilius įtraukiama informacija apie hipertekstinio dokumento išvaizdą, pavyzdžiui, teksto šriftą, spalvą, įtraukas, intervalus tarp pastraipų ir eilučių. Pagrindinis pakopinių stilių sukūrimo tikslas yra atskirti dokumento turinį nuo jo pateikimo (apipavidalinimo). Pakopinių stilių specifikacija taip pat apibrėžia stilių, įtrauktų į hipertekstinio dokumento turinio aprašą, ir skirtingų stilių suderinamumą.

Pagrindiniai hipertekstinio dokumento su pakopiniais stiliais privalumai: tą patį stilių failą galima pritaikyti keliems tinklalapiams, todėl lengviau atnaujinti arba pakeisti svetainės išvaizdą, didesnės (palyginant su HTML) formatų taikymo galimybės, tinklalapio pirminiame tekste sumažėja HTML kalbos gairių ir atributų skaičius.

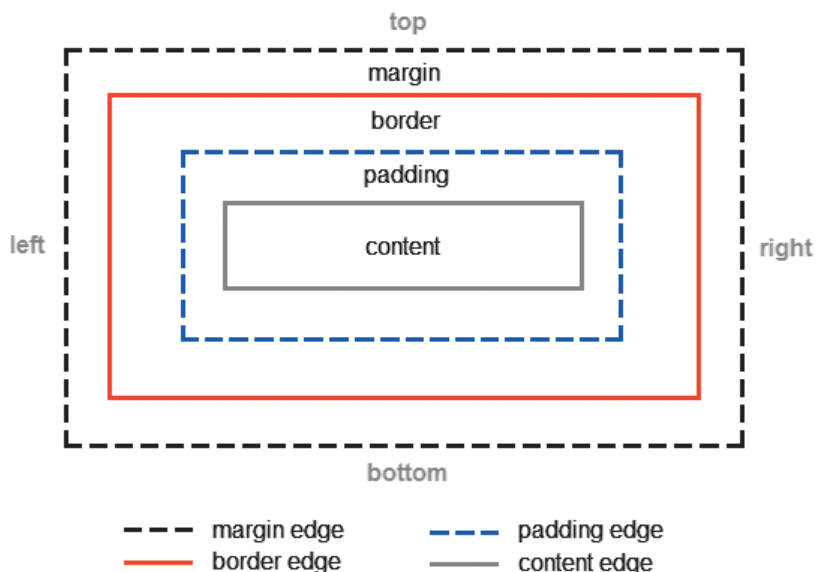
Pakopiniai stiliai būna išoriniai ir įterptieji. Jiems aprašyti naudojamos specialiai šiam tikslui sukurtos pakopinių stilių kalbos CSS ir XSL. Labiau paplitusi CSS kalba. Ji tinka HTML ir XML dokumentams. Kalba XSL tinka tik XML dokumentams. Tačiau ji tinka ir dokumentų konvertavimui. Pavyzdžiui, ją naudoja serveriai XML dokumentams konvertuoti į HTML/CSS formato dokumentus.

Dinaminiai stiliai leidžia dokumente pakeisti bet kokią HTML elementą. Stiliai kuriami arba kaip atributų deskriptoriai, arba su CSS (kaskadinių stilių sąrašai/pakopinių stilių schemas). Naudojant paprastus scenarijus HTML puslapių autoriai gali keisti dinaminių CSS atributų ir savybių reikšmes:

- elemento pavaizdavimas ir paslėpimas;
- teksto dydžio, šrifto, spalvos ir kitų elementų keitimas;
- elemento padėties ekrane keitimas.

3.8.1. Tinklalapio elementų išdėstymas

Naudojant CSS pakopinius stilius tinklalapio elementus galima išdėstyti įvairiais būdais, taip, kaip atrodo patogiausia vartotojui. Norint geriau suprasti CSS elementų išdėstymo principus reikia perprasti vadinamąjį „Lango modelį“ (6 pav.). Atvaizduojant kiekvienas dokumento elementas yra laikomas stačiakampe dėžute, kuri turi turinio plotą apsupimą „padding“, kraštinėmis. Paveiksle matyti, šios įvairios dalys.



6 pav. Pakopinių stilių „langelio“ modelis

Paraštės visada yra skaidrios. Sienų būna įvairių stilių. Fono elemento parametrai taikomi tik vidinėje sienų srityje, kuri apima „padding“ ir turinio sritis. Ilustracijoje užpildytas plotas rodomas šiek tiek tamsesne spalva, kad

būtų lengviau suprasti. Šie elementai yra neprivalomi, tačiau siekiant apskaičiuoti poziciją ir dydžius joms priskiriama numatytoji reikšmė 0 px, žinoma, jei nenurodoma kitaip. Skirtingi dydžiai gali būti nustatomi kiekvienai atskirai pusei (viršui, dešinei, apačiai ir kairiajai). Paraštės netgi gali įgyti neigiamas reikšmes.

Kiekvieno langelio plotis ir aukštis yra lygus išorinės paraštės pločiui ir aukščiui. Tai nėra būtinai turinio zonos plotis ir aukštis.

Languose gali būti sukurti bet kokie kiti langai, taip sukuriama jų hierarchija, pagrindinis langų išsidėstymas priklauso nuo naršyklės lango. Naršyklės langas yra šios hierarchijos pagrindinis elementas.

Su CSS taisyklėmis galime nurodyti visą puslapio dizainą. Įvairius rėmelius, jų storius, spalvas. Tekstų šriftus, spalvingumą, išdėliojimą. Įvairių atstumų tarp HTML elementų ir kitą dizainui skirtą informaciją. CSS technologija leidžia apibrėžti stilių bet kokiam HTML elementui ir jį priskirti visiems norimiems puslapiams. Norint pakeisti viso puslapio struktūrą užtenka pakoreguoti stiliaus failą ir visa puslapio išvaizda atsinaujina. Tai sutaupo daug laiko, kodas tampa lengviau redaguojamas, lengviau padaryti puslapio dizaino pakeitimus.

Naudojant išorinius CSS aprašus nesusidarko HTML kodas, todėl kodas tampa lengviau skaitomas. Paieškos sistemoms patinka tekstas, bet nepatinka kai puslapio didžiąją dalį sudaro kodas, o ne žmogui skaitomas tekstas, todėl reikėtų stengtis į patį puslapį nerašyti CSS taisyklių, o jas įtraukti pasinaudojant <link> žyme.

3.8.2. CSS sintaksė

CSS taisyklės sintaksė yra paprasta, ją sudaro trys dalys: selektorius, parametras ir jo reikšmė.

```

selektorius {
  parametras:reikšmė;
  parametras:reikšmė;
  parametras:reikšmė;
}

```

(34)

Tai yra kodo struktūros pavyzdys, kuriame vienam selektoriui aprašytos trys taisyklės. Taisyklių galima aprašyti tiek, kiek reikia. Tas pats parametras viename taisyklių bloke gali būti tik vieną kartą. Bloko pradžią nurodo atidarantys laužtiniai skliaustai „{“, o pabaigą – uždarantys laužtiniai skliaustai „}“.

Selektorius

Selektorius tai yra ta dalis, kuri parenka kokiems elementams bus taikomos CSS taisyklės. Selektorių su kiekviena nauja CSS kalbos versija atsiranda vis įvairesnių. Be to, ne visus selektorius supranta visos naršyklės.

CSS selektoriumi gali būti bet kuris HTML elementas, taip pat vietoje selektoriumi gali būti apibrėžta klasė. Pavyzdžiui:

```
p {text-align:center}
```

aprašo stilių HTML pastraipos elementui „p“ (<p>). Stilius gali būti sudaromas aprašant keletą savybių (atskirtų kabliataškiu):

```
p {text-align:center;color:red}
```

Internetinio puslapio pastraipų tekstas bus centruotas, raudonos spalvos.

3.8.3. Populiariausi selektoriai

Visi elementai

Labai dažnai prieš pradėdant rašyti CSS stilius, išvalomi visų elementų vidiniai ir išoriniai tarpai (angl. *padding*, *margin*). Toks aprašas visiems HTML elementams priskirtų stilius „padding“ ir „margin“ po 0 pikselių:

```
* {  
padding:0px;  
margin:0px;  
}
```

(35)

Elemento pavadinimas

Toks CSS kalbos aprašas nuo visų puslapyje esančių paveikslėlių numimtų rėmelį:

```
img {
  border:none;
}
```

(36)

Klasės selektorius

HTML elementui galima priskirti klasės atributą, o tai leidžia tam pačiam elementui turėti skirtingus stilius. Pavyzdžiui, norint, kad dvi pastraipos būtų skirtingai lygiuojamos:

```
p.right {text-align: right}
p.center {text-align: center}
```

(37)

HTML dokumente norimos pastraipos aprašomos taip:

```
<p class="right">Ši pastraipa lygiuosius
dešinės.</p>
```

ir

```
<p class="center">Ši pastraipa bus centruota.</p>
```

Elementui galima priskirti keletą klasių vienu metu, tačiau persidengiantys stiliai bus perrašyti:

```
<p class="right kitaklase">Pastraipa su dviem
klasėm.</p>
```

(38)

Labai patogu, kai klasės pavadinamos pagal tai, kokį stilių jos suteikia. Iš užrašo `p class="right"` galima suprasti, kad pastraipa bus lygiuojama prie dešinės paraštės.

Kitame pavyzdyje selektorius paima visus HTML elementus, kurie turi atributą „class“ su reikšme „klasė“ ir tų elementų teksto spalva tampa raudona:

```
.klase {
  color:#f00;
}
```

(39)

ID selektorius

Galima aprašyti stilių pagal elemento selektojų:

```
#selektorius {text-align: right}
```

Tada tik tam elementui su tuo selektoriumi bus taikomas aprašytas stilius:

```
<p id="selektorius">Ši pastraipa bus lygiuota  
dešinėje.</p>
```

(40)

Toks selektorius paima elementus su atributu „id“ ir reikšme „id“. Viename puslapyje gali būti tik vienas elementas su tokiu „id“, negali būti kelių su tokiu pat „id“ pavadinimu:

```
#id {  
  background-color:#0f0;  
}
```

(41)

Parametras

Parametro vietoje yra rašoma CSS kalbos komanda. Komandų yra apie 90, jos visos yra surašytos CSS žinyne. Viename taisyklių bloke kiekviena komanda negali kartotis, visos turi būti skirtingos, tačiau skirtingų gali būti tiek, kiek reikia. Daugelis parametų yra angliški žodžiai, todėl mokant anglų kalbą yra iš ties paprasta juos prisiminti.

Reikšmė

Reikšmės vietoje rašoma parametro reikšmė. Kiekvienas parametras turi turėti kokią nors jam tinkamą reikšmę, pavyzdžiui parametras „color“ gali turėti žodinę spalvos išraišką (pvz.: red) arba „hex“ koduote užrašytą spalvos kodą (pvz.: #ff0000; arba #f00;), arba „rgb“ išraišką (pvz.: rgb(255,0,0)). Kokių reikšmės gali įgyti parametras priklauso nuo parametro.

Komentarai

Kaip ir kiekvienoje kalboje, CSS kalboje taip pat galima komentuoti atskirų puslapio vietų rašomus taisyklių blokus:

```

/* css komentaras */
div {
    border:1px solid #f00;
}

```

(42)

CSS kalba yra kliento pusės, todėl bet kurio atidaryto puslapio CSS taisyklės galima pasižiūrėti (pvz.: tai galima atlikti tiesiog paspaudus dešinią pelės klavišą ant neaktyvios puslapio vietos ir pasirinkus meniu punktą „View Page Source“).

3.8.4. CSS įterpimas į HTML

CSS stilius į HTML dokumentą galima įterpti 3 būdais:

- išoriniu (byla *.css),
- vidiniu (CSS stiliai įrašomi pačiame HTML dokumente),
- žymoje pasinaudojant atributu „style“ (CSS įrašomi į kiekvieną HTML žymę).

Vienu metu gali būti naudojami visi būdai, taip pat galima naudoti kelis išorinius failus viename HTML dokumente. Tačiau rekomenduojama naudoti išorinių rinkmenų įterpimą, taip puslapis yra greičiau kraunamas, puslapį sudaro daugiau žmogui skaitomo ir naudingo teksto, o ne programinio kodo.

Išorinis

Naudojant išorinį įterpimo būdą yra sukuriamas CSS failas su plėtiniu *.css, pavyzdžiui stiliai.css ir jo viduje yra surašomos CSS kalbos taisyklės. Į HTML dokumentą jis yra įterpiamas naudojant žymę <link> puslapio <head> dalyje.

Pavyzdys:

```

<head>
  <link rel="stylesheet" type="text/css"
  href="stiliai.css" />
</head>

```

(43)

Galima įtepti tiek stilių rinkmenų, kiek reikia. Kiekvienai rinkmenai aprašoma nauja <link> žymė:

```
<head>
  <link rel="stylesheet" type="text/css"
href="pagrindinis.css" />
  <link rel="stylesheet" type="text/css"
href="menu.css" />
  <link rel="stylesheet" type="text/css"
href="straipsnis.css" />
</head>
```

(44)

Vidinis

Stilius galima įrašyti ir tiesiai į HTML dokumentą pasinaudojant žymę <style> ir jos atributu „type“ su reikšme „text/css“. Šios žymės viduje galima rašyti tas pačias stiliaus taisykles kaip išorinėje rinkmenoje. Skirtumas yra tik toks, kad pirmiausiai yra pritaikomos išorinės taisyklės, o tik tada – vidinės.

Pavyzdžiai:

```
<style type="text/css">
div {
  border:1px solid #ccc;
}
</style>
```

(45)

```
<head>
  <style type="text/css">
  <!--
  a {text-decoration: none}
  p {font-family: Verdana, Arial, Helvetica,
sans-serif; font-size: 11px; padding-top:
3px; padding-right: 6px; padding-bottom: 3px;
padding-left: 6px}
  .tekstas {font-family: Verdana, Arial,
Helvetica, sans-serif; font-size: 1px;
padding-right: 6px; padding-left: 6px}
  ...
  -->
</style>
</head>
```

(46)

Kitame pavyzdyje pateiktas kodas, kurio pagalba galima nupiešti Lietuvos vėliavą:

```
<html>
<head>
  <meta http-equiv="content-type" content="text/html; charset=UTF-8">
  <title> Lietuvos vėliava su CSS</title>
<style type="text/css">
  body {
    margin:0;
    padding:0;
  }
  #veliava {
    margin: 0 auto 0 auto;
    height: 300px;
    width: 450px;
  }
  #geltona {
    background-color: yellow;
  }
  #zalia {
    background-color: #239E46;
  }
  #raudona {
    background-color: red;
  }
  #geltona, #zalia, #raudona {
    height: 100px;
  }
</style>
</head>
<body>
<div id="veliava">
  <div id="geltona">
  </div>
  <div id="zalia">
  </div>
  <div id="raudona">
  </div>
</div>
</body>
</html>
```

(47)

Žymoje

Trečias metodas suteikti stilius yra rašyti juos tiesiai į žymės atributą „style“. Jo viduje galioja tos pačios taisyklės kaip ir išorinėse rinkmenose ar <style> žymoje. Svarbu prieš naudojant šį metodą išsiaiškinti, ar ta žymė turi „style“ atributą. Puslapio krovimo metu žymoje esančios taisyklės yra pritaikomos paskutinės.

Pavyzdžiai:

```
<div style="border:1px solid #ccc; border-bottom:none;"></div>
```

(48)

Kitame pavyzdyje pastraipa rašoma 14 dydžio geltonu „Verdana“ šriftu juodame fone, o atstumas nuo puslapio viršaus – 40 taškų.

```
<body>
  <p style="color: yellow; font-family: Verdana; font-size: 14px; background: black; padding-top: 40px;">Ši pastraipa rašoma geltonu
  Verdana šriftu juodame fone. Raidžių dydis -
  14, o atstumas nuo puslapio viršaus - 40 taškų.</p>
  <p>Tai nauja pastraipa be CSS.</p>
</body>
```

(49)

3.8.5. CSS hierarchija

Kiekviena naršyklė turi savo vidinius stilių aprašus, todėl sukūrus puslapį tik su HTML kalba, nebūtinai jis atrodys identiškai visose naršyklėse. Kai kurios naršyklės turi „padding“ nustatymą 8px <body> elementui, todėl puslapis yra atitraukiamas nuo viršaus. Taisyklių būna nurodyta ir daugiau. Yra nustatyta kokių šriftu bus atvaizduojamas tekstas, kokia spalva, kokio dydžio ir t. t.

Stilių priskyrimo tvarka

Kai naršyklė krauna puslapį, visų pirma yra įkeliamas HTML kodas, vėliau pritaikomos pačios naršyklės taisyklės, tada ieškoma išorinių stiliaus rinkmenų ir sukeliama juose esančios taisyklės, baigus – pritaikomos vidinės HTML

dokumente esančios taisyklės, kurios rašomos į `<style type="text/css">` žymę. Galiausiai yra pritaikomos taisyklės esančios viduje HTML elementų „style“ atributuose.

Stilių priskyrimo tvarka yra tokia:

1. įkeliamas HTML kodas;
2. pritaikomos naršyklės vidinės taisyklės pagal nutylėjimą (angl. *by default*);
3. pritaikomos išorinių *.css failų taisyklės;
4. pritaikomos vidinės, žymoje `<style type="text/css">` esančios taisyklės;
5. pritaikomos vidinės, žymų attribute „style“ esančios taisyklės.

Pavyzdžiui, turint HTML elementą „div“, jam galima priskirti skirtingus CSS stilius. Pavyzdį sudaro trys dalys, kiekvienoje dalyje prieš tai buvęs HTML kodas bus papildomas.

1 dalis

HTML kodas.

```

<html>
<head>
  <link rel="stylesheet" type="text/css"
href="stiliai.css" />
</head>
<body>
  <div>Viduje div elemento esantis
tekstas</div>
</body>
</html>

```

(50)

Iš 3 eilutės matyti, kad yra įtrauktas CSS failas pavadinimu stiliai.css, kurio vidus yra:

```

div {
  width:200px;
  background-color:#CCC;
  font-style:italic;
}

```

(51)

Tokiame puslapyje matomas 200 pikselių pločio su pilkos spalvos fonu (#ccc) „div“ blokelis ir juodu pasviru tekstu viduje.

2 dalis

HTML kodas papildomas su <style> žyme:

```
<html>
<head>
  <link rel="stylesheet" type="text/css"
href="stiliai.css" />
  <style type="text/css">
    div {
      font-style:none;
      margin:0px auto;
      height:100px;
    }
  </style>
</head>
<body>

  <div>Viduje div elemnto esantis te-
kstas</div>

</body>
</html>
```

(52)

Failo stiliai.css turinys nekeičiamas, jis išlieka koks buvo.

Dabar naršyklė pritaiko vidinius naršyklės stilius, tada išorinės rinkmenos stilius ir tada vidinius stilius. Vidiniuose stiliuose dubliavosi taisyklės, nes buvo aprašytas tas pats „div“ elementas, tik jau priskirtos kitokios reikšmės tiems patiems CSS parametrams. Kadangi vidinės taisyklės reikšmės buvo kraunamos vėliausiai, tai jos ir lieka užkrovus pilnai puslapį.

Taigi dabar turime „div“ elementą, kaip ir buvo – pilką spalvą jo fone, jis taip pat lieka 200px pločio, tačiau jo tekstas jau nebe pasviras, o paprastas ir jis tapo 100px aukščio bei jo pozicija yra lygiuojama horizontaliai centre.

3 dalis

HTML failas bus praplėstas naujomis CSS taisyklėmis „div“ elemento viduje.

```

<html>
<head>
  <link rel="stylesheet" type="text/css"
href="stiliai.css" />
  <style type="text/css">
    div {
      font-style:none;
      margin:0px auto;
      height:100px;
    }
  </style>
</head>
<body>

  <div style="border:10px solid #333; font-
style:italic;">
    Viduje div elemnto esantis tekstas
  </div>

</body>
</html>

```

(53)

CSS stiliaus failas stiliai.css išlieka toks pats:

```

div {
  width:200px;
  background-color:#CCC;
  font-style:italic;
}

```

(54)

Naršyklė užkrauna HTML kodą, tada savo vidinius stilius, tada išorinius, vidinius iš žymos <style> ir tada dar vidinius tiesiai iš HTML elemento atributo „style“.

Rezultatas: „div“ elementas, per vidurį horizontaliai, pilkas fonas, šriftas vėl tampa pasviręs ir dar aplink elementą prisideda 10 pikselių storio, vientisas ir beveik juodos spalvos rėmelis- #333.

3.8.6. CSS stilių parametrai

CSS fonas

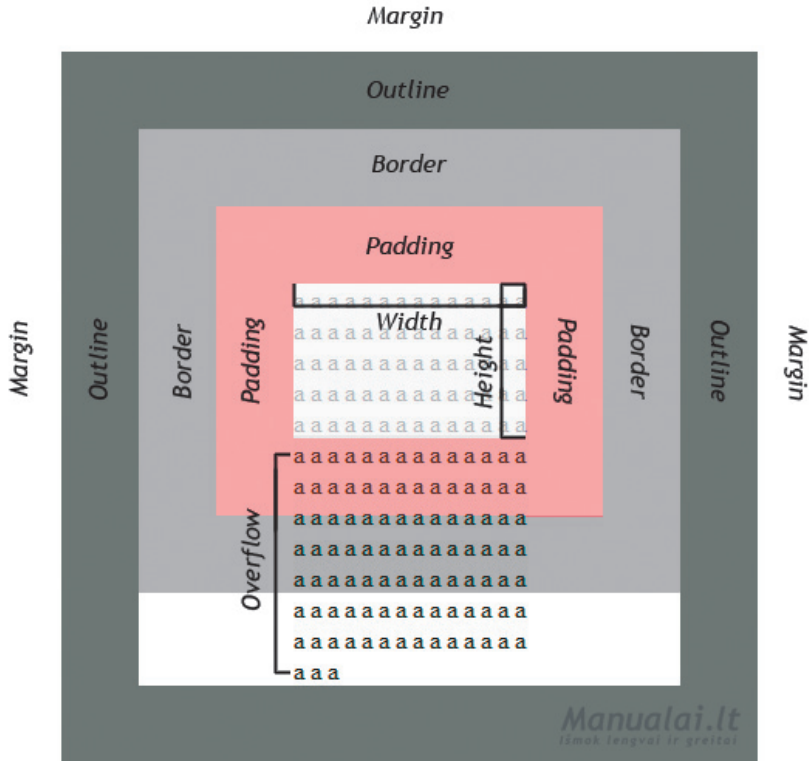
CSS kalboje foną galima nurodyti keliais būdais. Paprasčiausias yra su parametru „background-color“, kurio pagalba galima nurodyti fono spalvą. Kiti naudojami parametrai yra: „background-color“, „background-image“, „background-repeat“, „background-attachment“ ir „background-position“.

7 lentelė. Fono parametrai

Reikšmė	Aprašymas
background	Sutrumpintas visų sekančių parametų variantas.
background-color	Nurodo fono spalvą
background-image	Nurodo fono paveikslėlį
background-repeat	Nurodo ar kartoti fono paveikslėlį, jei reikia galima nurodyti tik vieną ašį.
background-attachment	Nurodo ar fono paveikslėlis yra „pririštas“ vietoje, ar juda kartu su puslapiu.
background-position	Nurodo fono paveikslėlio poziciją

CSS rėmai ir kontūrai – „width“, „height“, „padding“, „border“, „outline“, „margin“ ir „overflow“

CSS kalboje yra keletas parametų, su kuriais galite lengvai reguliuoti bet kokio elemento kraštus. Pagrindiniai rėmų, kontūrų ir atstumų tarp elementų nustatymai yra pavaizduoti 7 paveiksle ir aprašyti 8 lentelėje.



7 pav. Pagrindiniai rėmų, kontūrų ir atstumų tarp elementų nustatymai

Šaltinis: manualai.lt

8 lentelė. CSS rėmai ir kontūrai

Parametras	Reikšmė
width	Nurodo elemento plotį. Galima naudoti įvairius CSS matavimo vienetus. Į šią reikšmę neįeina nei „padding“, nei „border“, nei „outline“, nei „margin“ reikšmės.
height	Nurodo elemento aukštį. Galima naudoti įvairius CSS matavimo vienetus. Į šią reikšmę neįeina nei „padding“, nei „border“, nei „outline“, nei „margin“ reikšmės.

Parametras	Reikšmė
padding	Nurodo atstumą tarp elemento vidinio turinio ir jo rėmelio („border“). Jei yra nurodytas elemento parametras „background-color“, tada „padding“ erdvė būna padengiama ta pačia fono spalva, kaip ir centras. „Padding“ atstumą galima keisti kiekvienai pusei individualiai (viršuje, dešinėje, apačioje ar kairėje elemento pusėms).
border	Nurodo elemento rėmelį. Gali būti įvairių stilių, įvairių pločių. Galima keisti kiekvienos pusės rėmelį individualiai (viršuje, dešinėje, apačioje ar kairėje elemento pusėje).
outline	Nurodo antrinį rėmelį, išorinę elemento liniją esančią aplink rėmelį. Galimi tokie pat nustatymai kaip ir rėmeliui, galima keisti spalvą, stilių, storį. Visus nustatymus galima taikyti visoms arba kai kurioms pasirinktoms pusėms (viršuje, dešinėje, apačioje ar kairėje elemento).
margin	Nurodo atstumą tarp elementų. Tai yra erdvė už „outline“. Visus nustatymus galima taikyti visoms arba kai kurioms pasirinktoms pusėms (viršuje, dešinėje, apačioje ar kairėje elemento).
overflow	Nurodo kaip elementui elgtis, jei jo turinys reikalauja daugiau vietos, nei yra nurodytas jo aukštis.

CSS teksto nustatymai

CSS kalboje teksto konfigūracijai naudojami šie parametrai: „text-align“, „text-decoration“, „text-indent“, „text-transform“, „line-height“, „color“, „direction“, „letter-spacing“, „vertical-align“, „white-space“, „word-spacing“.

9 lentelė. CSS teksto reikšmės

Parametras	Reikšmė
text-align	Nurodo teksto horizontalų lygiavimą
text-decoration	Nurodo papildomas teksto dekoracijas
text-indent	Nurodo kiek atitraukti pirmą paragrafo sakinį nuo krašto
text-transform	Nurodo papildomas teksto transformacijas

Parametras	Reikšmė
line-height	Nurodo eilutės aukštį
color	Nurodo teksto spalvą
direction	Nurodo teksto kryptį
letter-spacing	Nurodo tarpų tarp raidžių dydį
vertical-align	Nurodo teksto vertikalų lygiavimą
white-space	Nurodo kaip elgtis su tarpais
word-spacing	Nurodo tarpų tarp žodžių dydžius

CSS šriftai

CSS kalboje teksto šriftą, dydį, stilių galima nurodyti šiais parametrais: „font-family“, „font-size“, „font-style“, „font-variant“ ir „font-weight“. Galima naudoti sutrumpintą šių visų parametru variantą – parametru „font“. Teksto spalvą galima nurodyti su parametru „color“.

CSS kodo pavyzdys:

```

.pirmas{
  font-family:Verdana, Geneva, sans-serif;
  font-size:12px;
}

.antras{
  font-family:Georgia, "Times New Roman", Times, serif;
  font-size:12px;
}

.trecias{
  font-family: "Courier New", Courier, monospace;
  font-size:12px;
}

.ketvirtas{
  font-family:Arial, Helvetica, sans-serif;
  font-size:12px;
}

```

(55)

```
font-style:italic;
font-variant:small-caps;
font-weight:700;
}
```

10 lentelė. CSS šriftų reikšmės

Parametras	Reikšmė
font-family	Nurodo teksto šrifto šeimą. Patariama naudoti tokius šriftus, kuriuos turi visi kompiuteriai, telefonai ir kita aparatūra, su kuria gali atverti jūsų kuriamą puslapį. Atskiriant kableliu galima nurodyti kiek reikia daug alternatyvių šriftų. Patiriamos reikšmės: font-family: „Verdana“, „Geneva“, sans-serif; font-family: „Georgia“, „Times New Roman“, „Times“, serif; font-family: „Courier New“, „Courier“, monospace; font-family: „Arial“, „Helvetica“, sans-serif; font-family: „Tahoma“, „Geneva“, sans-serif; font-family: „Trebuchet MS“, „Arial“, „Helvetica“, sans-serif; font-family: „Arial Black“, „Gadget“, sans-serif; font-family: „Times New Roman“, „Times“, serif; font-family: „Palatino Linotype“, „Book Antiqua“, „Palatino“, serif; font-family: „Lucida Sans Unicode“, „Lucida Grande“, sans-serif; font-family: „Comic Sans MS“, cursive;
font-size	Nurodo teksto aukštį.
font-style	Nurodo šrifto stilių. Galimos reikšmės: normal, italic, oblique arba inherit.
font-variant	Nurodo papildomas šrifto variacijas. Galimos reikšmės: small-caps, normal arba inherit.
font-weight	Nurodo šrifto storį. Galimos reikšmės: normal, bold, bolder, lighter, 100, 200, 300, 400, 500, 600, 700, 800, 900.
font	Nurodo šriftą ir jo kitus parametrus.

CSS nuorodos

Kiekviename puslapyje yra nemažai nuorodų. Nuorodos turi būti kitokios išvaizdos nei paprastas tekstas, tada vartotojui bus paprasta suprasti, kad

tai yra nuoroda ir ant jos galima paspausti. Kuriant nuorodas yra naudojami įvairūs „hover“ efektai, t. y. efektas užvedus pelės žymeklį ant nuorodos. Keletas nuorodų stilių keitimo taisyklių pateikiama 11 lentelėje.

