O’ZBEKISTON RESPUBLIKASI AXBOROT TEXNOLOGIYALARI VA KOMMUNIKATSIYALARINI RIVOJLANTIRISH VAZIRLIGI

TOSHKENT AXBOROT TEXNOLOGIYALARI UNIVERSITETI

FARG’ONA FILIALI

“ KOMPYUTER INJINIRINGI ” fakulteti

“DASTURIY INJINIRING” kafedrasi

“WEB Dasturlash”

fanidan

MARUZALAR MATNI

Farg’ona - 2015

Ushbu o’quv uslubiy ko’rsatma O’zbekiston Respublikasi Oliy va O’rta Maxsus ta’lim vazirligining Oliy O’quv yurtlari boshqarmasi 201__ yil _____da tasdiqlangan (ro’yxat t/r _____) namunaviy dasturi va “Dasturiy injiniringi” kafedrasida ishlab chiqilgan o’quv dasturi asosida tuzilgan va unga mos keladi

O’quv uslubiy ko’rsatma 5330500- Kompyuter injiniringi yo’nalishi talabalari uchun mo’ljallangan.

Tuzuvchilar: ass. Xashimov A

1-маъруза. Web иловаларни яратиш фанига кириш.

Режа:

I. Кириш.
II. Асосий қисм:
   1. Web-саҳифа, Web-сайт, Web-сервер;
   2. Интернет технологияси ҳақида
   3. Web-технология классификацияси;
   4. Разметкали тиллар: HTML, XML, XHTML, WML;
   5. Сценарийли тиллар. "Клиент-сервер" технологияси;
III. Хулоса.

Калит сўзлар: Web-саҳифа, Web-сайт, Web-сервер, HTML, XML, XHTML, WML, клиент-сервер технологияси.

Ишдан мақсад: Web-иловаларни яратиш фани бўйича ишлаш жараёнида талабаларда билим, ишлаш кўникмаси ва малакаларини шакллантириш ва бу жараённи амалга оширишда талабаларга маълумот беришни ташкил қилиш.

КИРИШ.

Бугунги кунда Интернетнинг оммавийлиги ҳақида гапириш ўринсиз. Интернет ҳаётимизнинг бир бўлагига айланди, биз унинг хизматларидан ҳар куни фойдаланишга одатландик. Ҳозирда ихтиёрий инсон Web-технологияларнинг инсон ҳаётининг талим, коммерсия, сиёсат, кўнгил очар, … бўлакларига кириб борганлигини тасаввур эта олади ва унинг гувоҳи ва фойдаланувчисига айланмоқда.

Интернет турли хил инсонларни ягона мақсад билан бирлашишига сабаб бўлмоқда. Ҳамма Интернет тармоғидан бирор турдаги ахборот олишга ҳаракат қилади. Шундай вақтлар келадики, ҳужжатни Интернетда чоп этиш малакаси ёзув машинасидан фойдаланиш каби ҳар бир, ҳатто ўрта маълумотга эга бўлган инсоннинг қўлидан келади.

Мазкур қўлланма Web-ҳужжатларни яратиш, уларни Интернетда чоп этиш, Web-ҳужжатни кўркамлаштириш, қизиқарли ва ўзига тортувчи қилиб яратиш, вақти келса маълумотларми янгилаш каби вазифаларни ўргатишга мўлжалланган.

Дастлабки Web-саҳифалар жуда содда тузилишга эга бўлиб, улар матнни форматлаш ва гиперкўрсаткичлардан таркиб топган эди. Web технологиялар ривожланиши натижасида Web саҳифалар таркибида Plug-in дастурлар

жойлаштирила бошланди, натижада Web саҳифаларга интер фаол хусусияти берилди. Web технологияларнинг ривожланишининг охирги натижаларидан бири бу скрипт тилларидир (Script Languages). Уларни ишлатишдан мақсад Web серверининг ишини енгиллаштириш, хар-хил ишлар учун Web серверини безовта қилмасдан, бундай масалаларни фойдаланувчи компютерининг ўзида яратишдир. Web технологиясининг охирги эришган ютуқларидан бири динамик Web саҳифалардир. Динамик Web саҳифалар CGI дастурлар билан бевосита боғлиқ бўлиб, CGI дастурлар серверда жойлашган ва сервер имкониятларини ишлатувчи дастурлардир. Улар серверга келган сўровларни қайта ишлайди ва қайта ишлаш натижасида янги Web саҳифа ҳосил бўлади.

Web саҳифа Интернет тармоқларида жойлашган файллар тўплами бўлиб, уларни сони соат сайин кўпайиб бормоқда. Бу файлларда маълумотларни турли хилларини: матн, график, тасвир, видео, аудио маълумотларни учратиш мумкин. Бугунги кунда Web Интернет ресурслари ичида энг оммавийси ҳисобланади. Чунки, аввалдан тайёрланган Web саҳифа орқали тегишли маълумотларни тўлдириш фойдаланувчининг қанчадан-қанча вақтини тежаш имконини беради. Шу боис математика ва информатика йўналишида таҳсил олувчи талабаларга Web технологияларни алоҳида курс сифатида ўқитила бошланди.

Ушбу қўлланма камчиликлардан холи эмас. Бундан ташқари турли ўқув муассасаларида ўқув ишлаб чиқариш амалиётини ўтказиш бўйича турли хил тажрибалар тўпланган. Биз қуйида Web-теҳнологиянинг асосий тушунчалари билан танишиб чиқамиз.

1. Web-саҳифа, Web-сайт, Web-сервер

Web-технологияни (Интернет-технология) ўрганишни Web-дизайннинг қуйидаги учта тушунчасини ўрганишдан бошлаймиз:

1. Web-саҳифа;
2. Web-сайт;
3. Web-сервер.

Технология грек тилидан (techne) таржима қилганда санъат, махорат, билиш маъноларини англатади, булар эса ўз навбатида жараёнлардир. Жараёнлар - бу қўйилган мақсадга эришиш учун маълум харакатлар мажмуасидир.

Web-саҳифа - ўзининг уникал адресига эга бўлган ва махсус кўриш дастури ёрдамида (браузер) кўрилувчи ҳужжатдир. Унга матн, графика, овоз, видео ёки анимация маълумотлар бирлашмаси - мултимедияли ҳужжатлар, бошқа ҳужжатларга гипермурожаатлар кириши мумкин.
Web-сайт - бир қанча Web-саҳифаларнинг мантиқий бирлашмаси.

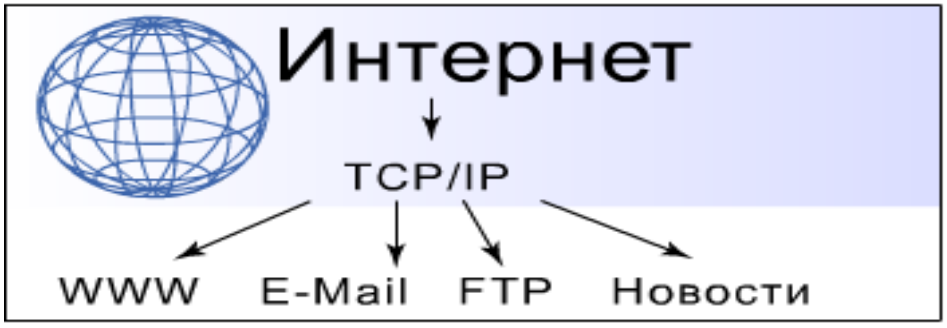
Web-сервер - тармоққа уланган компютер ёки ундаги дастур ҳисобланиб, умумий ресурсларни клиентга тақдим этиш ёки уларни бошқариш вазифаларини бажаради. Интернет тармоғини фойдаланувчиларга тармоқ ресурсларидан эркин фойдаланиш имкониятини берадиган Web- серверларсиз тасаввур этиб бўлмайди. Бундай серверларда Интернетда тақдим этилган ахборотнинг катта қисми жамланган.

Web-серверлар маълумотлар базалари ва мултимедияли маълумотларни бир бирига мослаштиради. Web-серверда Web-саҳифа ва Web-сайтлар сақланади.

Web-серверда мижоз компютери тизимини ташкил қилишнинг умумий тамойиллари нуқтаи назаридан мижоз-сервер технологиялари ишлатилади.