11 lentelė. CSS nuorodų reikšmės

Parametras	Reikšmė
text-align	Nurodo teksto horizontalų lygiavimą
text-decoration	Nurodo papildomas nuorodos dekoracijas
text-transform	Nurodo papildomas nuorodos transformacijas
line-height	Nurodo nuorodos teksto eilutės aukštį
color	Nurodo nuorodos teksto spalvą
direction	Nurodo nuorodos teksto kryptį
letter-spacing	Nurodo nuorodos tarpų tarp raidžių plotį
word-spacing	Nurodo tarpų tarp žodžių dydžius
border	Nurodo rėmelį aplink nuorodą
margin	Nurodo erdvę aplink nuorodos rėmelį
padding	Nurodo erdvę tarp nuorodos teksto ir rėmelio
outline	Nurodo erdvę tarp „border“ ir „margin“

Pseudoklasės

Su nuorodomis yra naudojamos jų stilių pseudoklasės:

- a:link – paprasta nuoroda, dar ant jos nebuvo paspausta;
- a:visited – nuoroda, kurioje vartotojas jau buvo;
- a:hover – nuoroda ant kurios vartotojas užėjo su pele;
- a:active – dabar aktyvi nuoroda.

Norint, kad nuorodų pseudoklasės gerai veiktų, ją aprašyti reikia būtent šia tvarka: *a:link*, *a:visited*, *a:hover*, *a:active*. Kitu atveju gali kilti netikslumų.

Pavyzdys:

```

<style type="text/css">
a:link, a:active, a:visited {
  padding:10px;
  line-height:25px;
  font-size:15px;
  font-weight:bold;
  background-color:#0CC;
  text-decoration:none;
  font-family:Verdana, Geneva, sans-serif;
}
a:hover {
  background-color:#0F6;
  font-style:italic;
  text-decoration:overline underline;
}
</style>
<a href="#">Pradžia</a>
<a href="#">Antras</a>
<a href="#">Trečias</a>

```

(56)

CSS sąrašai

Keletas pagrindinių nustatymų dažniausiai naudojami su HTML sąrašų elementais, tačiau naudojant sąrašų žymes ir keletą CSS stilių taisyklių, galima gauti visai išdėstytus meniu punktus.

Paprasti sąrašų stiliai

CSS sąrašų redagavimo parametrai leidžia keisti sąrašo elementų žymeklių poziciją, pačius žymeklius. Pagrindiniai parametrai pateikti 12 lentelėje.

12 lentelė. **Pagrindiniai paprastų sąrašų stilių parametrai**

Parametras	Reikšmė
list-style	Nurodo visus parametrus iš karto. Daugiau informacijos čia: list-style
list-style-type	Nurodo sąrašo indeksų tipą, kokio tipo skaitmenimis numeruoti ir pnš. Daugiau informacijos čia: list-style-type
list-style-	Nurodo sąrašo indeksų vietą. Daugiau informacijos čia:

Parametras	Reikšmė
position	list-style-position
list-style-image	Nurodo paveikslėlį, kurį reikia naudoti vietoje indeksų. Daugiau informacijos čia: list-style-image

Horizontalaus meniu pavyzdys
CSS stilių aprašai:

```

ul {
    float:left;
    width:100%;
    padding:0;
    margin:0;
    list-style-type:none;
}

a {
    float:left;
    width:6em;
    text-decoration:none;
    color:white;
    background-color:#378adf;
    padding:0.2em 0.6em;
    border-right:1px solid white;
}

a:hover {
    background-color:#9ae02c;
}

li {
    display:inline;
}

```

(57)

HTML kodas:

```

<ul>
  <li><a href="#">Pradžia</a></li>
  <li><a href="#">Apie mus</a></li>
  <li><a href="#">Kontaktai</a></li>
  <li><a href="#">Nuorodos</a></li>
</ul>

```

(58)

CSS lentelės

Gerai pritaikius CSS kalbos galimybes su lentelių HTML žymėmis, galima padaryti greitai kraunamas ir gražiai atrodančias lenteles. Žemiau pateikiamas pavyzdys lentelės, kuri padaryta pilnai su CSS ir HTML.

HTML kodas:

```

<table summary="Lentelė">
  <thead>
    <tr>
      <th scope="col">Antraštė</th>
      <th scope="col">1 stulpelis</th>
      <th scope="col">2 stulpelis</th>
      <th scope="col">3 stulpelis</th>
    </tr>
  </thead>
  <tfoot>
    <tr>
      <td colspan="4"><em>Pavyzdinė lentelė
su betkokiais duomenimis</em></td>
    </tr>
  </tfoot>
  <tbody>
    <tr>
      <td>Pavadinimas</td>
      <td>reikšmė</td>
      <td>reikšmė</td>
      <td>reikšmė</td>
    </tr>
    <tr>
      <td>Pavadinimas</td>
      <td>reikšmė</td>
      <td>reikšmė</td>
      <td>reikšmė</td>
    </tr>
    <tr>
      <td>Pavadinimas</td>
      <td>reikšmė</td>
      <td>reikšmė</td>
      <td>reikšmė</td>
    </tr>
    <tr>
      <td>Pavadinimas</td>
      <td>reikšmė</td>
      <td>reikšmė</td>
      <td>reikšmė</td>
    </tr>
  </tbody>
</table>

```

(59)

```

        <td>reikšmė</td>
        <td>reikšmė</td>
        <td>reikšmė</td>
    </tr>
</tbody>
</table>

```

CSS kodas:

```

table {
    border-collapse:collapse;
    font-family:"Lucida Sans Unicode","Lucida
Grande",Sans-Serif;
    font-size:12px;
    margin:15px;
    text-align:left;
    width:500px;
}
table thead th {
    background-color:#e8b9ff;
}

table th {
    background-color: #e8b9ff;
    color:#003399;
    font-size:13px;
    font-weight:normal;
    padding:8px;
}
table td {
    background-color:#f7d0ff;
    border-top:1px solid #FFFFFF;
    color:#666699;
    padding:8px;
}

table tfoot td {
    background-color: #f7d0ff;
    text-align:center;
}
table tbody tr:hover td {
    background:#fee9ff;
}

```

(60)

```

        cursor:pointer;
    }

```

Antraštė	1 stulpelis	2 stulpelis	3 stulpelis
Pavadinimas	reikšmė	reikšmė	reikšmė
Pavadinimas	reikšmė	reikšmė	reikšmė
Pavadinimas	reikšmė	reikšmė	reikšmė
Pavadinimas	reikšmė	reikšmė	reikšmė
<i>Pavyzdinė lentelė su betkokiais duomenimis</i>			

8 pav. Lentelės pavyzdys su CSS ir HTML

Šaltinis: manualai.lt

CSS spalvos










CSS spalvos yra tokios pat, kaip ir HTML. Spalvos apibrėžiamos naudojant RGB sistemą, dažnai pasitelkiant šešiolyktainę skaičiavimo sistemos žymėjimą (angl. *hex*).

RGB – spalvų maišymo sistema, kurioje naudojamos trys, žmogaus akių receptorių atitinkančios spalvos: raudona (R), žalia (G) ir mėlyna (B). RGB sistemoje spalva nagrinėjama, kaip spinduliavimas: visų trijų spalvų nulinės reikšmės atitinka juodą spalvą, o visų trijų maksimalios reikšmės – baltą. Maišant RGB spalvų poras, gaunamos CMY (žydra, purpurinė, geltona, angl. *Cyan, Magenta, Yellow*) sistemos spalvos.

RGB sistemoje spalvos užrašomos procentine dalimi nuo baltos (pvz., 100 %, 100 %, 0 % reikštų purpurinę) arba šešiolyktaine: #00FFFF reikštų cianą.

RGB sistema dažniausiai naudojama elektronikoje, kompiuteriuose. Ja koduojamas vaizdas kineskopų ekranuose, jį patogi naudoti programuojant.

13 lentelė. CSS pagrindinės spalvos

Spalva	HEX	RGB
	#000000	rgb(0,0,0)
	#ff0000	rgb(255,0,0)
	#00ff00	rgb(0,255,0)
	#0000ff	rgb(0,0,255)
	#ffff00	rgb(255,255,0)
	#00ffff	rgb(0,255,255)
	#ff00ff	rgb(255,0,255)
	#c0c0c0	rgb(192,192,192)
	#ffffff	rgb(255,255,255)

Kitas būdas aprašyti spalvas – naudoti jų vardus. Yra 150 skirtingų spalvų pavadinimų, kuriuos palaiko dauguma naršyklių.

Standartinių spalvų vardai

Visame pasaulyje yra sutarta 16 skirtingų spalvų vardų, kuriuos galima taikyti rašant tiek HTML, tiek CSS.

Spalvų sąrašas:

- aqua;
- black;
- blue;
- fuchsia;
- gray;
- green;
- lime;
- maroon;
- navy;
- olive;
- purple;
- red;
- silver;
- teal;
- white;
- yellow.

Norint naudoti kitas spalvas patariama naudoti „hex“ koduotes.

3.9. Svetainių atitikimo standartams testavimas ir taisymas

HTML atitikimą standartams galima patikrinti laisvai prieinamu HTML tikrintuvu:

<http://validator.w3.org/>

CSS taisyklingumą galima patikrinti šiuo adresu:

<http://jigsaw.w3.org/css-validator/>

HTML tikrinimo procedūros žingsniai:

1. Atveriamas <http://validator.w3.org/> puslapis, kuriame yra trys tikrinimo pasirinkimai:

- tikrinti svetainę nurodytu adresu (angl. *Validate by URL*),
- tikrinti iš kompiuterio įkeltą HTML dokumentą (angl. *Validate by File Upload*),
- tikrinti kodą tiesiogiai įrašant jį į formą (angl. *Validate by direct input*).

Pavyzdžiui, pasirinkus laukelį „Validate by URL“, suvedamas Socialinių mokslų kolegijos svetainės adresas: <http://www.smk.lt>. Paspaudus žemiau esantį „More Options“, atsiverusioje srityje galima pasirinkti papildomų tikrinimo galimybių (9 pav.). Tikrinimas vykdomas paspaudus „Check“ mygtuką.

9 pav. HTML klaidų tikrinimo langas

2. Atlikus tikrinimą pateikiama informacija apie tikrintą dokumentą. Antraštėje nurodytas dokumento tipas, koduotė, kiek yra neatitikimo specifikacijoms klaidų (10 pav.).

This document was successfully checked as HTML5!	
Result:	Passed, 1 warning(s)
Address:	<input type="text" value="http://www.snk.lt/"/>
Encoding:	utf-8 (detect automatically)
Doctype:	HTML5 (detect automatically)
Root Element:	html

10 pav. HTML klaidų tikrinimo rezultatai

Šiuo atveju matyti, kad nurodytas puslapis yra sukurtas naudojant HTML5 standartą ir klaidų neturi.

Kitu atveju, visos rastos klaidos pateikiamos žemiau ir nurodoma, kurioje eilutėje rasta klaida, o šalia kiekvienos klaidos pateikiamas trumpas jos paaiškinimas.

CSS tikrinimas yra analogiškas. W3C yra sukūrusi paveikslėlių, rodančių svetainės atitikimą standartams (14 lentelė). Šie paveikslėliai dedami į interneto svetainę norint parodyti, jog svetainė kuriama remiantis standartais. Ištaisius visas svetainės klaidas, <http://www.w3.org> pasiūlo įdėti tokius paveikslėlius kartu su HTML kodu.

Pagal aukščiau aprašytą pavyzdį galima atlikti savo turimo dokumento tikrinimą (angl. *Validate by File Upload*). Spragtelėjus „Browse“ pasirenkamas dokumentas ir spaudžiamas „Check“.

W3C tikrinimo paslauga trumpai paaiškina kiekvieną klaidą bei parodo eilutės numerį. Taip galima atsidarius svetainę savo mėgstamoje teksto rinkimo programoje nesunkiai surasti bei ištaisyti klaidas.

Geriausia svetainės kūrimo metu kiekvieną kartą pridėjus ar pakeitus elementą, tikrinti puslapį adresu <http://validator.w3.org/>. Taip sutaupoma laiko, o sukurtoji svetainė atitiks standartus.

Šie tikrinimo metodai netinka taisyti serverio dinamiškai generuojamiems puslapiams, sukurtiems PHP, SSI (angl. *Server Side Includes*), ASP (angl. *Active Server Pages*), JSP (angl. *JavaServer Pages*) ar panašiomis priemonėmis. Tokie puslapiai nesaugomi serveryje tuo pavidalu, kuriuo pateikiami naudotojams, tačiau kaskart sukonstruojami įterpiant puslapio turinio elementus į HTML dokumento šabloną. Norint pasiekti tokių generuotų dokumentų atitiktį standartams, reikia redaguoti šabloną bei užtikrinti, kad jame talpinamos dalys taip pat atitinka HTML specifikacijas. Taisymo būdas labai priklauso nuo

serveryje naudojamos programinės įrangos, todėl konkrečių taisymo instrukcijų nėra. Visgi visi pateikti principai gali būti pritaikomi ir tokioms svetainėms.

14 lentelė. W3C svetainės standartą atitinkančių paveikslėlių pavyzdžiai

Paveikslėlis	HTML kodas
	<pre><p> </p></pre>
	<pre><p> </p></pre>
	<pre><p> </p></pre>
	<pre><p> </p></pre>

Klausimai ir užduotys

1. Kokia yra HTML standarto paskirtis?
2. Kokias galimybes suteikia HTML5?
3. Kokie būna HTML redaktoriai?
4. Ar pakanka patikrinti, kaip veikia sukurta interneto svetainė tik „Internet Explorer“ naršyklėje? Kodėl?
5. Ar verta mokėti HTML? Kodėl?
6. Kokie yra pagrindiniai dokumentų formato ir perdavimo standartai?
7. Kuo skiriasi HTTPS nuo HTTP protokolo?
8. Ką skiria SSL technologija?
9. Kas yra pagrindinis HTML kalbos vienetas?
10. Kokia yra HTML elemento konstrukcija?
11. Kaip skirstomi HTML dokumentus sudarantys žymėjimo elementai?
12. Kokios keturios pagrindinės dalys sudaro HTML dokumentą?
13. Ar yra skirtumas interneto naršyklei, jeigu HTML dokumente bus parašyta `<body>`, `<BODY>` ir `<Body>`?
14. Kokios yra teksto formavimo komandos?
15. Kokio tipo nuorodos naudojamos HTML dokumente ir kaip jos aprašomos?
16. Kaip formuojami sąrašai HTML dokumente?
17. Kaip nurodomos spalvos?
18. Kaip formuojamos lentelės?
19. Kaip įtraukiami grafiniai vaizdai?
20. Kam skirti grafiniai žemėlapiai ir kaip jie formuojami?

21. Kokie yra dinaminio HTML privalumai?
22. Kokios HTML klaidos pasitaiko dažniausiai?
23. Kokie pagrindiniai HTML5 nauji elementai?
24. Kokia CSS standarto paskirtis ir privalumai?
25. Kaip CSS įterpiamas į HTML?
26. Kokiais prioritetais naršyklėje taikomi stiliai?
27. Kaip formuojamos lentelės su CSS stiliais?
28. Kaip vykdoma svetainių atitiktis standartams testavimas ir taisymas?
29. Rasti internete nedidelės apimties netvarkingą HTML dokumentą ir iš jo padaryti tvarkingą HTML dokumentą panaudojant CSS stilius.
30. Pasirinkta tema sukurti bent iš trijų skyrių susidedantį dokumentą tvarkinga HTML kalba:
 - Temų pavyzdžiai: „Biografija“, „Hobis“, „Gyvūnai“, „Gimtasis miestas“.
 - Patikrinti HTML kokybę, ištaisyti rastas klaidas.
 - Sukurtą dokumentą papildyti CSS stiliais.
 - Patikrinti HTML kokybę, ištaisyti rastas klaidas.

4. TINKLALAPIŲ KŪRIMO PRIEMONĖS IR VALDYMAS

4.1. „JavaScript“ skriptų kalba

„JavaScript“ – orientuota skriptų programavimo kalba, besiremianti prototipų principu. Dažniausiai kalba naudojama internetinių puslapių interaktyvumo realizacijai, bet taip pat naudojama ir kaip galimybė skriptais manipuluoti tam tikromis programomis. Kalba sukurta 1995 metais Brendano Eicho „Netscape“ kompanijoje ir pavadinta „Mocha“, o vėliau pervadinta „LiveScript“, galiausiai tapusi „JavaScript“. Vienas iš argumentų pervadinant kalbą buvo sintaksinis panašumas su „Java“ kalba.

„JavaScript“ yra skriptų kalba, kuri suteikia tinklalapiui gyvumo ir veikia daugelyje platformų. „JavaScript“ yra „clientside“ kalba, skirta darbui naršyklėje, todėl yra glaudžiai susijusi su HTML. Sintakse „JavaScript“ yra panaši į C, „Perl“, „Java“. Nors „JavaScript“ ir „Java“ kalbų pavadinimai panašūs, pačios kalbos nėra susijusios – abi kalbos perėmė C kalbos sintaksę, bet semantiškai jos labai skiriasi, taip pat visiškai nesuderinami yra jų objekciniai modeliai.

Po „JavaScript“, kaip svetainių skriptų rašymo kalbos sėkmės, „Microsoft“ sukūrė suderinamą kalbą „JScript“, kurios palaikymas įdiegtas jau „Internet Explorer“ naršyklės 3.0 versijoje, 1996 metų viduryje. Vėliau abi šios kalbos apjungtos į vieną „ECMAScript“ standartą. Nors „JavaScript“ ir „JScript“ sintaksiškai ir semantiškai suderinamos, naršyklės palaiko skirtingus dokumento objektinius modelius (DOM), dėl to skriptas, veikiantis vienoje naršyklėje, nebūtinai veiks ir kitoje.

„JavaScript“ kalbos sintaksė perimta iš C kalbos, su kitais komponentais bendraujama per sąsajas (dokumento objektinį modelį), palaikomas „Unicode“, reguliarios išraiškos (angl. *regular expressions*), taip pat teksto vykdymas naudojant „eval“ funkciją.

Paprasčiausias „JavaScript“ pavyzdys gali būti svetainėje esantis laikrodis, kuris kaskart atnaujina laiką. „JavaScript“ galima rašyti tiesiog pačiame HTML dokumente arba atskirame faile, o HTML dokumente, kuriame norima

matyti rezultata, užtenka nuorodos į dokumentą saugantį kodą. Paprastai „JavaScript“ kalbos kodas įtraukiamas į HTML puslapius, tokiu būdu išplečiant statinius HTML puslapius dinaminio skripto funkcionalumu – galimas anketų parametrų tikrinimas, naujų langų atidarymas, suskleidžiamos hierarchinės struktūros rodymas, išsiskleidžiantis meniu ir daug kitų interaktyvumo formų. „JavaScript“ kalba remiasi kelios pagrindinės svetainių kūrimo metodologijos – DHTML, AJAX, SPA. „JavaScript“ taip pat naudojamas įvairiuose įrankiuose – pavyzdžiui, „Adobe Acrobat“ ir „Adobe Reader“ programos leidžia naudoti skriptus PDF faile.

4.1.1. Pagrindiniai „JavaScript“ elementai

Nematomų simbolių (tarpo simbolis, „Tab“ simbolis, naujos eilutės simboliai) naudojimas šiek tiek skiriasi nuo C kalbos, nes čia tokie simboliai gali tiesiogiai veikti semantiką. Naudojama „kablitaško įterpimo“ technologija, t. y. bet kuri pilnai suformuota eilutė laikoma užbaigta – taip, lyg eilutės gale būtų padėtas kablitaškis. Tokiu būdu net neatskiriant atskirose eilutėse esančių sakinių kablitaškiais, skriptas bus sėkmingai vykdomas. Tačiau programuotojams visgi rekomenduojama tvarkingai dėti kablitaškus, kad pagerintų kodo skaitomumą ir išvengtų šalutinių kablitaškių įterpimo technologijos efektų.

Kintamieji yra dinaminų tipų, nebūtina kintamųjų aprašyti prieš naudojant. Funkcijoje išreikštinai (naudojant „var“ bazinį žodį) aprašytų kintamųjų galiojimo erdvė yra ribota šiai funkcijai, kiti kintamieji yra globalūs.

Kintamieji aprašomi, naudojant „var“ išraišką. Jeigu kintamasis aprašytas funkcijoje – jis yra lokalus (reikšmė išlieka funkcijos ribose, išnyksta, kai pasibaigia funkcijos veikimas), jeigu už funkcijos ribų – globalus dokumentui (reikšmė išlieka dokumento ribose, išnyksta jį uždarius). Duomenų tipas priskiriamas automatiškai, priskyrus reikšmę:

```
var a;
var b;
var c = true; //loginis tipas
a = 10; //sveikas skaičius
b = "10"; //tekstas
```

Išraiškos visada yra baigiamos kabliataškiu, eilutės pabaigoje gali būti rašomi komentarai, kurie rašomi po // ženklų. Bloko komentavimui galima naudoti /* ir */ kaip ir C++ kalboje.

Su kintamaisiais galimos operacijos:

```
a++ ; a=a+1; //padidėjimas vienetu,
a-- ; a=a-1 ; //sumažinimas vienetu,
a = !a //neigimas (NOT),
a = c //priskyrimas,
a+=c; a=a+c; //sudėtis, eilučių sujungimas,
a-=c; a=a-c; //atimtis,
a*=c; a=a*c //daugyba,
a/=c; a=a/c //dalyba,
a%=c; a=a%c dalybos likutis.
```

Matematiniai operatoriai

Matematiniais veiksmais atlikti su duotais kintamaisiais, skirti aritmetiniai operatoriai. Standartiniai aritmetiniai operatoriai yra: Sudėtis (+), Atimtis (-), Daugyba (*), Dalyba (/).

15 lentelė. **Matematiniai operatoriai**

Operatorius	Aprašymas	Pavyzdys	Rezultatas
+	Sudėtis	x=2; y=2; x+y;	4
-	Atimtis	x=5; y=2; x-y;	3
*	Daugyba	x=5; y=4; x*y;	20
/	Dalyba	x=20; y=4; x/y;	5
++	Didinimas vienetu	x=5; x++;	6
--	Sumažinimas vienetu	x=5; x--;	4

Palyginimo operatoriai

Žemiau pateiktoje lentelėje surašyti visi esami palyginimo operatoriai kurie naudojami „JavaScript“.

16 lentelė. **Palyginimo operatoriai**

Operatorius	Aprašymas	Pavyzdys
==	Lygybė	5==8 grąžins false
===	Lygybė (tikrinamos abi reikšmės ir tipai)	x=5 y=„5“ x==y Grąžina true x===y Grąžina false
!=	Nėra lygu	5!=8 grąžinama true
>	Daugiau negu	5>8 grąžina false
<	Mažiau negu	5<8 grąžina true
>=	Daugiau negu arba lygu	5>=8 grąžina false
≤	Mažiau negu arba lygu	5≤8 grąžina true

Loginiai operatoriai

Pateiktoje lentelėje surašyti visi loginiai „JavaScript“ operatoriai.

17 lentelė. **Loginiai operatoriai**

Operatorius	Aprašymas	Pavyzdys
&&	ir	x=6 y=3 (x < 10 && y > 1) grąžina true
	arba	x=6 y=3 (x==5 y==5) grąžina false
!	ne	x=6 y=3 !(x==y) grąžina true

„JavaScript“ kalba visi tinklalapio elementai sudėti tam tikra hierarchine tvarka. Kiekvienas elementas yra tam tikras objektas. Kiekvienas objektas turi tam tikrus metodus ir savybes. „JavaScript“ elementai sudaro kalbos pagrindą. Jie naudojami sudarant programos logines struktūras ir atliekant tam tikrus veiksmus.

Baziniai „JavaScript“ žodžiai:

```
break false if null var continue
for in true while else
function new type of with
```

Break – naudojamas veiksams nutraukti. Dažniausiai naudojamas blokuose „for“, „for...in“, „while“, kai reikia nutraukti ciklo vykdymą, t. y. išeiti iš ciklo jo neįvykdžius iki pabaigos:

```
for (I=0;I<=19;I++)
{If (I= =10) break;
document.write("I="+I"<BR><");}
//programa po break toliau vykdoma nuo čia
```

(61)

Continue naudojamas cikluose „for“, „for...in“, „while“. Šie elementai nurodo, kad iteracija stabdoma ir pereinama prie kitos iteracijos:

```
for (I=0;I<=15;I++)
{if (I= =10) continue;
document.write("I="+I+"<BR>");
}
```

(62)

Ciklas **for** naudojamas cikliniuose procesuose:

```
for (kintamasis= pradinė_reikšmė; tenkinimo
salyga; žingsnis) {
}
```

Pavyzdžiui, 100 kartų dokumente atspausdina paveiksluką:

```
for (var a=1 ; a<101; i++) {
document.write ("<img src = 'a.giff'");
}
```

(63)

Kitame pavyzdyje ciklas su nutraukimu „break“, kai $i \leq 5$:

```
for (var i=1; ; i+=10) {
document.write(i);
if ( i<= 5){
break;
}
}
```

(64)

For...in dažniausiai naudojamas kai reikia sužinoti įvairias objektų savybes.

Pavyzdžiui, kaip nagrinėjamą objektą nurodžius „navigator[i]“, bus pateikiamos naršyklės savybės:

```
function Test()
{
for (i in navigator)
{
document.write(i+"="+navigator[i]+"<BR>");
}
}
```

(65)

Ciklas **while** vykdomas tol, kol tenkinama sąlyga:

```
WHILE (sąlyga) {
išraiškos
}
```

(66)

Pavyzdžiui:

```
var n;
var x = "labas";
var substringas;
while (n<=x.length) {
substringas = x.substr(n,n)
document write("<P>" +substringas + </p>)
n++;
}
```

(67)

If...else tai sąlyginis sakiny, vartojamas tokiems veiksams, kurie priklauso nuo kokios nors sąlygos. Jei sąlygos sakinio loginė reikšmė yra „true“ (teisinga), atliekami veiksmai, aprašyti bloke „if“; jei loginė reikšmė „false“ (klaidinga), atliekami sakiniai bloke „else“.

Sąlygos gali būti rašomos dviem būdais. Pirmu atveju, jeigu sąlyga tenkinama, vykdomos išraiškos 1, jei ne – išraiškos 2. Kitu būdu kintamajam galima priskirti reikšmę: kai sąlyga tenkinama, priskiriama „true“ reikšmė, kai ne – „false“ reikšmė:

```

if (sąlyga){
išraiškos1
}
else{
išraiškos2
}

```

(68)

a = (sąlyga) ? true reikšmė : false reikšmė

Sąlygoje naudojami šie operatoriai:

```

== lygu,
!= nelygu,
< mažiau,
> daugiau,
<= mažiau arba lygu,
>= daugiau arba lygu,
&& konjunkcija (AND), turi būti tenkinamos visos
išvardintos sąlygos,
|| disjunkcija (OR), turi būti tenkinama bent
viena iš išvardintų sąlygų.

```

New sukuria naują tam tikros klasės egzempliorių. Jis naudojamas dviem būdais:

1. kintamajam priklausančiam standartiniam „JavaScript“ objektui sukurti;
2. kintamajam priklausančiam vartotojo objektui sukurti:

```
var now = new Date;
```

Return naudojamas funkcijos reikšmei gražinti. Pavyzdžiui, „return false“; gražinama funkcijos reikšmė FALSE.

Var – kintamajam aprašyti. Galima nurodyti ir kintamojo reikšmę:
var kintamojoVardas;

Valdymo struktūros

„JavaScript“ sąlyginių bei išrinkimo sakinių, taip pat ciklų „for“, „while“ bei „do .. while“ sintaksė analogiška C kalbos sintaksei. Taip pat igyvendintas „for .. in“ ciklas, supaprastinantis objekto atributų perrinkimą, naudojant išraišką „for“ (kintamasis „in“ objektas).

while – tai ciklas, kuris vykdomas tol, kol ciklo kintamojo reikšmė bus „true“. Ciklo sakiniai rašomi skliaustuose {}:

```
while (sąlyga) {
  Ciklo sakiniai
  .....
}
```

(69)

With naudojamas supaprastinti įvairius veiksmus su objektų savybėmis:

```
with (math) {
  alert (pi)
  alert (round(123.456))
}
```

(70)

Switch naudojamas, kai sakiny s gali įgyti keletą reikšmių, priklausomai nuo to atliekami tam tikri veiksmai.

```
switch (sakiny s) {
  case reikšmė1:{
    //veiksmai atliekami tada, kai sakiny s įgyja
    reikšmę, lygia label1
  }
  case reikšmė2:{
    //veiksmai atliekami tada, kai sakiny s įgyja
    reikšmę, lygia label2
  }
}
```

(71)

```
}  
default:{  
  //veiksmai atliekami tada, kai sakinyse neįgy-  
  ja nė vienos nurodytos reikšmės  
}
```

4.1.2. Standartiniai „JavaScript“ objektai

„JavaScript“ kalbos objektai suteikia galimybę susieti savybes su reikšmėmis. Yra keletas bazinių objektų (masyvas, loginiai kintamieji, data, funkcija, skaičius, matematinių operacijų klasė, reguliarios išraiškos, tekstinės eilutės). Kiti objektai aprašomi vykdymo metu.