Ҳозирги кунда оддий Web-серверни яратиш технологиясини анча оддий вазифа деб ҳисобласа бўлади. Асосий қийинчилик сервер саҳифасини бадиий безашдан иборат.

2. Интернет технологияси ҳақида



Интернет тармоғининг ишлаш принсIPи TCP/IP (Transmission Control Protocol/Internet Protocol - маълумотларни узатиш қайдномаси/ Интернет қайдномаси) компютер тармоғида маълумотларни узатиш қайдномалари мажмуининг номидир. TCP/IP жумласи ўз ичига Transmission Control Protocol(TCP) ва Internet Protocol(IP) қайдномалар номларини бирлаштириб олган қайднома бўлиб, у шундай қоидалар мажмуики, TCP/IP барча компютер ишлаб чиқарувчи компанияларнинг мосламавий ва дастурий таъминот хамкорлигини таъминлайди. Бу қоида жумладан, TCP/IP пакети билан ишловчи Digital Equipment фирмаси компютерларидан PC компютерларига мурожат қилишни кафолатлайди. TCP/IP очиқ қайднома, бу шуни билдирадики, қайднома хақидаги барча маълумотлар чоп этилган ва ундан барча очиқ фойдаланади.

Кўпчилик фойдаланувчилар TCP/IP ни битта дастур деб ўйлашади. Аксинча, у тармоқнинг бир вақтнинг ўзида маълумот узатиш учун ишлаб чиқилган, ўзаро боғланган қайдномаларнинг бутун бир дастурлар оиласидир. TCP/IP тармоқнинг дастурлар қисми бўлиб, у TCP/IP оиласидаги хар битта қисм маълум бир аниқ мақсадга қаратилган: электрон почталарини юбориш, системага олис

масофалардан киришни таъминлаш, файлларни манзилларга жўнатиш, хабарларга йўл кўрсатиш ёки тармоқлардаги бузилишларни талқин қилиш. TCP/IP Интернет глобал тармоғида кенг фойдаланувчи қайдномалардир. У хам йирик корпоратив тармоқларда, шунингдек, компютерлар сони оз бўлган локал тармоқларда хам қўлланилади.

TCP (Transmission Control Protocol). Қабул қилувчи ва узатувчи компютерларнинг мантиқий боғланишга асосланган маълумотлар узатишини қўллаб - қувватловчи қайднома.

IP (Internet Protocol)- Маълумотлар узатишни таъминлайди

Интернетнинг пайдо бўлиши тарихи 60-йилларнинг охирида Америка ҳукумати томонидан асос солинган ARPANet (Advanced Research Projects Agency ташкилоти) ҳисоблаш тармоғига бориб тақалади. Тармоқ ҳарбий ташкилотларга хизмат қилган.

1980 йиллар бошларида маълумотларни узатишни бошқариш протоколи TCP/IP (Transmission Control Protocol/ Internet Protocol) га асос солинди. Тахминан шу вақтда маълум бўлдики, TCP/IP дан турли миллий ва халқаро даражадаги компютер тармоқларини боғлашда фойдаланиш мумкин.

1989 йилнинг охирида ARPANet мукаммал ҳолга етиб келди, лекин бу вақтга келиб кўпгина унивецитетлар ва илмий муассасалар Интернетга уланган эдилар. 1990 йиллар бошларида корпорациялар ҳам Интернетдан электрон почта орқали маълумотлар алмашишда актив иштирок этардилар. У вақтларда Миллий Илмий фонд тижорат мақсадида Интернетдан фойдаланишни таъқиқлаган эди. 1991 йилда бу чеклаш бекор қилинади ва Интернетдан ташкилот, муассаса, ноҳукумат ташкилотларининг фойдаланиш даражаси ортди, шунингдек, тижорат мақсадида Интернетдан кенг фойдаланила бошланди (Интернет магазинлар, Интернет рекламалар ва ҳ.к.).

1993 йилда биринчи Web-браузер Mosaic пайдо бўлди.

Биз Интернет тармоғидаги Web-саҳифаларни кўришимиз учун WWW (World Wide Web) деб аталувчи сервисдан фойдаланамиз.

World Wide Web (WWW, Бутун дунё ўргимчак тўри) - бу клиент-сервер технологияси асосида ташкил этилган, кенг тарқалган Интернет хизматидир.

WWW (World Wide Web) - бу қанақадир Интернетдан ажратилган маълум бир жой эмас, компютер алоқа ўрнатадиган бирор нима ҳам эмас. Бутунжаҳон ўргимчак тўрини Интернет доирасидаги хизмат дейиш тўғрироқ. Web-серверлар деб аталувчи маълум протоколлардан, компютерлардан фойдаланиш орқали (чунки улар тармоққа уланган ва сервер дастурий таъминотига эга) Интернет хизмати йўлга қўйилади.

Компютер Web-сервер бўлиши учун Интернетга уланган ва сервер дастурий таъминоти (ДТ) га эга бўлиши етарли. Бу ДТ билан Windows, Mac OC, UNIX каби операцион системалар таъминлай олади. Web-сервер ҳар доим Интернетда “ўтиради” ва талаб қилинган томонга керакли информацияни жўнатади.

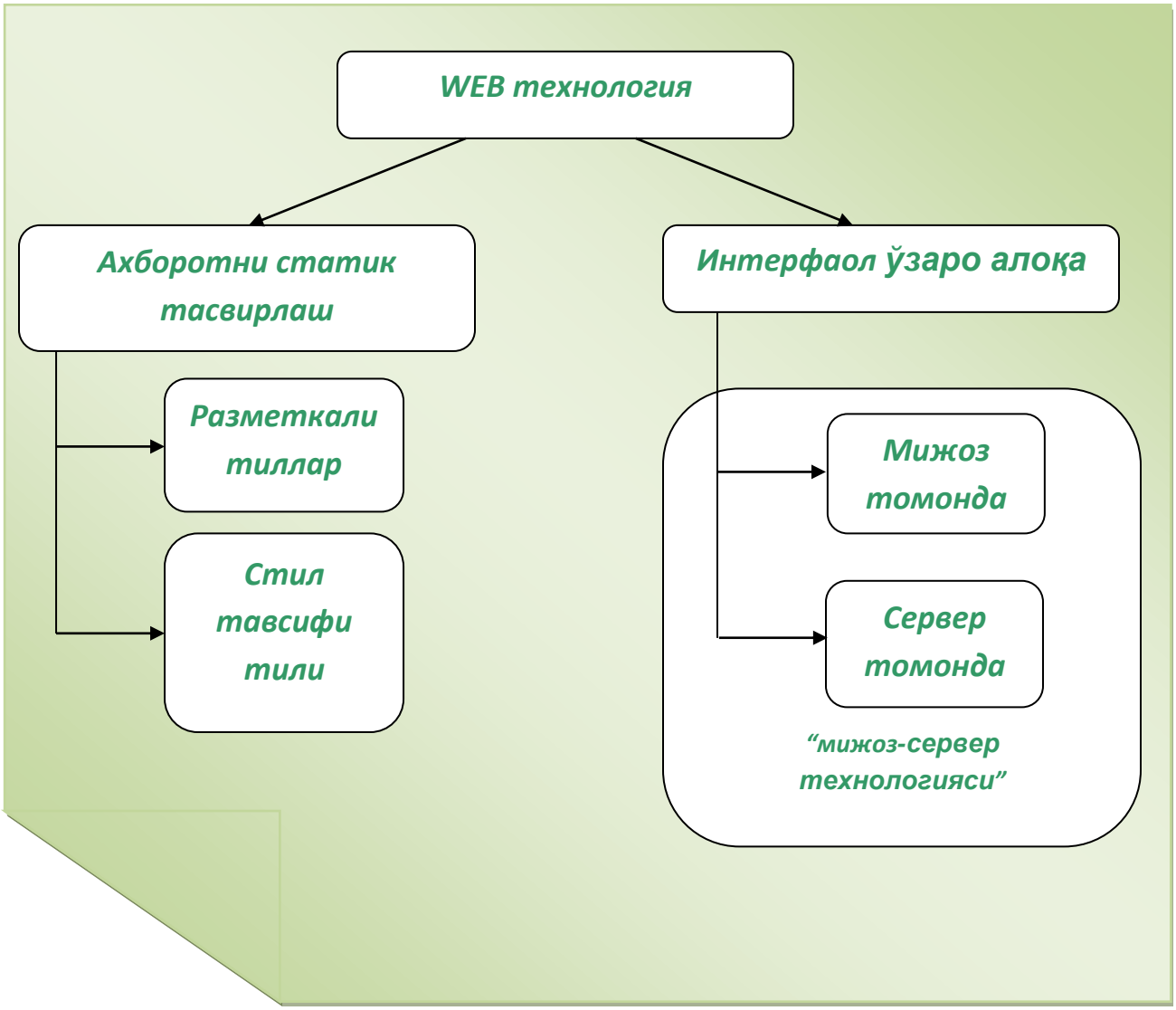
Webда ҳужжатлар билан ишлаш мумкин. Қуйида улардан баъзи бирлари билан танишиб чиқамиз.