Objektai aprašomi sukuriant konstruktoriaus funkciją. Kadangi „JavaScript“ kalba remiasi prototipais, yra objektų (ne klasių) paveldėjimas. Objektai paveldi savybes iš prototipų, bet galima vienam ar kitam objektui dinamiškai pridėti savybių (metodų ar kintamųjų), taip pat ir panaikinti savybes. Norint pridėti savybę visiems to tipo objektams, reikia ją pridėti prototipe, kitaip tai galios tik konkrečiam objektui, bet ne tipui.

Objektų naikinimą reguliuoja šiukšlių surinkimo mechanizmas, dėl to nebūtina rūpintis objektų šalinimu pabaigus naudoti.

Objektas **Boolean** naudojamas kitų tipų reikšmėms keisti į logines. Tokiam egzemplioriui sukurti naudojamas konstruktorius „new“. Metodas **valueOf** gražina loginio kintamojo reikšmę, o metodas **toString** – eilutės tipo reikšmę.

Objektas **Date** – naudojamas įvairiems veiksams su laiku ir datomis. Objektas turi daug metodų datai nustatyti, reikšmėms gauti.

Objektas **Function** naudojamas, kai reikia sukurti „JavaScript“ kodo eilutę, kuri skaitoma kaip funkcija. Kaip ir įprasta, funkcija gali turėti keletą parametrų ir vieną reikšmę. Aprašant parametrus nebūtina nurodyti kintamųjų tipų. Naujiems objektams kurti naudojamas konstruktorius „new“.

Objektas **Math** turi visą rinkinį savybių (matematinės, konstantos) ir metodų (trigonometrinės ir matematinės funkcijos). Kaip ir kiti objektai, jis gali būti panaudojamas atskirai ir kartu su kitais elementais.

Objektas **Number** naudojamas, kai reikia gauti įvairias skaitines reikšmes: didžiausias skaičius kompiuteryje, begalybė, t. y. konstantos turinčios tipą „not-a-number“ (NaN).

Objektas **String** – naudojant jį ir jo metodus galima dirbti su eilutėmis. Objektas turi nemažai metodų, skirtų atlikti įvairius veiksmus su eilutėmis, ir vieną savybę (angl. *length*), pagal kurią galima sužinoti eilutės ilgį.

Objektas **Navigator** naudojamas, kai reikia sužinoti informaciją apie naudojamą naršyklės modelį ir kitas savybes.

Objektas **Object** pagal „ECMAScript“ standartą garantuoja bendrą visų „JavaScript“ objektų funkcionavimą.

Pavyzdys:

```
// konstruktoriaus funkcija
function Taškas(x, y) {
    this.x = x;
    this.y = y;
}

// objekto sukūrimas
obj = new Taškas(12, 1000);

// atributo naudojimas
alert(obj.x);

// kitas būdas naudoti objekto atributus kaip      (72)
žodyno elementą
alert(obj["attributeA"]);

// dinamiškai pridėdamas atributas
obj.laikas = new Date();

// atributo išmetimas
delete obj.x;

// objekto sunaikinimas
delete obj;
```

Paveldėjimas ir prototipai:

```
function Base() {
    this.Override = function() {
        alert("Base::Override()");
    }

    this.BaseFunction = function() {
        alert("Base::BaseFunction()");
    }
}

function Derive()
{
    this.Override = function() {
        alert("Derive::Override()");
    }
}
Derive.prototype = new Base();
```

(73)

4.1.3. „JavaScript“ funkcijos

Function – naudojamas tam tikriems veiksams ir objektams sudaryti. Pati funkcija aprašoma tinklalapio antraštėje, po žodžio „function“ nurodomas funkcijos pavadinimas. Funkciją sudarantys operatoriai rašomi figūriniuose skliaustuose. Išskviečiama iš tinklalapio, nurodant pavadinimą ir skliaustus šalia funkcijos pavadinimo:

```
function Funkcijos_pavadinimas()
{
    Funkcija sudarantys operatoriai...
}
```

(74)

Kaip ir daugelyje kalbų, „JavaScript“ funkcijos – tai galimai parametrizuoti kodo blokai, galintys gražinti reikšmę. Funkcijos aprašymo pavyzdys:

```
function funkcijosVardas(arg1, arg2, arg3)
```

(75)

```

{
    sakiniai;
    return reikšmė;
}

```

„arg1“, „arg2“, „arg3“ – funkcijai perduodami argumentai arba parametrai. „Return“ reikšmė – funkcijos grąžinama reikšmė. Funkcija parametrus ir grąžinamų reikšmių gali ir neturėti.

Kiekviena funkcija yra objektas, kurio tipas yra „Function“, todėl bet kuria funkcija galima manipuluoti kaip ir kitais objektais, pavyzdžiui, naudoti kaip kitos funkcijos parametras. Kviečiant funkciją, nebūtina perduoti tiek parametrus, kiek išvardinta funkcijos aprašyme – neperduotų argumentų reikšmės bus neapibrėžtos.

Įvykiai

- „OnMouseOver“ – pelės žymeklio patalpinimas, užvedimas ant elemento,
- „OnMouseOut“ – pelės žymeklis palieka elementą,
- „OnMouseDown“ – bet kurio pelės klavišo paspaudimas,
- „OnMouseUp“ – bet kurio pelės klavišo atleidimas,
- „OnMouseMove“ – pelės žymeklio judėjimas ant elemento,
- „OnClick“ – kairiojo pelės klavišo paspaudimas,
- „OnDbClick“ – dvigubas kairio pelės klavišo paspaudimas,
- „OnKeyPress“ – klavišo paspaudimas ir atleidimas,
- „OnKeyDown“ – klavišo paspaudimas,
- „OnKeyUp“ – klavišo atleidimas.

Pavyzdžiai:

- Informacija apie dokumento paskutinio modifikavimo datą (kodas – „head“ dalyje naudojama funkcija, kuri iškviečiama mygtuko paspaudimu):

```

<html><head>
<script language="JavaScript">
<!--

```

(76)

```
function Labas() {
alert ("Puslapis paskutinį kartą buvo
taisytas: " +document.lastModified);
}
--> </script>
</head> <body>
<form> <input type="button" name="mygtukas"
value="Paspausk!" onClick="Labas()">
</form></body></html>
```

- Elemento spalvos keitimas, paspaudus pele ant elemento:

```
<html><head></head>
<body>
<p onClick="this.style.color='blue';"> Tai
dar tik pradžia </p>
</body>
</html>
```

(77)

- Teksto šrifto pavertimo keitimas, užvedus pele ant elemento ir paliekant elementą:

```
<html>
<head>
<script language="JavaScript">
function keistil (){
pirmas.style.fontStyle = "Italic";
}
function keisti2(){
pirmas.style.fontStyle = "Normal";
}
</script>
</head>
```

(78)

```
<body>
<p id="pirmas" style="font-
weight:normal;color:blue"
onMouseOver=" keistil ();"
onMouseOut =" keisti2 ();"> Tai tik
pradžia</p>
</body>
</html>
```

- Informacija apie elementą, kuriam generuojamas įvykis „onclick“:

```
<html>
<head></head>
<body onclick=
lick="alert(window.event.srcElement.id);">
<p>
<div id="Tekstas" >tekstas</div>

tekstas šiaip
</p>
</body>
</html>
```

(79)

- Funkcijos gali turėti argumentus. Čia sukuriami paveikslukai – objektai tam, kad iš anksto būtų įkeliami į kliento kompiuterį. Po to funkcija, pagal paveiksluko vardą „htmlNAME“ pakeičia jį į kitą paveiksluką „image“. Funkcijos naudoja 2 argumentus:

```
<html>
<head>
<script language="JavaScript">
<!--
if (document.images) {
pirmyn1 = new Image(90,25);
pirmyn1.src = "pirmyn.gif";
pirmyn2 = new Image(90,25);
pirmyn2.src = "pirmyn_x.gif";
atgal1 = new Image(90,25);
atgal1.src = "atgal.gif";
atgal2 = new Image(90,25);
atgal2.src = "atgal_x.gif";
}
function pav(htmlName,image) {
if (document.images) {
document.images[htmlName].src = eval(image +
".src")
}}
-->
</script></head><body>
```

(80)

```
<a href="pirmyn.html"
onMouseOver="pav('pirmyn','pirmyn2')"
onMouseOut=" pav ('pirmyn','pirmyn1')">
</a>
<br>
<a href="atgal.html"
onMouseOver="pav('atgal','atgal2')"
onMouseOut="pav('atgal','atgal1')"> </a>
</body>
</html>
```

- Puslapyje pateikiama data:

```
<html> <head> </head>
<script language="JavaScript">
function fulltime() {
var time=new Date();
document.clock.full.value=
time.toLocaleString();
setTimeout('fulltime()',500) ;
}
</script>
<body > <form name=clock>
<input type=text size=17 name=full>
</form>
<script language="JavaScript">
fulltime();
</script></body> </html>
```

(81)

- Laikrodis:

```
<html><head>
<script language="JavaScript">
var timer=null;
var timerrun=false;
function stoptime() {
if(timerrun)
clearTimeout(timer);
timerrun=false;
}
function starttime() {
```

(82)

```
stoptime();
showtime();
}
function showtime() {
var all=new Date();
var hours=all.getHours();
var minutes=all.getMinutes();
var seconds=all.getSeconds();
var timevalue=" " + ((hours>12) ? hours-12 :
hours)
timevalue += ((minutes<10) ? ":0" : ":") +
minutes
timevalue += ((seconds<10) ? ":0" : ":") +
seconds
timevalue +=(hours>=12) ? "P.M." : "A.M."
document.clock.next.value=timevalue;
timer=setTimeout('showtime()',1000);
timerrun=true;
}
</script>
<body bgcolor=ffffff text=ff0000 onLo-
ad="starttime()">
<center>
<form name=clock>
<input type=text name=next size=12 value=' '>
</form>
</body></html>
```

- Bėgantis tekstas būsenos eilutėje:
-

```
<html><head>
<script language="javascript">
function statusmessageobject(p,d) {
this.msg = message
this.out = " "
this.pos = position
this.delay = delay
this.i = 0
this.reset = clearmessage
}
function clearmessage() {
this.pos = position
}
```

(83)

```
var position = 100
var delay    = 40
var message  = "Labas rytas "

var scroll = new statusmessageobject()
function scroller() {
  for (scroll.i = 0; scroll.i < scroll.pos;
  scroll.i++) {
    scroll.out += " "
  }
  if (scroll.pos >= 0)
    scroll.out += scroll.msg
  else scroll.out = scroll.msg.substring(-
  scroll.pos,scroll.msg.length)
  window.status = scroll.out
  scroll.out = " "
  scroll.pos--
  if (scroll.pos < -(scroll.msg.length)) {
    scroll.reset()
  }
  setTimeout ('scroller()',scroll.delay)
}
</script></head> <body onLoad="scroller()">
</body></html>
```

4.1.4. Masyvai

Masyvai

Tipinė duomenų struktūra – masyvas, kuriame skaičiai (indeksai) susiejami su reikšmėmis. Galimybė susieti skaičių su reikšme yra ir kituose tipuose, bet masyvai turi ir specializuotą funkcionalumą (suliejimas, pridėjimas į galą ir pan.), taip pat masyvai turi masyvo dydį nusakantį atributą (angl. *length*).

Objektas *Array* apima masyvų sujungimo metodus, rūšiavimą ir perstatymą. Masyvas – kintamųjų rinkinys, kuriame svarbi elementų tvarka. Sukurti objektą „Array“ galima dviem būdais:

1. Konstruktoriumi „new“ sukuriamas masyvas, po to nurodomas jo dydis bei jį sudarantys elementai. Pirmiausia aprašomas elementas „mas“, naudojant konstruktorių „new“. Po to kiek-

vienam elementui suteikiama konkreti reikšmė. Panaudojus ciklą „for“, masyvo reikšmės spausdinamos vienoje eilutėje.

2. Kuriant masyvą galima iš karto aprašyti jo reikšmes.

Objektas „Array“ turi tokius metodus:

- **join** sujungia visus masyvo elementus į vieną eilutę;
- **reverse** keičia masyvo elementų tvarką: pirmasis elementas tampa paskutinis, o antrasis – priešpaskutinis ir t. t.
- **sort** rūšiuoja masyvo elementus.

Masyvo konstruktorius

Masyvo konstruktorius sudarytas taip:

```
new Array(masyvoIlgis)
new Array(elementas0, elementas1, ..., ele-      (84)
mentasN)
```

Masyvo elementai sukeliama į patį masyvą tokiu būdu:

```
[elementas0, elementas1, ..., elementasN]
```

Parametrai:

- masyvo ilgis – nustatomas masyvo ilgis. Šią reikšmę galima gauti panaudojus „length“ masyvo parametą. Jeigu nustatyta reikšmė nėra skaičius, tuomet sukuriamas masyvas su vienu įrašu, kur elemento reikšmė bus ta, kuri buvo nurodyta masyvo sukūrimo metu. Maksimalus masyvo ilgis leidžiamas iki 4,294,967,295.
- elementas0, elementas1, ..., elementasN – elementų sąrašas masyve.

Masyvo reikšmės sudarytos iš vieno kintamojo pavadinimo. Patartina nenaudoti nesudaryto masyvo objekto, o naudoti objekto sukūrimą. Pavyzdyje sukuriamas masyvo tiesioginis objektas. Masina kintamasis turi tris elementus ir masyvo ilgis bus 3:

```
var Masina = ["Ford", "BMW", "Opel"];
```

Galima sukurti ir tankų masyvą su dviem, ar keletą kintamųjų elementų pradedant masyvo pirminį raktą nuo 0, ir skaičiuojant didėjančia tvarka kiekvienam elementui. Žemiau pateiktame pavyzdyje sukuriamas tankus masyvas su trimis elementais:

```
var Masyvas = new Array("Labas", Kint, 3.14159);
```

Masyvo raktų indeksai

Kiekvienas elementas turi savo raktus, todėl galima kiekvieną kartą sukūrus masyvą paimti norimą elementą iš masyvo, nepriklausomai nuo to, kokie masyvo elementai dar yra. Pavyzdžiui, pirmiausiai sukuriamas toks masyvas:

```
var Masyvas = new Array("Vejas", "Lietus",  
"Ugnis");
```

Dabar galima kiekvieną elementą išvesti atskirai, pavyzdžiui:

- Masyvas[0] - grąžinamas pirmasis elementas: „Vejas“.
- Masyvas[1] - grąžinamas antrasis elementas: „Lietus“.
- Masyvas[2] - grąžinamas trečiasis elementas; „Ugnis“.

Nustačius vieną skaitinį parametą su „array“ konstruktoriumi, sukuriamas pirminis masyvo ilgis. Pateiktas pavyzdys sukuria masyvą su penkiais elementais:

```
var Masyvas = new Array(5);
```

Masyvo konstruktorius priklauso nuo pirminio skaitinio parametro. Nustačius reikšmę kaip skaičių, konstruktorius paverčia nurodytą skaičių į neaprašytą, 32-bit sveikąjį skaičių ir sugeneruoja masyvo „length“ parametą (masyvo ilgį).

Jeigu reikšmė nurodyta ne kaip skaičius, masyvas sukuriamas su vienu masyvo ilgiu ir nurodytas kintamasis masyvo objekte perkeliamas į elementą, kuriam priskiriamas pirminis raktas. Pateiktame pavyzdyje sukuriamas masyvas su 25 raktais, ir pirmiems 3 raktams priskiriamos reikšmės:

```
var Masyvas = new Array(25);  
Masyvas[0] = "R&B";  
Masyvas[1] = "Blues";  
Masyvas[2] = "Jazz";
```

(85)

Masyvo ilgio netiesioginis didinimas

Masyvo ilgis padidinamas kai įterpiamas koks nors elementas, tada masyvo ilgis tampa didesnis negu esamas masyvo ilgis. Žemiau pateiktas kodas sukuria masyvą su masyvo ilgiu 0, kai įterpiamas elementas į masyvą su raktu 99, tuomet masyvo ilgis gaunamas 100:

```
var spalva = new Array();
spalva[99] = "geltona";
```

Masyvų metodai

Lentelėje pateikti metodai, kurie keičia „JavaScript“ masyvą.

18 lentelė. Masyvą keičiantys metodai

Metodas	Aprašymas
Array.pop()	Grąžinamas paskutinis elementas ir ištrinamas iš masyvo.
Array.push()	Įdedami nauji elementai į masyvą ir grąžinamas naujas masyvo elementų ilgis.
Array.reverse()	Sukeičia elementus vietomis. Pradinis elementas tampa paskutinis ir t. t.
Array.shift()	Pašalina pirmą elementą iš masyvo ir palieka kitus.
Array.splice()	Pridedami arba atimami elementai iš nurodyto masyvo.
Array.sort()	Surūšiuojami masyvo elementai nurodyta tvarka.
Array.unshift()	Pridedami nurodyti elementai į masyvą pradžioje ir grąžinamas bendras jų ilgis.

Pridėtiniai metodai

Šie „JavaScript“ masyvo metodai nekeičia pačio masyvo, o tiesiog vaizduoja masyvą vienu ar kitu atveju.

19 lentelė. Keletas pridėtinių metodų

Metodas	Aprašymas
Array.concat()	Pridedami elementai iš vieno masyvo į kitą. Elementai pradedami nuo masyvo ilgio dydžio.
Array.join()	Sudeda visus elementus esančius masyve į nurodytą kintamąjį.
Array.slice()	Išvedami elementai iš masyvo su nurodytais indeksais.

Data (ang. Date)

„Date“ konstruktorius sudaromas tokiu būdu:

```

new Date()
new Date(milisekundės)
new Date(DatosKintamasis)
new Date(metai, mėnuo, diena
        [, valandos, minutes, sekundės,
        milisekundės])

```

(86)

Datos parametrai:

- milisekundės – skaitinė reikšmė atitinkanti skaičių milisekundėmis nuo Sausio 1 1970 00:00:00 UTC.
- datosKintamasis – kintamasis, kuris atitinka datą. Kintamojo reikšmė turi atitikti datą. Kintamasis turi būti tokio formato, kad atpažintų jį „parse“ metodas.
- metai, mėnuo, diena – skaitinės reikšmės, atitinkančios datos dalis. Kaip skaitinė reikšmė mėnuo atitinka nuo 0 iki 11, kur 0 atitinka Sausį, o 11 – Gruodį.
- valandos, minutės, sekundės, milisekundės – skaitinė reikšmė, kuri atitinka laiko dalis.

Nenustačius jokių argumentų, datos konstruktorius sukuria „Date“ objektą su šia diena ir dabartiniu laiku, kuris yra nustatytas veikiančiame kompiuteryje. Nustačius keletą argumentų, bet ne visus, nerasti argumentai yra nustatomi kaip 0. Tačiau būtina nustatyti bent metus, mėnesį, dieną, galima praleisti tik valandą, minutes, sekundes ir milisekundes. Data nustatyta milisekundėmis

yra nuo 1970 m. sausio 1 d. vidurnakčio. „Date“ objektas veikia nepriklausomai nuo esamos platformos, priima skaičius pagal UTC (universalus) metodus, taip pat puikiai ir lokalų laiko metodą. Lokalus laikas – tai kompiuterio laikas, kuriame vykdomas „JavaScript“ kodas. Dėl suderinamumo su tūkstantmečiu (skaičiuojamas nuo 2000 metų), patartina visada nustatyti tikslią datą (pvz.: naudoti 2014, o ne 14). Žemiau pateiktas kodas gražina intervalo skirtumą tarp „laikasA“ ir „laikasB“ milisekundėmis.

```
laikasA = new Date();
// Šioje vietoje ką nors atliekame.
laikasB = new Date();
skirtumas = laikasB - laikasA;
```

(87)

Pavyzdžiai:

- Keli būdai iškviešti „Date“ objektą:

```
siandien = new Date();
gimtadienis = new Date("December 17, 1995
03:24:00");
gimtadienis = new Date(1995,11,17);
gimtadienis = new Date(1995,11,17,3,24,0);
```

(88)

- Darbas su keliais nustatytais datos parametrais:

```
// Naudojame statinį metodą
var start = Date.now();
// Darome ką nors su puslapiu
DaromeKaNorsIlgai();
var end = Date.now();
var skirtumas = end - start; // Gražinamas
laikas milisekundėmis

// Jeigu mes turime Date objektą
var start = new Date();
// Darome ką nors su puslapiu:
DaromeKaNorsIlgai();
var end = new Date();
var skirtumas = end.getTime() -
start.getTime(); // Gražinamas skirtumas mi-
lisekundėmis
```

(89)

20 lentelė. Keletas datos objekto metodų

Metodas	Aprašymas
getDay()	Gražinama savaitės diena pagal nustatytą datą.
getDate()	Gražinama mėnesio diena pagal nustatytą datą.
getFullYear()	Gražinami dabartiniai metai, kurie nustatyti lokaliai.
getHours()	Gražinamos valandos pagal nustatytą datą.
getYear	Šis metodas nebegalioja ir yra pakeistas „getFullYear“ metodu.
getMilliseconds()	Gaunamos milisekundės pagal nustatytą datą.
getMinutes()	Gaunamos minutės iš nustatytos datos.
getMonth()	Gražinamas mėnesis pagal nustatytą datą.
getSeconds()	Gaunamos sekundės iš nustatytos datos.
setMinutes()	Nustatomos datos objekto minutės.
setMonth()	Nustatomas datos objekto mėnuo.
setSeconds()	Nustatomos datos objekto sekundės.
setTime()	Nustatomas datos objekto laikas.

Dokumentai (ang. *Document*)

„Document“ objektas pateikia tiesioginę prieigą prie HTML, XHTML ir XML dokumentų. „Document“ objektas pagrindinis įrankis yra DOM Core dokumento (anglų kalba) sąsaja. HTML dokumentai taipogi interpretuoja DOM HTMLDocument (anglų kalba) sąsają, kur yra daugiau sąsajų dirbančių su HTML dokumentais.

Pavyzdžiai:

- „document.cookie“ – vykdomas norint gauti „sausainėlius“;
- „document.alinkColor“ – vykdomas norint pakeisti nuorodos spalvą;
- „document.alinkColor“ – gražinama arba nustatoma nuorodos spalva;
- „document.anchors“ – gražinamas sąrašas, kiek dokumente yra inkarų (angl. *anchors*)

Istorija (ang. *History*)

„History“ objektas buvo sukurtas kaip URL masyvas, kuris pateikia pilną naršyklės aplankytų puslapių istoriją. Kai privatumo atskleidimo panaudojimas tapo akivaizdus, visiškas priėjimas prie esančio URL sąrašo buvo uždraustas ir „History“ objektas liko tik su „back()“, „forward()“ ir „go()“ metodais, leidžiančiais naršyklei judėti per istorijos masyvą neatskleidžiant jo turinio.

Pavyzdžiai:

- `history.back(vieta)` – grįžtama per nustatytą vietą atgal.
- `vieta` – nustatytas skaičius per kiek pozicijų grįžti atgal.
- Du būdai nueiti į buvusį puslapį, nenurodant jokios pozicijos ir panaudojus -1

```
<input type="button" value="Atgal"
onClick="history.back()" ">
  <input type="button" value="Atgal -1"
onClick="history.back(-1)" ">
```
- `history.back()` – postūmis per vieną, ar nurodytą poziciją atgal, nuo esamo puslapio.

Langai (ang. *Window*)

Naršyklės, kurios palaiko *tabuliacinius* langus (kiekviena nauja svetainė gali būti atvaizduojama naujame lange), turi savo atskirus „window“ objektus. „Window“ objektas nėra leistinas tarp dviejų *tabuliacinių* langų. Kai kurie metodai, tokie kaip „window.resizeTo“ ir „window.resizeBy“, veikia visuose languose neskaitant tabuliacijos.

21 lentelė. **Keletas „Window“ metodų**

Metodas	Aprašymas
<code>window.alert()</code>	Iškviečiamas pranešimo dialogas.
<code>window.blur</code>	Pašalina esamo lango akcentą.
<code>clearInterval()</code>	Sustabdomas vykdymas pasikartojančiais veiksmais nustatyta funkcija pasitelkiant <code>setInterval()</code> funkciją.
<code>clearTimeout()</code>	Sustabdomas vykdymas pasikartojančiais veiksmais nustatyta funkcija pasitelkiant <code>setTimeout()</code> funkciją.

Metodas	Aprašymas
confirm()	Rodomas pranešimo langas su „OK“ ir „Cancel“ mygtukais.
moveBy()	Perstumiamas naujas langas į naują vietą pridodant nustatytas koordinatas prie jau esamų (pikselių).
moveTo()	Perstumiamas naujas langas į nurodytą vietą.
open	Sukuriamas naujas langas pagal nustatytus kriterijus.
setInterval()	Vykdomas nurodytas kodas ar funkcija pagal nustatytą laiką.
setTimeout()	Vykdomas nurodytas kodas ar funkcija po nurodyto laiko.

4.2. Dažniausiai naudojamos tinklalapių kūrimo priemonės

Plačiausiai paplitusios ir geriausiai žinomos priemonės tinklalapiams kurti yra PHP, ASP.NET, „Java JSP“, „Python“ technologijos.

PHP Hypertext Preprocessor

PHP (<http://www.php.net/>) buvo ir yra viena labiausiai paplitusių tinklalapių programavimo kalbų pasaulyje. Daugelis svetainių yra parašytos būtent šia kalba. Atviras kodas, lankstumas, palaikymas, multiplatformiškumas kaip tik padarė šią kalbą labai patrauklia. Pirmasis šuolis nuo statinio Web 1.0 HTML turinio rodymo iki dinaminio generavimo kaip tik ir prasidėjo su šia kalba. Ji daugeliu atvejų primena C++, kadangi naudojasi panašia sintakse, yra objektiškai orientuota (OOP). Pavadinime „preprocessing“ pabrėžiama, kad kalba yra interpretuojama. Vartotojui paprašius, puslapio procesorius nuskaito HTML puslapį su įterptomis PHP žymėmis (žymėmis <php>), įvykdo kodą generuojantį dinaminį HTML turinį. Dalis našumo dėl to nukenčia, tačiau dėl savo paprastumo kalba yra labai populiari. Papildomai visada patariama naudotis kūrimo karkasais, šablonų bibliotekomis kaip „Dwoo“, „Smarty“, duomenų bazių abstrakcijos bibliotekomis „ADODB“ ir daugeliu kitų jau sukurtų įrankių.

Labiausiai paplitusi žiniatinklio serverių aplinka – „Apache“, veikianti tiek „Linux“ tiek „Windows OS“. Šiuo metu tinklalapių kūrimui populiariu

naudoti LAMP (angl. *Linux+Apache+MySQL+PHP*) arba WAMP (angl. *Windows+Apache+MySQL+PHP*) „kompleksą“, dar naudojami „EasyPHP“, XAMPP ir kt. Tai viskas iš programinės įrangos pusės, ko reikia, norint pradėti kurti tinklalapius su PHP. Kartu gaunama ir „MySQL“ reliacinė duomenų bazė bei dar papildomi įrankiai („phpMyAdmin“ duomenų bazėms administruoti/tvarkyti ir pan.). LAMP sistemos labai paplitusios dėl savo ekonomiško, nemokamų technologijų, todėl daug svetainių talpinimo įmonių kaip tik ir suteikia panašaus pobūdžio komplektaciją ir resursus už tikrai pigią kainą.

ASP.NET

ASP.NET (angl. *Active Server Pages*) yra tinklalapio struktūros technologija parduodama „Microsoft“, kurią programuotojai gali naudoti norėdami sukurti dinaminę internetinę svetainę, žiniatinklio konstrukciją arba paslaugą. Tai dalis „Microsoft .NET“ platformos „Microsoft“ *Aktyvių Serverio Puslapių* (angl. *Active Server Pages – ASP.net*) technologijos įpėdinis. ASP.NET integruota su *Bendros kalbos išpildymo aplinka* (angl. *Common Language Runtime – CLR*), leidžiančia programuotojams rašyti ASP.NET kodą bet kuria „Microsoft.NET“ kalba.

Principas toks pats kaip ir PHP – HTML kode įterptos žymės, kurias interpretuoja procesorius ir vykdo tarp jų parašytą kodą. ASP technologija veikia išskirtinai „Windows OS“ aplinkoje. Pati kalba senoje ASP versijoje – tiesiog „JavaScript“ rinkinys. ASP trūkumus „Microsoft“ pakeitė naujesne ir aktyviai plėtojama technologija ASP.NET (<http://www.asp.net/>). Pereita prie visiškai objektinių C# ir „Visual Basic“ kalbų, kurios žymiai lankstesnės, taip pat liko palaikoma ir atgaliniu būdu suderinama ASP „JavaScript“ atmaina. Naujas „Code Behind“ principas iškelia HTML ASP kodą (dizainą) į vieną failą, o C# kodą į atskirą, todėl dizaineris ir programuotojas gali dirbti vienu metu – taip įvedama papildoma tvarka. Nuo PHP skirtumas toks, jog svetainės kodas gali būti ir sukompiliuotas į bibliotekas – DLL (angl. *dynamic link library*) ir vykdomas tiesiogiai centriniame procesoriniame įrenginyje (angl. *CPU – Central Processing Unit*) komandų lygiu („native“ aplinkoje), kas teoriškai turėtų suteikti daugiau našumo, bet visada taip pat priklauso ir nuo realizacijos. Taip pat yra realizuoti standartiniai komponentai, validavimas, kurie palengvina ir pagreitina kūrimo procesą.