Ҳужжатлар билан ишлашни тезлаштириш

Ҳужжатлар билан ишлашни тезлаштириш мақсадида Webда маълум буйруқлар мавжуд. WWW буйруқлар рўйхати қуйидагилардир:

b	аввалги ҳужжатга қайтиш;
o	ҳужжатнинг кейинги саҳифасига ўтиш;
g **h**	Кўрсатилган ахборот манбаига бевосита ўтиш; сахифани чиқариш (ёрдам бериш йўли билан);
Ho	Бошланғич ҳужжатга қайтиш;
I	Жорий ҳужжатда бошқа ҳужжатларни мурожаатларини кўрсатиш;
m	Программадан фойдаланиш хақидаги маълумотларни экранга чиқариш
n	Аввалги ҳужжатдан кейинги мурожаатга ўтиш;
con	Ҳужжатга мурожаат билан ўтиш;
quit	WWWдан чиқиш;
V	Кўриб чиқилган ҳужжатларнинг рўйхатини чиқариш;
vcon	Бир сахифа пастга ўтиш;
return	Бир сахифа пастга ўтиш;
t	Ҳужжатнинг кейинги сахифасига қайтиш;
u	Ҳужжат ичида бир сахифага юқори чиқиш.

3. Web-технология классификацияси.



Web-теҳнологияни асосини қуйидаги иккита тушунча ташкил қилади:
1. Ахборотни статик тасвирлаш;
2. Интерфаол ўзаро алоқа.

Ахборотни статик тасвирлаш. Маълумотлар сегментида жойлашган маълумотлар статик маълумотлар деб аталади. Бунга асосий сабаб, улар учун хотира ишлаш жараёни давомида ажратиб қўйилади. Ишлаш давомида эса бу хотира ўзгармай қолади. Тўпламдаги хотира эса ишлаш давомида тўлдириб борилади ва зарур бўлган пайтда бу хотира бўшатиб қўйилиши мумкин.

Разметкали тиллар. Бу тил ёрдамида матнлар, график маълумотлар Web-саҳифа ҳужжатга жойлаштирилади ва бу ҳужжатни барча компютерда кўриш имконияти мавжуддир. Бундай махсус тиллар разметкали тиллар деб аталади.

Уларнинг асосий вазифаси - Web-саҳифага “маълумотларни жойлаштириш” ва улар орасидаги алоқани (гиперсалоқалар) таъминлашдан иборат.

Web-дастурлаш технологияларини, дастурларини, интерфаол ўзаро алоқа қисмини ҳам асосан иккита қисмга ажратиш мумкин:

1. клиент томонидаги дастурларлаш (слиент-сиде);
2. сервер томонидаги (сервер-сиде).

Клиент томонидаги сценарийлар фойдаланувчи томонидан киритилаётган маълумотларни тўғрилигини серверга мурожаат қилмасдан текширади. Кўп ҳолларда бу сценарийлар JavaScript ва VBScript тилларида ёзилади.

Сервер томонида бажарилиши керак бўлган сценарийлар одатда сайт папкасининг ичидаги махсус папкага жойлаштирилади.

4. Разметкали тиллар: HTML, XML, XHTML, WML.