Komponentinis kūrimas – analogija su standartinių „Windows“ formų kūrimu, todėl tokie dalykai kaip „login“ formos ir kita – jau realizuota „out-of-the-box“, telieka įdėti komponentą į tinklalapį. Norint kurti ASP.NET puslapius susidaro išpūdis, kad net nebūtina gerai žinoti HTML, kadangi daugelis HTML žymių tiesiog pakeičiamos standartine ASP žyme, kuri atitinka komponentą (pvz.: <asp:TextBox> žymė pakeičia HTML <input type=„text“> ir pan.). Serveris, apdorojęs puslapį, automatiškai pakeičia ASP žymes į atitinkamas HTML, kurias jau gali suprasti naršyklė. Puslapį galima įsivaizduoti kaip vieną didelę HTML formą, kadangi dažniausiai visi komponentai yra sudedami vienoje HTML puslapio formoje. Taip siekiama nuo visiškai būsenos neišsaugojančio (angl. *stateless*) HTML pereiti prie būseną išsaugojančios, nuoseklaus „su atmintimi“ puslapio koncepcijos. ASP taip pat palaiko šablonus (angl. *templates*), kurie realizuoti per „MasterPage/ChildPage“ puslapių struktūrą ir leidžia įtraukti vieną šabloną į kitą ir pan.

Susidūrus su ASP.NET matyti, jog XML taikomas labai plačiai – ypač aplikacijose ir serverio konfigūracijoje. XML prideda lankstumo, kaip gali būti sukonfigūruota tinklalapio aplikacija, tačiau kartais gali pasirodyti ir jo perteklius.

Pradėti programuoti ASP.NET aplikacijas galima parsisiuntus „Visual Studio“ įrankį. „Express“ versija prieinama nemokamai (<http://www.microsoft.com/visualstudio/en-us>). Tačiau, pirmiausia reiktų išmokti C# programavimo kalbą siekiant toliau įžengti į ASP.NET teritoriją. Sukurtas aplikacijas (puslapius) dažniausiai naudoja verslo sektorius (angl. *enterprise*), todėl ASP.NET prieglobos paslaugų firmų Lietuvoje nėra tiek daug, o pačios talpinimo paslaugos yra pakankamai brangios vien dėl to, jog „Windows“ licencijos kainuoja. Sukonfigūruoti savo serverį taip pat galima, jeigu prieinama „Windows Server OS“, kuri kartu jau integruoja ir IIS (angl. *Internet Information Services*), reikalingą ASP.NET veikimui. Įdiegus lieka tik pasirinkti ir nustatyti, jog norima sukurti žiniatinklio serverį. Lietuvoje dažniausiai ASP.NET labiau naudojamas didesnio verslo ar tarptautinio lygio svetainėms kurti.

„JavaEE JSP“

Nuo žiniatinklio sprendimų neatsilieka ir „Java“ plėtojanti JSP (angl. *Java Server Pages*) platforma „JavaEE“ tinklalapių aplikacijoms kurti. Kūrimo principai panašūs kaip ir prieš tai minėtų technologijų: aplikacija susideda iš

HTML JSP puslapių, kuriuose <jsp> žymomis aprašytas „Java“ kodas. Klientui pareikalavus puslapio „Java“ VM (virtuali mašina) apdoroja tai, kas parašyta ir sugeneruoja dinaminį turinį. Patys JSP puslapiai nėra atskirai talpinami serveryje, o supakuojami į taip vadinamą WAR (angl. *Web Application Archive*) failą, kuris yra JAR failas su papildomais MANIFEST failais ir papildoma XML konfigūracija. JSP puslapių alternatyva – „Java Servlet“, klasės, kurios turi apibrėžtą API ir apdoroja užklausas.

Bet kokio „Java“ kodo vykdymas leidžia serveryje išnaudoti visas „Java“ galimybes: panaudoti EJB (angl. *Enterprise Java Beans*) komponentus, multiplatformiškumą, prisijungti „legacy“ sistemas ir kt. „Java“ aplikacijos ypač paplito tarptautiniame versle ir taikomos gan rimtoms užduotims spręsti. Taip pat „JavaEE“ nė kiek nenusileidžia spartos atžvilgiu nuo kitų technologijų ir yra naudojamos realaus laiko situacijoms apdoroti, valdyti. Taip pat plėtojamas „Java Server Faces“ karkasas grafinėi sąsajai kurti ir t. t.

Pradėti kurti „JavaEE“ aplikacijas nėra labai sudėtinga, tereikia įdiegti „JavaEE“ palaikančių serverių, kurių yra nemažas pasirinkimas. Labiausiai paplitę ir naudojami „Apache Tomcat“, „JBoss“, „GlassFish“, „Spring“ bei „IBM WebSphere“ aplikacijų serveriai. Jie realizuoja aplikacijos konteinerį, kuriame ir veikia sukurta sistema. Taip pat serveriai turi valdymo bei monitoringo priemonės, konsoles. Pavyzdžiui, „Apache Tomcat“ pradeda veikti ypač greitai – per 2-3 sekundes bei pateikia neperkrautą ir patogų valdymo meniu, daug pavyzdžių. Masyvesnis variantas – „JBoss“, kuris parašytas „Apache Tomcat“ pagrindu, skirtas ypač rimtiems tarptautinio verslo projektams realizuoti.

Tinklalapių prieglobos paslaugos paremtos „JavaEE“ technologija kainuoja nemažai ir jų nėra labai daug (ypač retai pasitaiko Lietuvoje), kadangi nėra tiek paplitusios kaip, pavyzdžiui, PHP ir dažnai neįtraukiamos į paslaugų kompleksą. Vis dėlto, jei sprendimas reikalauja tokių technologijų, galima pasinaudoti dedikuotų serverių paslaugomis ar pan.

„Python/Ruby“

Šios pilnai objektinės OOP (angl. *Object-oriented programming*) kalbos greitai įgavo populiarumą dėl savo lankstumo ir paprastumo. Joms yra nemažai karkasų, kurie leidžia realizuoti norimas tinklalapių aplikacijas. „Python“ kalbai – „Django“, „Pylons“, „Grok“, „Zope“, o „Ruby“ – „Ruby on Rails“, „Ramaze“, „Merb“, „Halcyon“, „Wuby“ ir t. t. išties nemažai. Kuriant tinklala-

pius su „Python“ ir paėmus „Django“ (jį turi ir naudoja „Google AppEngine“), iškart gaunama šablonų sistema, standartinės formos, MVC (angl. *Model–View–Controller*) ir duomenų bazė „out-of-the-box“, belieka tik viską teisingai sukonfigūruoti. Pats puslapių apdorojimas skaidomas pagal MVC principus į „Controller“ klases – pirmiausia sukuriama norimo puslapio klasė, kuri paveldi bazinę klasę ir „index() metode“ apdorojama užklausa generuojamas dinaminis turinys. Taip sukurti puslapiai yra suregistruojami „Django“ sistemos sukūrimo metu, nurodant, koku keliu norima puslapį pasiekti. Principai nesudėtingi, gaunamos visos „Python“ funkcijos ir kalbos įmantrybės, todėl tinklalapių kūrimas tampa dar įdomesniu.

Pradėti kurti naudojant tokias sistemas nesudėtinga – tereikia įsidiegti „Python“ ar „Ruby“ ir parsisiųsti pateikiamą serverio sistemą. Pats HTTP serveris yra realizuojamas atitinkamos kalbos konstrukcijomis sukuriant specializuotą HTTP serverį. Šiomis sistemomis dirbančių žmonių skaičius kol kas nėra didelis, nes sudėtinga rasti tam tikslui specializuotą svetainių talpinimo serverį. Išėitis gali būti dedikuoti serveriai arba paieškos giganto „Google“ siūloma „AppEngine“ paslauga.

Kažkurios kalbos mokėjimas – tai tik tinklalapių kūrimo pradžia, o šių kalbų panaudojimo išmanymas – jau technologiniai dalykai. Interneto svetainių kūrimas yra labai plati ir nuolat tobulėjanti sritis. „jQuery“, „JavaScript“, CSS, AJAX, „Flash“, „Silverlight“ ir kitų technologijų naudojimas šiandien jau laikomas reikalavimu dinaminiam, išspūdingiems, konkurencingiems bei pilnavertiškiems puslapiams kurti.

Svetainių kūrėjai rūpinasi adresų užsakymu, svetainės patalpinimu serveryje, el. pašto dėžutės sukūrimu, ryšio su socialiniais tinklais integravimu („Facebook“, „Twitter“, „Google+“ ir t. t.).

4.3. PHP programavimas

PHP (angl. *PHP hypertext preprocessor*) – tai skriptinimo kalba (angl. *scripting language*), kuri pačioje pradžioje buvo orientuota tik į internetą, nors šiuo metu ją galima „drąsiai“ pavadinti programavimo kalba, kadangi ją naudojantis galima programuoti ne tik internetui. Pavyzdžiui su „php-gtk“ galima kurti pilnavertiškas „cross“ platformines programas su GUI (angl. *graphical*

user interface) bei konsolines programėles „mail wrapperius“ ir t. t. PHP skriptai yra interpretuojami ir įvykdomi serverio pusėje.

PHP skirtumas nuo „JavaScript“ yra tai, jog PHP skriptai yra atliekami serverio pusėje, o vartotojui yra gražinamas rezultatas HTML pagalba. Rezultatą vartotojas mato savo naršyklėje. Tuo tarpu, kai „Java Script“ yra pilnai perduodamas kliento kompiuteriui ir atliekamas toje dalyje.

PHP privalumai:

- visiškai nemokamas;
- tinka visoms platformoms;
- atviro kodo projektas;
- veikia daugelyje žiniatinklio serverių: „Apache“, IIS, PWS, „OmniHTTP“, „BadBlue“ ir t.t.
- labai lengva išmokti PHP programavimo pagrindus;
- pasižymi dideliu greičiu serverio pusėje bei dirbant su duomenų bazėmis;
- nedideliuose projektuose PHP paprastai galima įterpti į HTML dokumentą;
- kadangi PHP programuotojų yra be galo daug, daugumą jau parašytų skriptų galima rasti internete: „HotScripts“, „FreeScripts“, „PHPClasses.upperdesign.com“ ir t. t.

PHP skripto pavyzdys:

```
<html>
<head>
  <title>Pavyzdys</title>
</head>
<body>
  <?php echo " PHP skripto pavyzdys"; ?>
</body>
```

(90)

4.3.1. Žiniatinklio serveris „Apache“

„Apache“ serveris yra reikšmingas žiniatinklio plėtrai. „Apache“ buvo viena pirmųjų alternatyvų „Netscape Communications Corporation“ tinklo serveriui, dabar žinomam „Sun Java System Web Server“ vardu. Savo funkcionalumu ir našumu „Apache“ konkuruoja su kitais „Unix“ operacinės sistemos pagrindu kuriamais tinklo serveriais (<http://www.apache.org/>).

Projekto vardas pasirinktas iš pagarbos Amerikos indėnams apačiams, kurie yra gerai žinomi dėl savo išstvermingumo ir kovos įgūdžių. Projekto pavadinimą nulėmė ir tai, kad jis buvo kurtas iš NCSA HTTPd 1.3 pataisų serverio, kurį dar vadino „a patchy“.

Į „Apache“ tinklo serverį įtraukti moduliai, kurie išplečia serverio funkcionalumą – serverių programavimo kalbos (mod_perl, mod_python, Tcl, ir PHP), moduliai (mod_access, mod_auth ir mod_digest), SSL ir TLS (mod_ssl), tarpinio serverio modulio aptarnavimas, nuorodų (URL) redaktorius, duomenų filtravimo galimybės. „Apache“ pranešimų rinkmenos gali būti analizuojamos naudojant tinklo naršyklę ir kitas laisvai platinamas programas.

„Virtual hosting“ funkcija leidžia vienai „Apache“ programai aptarnauti kelias internetines svetaines. Pavyzdžiui, vienas kompiuteris su viena „Apache“ programa gali vienu metu aptarnauti kelias svetaines.

„Apache“ tinklo serveris naudojamas statinio ir dinaminio turinio svetainėms žiniatinklyje publikuoti. Dauguma internetinių programų yra sukurtos „Apache“ serverio pagrindu. „Apache“ naudojamas užduotims atlikti, kai turinys turi būti pasiekiamas saugiu ir patikimu būdu, pavyzdžiui, dalintis failais per internetą. „Apache“ tinklo serveris yra įtrauktas į populiarią LAMP tinklo serverio programų rinkinį.

„Apache“ HTTP serveris gali būti naudojamas pagal „Apache“ licenciją ir tai yra laisvoji atvirojo kodo programinė įranga. „Apache“ licencija leidžia naudoti programinę įrangą tiek atvirojo, tiek uždarojo kodo programoms.

Kuriant „Apache“ programinę įrangą siekiama sukurti tvirtą pagrindą įvairių tipų programinei įrangai taikyti. Kuriama platforma patikimoms sistemoms įvairiems uždaviniams spręsti. „Apache“ kūrėjų manymu, internetinio turinio viešinimo įrankiai turi būti visiems laisvai pasiekiami, o programinės įrangos bendrovės turi uždirbti kurdamos pridėtinės vertės paslaugas. Žiniatinklio protokolai, būdami atviri ir nepriklausomi, sukuria vienodas sąlygas dide-

liems ir mažiems rinkos dalyviams. Kita priežastis – tai noras įtraukti į programinės įrangos kūrimą jos vartotojus. Tas, kas sumokėjo už programinę įrangą, paprastai nenorės taisyti jos klaidų nemokamai.

„Apache“ turi dvi vartotojų grupes. Šis vartotojų pasiskirstymas nulėmė „Apache“ sėkmę dar iki laisvųjų programų bumo pradžios. Milijardai dokumentų, atverčiamų vartotojų naršyklėse, yra pateikiami „Apache“ programinės įrangos. Tokie vartotojai nedaro įtakos serveryje veikiančios programinės įrangos pasirinkimui. Dauguma iš jų nesidomi, kokia programinė įranga jiems tiekia dokumentus. Šie vartotojai sukuria didžiulę rinką. Didžiuliai resursai skiriami kuriant jiems skirtus produktus.

Kiti vartotojai – specialistai. Kiekvieną įdiegtą serverį tvarko bent vienas specialistas – jo administratorius, kuris išmano savo valdomą techniką, gilinasi į jos veikimo subtilybes. „Apache“ augimui didelę įtaką turi dvi tokios situacijos pasekmės. Specialistas yra pajėgus pasirinkti optimalų techninį sprendimą. Jis remiasi ne reklama, o techniniais kriterijais. Tai mažina nuosavų konkurentų investicijų į rinkodarą efektyvumą, daug sunkiau išstumti gerą produktą iš informuotų specialistų rinkos. Antroji pasekmė – sistemą eksploatuoja specialistai, siekiantys ir sugebantys ją tobulinti, kvalifikuotai padedantys kūrėjams.

4.3.2. Duomenų bazių valdymo sistema „MySQL“

Kiekviena informacinė sistema turi išsaugoti kažkokius duomenis. Kiekvienas patyręs programuotojas nesunkiai sukurs bet kokių duomenų įrašymo rinkmenose sistemą. Tačiau tokią sistemą optimizuojant didesniems duomenų kiekiams ir didesniems srautams tektų sugaišti nemažai laiko. Todėl geriau duomenų valdymą patikėti profesionalų sukurtai ir milijonų vartotojų išbandytai duomenų bazių valdymo sistemai. LAMP komplekte paprastai siūloma naudotis „MySQL“ – tai viena iš reliacinių duomenų bazių valdymo sistemų (liet. *RDBVS*, angl. *RDBMS*), palaikanti daugelį naudotojų, valdoma SQL kalba.

„MySQL“ yra atvirojo kodo programinė įranga (GPL ir kitos licencijos), vystoma ir palaikoma švedų kompanijos „MySQL AB“, kurios įkūrėjai –

švedai David'as Axmark'as, Allan'as Larsson'as ir suomis Michael'as „Monty“ Widenius.

2008 m. vasario 26 d. akcinę bendrovę „MySQL“ įsigijo korporacija „Sun Microsystems“, kuri anksčiau jau buvo įsigijusi „StarOffice“, pagal kurią sukūrė ir tobulina laisvąjį „OpenOffice.org“. „Sun Microsystems“ remia ir kitus laisvuosius atvirojo kodo produktus.

„MySQL“ galimybės

Kaip ir kiekvienos RDBVS, „MySQL“ duomenys prieinami per abstrakčias lenteles ir ryšius tarp skirtingų lentelių ar jų dalių. Duomenims įvesti, keisti, ieškoti bei lentelėms ir duomenų bazei valdyti yra naudojama SQL (angl. *Structured Query Language*) kalba.

„MySQL“ RDBVS veikia daugelyje platformų, ji dažnai pasirenkama programuojant internetines svetaines. Šiame sektoriuje su „MySQL“ bando konkuruoti ir „PostgreSQL“.

Pastaruoju metu „MySQL“ vis dažniau pritaikoma labai didelėse informacinėse sistemose. Pavyzdžiui, „Vikipedija“, kurios apkrova kartais viršija 10 tūkstančių užklausų per sekundę, arba vienas iš didžiausių JAV kabelinės televizijos tinklų „Cox Communications“, kurių duomenų bazėje – daugiau kaip 3 600 lentelių. Šiame sektoriuje pagrindinis „MySQL“ konkurentas yra „Oracle“.

Nors prieigai prie „MySQL“ duomenų bazių dažniausiai pasirenkama PHP kalba, ją taip pat galima pasiekti įvairiomis kitomis programinėmis priemonėmis: C, C++, C#, „Java“, „Perl“, „Python“ ir kitomis. Kiekvienai šių kalbų sukurtos specialios bibliotekos. Taip pat „MySQL“ duomenų bazėms yra sukurta ODBC sąsaja „MyODBC“, leidžianti duomenis pasiekti bet kuria kalba, neturinti specialios bibliotekos, tačiau palaikančia ODBC komunikavimo mechanizmą.

4.3.3. „MySQL“ valdymo įrankis phpMyAdmin

PHP kalba parašytas „MySQL“ valdymo įrankis „phpMyAdmin“, skirtas administruoti „MySQL“ duomenų bazes naudojantis naršykle nežinant SQL komandų bei tiesiogiai nesijungiant prie serverio.

1998 m. „MySQL“ sąsają PHP kalba pradėjo kurti Tobias Ratschiller, informacinių technologijų konsultantas ir kompanijos „Maguma“ įkūrėjas. Nuo

2000 m. jis nebedirbo prie projekto, bet „phpMyAdmin“ jau buvo tapęs populiariu „MySQL“ administravimo įrankiu su didele vartotojų ir tobulintojų bendruomene (<http://www.phpmyadmin.net/>).

Pastaruoju metu programa išversta į daugiau nei 50 kalbų, tarp jų ir į lietuvių. „phpMyAdmin“ yra įprastas LAMP komplekto elementas. Kalbant apie LAMP sudėtį „phpMyAdmin“ dažnai nepaminimas, nes jis turimas omenyje kaip „MySQL“ komponentas, nors ir yra atskiras kūrinys.

Prieš keletą metų „MySQL“ buvo platinama pagal dvi licencijas. Vienos sąlygos – asmeniniam naudojimui, kitos – įmonėms. Vėliau AB „MySQL“ paskelbė, kad visiems vartotojams jų pagrindinė licencija yra GNU GPL ir kad jie norėtų paskatinti kiekvieną skelbti savo programinę įrangą su šia licencija. Taip pat AB „MySQL“ siūlo naudotojui pasirinkti priimtinas sąlygas iš kitų licencijų sąrašo.

Norintys panaudoti „MySQL“ programinę įrangą sau priklausančiuose produktuose, gali pasirinkti tinkamą komercinę licenciją iš „OEM Embedded Database Products“ variantų sąrašo. Tai modernus verslo modelis, kai mokama už licencijos suteikiamas teises, bet ne už techninius sprendimus. Perkantiems ir nemokamai naudojantiems prieinami tie patys programos komponentai.

Ne visiems atvirojo kodo šalininkams tinkama GNU GPL licencija. Atvirojo kodo iniciatyvos (OSI) licencija yra artima GNU GPL, tačiau yra skirtumų, neleidžiančių GNU GPL išvestinio kūrinio platinti su OSI licencijomis. „MySQL“ nėra platinama su OSI licencija, tačiau yra parengta speciali GNU GPL su išimtimis, kurios leidžia išvestinius kūrinius platinti su OSI licencija.

Taip „MySQL“ programinė įranga platinama pagal visą licencijų įvairovę: nuo GNU GPL iki nuosavų, nesudarydama kliūčių kiekvienam naudotojui pasirinkti optimalų variantą.

4.3.4. Laisvasis įrankis PHP

PHP yra tinklalapių programavimo kalba (<http://php.net/>). „Apache“ modulis įvykdo PHP kalba parašytą skriptą ir vartotojui pateikia HTML dokumentą. Laisvai platinama daugybė programų PHP kalba, todėl informacinių sistemų kūrėjai paprastai nekuria programų nuo pirmos eilutės, bet ieško jau

sukurtų panašios paskirties programų ir patobulina jas bei pritaiko savo reikmėms.

Šis įrankis puikiai tinka ir yra dažnai naudojamas įvairios apimties projektams. PHP kūrėjų bendruomenė yra puiki terpė įvairių tipų atviriesiems projektams.

Maža universalių programų vartotojų dalis yra peržiūrinėjusi savo naudojamų programų pradinį tekstus. Dar mažesnė dalis yra juos redagavusi. Tokie vartotojai paprastai pasinaudoja nedidele sudėtingos programos funkcijų dalimi. Sudėtingesnėms internetinėms informacinėms sistemoms paprastai naudojama specializuota programinė įranga arba konkrečioms poreikiams pritaikyta universali programinė įranga. Abiem atvejais sistemą kurti turi mokantis programuoti specialistas. Žiniatinklio eros pradžioje daug programų tekdavo programuoti nuo pirmos eilutės. Viešai paskelbtuose programų tekstuose naudotojai gali ne tik pastebėti programos klaidas, bet ir pasiūlyti, kaip jas galima būtų ištaisyti. Tai natūralus bendradarbiavimas, nereikalaujantis jokių specifinių licencinių sutarčių. Pirminės programos kūrėjas nebūtinai gaus patobulintą versiją, tačiau patobulintos versijos autoriai turi analogiškus motyvus viešai atskleisti savo programas. Dauguma atvirųjų licencijų reikalauja, kad patobulinta programa būtinai turi išlikti atvira. Taigi šiam motyvui formaliai padeda atvirosios licencijos.

Daugelį PHP programų kuria pavieniai programuotojai. Jie turi galimybę parodyti savo darbą, rasti bendraminčių, kuriančių panašias programas visame pasaulyje. Tokių programų užsakovai dažnai nėra informacinių technologijų įmonės ir jie neturi komercinio intereso padaryti programą nuosava. Pradiniai kodo fragmentai dažnai platinami su licencijomis, neleidžiančiomis uždaryti išvestinio kūrinio.

Interaktyviose sistemose (pvz., el. parduotuvėse) vartotojo matomas vaizdas priklauso nuo jo atliktų veiksmų, todėl turi būti kuriamas programos. Net ir statinėse sistemose (pvz., el. laikraščiuose) patogiau sukurti kuo paprastesnę sąsają, kuri pateikia straipsnius į duomenų bazę. Tada programa kuria vaizdą ekrane prie straipsnio pridėdama kitus komponentus. Bet kokios programos išvedamus duomenis būtų galima perduoti „Apache“ serveriui, kuris juos pateiktų vartotojui. Tuo tikslu buvo sukurta CGI-BIN sąsaja. Žiniatinklio užduotims spręsti paplito „Perl“ kalba, turinti patogių eilučių apdorojimo įrankių, kurie dažnai praverčia formuojant HTML.

Vėliau buvo sukurtas „Apache“ PHP modulis. Sukurtos PHP kalbos sintaksė mažai skyrėsi nuo „Perl“. PHP tapo integruotas „Apache“ komponentas. Maždaug 2000 metais ne mažai el. sistemų buvo perrašyta iš „Perl“ į PHP. Tai suteikė ne tik techninių privalumų, bet ir lėmė patogesnę programavimą, nes pasitaikius klaidai PHP modulis galėjo pateikti išsamią informaciją, o „paki-bus“ „Perl“ programai „Apache“ serveris nežinodavo problemos priežasčių, todėl negalėdavo apie tai pranešti vartotojui.

Kaip veikia PHP modulis

„Apache“ serverio aplanke gali būti įrašomi ne tik HTML failai, kurie pateikiami vartotojui tokie, kokie yra, bet ir PHP failai, kurie perduodami PHP moduliui. Tokiu atveju PHP modulis analizuoja tekstą ir apdoroja tai, kas yra tarp ženklų „<?“ ir „?>“. Likusi dalis perduodama vartotojui nepakeista. Tekstą nuo „<?“ iki „?>“ PHP modulis interpretuoja kaip programą PHP kalba. Jei faile yra keli tokie intarpai, jie interpretuojami kaip viena programa, išsaugomos kintamųjų reikšmės. Pateikiamo dokumento PHP programos vietoje išvedamas programos pateikiamas tekstas.

Sudėtingesniuose projektuose failas dažnai prasideda ženklu „<?“ ir baigiasi „?>“, visą HTML tekstą kuria PHP programa. Į statinius puslapius kartais įterpiamos atskiros PHP komandos.

Programavimo kalbų „Perl“ ir PHP sintaksė

Labiausiai matomas sintaksės skirtumas tarp aptariamų ir senesnių kalbų yra kintamųjų pavadinimai. C ir kitų klasikinių kalbų kintamųjų pavadinimai būdavo sudaromi iš raidžių ir skaičių. „Perl“ ir PHP kalbose kintamojo pavadinimo pradžioje rašomas ženklas „\$“. Dėl šio skirtumo programos tekstas C ir PHP kalbomis atrodo labai skirtingas, tačiau „Perl“ ir PHP perėmė daugelį C kalbos konstrukcijų.

Priskyrimo sakiny:

```
$a = $b = 500;
```

Čia sujungti du priskyrimai. Šis sakiny iliustruoja iš C kalbos perimtą konstrukciją, kai priskyrimo sakiny turi reikšmę, analogišką priskirtajai, todėl ši reikšmė gali būti dar kartą priskiriama.

Aritmetiniai veiksmai:

```
$n = 2 * $a + $b / 300;
```

Rezultato išvedimas:

```
echo $n;
```

„Perl“ ir PHP turi sudėtingus eilučių apdorojimo mechanizmus.

Taip pat šiomis kalbomis patogiau atlikti paprasčiausius veiksmus – kurti naują eilutę įtraukiant kintamųjų reikšmes.

```
echo "Kintamųjų reikšmės yra: $a, $b ir $n";
```

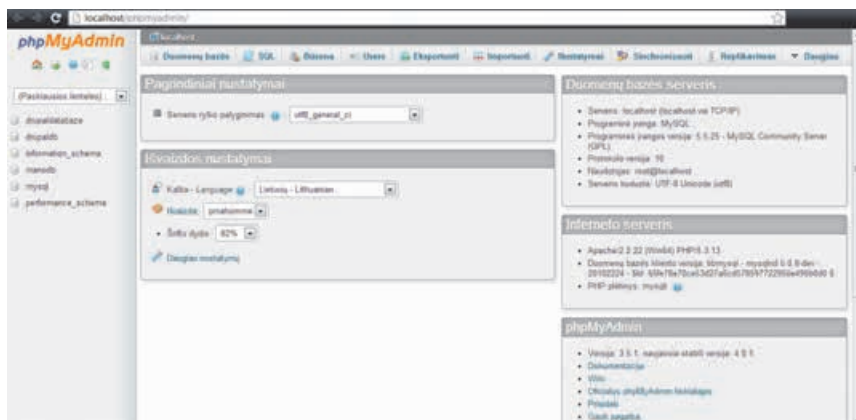
Programinė įranga įkeliama į serverį, kurį prižiūrinčios įmonės darbuotojai gali peržiūrėti serveryje esančių failų turinį. Mažesniuose projektuose programinė įranga ir serveris užsakomi iš skirtingų asmenų, todėl programos kūrėjas sunkiai galėtų apsaugoti programos pradinį tekstą. Kai kurie programuotojai šią problemą sprendžia apsunkindami trečiųjų šalių galimybes pasinaudoti jų programa. Dauguma užsakovų keliamų uždavinių nėra unikalūs. Ne vienas programuotojas jau yra kūręs ką nors panašaus į užsakomą programą, todėl prieš pradėdamas programuoti šioje srityje, verta paieškoti, kokios programos jau yra sukurtos. Ypač daug programuotojų visame pasaulyje kuria PHP programas. Internete laisvai prieinami ne tik programų tekstai, bet ir knygos apie PHP.

Reikalingų programų paketas:

- „Apache“ serveris, <http://www.apache.org>
- „MySQL“ duomenų bazių apdorojimo sistema, <http://www.mysql.com>
- „phpMyAdmin“ duomenų bazių administravimo sistema, <http://phpwizard.net/phpMyAdmin>
- PHP, <http://www.php.net>

4.3.5. PHP „MyAdmin“ duomenų bazių administravimo sistema

„phpMyAdmin“ – nemokama programa sukurta „php“ pagrindu ir skirta „MySQL“ duomenų bazių administravimui. „phpMyAdmin“ palaiko didelį kiekį įvairių operacijų. Dažniausiai naudojamos lentelių kūrimo, elementų priskyrimo ir panašios komandos, kurios pasiekiamos per vartotojo meniu.



11 pav. Pagrindinis „phpMyAdmin“ vartotojo sąsajos langas

„phpMyAdmin“ turi didelę dokumentaciją ir vartotojų kiekį, kurie gali dalintis žiniomis apie duomenų bazių kūrimą. Taip pat labai gerai išvystyta pagalbos vartotojams sistema. Siekiant palengvinti vartotojų darbą, pati programa buvo išversta į daugybę kalbų, tarp kurių yra ir lietuvių kalba.