Web-технологиянинг (Интернет-технология) Web-дизайн қисмини ўрганишни разметкали тил таснифи билан бошлаймиз.

Махсус тил мавжуд бўлиб, бу тил ёрдамида матнлар, график маълумотлар Web-саҳифа ҳужжатга жойлаштирилади ва бу ҳужжатни барча компютерда кўриш имконияти мавжуддир. Бундай махсус тиллар разметкали тиллар деб аталади. Уларнинг асосий вазифаси - Web-саҳифага “маълумотларни жойлаштириш” ва улар орасидаги алоқани (гиперсалоқалар) таъминлашдан иборат.

Разметкали тиллар қуйидагиларни ўз ичига олади:

HTML (HyperText Markup Language)

Дастлаб World Wide Web тизими матнли маълумотларни ва HTML ҳужжатларни кўришга мўлжалланган, матнни тахрирловчи тилга ўхшаш тизим бўлган. Айни дамда HTML тили WWW дага энг оммабоп тиллардан бири ҳисобланади. HTML тилида ёзилган маълумотлар ўз ичига матн файллар, график маълумотлар ва бошқаларни олади.

Ҳужжатлар орасидаги алоқани таъминлаш ва маълумотларни форматлаш воситалари тег (tag) деб аталувчи восита орқали амалга оширилади.

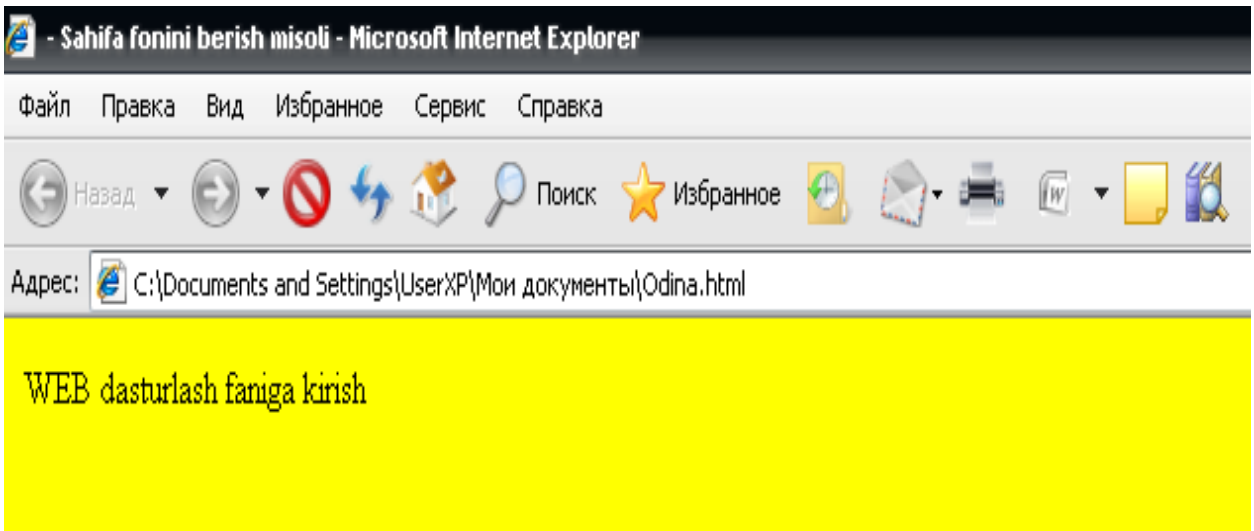
Web-саҳифанинг матн ва теглари аралаш равишда HTML-ҳужжат деб аталувчи файлининг ичига жойлаштирилади. Қандай тегни қўллаганингизга қараб браузер ойнасида маълумотлар турлича кўринади. HTML ҳужжатга маълумотларни жойлаштириш ва таҳрирлаш учун юзлаб теглар мавжуд. Масалан, ва теглари абзацни ташкил этади, ва жуфт теглари эса, матнни ёзма (курсив) ҳолда кўрсатиш учун қўлланилади. Шу билан бирга гиперматнли SSIлкалар теглари ҳам мавжуд. Ушбу элементлар фойдаланувчига гиперматн устига сичқонча курсори босилганда бошқа ҳужжатга боғланиш имконини беради.