4.3.6. „MySQL“ duomenų bazių apdorojimo sistema

„MySQL“ – viena iš reliacinių duomenų bazių apdorojimo sistemų (angl. RDBMS – *Relational Database Management System*). Duomenų bazių sistema yra skirta tvarkyti, rūšiuoti ir manipuluoti informaciją. Kadangi „MySQL“ duomenų bazė leidžia įtraukti, keisti ir ištrinti informaciją iš kelių lentelių vienu metu pagal nurodytus kriterijus, ji priskiriama reliacinėms duomenų bazėms (angl. *relational*).

„MySQL“ sistemoje duomenys yra talpinami lentelėse. Kiekviena lentelė yra suskirstyta į eilutes ir stulpelius. Kiekviena eilutė atitinka vieną įrašą. Įrašas gali turėti kelių rūšių informaciją, kuri priklauso nuo stulpelių tipų ir pavadinimų. Duomenų bazėje privalo būti pirminis raktas, kuris gali būti vienintelis atributas arba gali būti sudėtinis raktas (raktas su keliais laukais), o visi rakto atributai turi turėti reikšmes (negali būti pirminio rakto reikšmė NULL). Dažniausiai pirminiu raktu duomenų bazėje yra priskiriami ID laukelio duomenys, kuriuos pagal nutylėjimą priskiria automatiškai įvedant duomenis.

4.4. Optimizavimas paieškos sistemoms

Sukūrus ir patalpinus internetinę svetainę, apie ją žinos tik nedidelė dalis tiek esamų, tiek potencialių vartotojų. Gera pozicija „Google“ paieškos sistemoje – daugiau lankytojų, būsimų klientų, kurie lems pardavimus ir pelną. SEO pagrindinis uždavinys yra pasiekti ir atvesti į svetainę tikslinę auditoriją pasinaudojus paieškos sistemomis.

Pagrindinis paieškos tikslas – gauti informaciją atitinkančią užklausą. Paieškos užklausa gali turėti įvairias formas. Vienas pagrindinių marketingo strategijų elementų – tai kurti interneto rinkodaros sistemą, paremtą paieškos svetainių reitingu ir suprantant tikslinės auditorijos psichologiją. Tam būtina suprasti, kaip „ieškotojas“, o ir ypač tikslinė rinka, naudoja paieškos sistemas tam, kad tiksliau pasiektų ir laikytų naudotojus.

Paieškos sistemos tobulėja jau daug metų, bet pirminiai principai atliekant paiešką liko nepakitę. Pirmaujančių pozicijų nepalieka „Google“ ir „Yahoo“, todėl šių paieškos sistemų įvertinimas tinklalapiams yra labai svarbus.

4.4.1. Tinklalapių reitingavimas

„Google“ vertina svetainę bei nustato jos atitiktį paieškai keliais būdais. Vienas iš pagrindinių, tai „puslapio veiksnio“ metodas, t. y. peržiūri ar tinkamai panaudoti raktažodžiai visose strateginėse svetainės puslapių vietose. Jie būtini tam, kad „Google“ žinotų apie ką ši svetainė ir galėtų ją pateikti paieškos sistemų rezultatuose, remdamasi būtent šiais raktažodžiais. Strateginės

puslapio vietos minimos šios – antraštė, žymos, vidinės nuorodos, išorinės nuorodos, „inkaro“ tekstas, paryškintas paverstas tekstas, HTML sąrašai, ALT žymos, paveikslėlių pavadinimai, dinaminiai „duonos trupiniai“, pavadinimas, aprašymas, raktiniai žodžiai, antraštės, turinys, puslapio pavadinimas.

Kitas ne mažiau svarbus metodas – tai vidinių bei išorinių nuorodų talpinimas savo bei kitose aukšto reitingo, būtinai tokius pat raktažodžius turinčiose, svetainėse.

Paieškos varikliai uždirba pinigus rodydami skelbimus. Daugeliu atvejų, tai yra visas jų pelno modelis. Vadinas, kad siekdami užsidirbti, jie turi parodyti skelbimus kaip galima didesniam žmonių kiekiui. Todėl paieškos sistemos yra suinteresuotos teikti geriausius ir tinkamiausius paieškos rezultatus.

4.4.2. Teisingų raktažodžių parinkimas

Kiekvienas paieškos sistemos optimizavimo ekspertas žino, kad lemiamą reikšmę puslapių optimizavime atlieka raktažodžių parinkimas. Nepaigailėjus laiko iškart surasti gerus tikslinius raktažodžius, vėliau bus pasiektas aukštas svetainės reitingas paieškos sistemose:

- puikiai optimizuotas tinklalapis, nes rašytojai bei kiti turinio kūrėjai jausis patogiai naudodami gerai parinktus raktažodžius, įterpdami juos į naujai kuriamą tekstą;
- daug daugiau skaitytojų pamatys tinklalapio aprašą, nes raktažodžiai puikiai atitiks svetainės turinį;
- daug daugiau lankytojų apsilankys svetainėje, nes tiksliniai raktažodžiai pritrauks tikslią auditoriją.

Raktažodis negali būti sudarytas tik iš vieno žodžio, nes jis yra konkurencingas ir abstraktus, t. y. tinkamas įvairiose nekonkrečiose srityse, o tai ap sunkina paieškos sistemos uždavinį pateikti tikslingą informaciją. Pavyzdžiui, raktažodis „nekilnojamas turtas“ tinkamas tik kai prekiaujama nekilnojamoju turtu visame pasaulyje, kitu atveju didelis kiekis lankytojų užeis į svetainę bei ieškos tai, ko nėra pasiūlyta. Todėl tokius raktažodžius galima įterpti tik tuo atveju, jei padės sustiprinti kitų raktažodžių visumą.

2-3 žodžių frazės atsiųs didžiausią lankytojų srautą. Raktiniai žodžiai atspindi populiarias frazes, o jų kiekis turėtų būti apie 10-20, žinoma, kiekis

nuolat didės kartu su svetaine. Kiekvienam raktažodžiui taip pat svarbi žodžių daugiskaita ir vienaskaita.

Išvestiniai raktažodžiai – tai žodžiai su vienoda šaknimi, pavyzdžiui, laivas, laivininkystė, burlaivis ir t.t. Į tai reikėtų atkreipti dėmesį, nes reikia naudoti tokias frazes, kurias lankytojai naudoja, tačiau nebūtina panaudoti visas įmanomas šaknies versijas, jei jos neatitinka svetainėje esamo turinio.

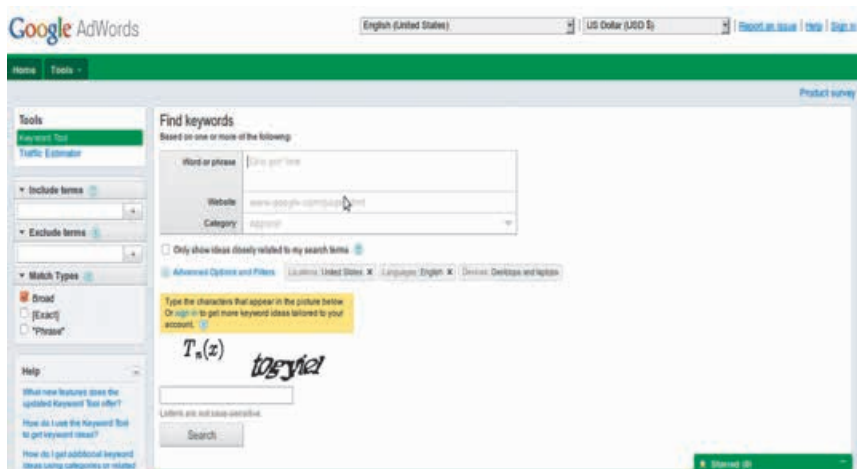
Raktažodžiai su klaidomis, tai kita panaši problema, nes ne visada pavyksta įvesti paieškos žodžius be klaidų, nors ir tikimasi teisingų rezultatų. Todėl raktažodžių su bendrai daromomis klaidomis įrašymas padės pritraukti daugiau lankytojų į svetainę.

4.4.3. Raktiniai žodžiai „Google“ įrankių pagalba

Nors internete gana daug raktažodžių paieškos programų, tai daugiau menas nei mokslas. Geriausias būdas pradėti nuo tų, kurie akivaizdūs ir ypač aktualūs svetainės turiniui t. y. informacijai, parduodamai prekei ar paslaugai. Turint tokius raktažodžius, likęs procesas taps lengvesniu.

Kita svarbi raktažodžių grupė yra terminai, kurie naudojami aprašyti dalykus panašius, bet nebūtinai aprašomus svetainėje. Pavyzdžiui, jei raktinis žodis yra „SEO optimizavimas“, susijęs terminas galėtų būti „paieškos sistemos marketingas“. Šie terminai stiprins svetainę, kai bus įtraukiamos vidinės bei išorinės nuorodos (dvipusė reklama).

Rasti raktažodžius susijusius su svetainės terminais arba iš raktažodžių sąrašo, galima naudojant „Google AdWords“ raktažodžių įrankį (12 pav.).



12 pav. Raktinių žodžių įrankis „AdWords“

„Google AdWords“ įrankis skirtas reklaminių vietų pagal raktažodžius paieškai, bet jis taip pat puikiai tinka raktažodžių tyrimui ir išaiškinimui. Įrankis nustato paieškos kiekį specifiniam žodžiui ar frazei. Tačiau dažnai gaunamas klaidingas rezultatas vien tik todėl, kad nemokama ar nesugebama panaudoti skirtingų programos paieškos filtrų.

Labai svarbu, ieškant raktažodžio ar frazės, nustatyti tinklapio kalbą ir regioną, nes kitu atveju bus gauti duomenys tik nurodyta kalba iš pažymėtos valstybės ar šalies. Tuo atveju jei veikla vykdoma kitoje šalyje, įjungus papildomą paiešką galima teritoriją ar kalbą patikslinti. Pažymėjus „Match Types“ gaunamas tikslus užklausų kiekis tam tikram regionui.

Filtras „Frazė“ (angl. *Phrase*) parodo, kokiuose išsireiškimuose (frazėse) šis žodis naudojamas. Jei reikalinga tiksli informacija, kiek paieškų atliekama būtent konkrečiam žodžiui, visada naudojamas pažymėtas tikslus (angl. *Exact*) langelis. Kadangi kai kurie terminai ieškomi globaliu mastu, „Google“ diferencijuoja paiešką į globalią paiešką ir lokalią (vietinę) paiešką. Globali paieška suprantama kaip paieška visame pasaulyje per paskutinius 12 mėnesių.

Antraštės aprašymas – tai antraštės santrauka, kurioje turi būti raktažodžių frazė. Ta pati frazė turi būti panaudota antraštėje ir aprašyme.

Puslapio turinys yra labai svarbi paieškos sistemų optimizavimo dalis. Daugelis paieškos variklių, tokių kaip „Google“, išskiria turinio tekstą juostele

ir naudoja jį ne tik paieškos rezultatuose, bet naudoja svetainės subjekto bei objekto nustatymui, nustato kokie žodžiai ar frazės naudojamos dažniausia puslapi nustatant, kurie žodžiai ir frazės naudojami labiausiai, ir priskiria puslapio temą. Tokiu būdu „Google“ atranda tinkamiausius puslapius paieškos rezultatams pateikti. Turinio rašytojas kiekviename puslapyje turi įterpti raktažodžius. Raktažodžių tema turi eiti per antraštinę dalį, aprašymą, raktinius žodžius, santrauką, turinį, nuorodas ir pavadinimą. Raktažodžių temos pritaikymui, reikia laikytis individualių raktažodžių, skirtų būtent tam puslapiui.

Kiti raktažodžių elementai gali būti įtraukti į navigaciją, naujienų antraštes, dienoraščių antraštes, išorines nuorodas ir reklamą. Kai tai suderinama su „meta“ pavadinimu, aprašymu ir raktažodžiais, atitiktis paieškos sistemų rezultatams padidėja ir svetainė bus rodoma paieškos rezultatų puslapyje aukščiausioje vietoje.

4.4.4. HTML5 gali padėti SEO

1 problema: numeracija

Jei svetainėje yra daugybė puslapių su pakartotinai naudojama informacija (pvz.: tam tikros kategorijos produktų sąrašas), reikalinga indeksacija. Kita vertus, turinio dubliavimas gali lemti nedidelį išduodamos informacijos kiekį.

Čia gali padėti nuoseklios nuorodos (angl. *sequential links*). Pagal standartą, dokumentų seka yra toks dokumentų rinkinys, kai kiekvienas iš jų turi prieš jį einantį ir po jo einantį dokumentą. Dokumentas, be po jo sekančio dokumento, yra sekos pabaiga.

Nuoseklios nuorodos yra realizuojamos naudojant atributą „rel“. Tarkime turima numeracija:

```
< a href = „produktai.php? puslapis = 3“ >  
  Ankstesnis puslapis </a>  
< a href = „produktai.php? puslapis = 5“ >  
  Kitas puslapis </a>
```

(91)

Reikia tik pridėti „rel“ atributus:

```

< a href = „produktai.php? puslapis = 3 rel
    =“ ankstesnis „> Ankstesnis puslapis </a>
< a href = „produktai.php? puslapis = 5“ rel
    = „kitas“ > Kitas puslapis </a>
  
```

(92)

Naršyklei tai sako, kad indeksuojami puslapiai yra susiję. Kai kuriais atvejais šio veiksmo nauda nėra tiksliai žinoma, pvz., jei kalbama apie didelio straipsnio suskaidymą į puslapius. Svarbiausia yra tai jog kuo tiksliau naršyklei aprašomas puslapio turinys, tuo tiksliau naršyklė gali jį įvertinti.

2 problema: puslapio struktūra

Ilgą laiką svarbiausias buvo toks metodas puslapiams sukurti: HTML naudojamas turiniui, CSS – stiliaus sukūrimui ir pristatymui, „JavaScript“ – papildomoms funkcijoms įgyvendinti. Tačiau HTML neužteko išraiškingų priemonių suskirstyti turinį į kategorijas. Problema iš dalies pradėta spręsti naudojant mikroformatus.

Viena iš HTML5 ypatybių – papildomų informacinių žymų rinkinys, kurios leidžia tiksliau patalpinti turinį. Tarp tokių žymų yra <section> <header> <footer> <article> <hgroup> <aside> <nav>.

Pavyzdžiui:

```

< div id = „myarticle“ >
...
</div>
< div id = „extrafacts“ >
...
</div>
  
```

(93)

Virsta į

```

<article>
...
</article>
<aside>
...
</aside>
  
```

(94)

Tokiu būdu naršyklei bus perduota papildoma informacija apie puslapio turinį, į kurį bus atsižvelgiama formuojant fragmentą.

Be to, HTML5 leidžia naudoti kelias H1 informacines žymas viename puslapyje. Todėl galima teisingai pažymėti vienodai svarbią informaciją, esančią tame pačiame puslapyje. Tai realizuojama tuo pat metu kelių naujų žymų deriniu:

```
<section id="finance">
<header>
<h1></h1>
</header>
</header>
...

<footer>
</footer>
</section>
```

(95)

```
< skirsnyje id = „Pramogos“ >
<header>
<h1> </h1>
</header>
...

<footer>
</footer>
```

Naudojant `<section>` turinys dalinamas į skyrius, kiekvienam skyriui suteikiama pirmo lygio antraštė, papildomai pažymėta informacine žyma `<header>`.

Stiliaus formatavimas šiose naujose žymose su CSS pagalba puikiai veikia „Firefox“, „Google Chromo“ ir „Safari“, bet IE vartotojams reikės papildomai įdiegti nedidelį „JavaScript html5shiv“.

3 problema: vidaus paieškos puslapiai

Esant vidaus paieškos mechanizmui, sukurti puslapiai yra uždaromi nuo paieškos robotų „robots.txt“ pagalba, nes pateikiant informaciją beveik visada atsiranda jau turimo tinklalapyje turinio dublikatas. Tačiau daugeliu

atvejų, vidaus paieška yra naudojama kaip navigacijos priemonė. Vidaus paieškos pateikimo indeksacija vis dėlto yra reikalinga.

HTML5 įvertina tai ir siūlo naudoti atributą „rel“ tokiu būdu:

```
< a href = „/ search.php“ rel = „ieškoti“ >
paieška svetainėje</a>
```

Tokiu būdu naršyklei pranešama daugiau informacijos apie indeksuojamo tinklalapio esmę. Be to, vidaus paieškai galima priskirti dar daugiau funkcijų naudojant „OpenSearch“ technologijų rinkinį, kurį palaiko „Firefox“ (2.0), „Google Chrome“ ir „Internet Explorer“ (7.0). Šios technologijos leidžia iš dalies integruoti įtrauktą paiešką svetainėje į naršyklės paiešką.

4 problema: mikroformatai! = schema.org

Mikroformatai ir RDF užtikrina galimybę įterpti į puslapius metaduomenis struktūriškai aprašančius turinį. Tai leidžiama tokioms naršyklėms kaip „Yandex“ ir „Google“ generuoti patrauklesnius fragmentus, turinčius daugiau informacijos.

Optimizuojant pagal „Yandex“, neverta jaudintis, nes „Yandex“ naudoja mikroformatus, o naudojant „Google“, galima panaudoti keletą standartų tuo pačiu metu.

5 problema: AJAX ir URL

AJAX technologija nebuvo suprojektuota atsižvelgiant į SEO reikalavimus, todėl padaryti taip, kad būtų patogų ir vartotojams, ir optimizuotojams yra ne taip paprasta.

Problema iš dalies išsprendžiama „prikabinus“ AJAX kvietimus prie informacinių žymų <a>, kad naršyklė galėtų indeksuoti ieškomą turinį. Tačiau tai yra tik dalinė priemonė, nes gauto turinio adreso neįmanoma nukopijuoti į mainų sritį, norint pasidalinti juo per socialinius tinklus.

HTML5 pagelbės suteikdamas priemones dinamiškam URL pakeitimui naršyklės adreso juostoje neperkraunant puslapio. Tam yra naudojamas `history.pushState()` bei jo metodai `replaceState()` ir `popState()`.

Pavyzdys kaip naudoti šį metodą:

```
var stateObj = {foo: „bar“};  
history.pushState(stateObj, „puslapis 2“,  
„bar.html“);
```

Metodas `history.pushState()` ne tik iš karto keičia URL, bet taip pat ir įrašo jį į naršyklės apsilankymų istoriją taip, kad vartotojas gali naršyti naudodamasis istorija pirmyn ir atgal.

4.5. Tinklalapio šablonai

Tinklalapio šablonas – tinklalapio dizainas, paruoštas naudoti kaip pagrindas sparčiam ir kokybiškam tinklalapio vystymui. Turint šabloną, reikia pridėti tik tekstą ir nuotraukas, kad tinklapis būtų užpildytas informacija. Tinklalapio šablonas gali būti naudojamas tiek asmeniniam tinklapiui, tiek didelės įmonės tinklapiui, e. parduotuvei, ar portalui.

Norint panaudoti tinkamą šabloną svetainei, pirmiausia reikia žinoti iš ko sudaryti šablonai. Dažniausiai šablonuose būna HTML puslapiai ir paveikslėliai, tačiau būna ir rimtesnių šablonų, kurie naudoja įvairius šriftus, stilių failus, „flash“ animacijas ir kitus komponentus. Dažniausiai šablonas būna supakuotas, atsisiųstas šablono ZIP failas išpakuojamas ir pamatoma vidinė jo struktūra. Paprasčiausias šablonas būna toks, kai visi failai sudėti į vieną katalogą. Tai patogiu, jei komponentų būna nedaug, nes viskas vienoje vietoje. Tačiau tai visiškai netinka sudėtingiems ir dideliems šablonams. Tuo tikslu naudojama hierarchinė katalogų struktūra.

Pagrindiniai šablonų naudojimo privalumai:

- Greičiau. Naudojant šabloną tinklalapio kūrimas pagreiteja iki 5-10 kartų. Internete galima rasti daug nemokamų arba už minimalią kainą parduodamų tinklalapių šablonų pagal poreikį.
- Aukštesnė kokybė. Didžioji dalis platinamų šablonų yra sukurta profesionalių dizainerių, kurių darbai pasižymi dizaino kokybe, paveikslėlių suspaudimo lygiu ir optimizuotu HTML

kodu. Pradedantiesiems kūrėjams tinklalapio šablonas labai palengvina darbą.

- Pigiau. Profesionalių dizainerių sukurtas išskirtinis dizainas yra brangus. Tačiau yra publikuojama daug tinklalapio šablonų nemokamai.
- Paprasčiau. Kiekvienas šablonas yra automatiškai ir rankiniu būdu patikrintas ar nėra palikta klaidų bei kaip suformuojamas visas bendras vaizdas. Patį šabloną labai lengva redaguoti. Iškilus klausimams, atsakymą galima rasti viešose diskusijose.

Šablonų tipai:

- Tinklalapio šablonas suprogramuotas ir suskaidytas naudojant HTML kalbą ir PSD formato („Adobe Photoshop“) šaltinių failus. Dauguma šių šablonų susideda iš dviejų puslapių: pradinio ir žemesnio lygio puslapio. Keičiant „kompanijos vardas“, tekstą ir visą turinį bei pridėdant savo paveikslukus, gaunamas profesionaliai atrodantis puslapis.
- CSS šablonas paremtas CSS technologija, kuri leidžia tinklalapių kūrėjams labai lengvai formuoti ir stilizuoti tinklalapį. Naudojant CSS galima nurodyti pagrindinius naudojamo teksto parametrus visam tinklalapiui, redaguoti visą tekstą, grafiką ir nuorodų stilius visuose tinklalapio puslapiuose, keičiant tik stilių lentelės failą.
- Logotipo šablonas. Tai įvaizdžio dalis ir raktas į sėkmę, todėl jo kokybė turi būti itin aukšta.
- „Flash“ animuotas šablonas savyje naudoja „Flash“ animaciją. Šiame šablone naudojama animuota antraštė, kurioje pateikiama patraukli gyvo veiksmo ir tinkamo garso multimedijos kombinacija.
- „Flash“ įvadinė antraštė. Efektyvi „Flash“ prezentacija naudojama perteikti tinklalapio ir veiklos idėją būsimiems tinklalapio vartotojams. Tai lankytojus sužavintis informatyvus vaizdo filmukas.

- „Flash“ tinklalapio šablonas pagrįstas „Flash“ technologijomis, kurios atveria galimybes tinklalapių dizaino srityje. „Flash“ tinklalapis suteikia solidumo ir individualumo.
- PHP-Nuke apvalkalas naudojamas PHP-Nuke generuojamiems tinklalapiams. PHP-Nuke yra automatizuota naujienų pateikimo sistema naudojama internete ir intranete. Naudojimas nereikalauja jokių techninių įgūdžių ir apvalkalo integracija užima vos keletą minučių.
- „Fireworks“ šablonas yra sukurtas naudojant „Macromedia FireworksTM“. „Macromedia Fireworks“ yra grafinis redaktorius, integruotas su kitais „Macromedia“ platinamais produktais.
- 3 spalvų šablonas suteikia galimybę tinklalapiui pasirinkti spalvų schemą. Visi šaltinių failai („html“, „psd“ and „fla“) yra sukurti naudojant 3 skirtingų spalvų schemas, tereikia pasirinkti labiausiai patinkančią spalvų schemą.
- Pilnas tinklalapis – išplėstinis tinklalapio šablonas, kuris turi pilną rinkinį žemesnio lygio tinklalapių su atitinkamais išdėstymais ir paveiksluokais, kurių gali pririnkti.
- Pilnas paketas nėra vien tik tinklalapio dizainas – tai sudėtinis identiško paketas. Jis susideda iš žemesnio lygio puslapių, paieškos rezultatų lango, papildomų iššokančių langų dizaino ir reklaminių antraščių, kurios derinamos prie tinklalapio.

4.6. Turinio valdymo sistemos

Interneto svetainės gali būti sutartinai suskirstytos į dvi grupes – statistines ir valdomas. Statistinėse svetainėse nėra numatomas informacijos atnaujinimas. Tokie projektai vykdomi trumpalaikiam tam tikros prekės reklamavimui. Pavyzdžiui, naujo mobilaus telefono modelio prezentacija interneto svetainėje.

Statistinės svetainės palaikymui reikia kreiptis į jos projektuotoją net įvedant menkiausius pakeitimus, o tai susiję su nuolatinėmis finansinėmis išlai-

domis. Per trumpą laikotarpį statistinės svetainės palaikymo sąnaudos gali ke-
liasdešimt kartų viršyti jos sukūrimo sąnaudas.

Todėl internete atsirado naujas interneto svetainių kūrimo principas,
kurio idėja, atskirtas nuo įforminimo turinys gali būti keičiamas, nesikreipiant į
svetainės projektuotoją. Tam tikslui pradėta naudoti turinio valdymo sistema.
Tokios interneto svetainės vadinamos valdomomis svetainėmis.

Turinio valdymo sistemos (TVS, angl. CMS – *Content Management System*) leidžia administruoti svetainę be programavimo žinių, padeda paskirs-
tyti svetainės priežiūros darbus, automatiškai renka statistiką apie svetainės
populiarumą, veda informacijos pateikimo bei peržiūros apskaitą, padeda suži-
noti svetainės lankytojų nuomonę. Tinklalo turinį galima atnaujinti iš visur
kur yra interneto prieiga, administravimo teisės gali būti suteiktos ir daugiau
žmonių. Įprastai su turinio valdymo sistema galima gauti naujienų modulį, sve-
čių knygą, dažniausiai užduodamų klausimų funkciją, galeriją, formų modulį,
leidžiantį atlikti apklausas, formuoti užsakymus, kontaktuoti su tinklalo admi-
nistratoriumi ir t. t. Svetainę galima papildyti ir reklaminio skydeliu, moduliu ar
tinklaraščiu (angl. *Blog*).

Turinio valdymo sistemos – įvairūs programiniai įrankiai, palengvi-
nantys informacinių sistemų turinio (tekstinį ir grafinį) valdymą taip, kad suku-
riant bei keičiant turinį ar jo struktūrą, nereikėtų jokių specialių kompiuterinių
(ar net programavimo) žinių.

Vienas pagrindinių turinio valdymo sistemos tikslų – užtikrinti pa-
prastą sistemas ar interneto svetainės administravimą. TVS naudojamos ne tik
svetainių palaikymui, bet ir duomenų bazių palaikymui, failų bei dokumentų
apskaitai, informacijos įforminimui intraneto sistemose.

TVS gali būti tiek asmeninėms interneto svetainėms (angl. *Personal TVS*), tiek korporatyvinės turinio valdymo sistemos (angl. *Enterprise TVS*)
didelių portalų valdymui. Funkcijų gausa leidžia pasirinkti labiausiai poreikius
atitinkančias turinio valdymo sistemas:

- Svetainių turinio valdymo sistemos (angl. *Web CMS*) padeda
įmonei automatizuoti įvairius tinklalapių kūrimo, valdymo ir
platinimo (pateikimo) aspektus. Tinklalo pateikimas (inter-
nete HTML) yra pagrindinis formatas, tačiau gali būti WAP,
PDF, spausdinimo režimui ir pan.

- Transakcinės turinio valdymo sistemos (angl. *Transactional CMS*) padeda organizacijoms valdyti e. komercijos veiksmus, operacijas (transakcijas).
- Integruotos turinio valdymo sistemos (angl. *Integrated CMS*) su duomenų bazių valdymo sistema valdo organizacijos dokumentus ir turinį.
- Leidybinės turinio valdymo sistemos (angl. *Publication CMS*) skirtos leidinių (vadovėliai, knygos, vadovai) gyvavimo ciklui (angl. *content life cycle*) valdyti.
- Mokymo turinio valdymo sistemos (angl. *Learning CMS*) organizuoja mokslo programos ciklą (angl. *learning life cycle*) žiniatinklyje.
- Korporatyvinės turinio valdymo sistemos (angl. *Enterprise CMS*) (skirtos įmonėms, organizacijoms) dažniausiai mišrios, nes konkreiti veiklos specifika diktuoja specializuotus funkcinius poreikius, todėl turinio valdymo sistema turi būti atitinkamai suprogramuota.

Atvirojo kodo TVS yra nemokamos ir ne ką prastesnės už komercines, ir dažniausiai skiriasi tik tuo, kad neturi oficialios techninės pagalbos paslaugos. Norint įdiegti TVS, reikia turėti truputį žinių apie „Unix“ šeimos OS (neretai tinka ir „Windows“) ir vietos serveryje, kuriame įdiegta PHP (neretai reikia ir „MySQL“). Norint naudoti „MySQL“, jai reikia atskiro prisijungimo vardo. Failai serveryje dažniausiai tvarkomi per FTP, bet gali būti ir kitų variantų.

Įdiegus TVS, turintys programavimo pagrindus, gali ir patys nesunkiai šį tą pakoreguoti.

4.6.1. TVS struktūra ir naudojimas

Kad portalo sistema efektyviai funkcionuotų, reikia įvairių specialistų indėlio. Programuotojų darbą atlieka atvirojo kodo judėjimo savanoriai. Taip pat galima rasti gerų dizaino/stiliaus pavyzdžių (angl. *skin/theme*). Daugiausia darbų tenka administratoriui: pasirinkti tinkamus sistemos nustatymus (konfigūraciją), pasirinkti, kurių modulių reikia, sukurti vartotojų grupes bei nustatyti jų teises naudotis vienais ar kitais moduliais.

Apibendrinta TVS struktūra:

- Sistemos branduolio architektūra lemia administravimo patogumą, serverio darbo greitį bei išplėtimo (modulių prijungimo) galimybes – nuo to labai priklauso TVS populiarumas.
- Moduliai – tai papildomas funkcionalumas, kuris integruojamas į bendrą sistemą.
- Blokai – modulio informacijos santraukos. Jų dėka galima viename puslapyje lengva pateikti svarbiausią ar naujausią informaciją iš įvairių modulių.

Dažniausiai naudojama technologija PHP+MySQL+Smarty:

- PHP – HTML kodo generavimui realiu laiku bei veikimo logikos (modelio) realizavimui;
- „MySQL“ – duomenų saugojimui bei apdorojimui (pvz., rikiavimui, paieškai);
- „Smarty“ – tai PHP biblioteka, šablonų mechanizmas, kuris palengvina struktūrizuotą duomenų vaizdavimą (gali būti ir kiti šablonų mechanizmai).

TVS įvairovė: paprasčiausios naujienų sistemos, forumai, galerijos/failų tvarkymas, kalendoriai, komercija, nuotolinis mokymas, portalų sistemos (nedideliams, vidutiniams, plataus naudojimo portalams), pora TVS, kurtų ne su PHP, Lietuvoje siūlomos TVS.

Prieš apsisprendžiant, kurią TVS naudoti, galima išbandyti kelias TVS vartojimo bei administravimo aspektu. Šią galimybę siūlo <http://www.opensourcecms.com/> svetainė.

TVS sisteminiai reikalavimai

- Visas TVS palaiko „Linux“ bei „Win32“ tipo sistemos. TVS gali būti suderintos ir su kitomis „Unix“ tipo sistemos („Solaris“, HP-UX, „FreeBSD“ ir kt.).
- Duomenų bazėms gali būti naudojama: „MySQL“ versija 3.23 arba naujesnė, „PostgreSQL“ versija 7.1.3 arba naujesnė, taip pat „Microsoft SQL Server 7/2000“ bei „Microsoft Access“ (via ODBC).

- „Apache“ HTTPD bent 1.3 versija arba kitas žiniatinklio serveris, palaikantis PHP.
- PHP 4.0.6 versija arba naujesnė. PHP turi būti sukompiliuotas kaip „Apache“ modulis. Kai kurios TVS reikalauja:
 - „MySQL“ išplėtimo (suderinamumui su PHP).
 - „Zlib“ – programa duomenų suspaudimui ir saugojimui, o „ImageMagick“ – darbui su paveikslėliais.
- Kai kurių TVS dokumentacijose rekomenduojama naudoti išskirtinai „Internet Explorer 5.5“ ir naujesnes naršyklės, tačiau kitų produktų administracinės dalies priėjimui galima naudoti ir kitas naršyklės.
- Beveik visoms TVS sistemoms reikia PHP ir „MySQL“ komponentų, tačiau kita reikalinga programinė įranga gali skirtis. Tokiais atvejais patariama skaityti produkto dokumentaciją.
- Ne visų TVS gamintojai pateikia savo sistemos aparatūrinius reikalavimus. Minimalūs reikalavimai tarp skirtingų sistemų gali skirtis: nuo CPU 166MHz ir 64MB RAM iki CPU 500MHz su 250MB RAM. Tačiau taip pat keliami reikalavimai priklauso ir nuo to, koks bus svetainės lankytojų skaičius, ką atlieka ir naudoja TVS ir kt.

4.6.2. Vartotojo sąsaja

Norint dirbti su turinio valdymo sistema kaip vartotojui, reikia kompiuterio su interneto ryšiu, taip pat reikia žinoti savo vartotojo vardą ir slaptažodį. Nereikia jokių papildomų žinių, susijusių su operacinėmis sistemomis, tarnybinėmis stotimis ar dizainu.

Puslapių kūrimas, naudojant turinio valdymo sistemas yra labai paprastas. Teksto apdorojimas vyksta „MS Word“ aplinkos tipo redaktoriuje, kur galima rašyti tekstą, kurti lenteles, grafikus. Norint perkelti duomenis iš „MS Office“ dokumento, tereikia viską nusikopijuoti į iškarpinę (angl. *clipboard*) ir vėliau viską įterpti į puslapį.

Puslapių apdorojimas vyksta naudojant patogią „Windows Explorer“ tipo navigaciją, kur galima kurti, trinti, pervadinti ir rūšiuoti puslapius. Turinį galima skelbti nustatytu laiku, t. y. galima nurodyti, kada turinys bus rodomas puslapyje arba kada bus iš jo šalinamas.

Naujų puslapių apjungimas su šablonu: puslapio dizainas, funkcionalumas, navigacija ir turinys apjungiamas išsirenkant tam tikrus „blokus“ iš jų talpyklos ir įkeliant juos į atitinkamą puslapio vietą. Informacijos išdėstymas gali būti lengvai ir greitai keičiamas. Tereikia paimti ir nutempti norimą informacijos bloką į norimą vietą. Kai kurias TVS galima išplėsti iki verslui naudojamam modeliui: duomenų bazių valdymui, dokumentų apdorojimui ir daugeliui kitų.

4.6.3. TVS privalumai

Turinio valdymo sistema yra visapusė sistema, kuri naudojama profesionalių puslapių kūrimui ir priežiūrai. Gerokai sumažėja informacijos pakeitimo puslapyje laikas ir kaina, o tai užtikrina greitą besikeičiančios verslo informacijos perkėlimą į interneto svetainę. Naudojamos turinio valdymo sistemos ir jose esantys papildomi komponentai, suteikia svetainei daugiau funkcionalumo. Nenaudojant TVS reikėtų perrašyti visą kodą ar dalį kodo, o tai užimtų daug laiko ir brangiai kainuotų.

Kai kurios TVS yra atviro kodo, jas galima redaguoti savo nuožiūra, naudojant standartinius programavimo įrankius ir kalbas (pvz.: C++). TVS leidžia patiems programuotojams patobulinti kodą, nes tai yra greita ir lengva. Žemiau pateikiamas PHP kodo pavyzdys vieno iš komponentų, kurį sudaro teksto redaktorius, paveikslėlių įkėlimas ir nesudėtingas pritaikymas vartotojui (angl. *customization*):

```

<?
$cache = 0;
$info = 'Allow s to enter any text.'; $prop-
erties = Array(
  'header' => Array(
    'label' => 'Header',
    'type' => 'string',
  )
)

```

(96)

```
'value' => '',  
) ,  
'text' => Array(  
'label' => 'Text',
```

Be „Windows“ operacinės sistemos, TVS pritaikytos dirbti ir su kitomis nemokamomis operacinėmis sistemomis, o papildomos programos, reikalingos TVS darbui palaikyti serveryje, taip pat dažniausiai būna nemokamos, todėl nereikia investuoti daug pinigų į papildomą programinę įrangą.

Nereikia daug žinių norint naudoti turinio valdymo sistemas, jas nesunku perprasti ir naudoti.

Dažniausiai turinio valdymo sistemos pateikti naujienas, straipsnius, informaciją ar kitokios paskirties tekstus. TVS taip pat gali būti naudojamos pateikti turinį ir objektus, kitokio nei tekstinis tipo, kaip pavyzdžiui: formatuotas tekstas, paveikslėliai, filmai, „flash“ ir failai.

Kryžminės nuorodos

Turinį galima suskirstyti į kategorijas, tas pats straipsnis gali būti priskirtas kelioms kategorijoms. To paties turinio straipsniai taip pat gali būti susiejami su keliais raktiniais žodžiais. Straipsnis automatiškai sukuria raktinių žodžių abėcėlinę rodyklę, kurioje galima rasti raktinius žodžius ir straipsnius, kuriuose jie naudojami. Temos (angl. *topics*) naudojamos atrinkti norimą straipsnį ar turinio tipą.

Formatuotas tekstas

Svetainėse dažniausiai pateikiamas formatuotas tekstas. Galima naudoti pastorintą, pasvirusį, pabrauktą šriftus ir kitas pagrindines teksto formavimo priemones. Taip pat galima naudoti lenteles ir sudėtingą teksto formavimą, įterpti žymes (angl. *tags*).

Vartotojo žymės

Jei reikia padarytos pagal užsakymą žymės, pavyzdžiui kompanijos logotipo, galima susikurti savo paties žymę. Tai leidžia pakeisti dokumento išvaizdą taip, kaip nori pats autorius, su tokiomis žymėmis, kokių jam pačiam reikia.

Paveikslėliai

Į puslapį galima įterpti paveikslėlius. Jie gali būti įterpti bet kurioje teksto vietoje. Paveikslėlių dydis nustatomas automatiškai pagal tai, kokio dydžio reikia. Iš anksto nustatyti dydžiai yra: mažas, vidutinis, didelis ir originalo dydžio. Paveikslėliai saugomi duomenų bazėje, todėl galima juos panaudoti ir vėliau.

Daugialypės terpės objektai

Galima įterpti objektus iš jiems skirto katalogo ir naudoti bet kurioje teksto vietoje. Objektų pavyzdžiai: „Flash video“, „Quicktime movie“ ir kiti. Daugialypės terpės objektai yra saugomi.

Failai

Failai saugomi jiems skirtoje duomenų bazėje ir gali būti prisegami prie straipsnių. Nuorodos į failus parsisiuntimui gali būti įterptos bet kurioje teksto vietoje.

Šablonai

Straipsnis atvaizduojamas pagal šabloną, kad būtų galima pakeisti visą išdėstymą. Šablonai sukurti kiekvienam objektui ir žyme, kurie tik gali būti įterpti straipsnyje.

4.6.4. Turinio valdymo sistema „Joomla“

„Joomla“ yra pasaulyje pripažinta atviro kodo turinio valdymo sistema (TVS), kuri padeda kurti internetinius tinklalapius ir kitas galingas internetines aplikacijas. „Joomla“ yra nemokama, atvira ir prieinama visiems pagal GPL licenciją (www.joomla.org).

Pavadinimas „Joomla“ yra paimtas iš Swahili (Pietų Afrika) kalbos žodžio „Jumla“ ir reiškia „visiškai kartu“. Tai daugiamečio darbo rezultatas, atsiradęs vietoje anksčiau vartoto „Mambo“. Tai PHP programavimo kalba parašytų skriptų rinkinys.

„Joomla“ nėra sudėtinga valdymo atžvilgiu, nes yra sukurta plačiai vartotojų grupei. Ši TVS nereikalauja iš vartotojo ar administratoriaus ypatingo HTML išmanymo. Ji yra ypač patikima, nesudėtingai įdiegiama ir valdoma.

„Joomla“ naudojama visame pasaulyje nuo paprasto, asmeninio tinklapio iki sudėtingų korporacinių internetinių aplikacijų. Štai tik keletas jos pritaikymo sričių:

- korporaciniai internetiniai tinklalapiai ar portalai;
- internetinė prekyba;
- smulkaus verslo portalai;
- ne pelno siekiantys ir organizaciniai portalai;
- vyriausybės aplikacijos;
- korporacijų vidaus ir išorės valdymas;
- mokyklų ir bažnyčių tinklapiai;
- asmeniniai ar šeimos tinklapiai;
- bendruomenės organizavimo portalai;
- žurnalai, laikraščiai ir t. t.

Su „Joomla“ galima ypatingai lengvai valdyti kiekvieną tinklalapio aspektą, pradedant nuo naujienų ar paveikslėlių įdėjimo ir baigiant produktų atnaujinimu ar internetinėmis rezervacijomis.

„Joomla“ savybės:

- visiškas duomenų bazės komponentų ir svetainės valdymas;
- naujienų, prekių ir serviso skyriai;
- skyrių skaičius bet kada gali būti papildytas;
- visiškas blokų valdymas, įskaitant kairįjį, dešinįjį ir viršutinį meniu blokus;
- vaizdų, reikalingų administruojant svetainę, įkrovimas į asmeninę biblioteką naršyklės pagalba;
- dinaminiai apklausų, balsavimų, forumų moduliai su rezultatų demonstravimu;
- Suderinamumas su „Linux“, „FreeBSD“ serveriu, „Solaris“ ir AIX.

Plačios valdymo galimybės:

- objektų eiliškumo keitimo galimybė, įskaitant naujienas, DUK straipsnius ir t. t.;

- svarbių naujienų generatorius;
- galimas straipsnių naujienų siuntimas;
- objektų hierarchija: galimybė kurti skyrius, kategorijas, ir puslapius norima tvarka;
- vaizdų biblioteka: galimybė saugoti savo PNG, PDF, DOC, XLS, GIF formato dokumentus tiesiog svetainėje palengvinant jų naudojimą ateityje;
- automatinis paieškos kelio taisymas;
- naujienų juostų valdymo valdytojas (360 įvairių informacinių tarnybų);
- bet kuri svetainės objektą galima „siųsti draugui“ arba „spausdinti“;
- įdiegtas panašus į „Word Pad“ teksto redaktorius;
- galimybė valdyti vartotojų prieigą prie tam tikrų svetainės funkcijų ir skyrių;
- apklausų ir balsavimų organizavimas tiek pasirinktame puslapyje, tiek ir visoje svetainėje;
- asmeninių puslapių modulis – galimybė atnaujinti savo svetainę;
- šablonų valdytojas – galimybė per kelias sekundes parsisiųsti šabloną ir jį įdiegti savo svetainėje;
- išankstinės peržiūros galimybė prieš medžiagos publikaciją;
- juostinės reklamos valdymo sistema.

Pagrindinis „Joomla“ paketas net ir ne programuotojams yra labai lengvai įdiegiamas. „Joomla“ turi augančią bei aktyvią bendruomenę, kurią forumuose sudaro daugiau nei 40 000 draugiškų vartotojų bei kūrėjų, kurie visuomet pasiryžę padėti.

Įdiegus ir paleidus „Joomla“ visiškai paprasta, net netechniniams vartotojams, sukurti ar redaguoti turinį, atnaujinti paveikslėlius ar valdyti svarbią informaciją, nuo kurios priklauso organizacijos veikla. Kiekvienas gali lengvai išmokti valdyti „Joomla“ tinklalapį.

Pagrindinis „Joomla“ paketas puikiai tinka visiškai valdyti tinklalapį. Daugeliui žmonių tikrasis „Joomla“ veidas atsiskleidžia panaudojant karkasą,

kuris leidžia tūkstančiams kūrėjų visame pasaulyje kurti galingus priedus ir papildymus, taip dar labiau praplečiant „Joomla“ galimybes. Keletas iš prieinamų papildymų pavyzdžių: dinaminiai formų kūrimo įrankiai, verslo ar organizacijų katalogai, dokumentų valdymas, paveikslėlių ir multimedijos galerijos, elektroninė komercija ir prekių krepšelių varikliai, forumai ir pokalbių programinė įranga, kalendoriai, tinklaraščių įrankiai, vartotojų ir tinklo apkrovimo analizavimo įrankiai, el. pašto naujienos, duomenų kaupimo ir pranešimo įrankiai, logotipų reklamavimo sistemos, prenumeratos paslauga ir kt.

Jeigu reikalingos papildomos savybės, kurios nėra numatytos pagal nutylėjimą, galima lengvai įdiegti išplėtimų: komponentai, moduliai, įskiepiai („pluginai“), šablonai ir kalbos. Kiekvienas šių išplėtimų skirtas tvarkyti konkrečias funkcines galimybes.

Komponentai (elementai)

Komponentai yra stambiausi ir sudėtingiausi išplėtimai iš visų, jie gali būti laikomi kaip mini taikomosios programos. Šiuose komponentuose yra dvi sekcijos – administratoriaus sekcija (angl. *Back-end*) ir svetainės sekcija (angl. *Front-end*). Pavyzdžiui, „com_registration“ yra komponentas, kuris tvarko vartotojų registraciją, vartotojai gali užsiregistruoti kaip nariai per svetainės sąsają, o administratorius gali redaguoti užsiregistravusius vartotojus. Komponentai yra labai svarbi svetainės dalis, kadangi komponentai valdomi meniu elementų, o kiekvienas meniu elementas paleidžia komponentą.

Pavyzdžiui: „com_content“, „com_registration“.

Moduliai

Moduliai yra paprastesni ir lankstesni išplėtimai naudojami svetainių vaizdavimui. Kartais moduliai susiejami su komponentais, pavyzdžiui, kaip „latest news“ modulis susiejamas su „com_content“ ir pateikia nuorodas į naujausius turinio elementus. Šie moduliai labiausiai žinomi kaip „dėžutės“ (angl. *boxes*), kurie sukomponuojami kartu su komponentais, pavyzdžiui, prisijungimo modulis. Apatinė antraštė taip pat yra modulis. Moduliai priskiriami meniu elementui. Taigi, galima nuspręsti, rodyti ar paslėpti prisijungimo modulį, priklausomai nuo to, kuris komponentas (menu elementas) yra naudojamas.

Vis dėlto, modulius nebūtina susieti su komponentais, jie neprivalo būti su kažkuo susieti ir gali būti nekintamas HTML ar tekstas.

Pavyzdžiui: „mod_banners“, „mod_mainmenu“.

Įskiepai

Įskiepai yra sudėtingesni išplėtimai ir įvykių apdorojimo elementai. Bet kokios „Joomla“ dalies, branduolio, modulio ar komponento, vykdymo metu gali būti sužadintas įvykis. Kai įvykis yra sužadinamas, vykdomi tie įskiepai, kurie registruoti toje taikomojoje programoje apdorojantys konkrečių įvykių.

Pavyzdžiui: „content.searchbot“, „tinymce“.

Šablonai

Šablonas yra „Joomla“ pagrindu sukurtos svetainės dizainas, kurio pagalba galima keisti svetainės išvaizdą. Šablonai turi specialius laukus, kuriuose matomi komponentai ir moduliai. Šablonus lengva kurti ar pritaikyti, jie suteikia maksimalų lankstumą kuriant svetainės stilių.

Kalbos

Kalbos gali būti sukomplektuotos dviem būdais – kaip branduolio komplektas arba kaip išplėtimo komplektas. Šie failai apima „raktas/reikšmė“ (angl. *key/value*) poras, o šios poros pateikia statinių teksto eilučių (sekų) vertimą, kurios susietos su „Joomla“ kodu. Šie kalbų komplektai veikia tiek sąsajos, tiek administratoriaus pusėje. Šiuose kalbų komplektuose taip pat yra XML meta failas, kuriame aprašoma kalbos ir šrifto informacija, skirta naudoti PDF turinio generavimui.

Turinio valdymo sistemą „Joomla“ sudaro sistemos branduolys ir prie jo jungiami moduliai, komponentai, įskiepai, kurie leidžia išplėsti tinklalapio galimybes ir funkcionalumą.

SEO nustatymai

SEO įrankis SEF (angl. *Search Engine Friendly*) generuoja nuorodas ir antraštes pagal straipsnių ir prekių pavadinimus. Įrankis leidžia pakeisti tink-

lalapio sudėtingą URL į paprastą ir lengvai perskaitomą tiek žmogui, tiek paieškos sistemoms.

„Joomla“ saugumas

Pagrindiniai patarimai kaip apsaugoti savo tinklalapį, kurtą naudojant TVS:

- Turi būti daromos tik duomenų kopijos, o ne visos sistemos. Pavyzdžiui, išvalyti visus laukus ir visiškai įdiegti TVS. Galima išsaugoti, tiksliau pasidaryti atsarginę kopiją ir dizaino šablono, tačiau prieš jį diegiant būtina patikrinti ar jis nėra pažeistas.
- Mokantys skaityti ir suprasti LOG failus, gali nustatyti, kodėl ir kaip tinklalapis buvo pažeistas.
- Naudoti visus naujus šiuolaikinius plėtinius, kurie turi papildomą apsaugos paketą nuo „įsibrovėlių“.
- Naudoti stiprius slaptažodžius, kuriuos bus kur kas sunkiau „nulaužti“.
- „Registre_globals“ turėtų būti išjungtas.
- Jei nenaudojami jokie skriptai, kurie reikalauja „allow_url“, reikia jį išjungti.

Jeigu spėjama, jog tinklalapis buvo atakuotas „įsibrovėlių“, pirmiausia patartina daryti taip:

1. Patikrinti failų prieigos teises.
2. Patikrinti ar prieglobos tiekėjas nepaleido kokio atnaujinimo, kuris gali pastoviai lįsti į tinklalapį, nuolatos ką nors atnaujin-damas.
3. Pagalvoti, ar nebuvo daryta jokių pakeitimų. Pavyzdžiui, ar buvo daryti kokie nors duomenų bazės ar administravimo pa-keitimai, ar padaryti pakeitimai „htaccess“?

Kartais manoma, jog puslapis buvo pavogtas, nors tai gali būti tik aki-vaizdžios konfigūravimo klaidos.

Jei manoma, kad svetainė tikrai buvo atakuota hakerių, reikia paklaus-ti paslaugų teikėjo, ar jie turi kokios nors informacijos, kurią galėtų suteikti.

Keletas skriptų yra žinomi ir labai mėgstami programišių, pavyzdžiui „phpBB“. Rekomenduojama pažiūrėti, kokie skriptai yra paleisti, patikrinti ar jie turi kokių žinomų pažeidimų.

Programišių įsiveržimo priežastis gali būti ta, kad į sistemą buvo įdiegti papildomi komponentai ar moduliai, kurie jau buvo pažeisti įsibrovėlių. Yra daugybė forumų apie komponentus ar modulius, todėl galima išsiaiškinti ar jų skriptuose aptikta kokių skylių.

Turint atsarginę kopiją, svetainę galima atstatyti. Nors dalis informacijos bus prarasta, tačiau tinklalapis bus išsaugotas. Reikia pirmiausia ištrinti laikmenas, kurie nepriklauso tinklalapiui, tuomet tikrinti tik iš savo saito, kad būtų galima atstatyti savuosius. Kitu atveju, viską reikia išvalyti ir iš naujo įdiegti turinio valdymo sistemą. Programišiai gali palikti skriptų, kurie paslėpti giliai tinklalapyje, o tai suteiks galimybę pabandyti vėl įsiveržti į sistemą.

Klausimai ir užduotys

1. Kokia „JavaScript“ paskirtis?
2. Kokie yra baziniai „JavaScript“ žodžiai?
3. Kaip užrašomi „JavaScript“ kintamieji?
4. Kokie yra „JavaScript“ matematiniai, palyginimo ir loginiai operatoriai? Kaip jie naudojami?
5. Kaip aprašoma sąlygos ir valdymo struktūros?
6. Kokie yra standartiniai „JavaScript“ objektai ir kam jie skirti?
7. Kaip aprašomos „JavaScript“ funkcijos?
8. Kokios dažniausiai naudojamos tinklalapių kūrimo priemonės?
9. Kokios „php“ programavimo savybės?
10. Kas skirtas „Apache“?
11. Apibūdinkite „Apache“ vartotojų grupes.
12. Kam skirtas „MySQL“ ir „phpMyAdmin“?
13. Kokios vartotojų grupės gali nemokamai atsisiųsti naujausią „MySQL“ versiją?
14. Kokiame serveryje gali būti naudojama „MySQL“?
15. Kas yra PHP?
16. Kam skirtas SEO? Kokie SEO principai?
17. Kaip HTML5 gali padėti SEO?
18. Kas yra tinklalapio šablonas?
19. Išvardinkite pagrindinius tinklalapio šablono naudojimo privalumus.
20. Kokia TVS struktūra?
21. Kokios TVS pagrindinės savybės?

22. Kuo pasižymi TVS vartotojo sąsaja?
23. Kokiems projektams galima naudoti „Joomla“?
24. Kokius išplėtimus turi turinio valdymo sistema „Joomla“?
25. Ką daryti, jeigu manoma, jog svetainė buvo atakuota įsibrovėlių?
26. Surasti internete bent 5 įvairios paskirties laisvai platinamas „JavaScript“ programėles, jas išbandyti savo kompiuteryje, atlikti pakeitimus savo nuožūra.
27. Parašyti sąlygą su PHP: jeigu kintamasis \$a daugiau už \$b, spausdintų tekstą: "a daugiau už b".
28. Sukurti funkciją, kuri sujungia 3 išorinius kintamuosius ir grąžina tų kintamųjų sąjungos reikšmę.
29. Sukurti funkciją, kuri atliktų sudėties, atimties, daugybos ar dalybos veiksmus, perduodant jai 3 parametrus. Funkcija iškviečiama tokiu būdu: `<?php funkcija ("veiksmas", skaičius1, skaičius2); ?>`
30. Sukurti dokumentą su PHP, kurio antraštėje ir dokumento tekste būtų nurodyta, kiek milisekundžių praėjo nuo vidurnakčio iki tol, kol dokumentas buvo užklaustas. Abu šie skaičiai turėtų sutapti.

5. TINKLALAPIŲ DIZAINAS

5.1. Tinklalapių dizaino projektas

Tinklalapių dizainas – viena iš svarbiausių interneto svetainės dalių. Labai svarbu, kad grafinis apipavidalinimas būtų patrauklus, gerai pristatantis įmonę ar projektą bei padedantis lengvai įsisavinti informaciją.

Prieš kuriant internetinę svetainę, reikia turėti kuriamos svetainės viziją. Interneto tinklalapio išvaizda yra vienas svarbiausių kriterijų siekiant pritraukti lankytojus bei atspindėti tinkamą įmonės įvaizdį internete.

Pagrindiniai tinklalapių dizaino darbų etapai:

- idėjos ieškojimas;
- firminio stiliaus kūrimas (logotipas ir kt.);
- grafinis sprendimas (dizaino paaišymas);
- baigiamasis etapas (dizaino tvarkymas).

Firminis stilius – įmonės veidas, pagal kurį įmonė vėliau atpažįstama (logotipas, firminiai blankai, vizitinės kortelės, plakatai, kalendoriai, pakuotės ar suvenyrai). Firminis stilius – išimintinas, originalus, klientui patrauklus ir atitinkantis įmonės tikrąjį veidą, išskiriantis ją iš konkurentų masės. Logotipas – tai grafinis kompanijos atspindys, išskiriantis ją iš konkurentų, sukurtas greitam atpažinimui. Logotipas ir firminis stilius, tai kompanijos bei produkto rinkodaros dalis.

Interneto svetainės firminis stilius nesusideda vien iš logotipo, tai viso tinklalapio idėjinis turinys. Įmonei interneto svetainė, tai tarsi jos vizitinė kortelė, o firminis stilius tai balsas internete, spaudoje ir visuomenėje. Dizaineriai gali padėti ir parinkti originalų spalvų, šriftų, grafikos elementų derinį, pagal kurį klientas susidarys teigiamą įspūdį apie įmonę ir siūlomus produktus.

Internetinės svetainės dizainas yra įmonės veidas internete. Neretai kuriant svetainę persistengiama, siūlomas prekes ir paslaugas užgožia animacija ir per didelis informacijos kiekis, kuriame vartotojas neretai pasiklysta. Svetainės kūrėjo tikslas turėtų būti sukurti unikalų svetainės dizainą, kuris atitiktų darbo specifiką, aiškiai suprantamą ir valdomą bei patraukiančią vartotojo akį. Papildomi moduliai turi padėti lengviau užmegzti ryšį tarp įmonės ir vartotojo. In-

ternetinės svetainės tinklalapio turinys turi būti paruoštas taip, kad būtų tinkamai atvaizduojamas visose internetinėse naršyklėse būtent taip, kaip reikia. Svetainės optimizavimas paieškos sistemoms turėtų pritraukti daugiau lankytojų į tinklalapį.

Reklaminius skydelius (angl. *banner*) – reklaminius plotus internetinėje svetainėje su reklamuojama vaizdine medžiaga. Reklamjuostės yra gražus paveiksliukas ar animacija, su nuoroda į reklamuojamą internetinę svetainę ar produkto išsamesnį aprašymą.

Skydelio efektyvumas: atlikta daugybė tyrimų, įrodančių, kad kuo didesnis skydelio plotas, tuo didesnę poveikį jis turi prekės ženklo reklamai. Šiuolaikinėmis „Flash“ technologijomis sukurtos „SkyScraper“, „Billboard“ ir kitų formatų reklamjuostės, leidžia reklamos užsakovams sukurti papildomą turinį ir interaktyvumą, neperžengiant standartinių reklamjuosčių rėmų.

Programavimas apjungia meno, fundamentalių mokslų ir inžinerijos elementus. Siaurąja prasme programavimas suvokiamas kaip kodavimas – vieno ar kelių, tarpusavyje susijusių algoritmų realizavimas viena iš programavimo kalbų.

Gerai suprogramuotas interneto tinklalapis yra funkcionalus ir patogus tiek lankytojui, tiek klientui. Šiuolaikiško tinklalapio galimybės yra lengvai išplečiamos, o informacija jame keičiama greitai ir paprastai.

Didelis projektas turi būti apgalvotas ypač kruopščiai. Svetainės kūrėjo užduotis – suderinti dizainą ir programavimo sprendimus, atsižvelgiant į tinklalapio tobulinimo bei plėtotės galimybes. Svetainė turi būti originalus, modernus, dinamiškas ir estetiškai patrauklus tinklalapis.

5.2. Interneto svetainės dizaino formulė

Gero interneto svetainės dizaino formulė susideda iš kelių pagrindinių dalykų – estetikos, patogios ir aiškios navigacijos, profesionalaus, apgalvoto informacijos suformavimo bei išsamiai ir įdomiai perteiktos tematikos.