Бутунжахон ўргимчак тўрининг асосий ва HTML нинг таркибий қисмини гиперматнлар ва гипермурожаатлар ташкил этади. Махсус командалар ёрдамида матннинг маълум қисми шундай ажратиладики, натижада ўша матн устига сичқон тугмаси босилса бошқа матн ёки сахифа очилади. Бундан ташқари мултимедия воситаларининг ишлаб кетиши ёки бўлмаса, маълумотни дискда сақлаш таклифи ҳам берилиши мумкин.

Қуйида биз HTML тилида тузилган дастур коди билан танишиб чиқамиз:

```
<HTML>
        <HEAD>
                <TITLE> -  Sahifa fonini berish misoli </TITLE></HEAD>
        <BODY BGCOLOR = YELLOW TEXT = BLACK LINK = RED
        VLINK = PURPLE ALINK = GREEN>
                        WEB dasturlash faniga kirish
        /BODY>
</HTML>
```

Бу дастурни ишга тушириш натижасида қуйидаги ойна очилади:



Гиперматн ёки гипермурожаат бирор бир тасвирга ҳам қўйилиши мумкинки унинг устига босилганда ҳам юқорида айтилган ҳолатлар рўй бериши мумкин.

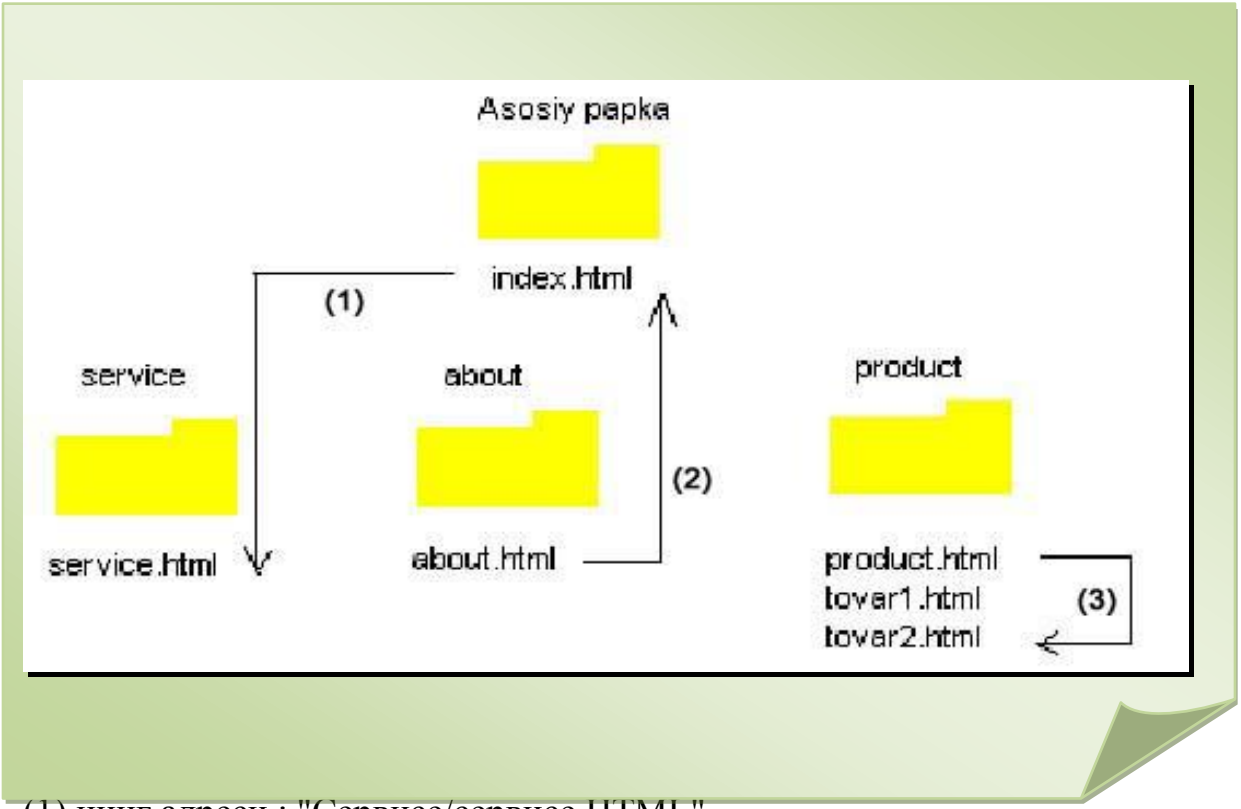
Ҳар бир Web-сахифа ўзида бир нечта гиперматн ёки гипермурожаатларни мужассам этиши мумкин.