Grubiausios interneto svetainių dizainerių klaidos:

1. Skaitomumo problema.
2. Nestandartinės nuorodos.
3. „Flash“.

4. Turinys parašytas ne interneto svetainei.
5. Bloga paieška.
6. Naršyklių nesuderinamumas.
7. Milžiniškos formos.

1. Skaitomumo problema

Viena didžiausių klaidų yra netinkamas šrifto parinkimas. Didžioji dauguma lankytojų skundžiasi dėl mažo šrifto dydžio arba dydžio fiksavimo, taip pat dėl per mažo kontrasto tarp šrifto ir fono spalvų.

2. Nestandartinės nuorodos

Pagrindinės nuorodų taisyklės:

- Svetainėje būtina aiškiai parodyti, kur yra nuorodos: jei tai yra tekstas, tegul jis būna kitos spalvos ir pabrauktas (nenaudokite pabraukimų įprastame tekste).
- Neslėpti skirtumo tarp lankyto ir dar nelankyto nuorodų.
- Informuoti vartotoją apie tai, kas slypi po nuoroda. Reiktų parašyti bent kelias užuominas apie nuorodos turinį, kad jas suprastų tiek vartotojas, tiek paieškos robotas. Aprašant nuorodą reikia vengti nieko nesakančių frazių kaip „spauskite čia“.
- Vengti „JavaScript“ ir kitų nuorodą „apsunkinančių“ priemonių, kurios keičia nuorodos naudojimo įpročius.
- Neatidarinėti nuorodų naujuose languose (išskyrus PDF ir kitas bylas).

3. „Flash“

Nemaža dalis „Flash“ darbų erzina vartotojus, o puslapių įžanga buvo ypatingai kenksminga. „Flash“ yra programinė aplinka, todėl ji turi suteikti papildomas galimybes vartotojui, kurių nesuteikia tradicinė statinė svetainių aplinka. Nereikia naudoti „Flash“ vien tam, kad svetainėje kas nors „šokinėtų“ ar „bėgiotų“. Jei svetainės turinys nuobodus, verta perrašyti tekstus, kad jie būtų įdomesni, pasamdyti profesionalų fotografą, kad padarytų spalvingesnių nuotraukų. Dauguma žmonių animaciją lygina su netinkamu turiniu. „Flash“

naudojimas navigacijoje beveik visada klaidingas. Žmonės visada mieliau nauduos nuspėjamą navigaciją ir statinį meniu.

4. Turinys parašytas ne interneto svetainei

Rašyti internetui reiškia:

- rašyti trumpai;
- rašyti taip, kad lengva būtų apžvelgti;
- rašyti tik „į temą“ (o ne pildyti erdvės tuščia reklama).

Interneto svetainės tekstas turi atsakyti į vartotojų užklausimus. Tam reikia naudoti paprastą kalbą be sudėtingų terminų (paprastais žodžiais išreikštas turinys palengvins ir „paieškos sistemų darbą“, todėl, kad vartotojai ieško informacijos naudodami savo kalbą).

5. Bloga paieška

Visų kitų klaidų taisymas yra lengvas lyginant kiek resursų sunaudos geros paieškos sukūrimas. Nepaisant išlaidų, tai apsimoka todėl, kad paieška yra esminis darbo įrankis vartotojui, ir kiekvienais metais šis įrankis tampa vis svarbesniu ir svarbesniu.

6. Naršyklių nesuderinamumas

Nors vis daugiau vartotojų pradeda naudotis kitomis interneto naršyklėmis, tokiomis kaip „Chrome“, „Firefox“, „Opera“, „Safari“, tačiau vis dar didžioji dauguma naudojami „Internet Explorer“ naršykle.

7. Milžiniškos formos

Vis daugiau žmonių skundžiasi svetainių formomis. Formos labai dažnai naudojamos internete, jos būna didelės, su daugeliu klausimų ir variantų. Visiškai išvengti formų neįmanoma, tačiau vertėtų padaryti taip, kad tas susitikimas su forma lankytojui būtų kuo trumpesnis.

Penkios formų naudojimo taisyklės:

- Išimti visus nereikalingus klausimus. Pavyzdžiui, ar tikrai reikia žinoti kaip kreiptis į vartotoją (tai vyras ar moteris, ištekėjusi ar ne)?

- Privalomais padaryti tik tuos laukelius, kurie yra tikrai reikalingi.
- Naudoti automatinio pildymo funkciją naršyklėje. Tam reikia naudoti HTML kode tradicinius laukelių pavadinimus („name“, „address“ ir t. t.).
- Nustatyti automatinį žymeklį pirmame laukelyje. Taip sutaupomas vienas pelės paspaudimas.
- Lanksčiai palaikyti įvairius formatus. Pavyzdžiui, telefonų numerių, kreditinių kortelių numerių ir t. t. Juk nesunku suprogramuoti, kad sistema „išimtų“ nereikalingus tarpus ir brūkšnius.

5.3. Tinklapių dizaino kūrimo procesas

Tradicinis linijinis dizaino kūrimo procesas susideda iš trumpo aptarimo, tyrimo ir išvalgo, dizaino proceso kūrimo, sprendimo ir galutinio produkto.

Interneto svetainės dizaino kūrimo procese dažnai dalyvauja keletas žmonių, kūrimo proceso metu pasitaiko dizaino atnaujinimų bei bendradarbiavimo tarp skirtingų specialistų. Siūloma šį linijinį procesą truputį pakeisti, karfojant kiekvieną dizaino proceso žingsnį, kol jis bus baigas ir tik tada pereiti prie tolimesnio proceso žingsnio. Toks dizaino procesas supaprastina interneto svetainės kūrimą.



13 pav. Interneto svetainės dizaino kūrimo procesas

1. Trumpas aptarimas

Trumpas aptarimas susideda iš šių dokumentų:

- Kliento atsiųstas pasiūlymas.
- Techninis aprašas (dažniausiai sukuria dizaineris), kuriame bendrai aprašomi techniniai projekto reikalavimai bei apimtis.

- Kūrybinis aprašas yra sukuriamas kaip atsakymas į kliento pasiūlymą ir naudojamas kūrybinio proceso metu.
- Įžvalgų aprašas – kelių sakinių dokumentas, sukuriamas pasitarus su klientu. Dokumente tiksliai ir trumpai aprašomas projektas.

2. Tyrimas ir įžvalgos

Šiame internetinės svetainės dizaino proceso etape atliekamas tyrimas.

Tyrimo metu reikėtų atsakyti į panašius klausimus:

- Kaip atrodys pagrindinis svetainės puslapis?
- Kiek individualių vartotojų laukiama per tam tikrą laiko tarpą, pvz., mėnesį?
- Kaip vartotojai naršys interneto puslapyje (galimi populiariausi scenarijai)?
- Kas žinoma apie internetinės svetainės būsimus individualius vartotojus?
- Ko individualus vartotojas tikisi iš svetainės?
- Ko iš svetainės tikisi tam tikra tikslinė vartotojų grupė?

„Swot“ analizė leidžia įvertinti internetinės svetainės stipriąsias bei silpnąsias puses, galimybes ir pavojus. Vertėtų sukurti klausimyną ir apklausti kuo daugiau potencialių internetinės svetainės vartotojų. Svarbu nustatyti rinkos segmentaciją – žmonių grupes, turinčias bendrų savybių, tokių kaip gyvenama vieta, amžius, lytis, pomėgiai.

Prieš kuriant dizainą labai svarbu kuo daugiau surinkti tikslios informacijos apie norimą dizainą užduodant tinkamus klausimus, kad tektų mažiau jį perdarinėti. Norint sukurti kokybišką interneto svetainę nėra būtina turėti atsakymus į visus klausimus, tačiau atsakymai į didžiąją dalį klausimų padės greičiau ir kokybiškiau sukurti tinkamą dizainą.

3. Idėjų generavimas

Paprasčiausias idėjų generavimo būdas yra naudoti minčių lietu, t. y. sugalvoti idėją ir iš karto ją užrašyti bei nieko nelaukiant eiti prie kitos. Idėjų tinkamumas ir analizavimas šiuo atveju nesvarbus. Yra ir kitų idėjų generavimo būdų (minčių žemėlapiai, idėjų sesijos su bendradarbiais ar klientu). Į šį proce-

są galime įtraukti klientą. Taip jis bus labiau informuotas apie procesą ir pakreips projekto eigą jam patinkančia linkme. Sugeneruotos idėjos aptariamasi su klientu ir įtraukiamasi į kūrybinį aprašą.

4. Sprendimas

Surinkus duomenis iš idėjų generavimo žingsnio, pridėjus tyrimo duomenis bei aprašus, priimamas sprendimas, kokia kryptimi bus plėtojamas interneto svetainės dizainas. Toliau kuriamasi internetinės svetainės prototipas, kurį galima pradėti piešiniu ant popieriaus ar grafine programa („Photoshop“, „Gimp“ ar „Paint.NET“). Tikslas yra kuo greičiau ir paprasčiau sukurti dizainą, kad vėliau jį galima būtų tobulinti. Sukūrus patinkantį prototipą, reikia leisti jį patikrinti (bendradarbiui, draugui ar šeimos nariui). Svarbiausia išsiaiškinti, kur yra silpnosios internetinės svetainės vietos ir jas ištaisyti. Toliau sukuriamas interneto svetainės pagrindinių puslapių dizainas, atsižvelgiant į logotipą, spalvas ir šriftus.

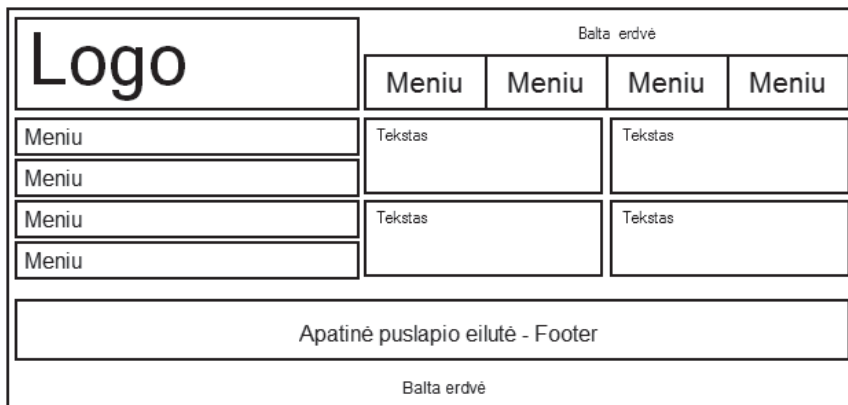
5. Galutinis produkto paleidimas

Panaudojus surinktą informaciją ankstesniuose žingsniuose ir atsižvelgus į sukurtą prototipą, lieka sukurti veikiančią produktą.

Dizaino kūrimo procesų yra įvairių. Juose visuose minimi tie patys elementai: aptarimas, tyrimas, dizaino kūrimas, pataisymai ir galutinis produkto paleidimas. Skirtingi dizaino procesai daugiau dėmesio skiria tam tikram proceso elementui. Svarbiausia, kad tinkamas dizaino procesas būtų pritaikytas specifiniam projektui.

5.4. Tinklalapio dizaino elementai

Kiekvienas tinklalapis yra sudarytas iš tam tikra tvarka išdėliotų dizaino elementų. Visi interneto puslapiai sudaromi kūrybingai, išdėliojant vos penkis dizaino elementus.



14 pav. Tinklalapio maketo pavyzdys

Logotipas

Geriausias būdas tapti išskirtiniu ir įsimintinu yra geras logotipas, kuris dažniausiai būna kairiajame viršutiniame tinklalapio kampe. Logotipas turėtų būti toje pačioje kiekvieno atskiro interneto svetainės puslapio vietoje. Paprastai spustelėjus ant logotipo sugrįžtama į pagrindinį interneto svetainės puslapį.

Navigacija

Svarbiausia interneto puslapio navigacijos elemento funkcija yra patogumas. Žmonės navigacijos elementus pripratę matyti puslapio viršuje arba kairėje pusėje. Planuojat navigacijos elementus, svarbiausia prisiminti, kad jie turi būti kuo arčiau puslapio viršaus. Geriausia navigacijos elementus sutalpinti matomame puslapyje taip, kad nereikėtų slinkti žemyn.

Turinys

Interneto puslapiuose lankomasi dėl juose pateikto turinio. Neradus dominančios informacijos, tinklalapis paliekamas per kelias sekundes, kad ir koks gražus būtų svetainės dizainas. Svarbiausi ir lankytojams įdomiausi informacijos blokai turi būti išdėstyti galvojant apie interneto puslapio skaitytojų sudominimą.

Apatinė puslapio eilutė (angl. *footer*)

Puslapio apačioje esanti poraštė talpina kontaktinę, teisinę informaciją. Dažnai įkeliamos nuorodos į svarbiausias puslapio dalis. Apatinę puslapio eilutę svarbu sukurti taip, kad interneto svetainės lankytojas suprastų atsidūręs puslapio apačioje.

Balta erdvė (angl. *white space*)

Interneto puslapyje esantys elementai turi būti atskirti balta erdve. Tai interneto puslapio erdvė, neužpildyta tekstu ar grafiniais elementais. Dažnai daroma klaida, kai kiekvienas puslapio pikselis perpildomas nuotraukomis, paveikslėliais ar tekstu. Tokiu atveju pasidaro sunku naršyti bei rasti dominančią informaciją.

5.5. Tinklalapio dizaino spalvos

Spalvų dizainas yra grafinio projekto dalis, tai labai subjektyvu. Vieniems tamsiai pilkos spalvos paletė su ryškiai rožine spalva gali atrodyti puikiai, o kitiems tai būtų visiškai neteisinga. Kuriant dizainą labai svarbu yra jo spalvos, svarbu parinkti jas tinkamai.

5.5.1. Spalvų teorija

Spalvos sukelia emocijas, kurios vienaip ar kitaip paveikia mūsų būseną ir įprasmina tam tikras asociacijas. Štai kaip žmogaus smegenys reaguoja į tam tikras spalvas:

- mėlyna spalva asocijuojasi su sėkme, rimtumu, ramybe, profesionalumu, patikimumu;
- raudona yra visiška priešingybė mėlynai spalvai: pavojus, aistra, meilė, nuotykių, perspėjimas;
- žalia siejama su sveikata, sėkme, pinigais, harmonija, gyvenimu, gamta, natūralumu;
- geltona asocijuojasi su švara, paprastumu, grynumu, nekalrumu;

- oranžinė – su kūryba, jaunatve, linksmybėmis, komfortu;
- rožinė/rausva spalva siejama su nekaltumu, švelnumu, saldumu, jaunatviškumu;
- violetinė – su fantazija, sapnais, prabanga, teisingumu;
- balta – su nekaltumu, teisybe, dorove, paprastumu, švara;
- juoda spalva siejama su mistika, rimtumu, tamsa, paslaptimi;
- pilka spalva asocijuojasi su neutralumu, lygybe, uždarumu.

Pavyzdžiui, išpėjamieji ir draudžiamieji kelio ženklai, klaidų kompiuteryje pranešimai visada būna raudonos spalvos, kuri iškart asocijuojasi su pavojumi. Tuo tarpu žalios spalvos pranešimai informuoja apie kokią nors sėkmingai įvykdytą operaciją.

Tačiau ta pati spalva gali reikšti keletą dalykų, todėl negalima kiekvienai spalvai priskirti konkrečią, vienintelę asociaciją. Be to, skirtingose šalyse spalva gali turėti skirtingas prasmes. Pavyzdžiui, kinams balta spalva yra susijusi su laidotuvėmis.

Spalvų teorija gana sudėtinga, ją studijuoja mokslas, menas ir psichologija. Spalvos sukelia emocijas, kurios vienaip ar kitaip paveikia mūsų būseną ir įprasmina tam tikras asociacijas. Sakoma, jog pamačius raudoną spalvą truputį pakyla kraujospūdis.

Kuriant tinklalapio dizainą reikėtų gerai pagalvoti ir pasirinkti tinkamą spalvą arba jų derinį. Egzistuoja kelios spalvų derinių kūrimo schemas:

- analoginė;
- monochromatinė;
- papildomoji;
- skiriamoji papildomoji;
- trijų sudaromoji.

Spalvų gamos parinkimas yra sunkus ir daug laiko užimantis darbas. Kai kuriais atvejais būna ypatingai sunku pasirinkti spalvas, nes vienos atrodo per šviesios, kitos per tamsios ir panašiai. Todėl spalvų pasirinkimas dažnai gaišina laiką vien dėl to, kad nepasirenkama tuo metu atrodžiusi tinkama spalva ir nepereinama prie kitų darbų, spalvos koregavimą paliekant vėlesniam laikui.

Patyrę dizaineriai teigia, kad spalvos pasirenkamos ne tik dėl estetikos, bet ir dėl to, kad taip galima pasiekti tam tikrų norimų rezultatų.

Atspalvis, sodrumas, tamsumas, šviesumas gali perduoti tam tikrą žinią. Jei spalva kažką reiškia, ji turi galią perduoti tai ją matantiems.

Spalvų teorija yra susijusi su spalvų deriniais per sąryšių kūrimą. Sąryšiai yra kuriami iš spalvų rato pagal spalvų poziciją. Spalvų teorijos sudėtingumas juntamas atsižvelgiant į įvairius atspalvius, šešėlius ir tonus.

5.5.2. Spalvų ratas

Pats svarbiausias spalvų teorijos modelis yra spalvų ratas. Standartiniame RYB (angl. *Red, Yellow, Blue*) spalvų modelio rate yra trys pirminės spalvos: raudona, geltona ir mėlyna. Šių spalvų negalima išgauti maišant kitas. Teoriškai visoms spalvoms išgauti užtektų trijų pagrindinių spalvų – ciano mėlynos, rožinės raudonos, geltonos bei juodos ir baltos – spausdintuvai veikia būtent tokiu principu.

Tinklalapių dizaineriai naudoja tris pagrindines spalvas: raudoną, mėlyną ir žalią, kitaip žinomas kaip RGB (angl. *red, green, blue*). Maišydami šias spalvas elektroniniai įrenginiai atkuria visą spalvų spektrą. Interneto puslapių kūrimui naudojamas būtent toks RGB modelis, nes elektroniniai įrenginiai, mūsų atveju – monitoriai, būtent taip atkuria spalvas.

Pirminės (pagrindinės) spalvos

Spalvų ratas – tai vizualinė spalvų reprezentacija, kurioje susiję atspalviai vaizduojami aplink apskritimą. Spalvų ratas formuojamas iš trijų pagrindinių spalvų, sudarančių lygiakraštį trikampį: raudona, geltona ir mėlyna. Pirminės spalvos – esminės spalvos, kurių negalima išgauti maišant kitas spalvas.

Maišomos dvi pirminės (pagrindinės) spalvos kuria antrines spalvas. Tretinės spalvos yra sukuriamos kombinuojant pirmines ir antrines spalvas (15 pav.).



15 pav. Pirminės, antrinės ir tretinės spalvos

Antrinės spalvos

Kai viena pirminių spalvų yra sumaišoma su kita, gaunama antrinė spalva: oranžinė (gaunama sumaišius raudoną ir geltoną spalvas), žalia (geltona + mėlyna), violetinė (mėlyna + raudona).

Antrinės spalvos taip pat vaizduojamos trikampiais, esančiais šalia pirminių spalvų trikampio. Toks spalvų grupavimas vadinamas absorbciniu, kadangi gaunamas rezultatas – antrinė spalva sugeria daugiau šviesos nei pagrindinė.

Tretinės spalvos

Šios spalvos sukuriamos sumaišius vieną antrinę ir vieną pirminę spalvą, taigi iš viso tris (kadangi antrinę spalvą sudaro dvi pirminės), pavyzdžiui – mėlyna + violetinė.

Spalvų rate kiekviena ši trečioji spalva bus greta esančių spalvų kombinacija. Trečiosios spalvos yra šešios: geltonai oranžinė, raudonai oranžinė, raudonai violetinė (rausva, rožinė), mėlynai violetinė, mėlynai žalia ir geltonai žalia („salotinė“). Spalvų seka šiame rate yra ta pati, kaip ir vaivorykštėje. Naudojantis ja, galima surasti visas įmanomas spalvų kombinacijas.

Spalvų ratas yra vizualus spalvų išdėstymas pagal jų chromatinį santykį. Rato išdėstymas prasideda nuo pagrindinių atspalvių, išdėstytų tolygiai vienas nuo kitos, toliau sukuriami perėjimai tarp pirminių naudojant antrines ir tretines spalvas (16 pav.).



16 pav. Spalvų ratas

Galima išskirti dar daugiau atspalvių, tačiau populiariausias yra dvylikos spalvų ratas. Šis dvylikos spalvų ratas parodo pagrindinius spalvų derinimo principus. Jie yra gana paprasti, spalvų ratas leidžia juos realiai pamatyti ir taip geriau juos įsisavinti.

Pasirinktos spalvos iš spalvų rato dera tarpusavyje. Tai spalvų paletės pagrindas, spalvų sąveika. Šių spalvų deriniai yra labai svarbūs kuriant spalvų paletes. Galima pasikliauti nuojauta, tačiau dažniausiai šie sprendimai yra grindžiami patirtimi, šių spalvų deriniai naudojami visur kasdiniame gyvenime.

Šiame rate pateikiama spalvų progresija eiliškumo tvarka, todėl tai naudingas įrankis kuriant harmoningus spalvų derinius tapybos darbuose, dekoruojant interjerą ar kuriant vizualinius projektus. Spalvų ratas yra ypač naudingas dizaine, kai norima sudaryti tam tikrą spalvų derinį arba kombinaciją, sužinoti, kaip susijusios spalvos tarpusavyje.

Spalvų ratas padeda suprasti ne tik įvairių spalvų santykius, bet ir spalvų klasifikaciją.

Substraktyvios ir adityvios spalvos

Pirminės spalvos gali būti skirstomos į adityvias ir substraktyvias spalvas. Kompiuteryje spalvos yra sukuriamos naudojantis adityvių spalvų metodu. Adityvių spalvų maišymas prasideda nuo juodos (#000000) ir baigiasi

balta (#ffffff), kuo daugiau spalvų yra pridėjama, tuo rezultatas yra šviesesnis ir artėja prie baltos spalvos. 17 paveikslo kairėje pateiktos subtraktyvios spalvos, kurios sumaišytos sukuria juodą spalvą, o dešinėje – pateiktos adityvios spalvos, kurios sumaišytos sudaro baltą spalvą.



17 pav. **Subtraktyvios ir adityvios spalvos**

5.5.3. Spalvų tipai

Spalvos yra skirstomos pagal atspalvį, sodrumą, šviesumą arba skais-tumą. Sodrios, arba kitaip tamsios spalvos, yra gaunamos pridėjus pilkos. Švie-sios arba skaisčios spalvos gaunamos pridėjus baltos.

Grynos spalvos gaunamos pridėjus pilkos, kuri yra juodai baltame spektre per patį vidurį tarp baltos ir juodos.



18 pav. **Grynos spalvos**

Šviesios spalvos yra gaunamos pridėjant šviesiai pilkos spalvos iš juodai balto spalvų spektro.



19 pav. **Šviesios spalvos**

Tamsios arba sodrios spalvos yra gaunamos į spalvą pridodant tamsiai pilkos spalvos.



20 pav. Tamsios arba sodrios spalvos

Šaltos spalvos gaunamos pridėjus mėlynų atspalvių, o šiltos spalvos – pridėjus raudonų atspalvių.

Kontrastas taip pat labai svarbus, nes parašius šviesų tekstą ant šviesaus fono nieko nesimatys. Taip pat nieko nesimatys, jeigu ant tamsaus fono bus parašytas tamsus tekstas.

Spalvas galima derinti pasirinkus vieną pagrindinę spalvą, pridodant kelias papildomas spalvas. Spalvų derinimui galima naudoti populiarių tarp dizainerių „ColorSchemedesigner“ įrankį (<http://colorschemedesigner.com>). Šis įrankis yra gerai suderintas su spalvų teorija ir leidžia rinktis iš kelių spalvų variantų, priklausomai, kiek spalvų reikia.

5.5.4. Spalvų deriniai

Papildomoji spalvų schema (angl. *complementary*) – tai spalvų rate viena prieš kitą išsidėsčiusios spalvos (pavyzdžiui, raudona ir žalia ar raudonai violetinė su geltonai žalia).

Papildomų spalvų pasirinkimai remiasi kontrastingomis spalvomis. Kartais jie atrodo siaubingi ir tiesiog netinkami, tačiau kartais tai tiesiog atsakymas. Šių spalvų atspalviai puikiai tinka spalvų išryškinimui. Priešingos spalvos parinktos kartu sudaro mirgėjimo efektą.

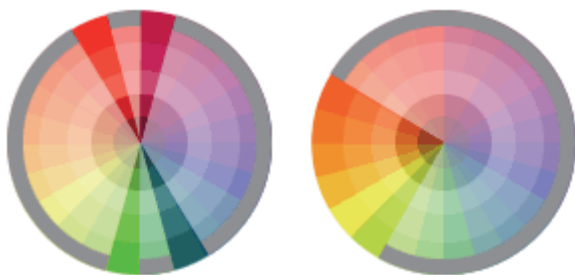
Natūralių atspalvių spalvų derinys sukuria maksimalų kontrastą ir tuo pačiu – maksimalų stabilumo išpūdį.

Skiriamoji papildomoji spalvų schema – vienas atspalvis ir du kiti vienodai nutolę papildantys atspalviai. Ji apjungia analoginę ir papildomąją spalvų sudarymo schemas.



21 pav. **Papildomoji ir skiriamoji papildomoji spalvų schemas**

Dviguba papildomoji spalvų schema – dviejų papildančių spalvų aibė; atstumas tarp pažymėtų papildančių spalvų porų sąlygoja bendrą kontrastą ga-
lutinėje kompozicijoje.



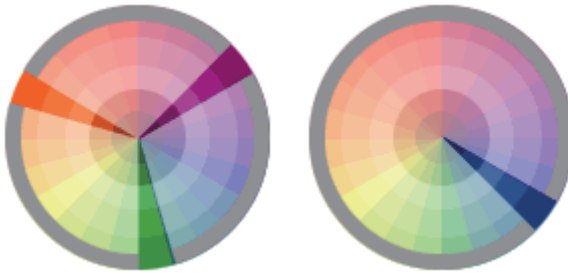
22 pav. **Dviguba papildomoji ir analoginė spalvų schemas**

Analoginė spalvų schema – spalvos, kurios yra gretimos viena kitai spalvų rate (dažniausiai 3 spalvos, pavyzdžiui: geltonai žalia, geltona ir geltonai oranžinė arba geltona, žalsvai gelsva, žalia). Šie atspalviai panašiausi vienas į kitą ir nesudaro kontrasto, todėl sukelia harmonijos pojūtį. Dažniausiai viena iš šių trijų spalvų dominuoja.

Toks panašumas padeda apjungti atskirus kompozicijų elementus. Kompozicijose svarbus tiek kontrastas (gyvumas), tiek harmonija: jei elementai per daug panašūs, tai sukelia nuobodulį, o jei aplink vien ryškūs kontrastai, sunku koncentruoti dėmesį ir atrasti prasmę. Skirtingus atspalvius galima susieti naudojant vienodo intensyvumo spalvas.

Trijų sudaromoji (triados – angl. *triadic*) spalvų schema yra 3 vienodai viena nuo kitos spalvų rate nutolusios spalvos (pavyzdžiui, žalia, oranžinė, violetinė). Trijų grynų kontrastingų spalvų kombinacijos sukuria harmoningus derinius. Tačiau triados dėl ryškumo gali sukurti įtampą dizaine, todėl jas naudoti reikia atsargiai.

Monochromatinė spalvų schema – spalvos, kurios yra konkrečios spalvos atspalviai arba šešėliai. Vienspalviai pasirinkimai yra tiesiog viena spalva iš spalvų rato. Pasirenkama viena spalva ir išvedami nauji atspalviai šviesinant ar tamsinant pasirinktą spalvą.



23 pav. **Triados ir monochromatinė spalvų schemas**

Dar gali būti naudojama *keturkampė* spalvų schema (angl. *tetrad*), kurioje yra keturios vienodai viena nuo kitos paletėje nutolusios spalvos. Tokia paletė yra veiksmingiausia, jeigu viena spalva padaroma dominuojančia.

5.5.5. Spalvų derinimo principai

Viena kitą papildančios trys spalvų poros, raudona ir žalia, geltona ir violetinė, mėlyna ir oranžinė, visada dera. Derinant ryškias pirmines ir antrines spalvas, esančias viena priešais kitą spalvų rate, sukuriamas dramatiškas išpūdis, o norint kuklesnio rezultato, patartina naudoti tų pačių spalvų pastelinius atspalvius.

Žmogaus akis turi savybę kiekvienai spalvai ieškoti atsveriančio atspalvio. Jeigu dvi vienodo ryškumo viena kitą atsveriančios spalvos yra sugretinamos, į jas sunku žiūrėti, nes žvilgsnis krypsta tai į vieną, tai į kitą spalvą.

Geltona ir violetinė papildo viena kitą ir tuo pačiu – labiausiai kontrastuoja. Rausvai oranžinė ir mėsvai žalia papildo viena kitą, tačiau sukuria stiprų šilta-šalta kontrastą. Raudona ir žalia papildo ir atsveria viena kitą.

Atspalvių intensyvumas

Gryni atspalviai turi savo intensyvumą, skiriasi savo ryškumu, tonų santykiu. Gryna geltona – labai šviesi, o gryna mėlyna spalva – labai tamsi. Kompoziciją sudarant tik iš kelių vienodo intensyvumo tonų, sukuriamas tvarkos, aiškumo ir energijos pojūtis.

Rezultato vaizdingumas bei perspektyvos buvimas ar nebuvimas priklauso nuo to, kaip derinamos spalvos ir kokie atspalviai naudojami. Pavyzdžiui, naudojant vienodo intensyvumo atspalvius, sukuriamas plokščias vaizdas, tuo tarpu jungiant skirtingus atspalvius ir šešėlius, gaunamas perspektyvinis vaizdas.

Pilka spalva

Pilka spalva ir visi jos atspalviai tarp juodos ir baltos yra neutralūs, indiferentiški, achromatiniai. Sugretinti su kita spalva, pilki atspalviai veikia kaip fonas, išryškinantys silpnesnius atspalvius ir sugerdami pernelyg kontrastingų derinių spalvingumą.

Achromatinės spalvos yra abstrakčios ir vienodos, chromatinės – gyvybingos ir sudėtingos. Šalia chromatinių spalvų achromatinės praranda savo savybes, todėl norint išlaikyti achromatinės spalvos neutralumą, gretimų spalvų intensyvumas turi skirtis. Pavyzdžiui, jeigu raudona spalva sugretinama su vienodo intensyvumo pilka, pilka atrodys žalesnė. Šį efektą panaikinti galima pasirinkus šviesesnę ar tamsesnę pilką atspalvį.

Sunkiau atskirti tamsius vienos spalvos atspalvius, tuo tarpu šviesūs atpažįstami geriau.

Spalva ir erdvė

Tamsiame fone visi šviesūs atspalviai atrodys ryškesni, didesni, artimesni, tuo tarpu baltame fone – atvirkščiai. Taip pat visuomet šviesūs atspalviai atrodys nutolę, o tamsūs – artimesni.

Šilti ir šalti atspalviai yra vienodo „svorio“. Šilti tonai atrodo artimesni, o šalti – tolimesni. Sujungus šias savybes su šviesių ir tamsių tonų ypatybėmis, galima dar labiau akcentuoti perspektyvinį vaizdą ar jį sumažinti.

Juodame fone žalia ir mėlyna spalvos atitolina vaizdą, o raudona ir oranžinė – priartina. Pašviesinus raudoną ir oranžinę spalvas, ši jų savybė dar labiau sustiprėja. Lygiai taip pat tamsinant mėlyną ir žalią, jų savybė atitolinti vaizdą – stiprėja. Jeigu mėlyna ir žalia šviesinamos, jos ima vaizdą artinti.

Grynosios spalvos, lyginant su maišytomis, atrodo artimesnės. Jei į kompoziciją įtraukiami ir šiltų-šaltų bei šviesių-tamsių spalvų deriniai, perspektyvos vaizdavimas taip pat atitinkamai keičiasi.

Lyginant didelį raudoną plotą ir mažą geltonos spalvos plotelį, raudona spalva taps fonu, taigi geltonos spalvos plotas išryškės, atrodys artimesnis. Pamažu didinant geltonos spalvos plotą iki didesnio už raudoną, geltona spalva taps fonu, o raudona – akcentu.

Šiltos ir šaltos spalvos

Kiekviena spalva turi polinkį sukelti šiltą ar šaltą įspūdį. Šis polinkis ne visuomet akivaizdus, tačiau nulemia spalvų derinimo principus. Raudoni ir geltoni atspalviai laikomi šiltais, o mėlyna spalva – šalta. Derinant spalvas, reikia atsiminti, kad šiltos ar šaltos spalvos atspalviai skiriasi vienas nuo kito, nors šiltos spalvos ir pats „šalčiausias“ tonas visuomet bus „šiltesnis“ už šaltus atspalvius.

Maišant antrines spalvas, svarbu ne tik proporcijos, o ir tai, kokias savybes įgaus skirtingi atspalviai.

Harmoningi spalvų deriniai

Naudojant neutralių ir natūralių spalvų paletę kuriama rami, patogi, harmoninga atmosfera, dažnai įkvėpta gamtos.

Panašią chromatinę vertę turintis spalvų derinys bus harmoningas (nesvarbu, kad spalvos bus skirtingos). Pagal šiuos panašios chromatinės vertės principus sukurtas dizainas leidžia sujungti daugiau įvairių spalvų, kurios iš tolo atrodo kaip harmoninga visuma.

Neutralūs spalvų deriniai

Neutralios spalvos – pilkų, balsvų, kreminės ir rusvų atspalvių derinys. Šios spalvos paprastai dera prie daugumos kitų spalvų, todėl puikiai tinka kaip fonas. Ramių, neutralių spalvų paletė padeda sukurti jaukumą.

Dizainas be spalvų

Juodos ir baltos spalvų derinys – specialus poligrafijos dizainas. Visiškai koncentruojant dėmesį toniniam dizainui galima šalinti spalvą ir gauti įspūdingą dizainą.



24 pav. Nespalvoto dizaino pavyzdys

Šaltinis: <http://www.subtraction.com>

Pateiktame pavyzdyje matyti, kaip kuriant tik su juoda spalva galima sukurti unikalų ir patrauklų dizainą (24 pav.). Juodos ir baltos spalvos bei atspalviai suderinti su puslapiu erdve.

5.6. Tipografijos svarba tinklalapiuose

Tipografiją galima laikyti atskira sritimi, nagrinėjančia sąryšį tarp teksto ir kitų tinklalapio dalių. Svarbiausia pamatyti, kiek tam tikruose tinklala-

piuose gero dizaino lemia tekstas. Galima būtų išimti visus paveikslėlius ir įvertinti teksto išdėstymą. Kai kurie mano, kad tinklalapis, sudarytas daugiausiai iš teksto, yra nuobodus. Kaip bebūtų, sukurti tinklalapį naudojant vien tik tekstą yra įmanoma. Nors paveikslėliai yra kur kas vertingesni, tačiau juos reikia derinti su esamu tekstu bei palikti aplink juos tuščio ploto. 99 % esamų dizainų pagrindas – tekstas ir tuščia erdvė aplink jį.

5.6.1. Kontrastas

Taikant šriftą, svarbus jo skaitomumas. Visuomet reikia užtikrinti tai, kad tekstas būtų lengvai skaitomas. Vienas iš būdų išsiaiškinti, ar teksto spalva kontrastuoja su aplinkos spalvomis, yra išjungti visas spalvas vaizdų redagavimo programoje. Taip bus matyti, ar parinktas tinkamas spalvų kontrastas. Tipografas Robertas Bringhurstas yra sakęs, kad tipografija yra tam, kad būtų galima atspindėti turinį.

Taip pat skaitomumui svarbu ir teksto dydis. Patartina nenustatyti pagrindinio teksto dydžio mažesnio nei 10-12px ir geriau būtų padaryti jį didesnį, jei tik tai yra įmanoma. Mažas šriftas gali būti gerai skaitomas dideliame ekrane, tačiau jis taps visiškai neįskaitomas mažame planšetinio kompiuterio ekrane.

5.6.2. Hierarchija ir erdvė

Vienas iš geriausių būdų suskaidyti turinį – padaryti skirtingų sričių teksto dydžius, skirtingus pagal tai, kokios svarbos yra tam tikras tekstas. Taip lankytojams bus aišku, ką skaityti svarbiau. Ir jei jie skuba, jiems bus paprasčiau orientuotis tinklalapyje.

Hierarchija gali būti sudaryta ne tik taikant teksto dydžius, bet ir jo stilių. Pavyzdžiui, vienas tekstas gali būti išryškintas naudojant visas didžiąsias raides, kitas gali būti parašytas kursyvu ar pabrauktas. Taip pat serifinis tekstas gali būti maišomas su neserifiniu tekstu.

Nereikia bijoti palikti tuščios erdvės. Tuščia erdvė padeda susikoncentruoti ties tekstu. Kitas svarbus dalykas – eilučių aukštis, kuris turėtų

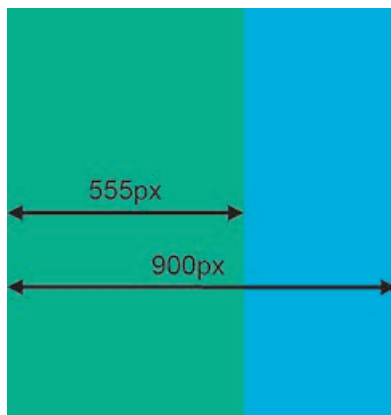
būti bent 140 % teksto dydžio. Geri dizaineriai skiria daug laiko siekdami sudaryti balansą tarp teksto ir tuščio ploto.

5.7. Dizaino taisyklių naudojimas tinklalapių dizaine

5.7.1. Aukso proporcijos naudojimas

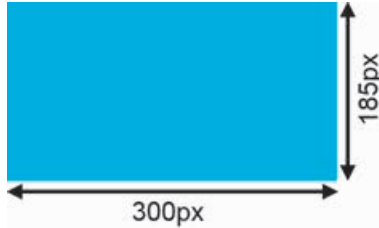
Geras dizainas nebūtinai turi būti ryškiai spalvotas ar perpildytas dizaino elementų. Svarbiausia – paprastumas, aiškumas ir informacijos prieinamumas. Norint tai pasiekti, galima taikyti aukso proporcijos taisyklę. Aukso proporcija arba 1.618 yra tas stebuklingas skaičius, kuris apjungia matematiką ir grožį.

Aukso proporcijos panaudojimo formulė yra paprasta. Pavyzdžiui, pasirinkus dviejų dalių puslapį (pagrindinė, skirta tekstui ir paveikslėliams, ir šoninis meniu), kurio plotis 900px, pagrindinei daliai reiktų skirti: $900\text{px} / 1.618 = 555\text{px}$, o šoniniam meniu lieka $900\text{px} - 555\text{px} = 345\text{px}$ (25 pav.).



25 pav. 900px pločio dviejų dalių puslapio pavyzdys

Taip pat galima apskaičiuoti interneto puslapyje esančio elemento dydį. Jeigu reikia 300px pločio elemento, tuomet $300\text{px} / 1.618 \approx 185\text{px}$. Taip gaunamas 300px skersmens ir 185px aukščio elementas (26 pav.).



26 pav. Interneto puslapyje esančio elemento dydis

5.7.2. Trijų trečiųjų taisyklės taikymas

Viena dažniausiai naudojamų taisyklių tinklalapių dizaine yra trijų trečiųjų taisyklė (angl. *rule of thirds*). Ji taikoma pagal šią taisyklę sukurtą tinklėlį uždedant ant naujai kuriamo interneto puslapio. Tokiam tinklėliui nereikia naudoti jokių specialių taisyklių. Tik puslapis turi būti padalintas į 9 lygius keturkampius. Trijų trečiųjų taisyklė jau yra naudojama kelis šimtus metų, o pirmą kartą buvo aprašyta 1797 metais knygoje „Remarks on Rural Scenery“ (John Thomas Smith). Jis aprašė, kaip panaudoti 3×3 koncepciją dailėje ir pirmasis panaudojo trijų trečiųjų taisyklės pavadinimą, kuris prigijo tarp menininkų, fotografų ir dizainerių.



27 pav. Trijų trečiųjų taisyklė

Kaip tinklelis atrodanti trijų trečiųjų taisyklė padeda suprasti, į kurias vietas paveiksle, erdvėje ar net interneto puslapyje atkreipiamas dėmesys. Visų pirma, dėmesys krinta ant susikertančių linijų (27 pav.). Tobulą simetriją galima naudoti, bet trijų trečiųjų taisyklė vis tiek veikia ir dėmesys kris į tam tikrus taškus.

Klausimai ir užduotys

1. Kas sudaro tinklalapio dizaino kūrimo procesą?
2. Kas sudaro svetainės dizaino kūrimo procesą?
3. Kokie yra 5 pagrindiniai tinklalapio dizaino elementai?
4. Kokios yra spalvų reikšmės?
5. Pagal ką skirstomos spalvos?
6. Kiek spalvų sudaro spalvų ratą?
7. Kaip sudaromos antrinės ir tretinės spalvos?
8. Kokie yra spalvų deriniai?
9. Kokie taikomi spalvų derinimo principai?
10. Kaip taikoma aukso proporcija kuriant tinklalapius?
11. Kaip taikoma trijų trečiųjų taisyklė tinklalapio dizaine?

TERMINŲ ŽODYNĖLIS

AKL	„Atviras kodas Lietuvai“ – asociacija, vienijanti atvirojo kodo ir laisvųjų programų judėjimų šalininkus Lietuvoje.
Apache	atvirojo kodo programinės įrangos kūrimo projektas ir pagrindinis to projekto produktas – daug metų pirmaujantis žiniatinklio serveris.
Atviroji programinė įranga	terminas, reiškiantis programinę įrangą, kuri platinama licencijomis, neribojančiomis jos kopijavimo ir tobulinimo, analogiškas terminui – laisvoji atvirojo kodo programinė įranga.
CSS	(angl. <i>Cascading Style Sheets</i>) – pakopinių stilių schemos. CSS leidžia tiksliai aprašyti HTML dokumentų išvaizdą bei išdėstymą.
DBVS	(angl. <i>Database management system (DBMS)</i>) – duomenų bazių valdymo sistema. Dažniausiai kalbama apie reliacines duomenų bases.
Flash	tinklalapių kūrimo technologija, leidžianti kurti judančius objektus.
FLOSS	(angl. <i>Free Libre Open Source Software</i>) – laisvoji atvirojo kodo programinė įranga arba atviroji programinė įranga.
FSF	(angl. <i>Free Software Foundation</i>) – laisvosios programinės įrangos fondas.
HTML	(angl. <i>HyperText Markup Language</i>) – hiperteksto dokumentų aprašymo kalba. HTML yra vienas iš svarbiausių žiniatinklio atvirųjų standartų.
IETF	(angl. <i>Internet Engineering Task Force</i>) – atvira standartų organizacija, kurianti interneto standartus ir skatinanti juos naudoti. Intensyviai bendradarbiauja su ISO ir W3C.
Internetinė svetainė, tinklalapis	(angl. <i>website, site</i>) – daugybė tarpusavyje hipernuorodomis susietų internetinių puslapių (tinklalapių).

ISO	(angl. <i>International Organization for Standardization</i>) – tarptautinė standartizacijos organizacija, kurią sudaro valstybių nacionalinių standartizacijos institucijų atstovai.
MySQL	laisvoji atvirojo kodo DBVS. AB MySQL yra šios DBVS kūrimą prižiūrinti akcinė bendrovė.
OEM	(angl. <i>Original Equipment Manufacturer</i>) – programinė įranga, parduodama kartu su naujais kompiuteriais ir kitais įrenginiais.
OS	(angl. <i>Operating System</i>) – operacinė sistema.
OSI	(angl. <i>Open Source Initiative</i>) – atvirojo kodo iniciatyvos organizacija.
PHP	(angl. <i>Hypertext Preprocessor</i>) – HTML skriptų (programavimo) kalba.
PĮ	programinė įranga.
PSD	„Adobe Photoshop“ šaltinių bylos formatas.
RFC	(angl. <i>Request For Comment</i>) – IETF organizacijos techninių ir organizacinių pastabų apie internetą rinkinys. Pastaruoju metu taip savo dokumentus vadina ir kitos organizacijos, pavyzdžiui, „Vikipedija“.
Šablonas	paruošta svetainė, kurią galima naudoti kaip pagrindą greitam ir geros kokybės puslapių kūrimui.
SQL	(angl. <i>Structured Query Language</i>) – struktūrizuota užklausų kalba, skirta duomenims aprašyti ir jais manipuluoti reliacinių duomenų bazių valdymo sistemose.
TVS	(angl. CMS – <i>Content Management System</i>) – turinio valdymo sistema.
Vikis	(angl. <i>Wiki</i>) – bendra tekstų redagavimo sistema, leidžianti daugeliui autorių kartu redaguoti tekstą naudojantis nesudėtingais sintaksės žymenimis. Pirmoji tokia sistema – 1994 m. sukurtas žiniatinklis „WikiWikiWeb“.

- W3C (angl. *World Wide Web Consortium*) – tarptautinė organizacija, kurianti žiniatinklio standartus.
- WWW (angl. *World Wide Web*) – pasaulinis žiniatinklis, visuma internete prieinamų resursų, pateikiamų W3C aprašytais atviraisiais formatais ir protokolais.
- XHTML (angl. *eXtensible HyperText Markup Language*) – išplečiama teksto su nuorodomis žymėjimo kalba. Patobulinta HTML kalbos versija, aprašyta laikantis XML reikalavimų. XHTML dokumentus galima apdoroti daugybe egzistuojančių įrankių, skirtų darbui su duomenimis XML formatu. Iš XHTML pašalintos nereikalingos HTML dalys, įdiegtas papildomas funkcionalumas (tobulesnis formų veikimas).
- XML (angl. *eXtensible Markup Language*) – išplečiama dokumentų žymėjimo kalba. W3C rekomenduojama bendros paskirties duomenų struktūrų ir jų turinio aprašymo kalba.
- Žiniatinklis lietuviškas WWW terminas.

LITERATŪRA IR ŠALTINIAI

1. AdWords raktažodžių paieškos įrankis [interaktyvus]. [Žiūrėta 2013-04-26]. <<https://adwords.google.com/select/KeywordToolExternal>>
2. Apache oficialus puslapis. [Žiūrėta 2013-04-29]. <<http://www.apache.org/>>
3. *Atvirųjų standartų vartotojo vadovas*. Atviras kodas Lietuvai, Latvijos atvirojo kodo asociacija. Vilnius, 2005.
4. Boulton M. *Designing for the Web. 2010 Five Simple Steps LLP*. [Žiūrėta 2013-04-26]. <<http://designingfortheweb.co.uk/book>>
5. Cover D. *Search Engine Optimisation Secrets*, Wiley Publishing, Inc. 2011, ISBN: 978-0-470-55418-0.
6. CSS pamokos lietuviškai. [Žiūrėta 2013-05-27]. <http://kodai.manualai.lt/css.html>
7. Dagienė V., Grigas G., Jevsikova T. *Enciklopedinis kompiuterijos žodynas*. Vilnius: TEV, 2008.
8. Hot Scripts. [Žiūrėta 2013-05-15]. <<http://www.hotscripts.com/>>
9. HTML Pamokos. [Žiūrėta 2013-04-15]. <http://www.webpamokos.lt/html_pamokos>
10. HTML5 Introduction. [Žiūrėta 2013-05-14]. <http://www.w3schools.com/html/html5_intro.asp>
11. Informacija apie web dizainą [Žiūrėta 2013-06-20]. <<http://www.frogsign.lt/category/dizainas/>>
12. Johnson D. *The classList API*. 2013. [Žiūrėta 2013-04-19]. <<http://html5doctor.com>>
13. Lietuviškas „Joomla“ palaikymo tinklalapis. [Žiūrėta 2013-05-19]. <www.lithuanianjoomla.com>
14. Lynch P. J., Horton S. *Web Style Guide Online*. [Žiūrėta 2013-05-07]. <<http://webstyleguide.com/wsg3/>>
15. Maskeliūnas S. Ontologijų panaudojimo galimybės, kuriant sudėtingas sistemas. *Informacinės technologijos. Konferencijos pranešimų medžiaga*. KTU, Kaunas, 2001.

16. Merrill D. *Mashups: The new breed of Web app*. 2009. [Žiūrėta 2013-05-10]. <<http://www.ibm.com/developerworks/xml/library/x-mashups.html>>
17. Murphey R. *jQuery Fundamentals*. 2012. [Žiūrėta 2013-06-12]. <<http://jqfundamentals.com/legacy>>
18. Nielsen J. *2000. Is Navigation Useful?* [Žiūrėta 2013-05-09]. <<http://www.nngroup.com/articles/is-navigation-useful/>>
19. O'Reilly T. *What Is Web 2.0. 2005*. [Žiūrėta 2013-04-20] <<http://oreilly.com/web2/archive/what-is-web-20.html>>.
20. *OpenSourceCMS, atviro kodo turinio valdymo sistemų puslapis*. [Žiūrėta 2013-04-26]. <<http://www.opensourcecms.com/>>
21. Pagrindiniai „JavaScript“ elementai. [Žiūrėta 2013-06-12]. <<http://www.programva.com/lt/html-http-pagalba/360-javascript-teorija-objektai-rezervuoti-zodziai>>
22. Perens B. *Microsoft and Apache - What's the Angle?* [Žiūrėta 2013-04-26]. <<http://itmanagement.earthweb.com/osrc/print.php/3762786>>
23. PHP oficialus puslapis. [Žiūrėta 2013-05-10]. <<http://php.net/>>
24. PHP pagrindai. [Žiūrėta 2013-06-08]. <http://kodai.manualai.lt/php/pagrindai.html>>
25. Pilgrim M. *Dive into HTML5*. [Žiūrėta 2013-04-26]. <<http://diveintohtml5.info>>
26. Slideshare Inc. *Seo patarimai ir apžvalgos* [interaktyvus]. 2011. [Žiūrėta 2013-06-28]. <http://www.slideshare.net/martynasjokubauskas/seo-patarimai-ir-apvalgos-seo-paslaugos>>
27. Spalvų derinimo pagrindai. [Žiūrėta 2013-06-20]. <<http://www.prodesign.lt/2010/11/25/spalvu-derinimo-pagrindai/>>
28. Vidžiūnas A., Barzdaitis V. *Interneto svetainių ir tinklalapių kūrimas*. Kaunas, 2005.
29. Way J. *28 HTML5 Features, Tips, and Techniques you Must Know*. 2011. [Žiūrėta 2013-04-29]. <<http://net.tutsplus.com/tutorials/html-css-techniques/25-html5-features-tips-and-techniques-you-must-know/>>
30. Wrigh R. Top Performing SEO/SEM Strategies, February 04, 2008.

PRIEDAI

1 priedas. Reikšmingos interneto istorijos datos

Metai	Įvykis
1957	SSRS paleidžia pirmąjį dirbtinį Žemės palydovą „Sputnik“. Atsakdamas į tai JAV Gynybos departamentas įkuria Perspektyviųjų gynybos tyrimo projektų agentūrą, angl. <i>Advanced Research Projects Agency</i> (ARPA), kuri padėtų JAV pirmauti taikant mokslo ir technologijų sritis karo pramonei.
1958	„Bell Telephone“ išleidžia pirmąjį modemą, kuris telefono linijomis perduoda skaitmeninius duomenis iki 300 bitų per sekundę.
1960	JAV Gynybos departamentas iškėlė idėją sujungti kompiuterius į tinklą, kuriam neturėtų įtakos atsitiktinės klaidos ir kuris būtų tinkamas ryšiams, ypač karo atveju (pažeidus bet kurią ryšio liniją, tinklo darbas nenutrūktų). Šį projektą turėjo įgyvendinti ARPA.
1969	„ArpaNet“ tinklu, kuris laikomas interneto prototipu, buvo sujungti keturi kompiuteriai.
1970	ARPA sukūrė interneto protokolą IP.
1971	Ray Tomlinsonas (g. 1941 m.) iš BBN sukuria elektroninio pašto programą žinutėms siųsti paskirstytame tinkle.
1972	Sukurta „Telnet“ specifikacija (RFC 318).
1973	Sukurta FTP specifikacija (RFC 454). Pirmieji tarptautiniai prisijungimai prie „ArpaNet“ – Londono universiteto koledžas ir Norvegijos Karališkoji radarų bendrija. Bobas Metcalfe'as disertacijoje pateikia „Ethernet“ idėją.
1976	„AT&T Bell Labs“ sukuria failo kopijavimo iš kompiuterio į kompiuterį programą UUCP („Unix to Unix cp“, kur „cp“ yra „Unix“ sistemos failų kopijavimo komanda). UUCP po metų išplatinama su „Unix“ sistema. Vėliau UUCP skirta programinė įranga kuriama ir kitoms sistemoms, UUCP tampa protokolu (angl. <i>Unix to Unix Copy Protocol</i>).

1977	Sukurta el. pašto specifikacija (RFC 733).
1979	Tomas Truscottas, Jamesas Tice Ellisas (1956–2001 m.) ir Stevenas M. Bellovinas panaudoję UUCP įkuria diskusijų grupių sistemą „UseNet“.
1982	DCA ir ARPA „ArpaNet“ tinkle įdiegiamas TCP ir IP junginys (TCP/IP) ir JAV gynybos departamentas paskelbia jį standartu. Sukuriama EGP (angl. <i>External Gateway Protocol</i>) specifikacija (RFC 827).
1983	Paulas Mockapetris sukuria domeno vardų serverį (DNS), RFC 882 ir RFC 883.
1984	Įsigali DNS sistema. Mazgų skaičius viršija 1 000. Amerikos gynybos departamentas paskelbia TCP/IP standartiniu visų kairnio pobūdžio tinklų protokolu. „Usenet“ tinkle atsiranda moderuojamų diskusijų grupių.
1986	Craigas Partridge'as sukuria MX įrašo formatą – tai leidžia ne IP tinklams turėti domenų vardus. Sukurtas NNTP protokolas.
1987	Billas Atkinsonas pritaikė hipertekstą „Apple“ kompiuteriams.
1988	Interneto tinklu prašliaužia „kirminas“ ir užkrečia 6 000 iš tuo metu veikusių 60 tūkst. mazgų. Sukurta pirmoji elektroninio pašto programa „Eudora“, pirmą kartą pritaikyta Iliojaus universitete.
1989	Europos interneto tinklams administruoti įkuriamas RIPE. Mazgų kiekis viršija 100 tūkst. ARPANET veikla nutrūko, susikūrė dabartinis interneto tinklas.
1991	Philipas Zimmermanas išleidžia PGP (angl. <i>Pretty Good Privacy</i>), skirtą kompiuterių tinklais perduodamiems duomenims šifruoti.
1992	Interneto mazgų kiekis viršijo milijoną. Lietuvai suteikta „lt“ aukščiausiojo lygio adresų sritis. Lietuvoje pradėjo veiklą pirmieji interneto paslaugų tiekėjai UAB „Elmeta“ ir KUB „Aiva sistema“.
1993	Naršyklės „Mosaic“ startas. Ziniatinklis (WWW) vystosi nematytu tempu.
1994	Internetu pasirodo el. parduotuvės ir pirmasis virtualusis bankas.

1995	„http“ tapo pirmaujančiu tinklo protokolu pagal perduodamą duomenų srautą. Paskelbta „Apache 1.0“ versija. Įsteigta „Netscape“ kompanija. Domenų vardai tapo mokami.
1996	Vasario 23 d. buvo išleistas pirmojo lietuviško interneto žurnalo „Vartiklis“ pirmasis numeris.
1998	Du Stenfordo studentai įkūrė paieškos sistemą „Google“.
2003	Pirmas lietuviškas straipsnis „Vikipedijoje“, lietuviškos laisvosios enciklopedijos kūrimo pradžia. Sukurta pelno nesiekianti organizacija „Mozilla Foundation“.
2007	Mobiliojo ryšio bendrovė „Omnitel“ save ir Lietuvą pristato beveik 10 mln. vartotojų turinčiame virtualiame pasaulyje „Second Life“.
2008	Ministro Pirmininko ir ministrų atsakymai į žurnalistų klausimus bei LR Vyriausybės spaudos konferencijos skelbiamos vaizdo įrašų publikavimo sistemoje „Youtube“.

2 priedas. Specialūs simboliai

Simbolis	Kodas	Aprašymas
"	"	Citatos pradžios ar pabaigos ženklas
&	&	Vadinamasis loginis (matematinis) „ir“ (and)
<	<	Ženklas mažiau
>	>	Ženklas daugiau
'	 	Ženklas reiškiantis vietą, kurioje negalimas perkėlimas ar eilutės dalijimas, bet yra tarpas
–	­	Trumpas brūkšniukas eilutės vidurio linijoje
©	©	Autorinių teisių (copyright) ženklas
®	®	Registruoto firmos ženklo (registered trademark) simbolis
™	™	Firmos ženklo (trademark) simbolis
¹	¹	Mažas vienetas virš eilutės vidurio linijos (superscript). Pavyzdžiui: x ¹
²	²	Mažas dvejetas virš eilutės vidurio linijos (superscript)
³	³	Mažas trejetas virš eilutės vidurio linijos (superscript)
¶	¶	Pastraipos pabaigos ženklas
•	·	Taškas eilutės vidurio linijoje
«	«	Dvigubas ženklas mažiau
»	»	Dvigubas ženklas daugiau
¼	¼	Viena ketvirtoji
½	½	Viena antroji
¾	¾	Trys ketvirtosios
¡	¡	Apverstas aukštyn kojomis šauktukas
¢	¢	Cento ženklas
£	£	Svaro sterlingų ženklas
¤	¤	Bendras valiutos ženklas

¥	¥	Jenos ženklas
 	¦	Ištisinis vertikalus brūkšnys
§	§	Pastraipos (skyriaus) ženklas
¬	¬	Loginės operacijos „ne“ (neigimo) ženklas
°	°	Laipsnių ženklas, pavyzdžiui C°
±	±	Ženklas "plus-minus"
μ	µ	Ženklas, reiškiantis priešdėlį „micro“ (graikiška raidė miu)
×	×	Daugybos ženklas
÷	÷	Dalybos ženklas

UŽRAŠAMS

UŽRAŠAMS

Laima Ričkutė

Tinklalapių kūrimas, dizainas ir valdymas. Mokomoji priemonė. – Klaipėda: Socialinių mokslų kolegija, 2013. – 240 p., 27 iliustr.

ISBN 978-9955-648-10-9

Tiražas 200 egz.

Išleido

VšĮ Socialinių mokslų kolegija
Nemuno g. 2, LT-91199 Klaipėda

info@smk.lt

www.smk.lt

Spausdino

Spaustuvė „Druka“

Mainių g. 5, LT-94101 Klaipėda

info@druka.lt

www.druka.lt