Гипермурожаатлар Web-сайтлар бўйлаб харакатнинг асоси хисобланади. Мурожаатни танлаганда фойдаланувчи браузер ойнасига юкланувчи ёки ёрдамчи программани ишга тушурувчи URL билан боғланган адресга «тушиб» қолади. Баъзан гипермурожаат натижаси э-маил ёки ФТП серверга йўлланма берувчи янги Web-сахифани очилишига олиб келади.Фойдаланувчи мурожаатни танлаб олиши учун Web-дизайнер уни яратиши керак.

Гипермурожаат яратиш учун <a> (anchor, якор) элементидан фойдаланилади. У ўзида йўлланма берувчи URL адресни кўрсатувчи ҳреф атрибути билан тўлдирилади. Шунинг учун гипермурожаатни яратиш учун URL адресни аниқлаб олиш керак.

Гипермурожаат яратиш учун (анчор, якор) элементидан фойдаланилади. У ўзида йўлланма берувчи URL адресни кўрсатувчи ҳреф атрибути билан тўлдирилади. Шунинг учун гипермурожаатни яратиш учун URL адресни аниқлаб олиш керак.

Гипермурожаатни яратишда агар Интернетдаги хизмат ёки адресдан фойдаланмоқчи бўлсак албатта унинг тўлиқ адресини кўрсатиш шарт. Агар ўзимизда бор бўлган Web-сахифалардан гипермурожаатлар яратмоқчи бўлсак баъзи бир ишни осонлаштирувчи холатлар мавжуд:



(1) нинг адреси : "Сервисе/сервисе.HTML"

Асосий папкадан ихтиёрий ички папкадаги веб-сахифага мурожаат : "Папка_номи/файл_номи.HTML" кўринишда берилади.

(2) нинг адреси : "../индех.HTML"

Ихтиёрий ички папкадан асосий папкадаги асосий Web-сахифага мурожаат: "../асосий_файл.HTML" кўринишда берилади.

(3)нинг адреси : "Товар2.HTML"

Бир папкадаги веб-сахифалардан бир-бирига мурожаат : "файл_номи.HTML" кўринишда берилади

Интернет хизматларининг кўпчилигига доступ (рухсат, йўл) адресация схемаси (URL) ёрдамида қизиқтирилган ихтиёрий ҳужжатни топиш имкониятини беради. Ҳар бир тур бошқасидан фарқ қилувчи ўзининг формат адресига эга.

URL дан фойдаланиб, Web-браузерлар ёрдамида ихтиёрий ҳужжат ва хизматларга доступ олиш мумкин. URL қуйидаги тартибда ёзилади:

Protokol://internet_adres /yo'l /fayl_nomi.kengaytma

ёки **Protokol://internet_adres**

URL га мисол:

http://www.microsoft.com/windows/index.html

Бу ерда:

http:// - протокол;

www.microsoft.com – **internet_adres**(Microsoft компаниясининг Web- серверининг номи)

/windows/ - йўл

index - файл_номи

html - кенгайтма

URL да қўлланиладиган протоколлар рўйхати:

Протокол номи	Протокол нимага доступ бериши мумкинлиги
http://	HTTP (веб) серверларига
https://	Шифрланган баъзи бир HTTP (Web) серверларга
file://	Фойдаланувчи қаттиқ дискидаги файлларга
ftp://	FTP сервер файлларига
gopher://	Гопҳер меню ва файлларига
news://	Усенет янгиликлар серверлари группасига

news:	Аниқ Усенет янгиликлар группасига
mailto:	Аниқ электрон почта адресига
telnet:	Телнет удален серверига

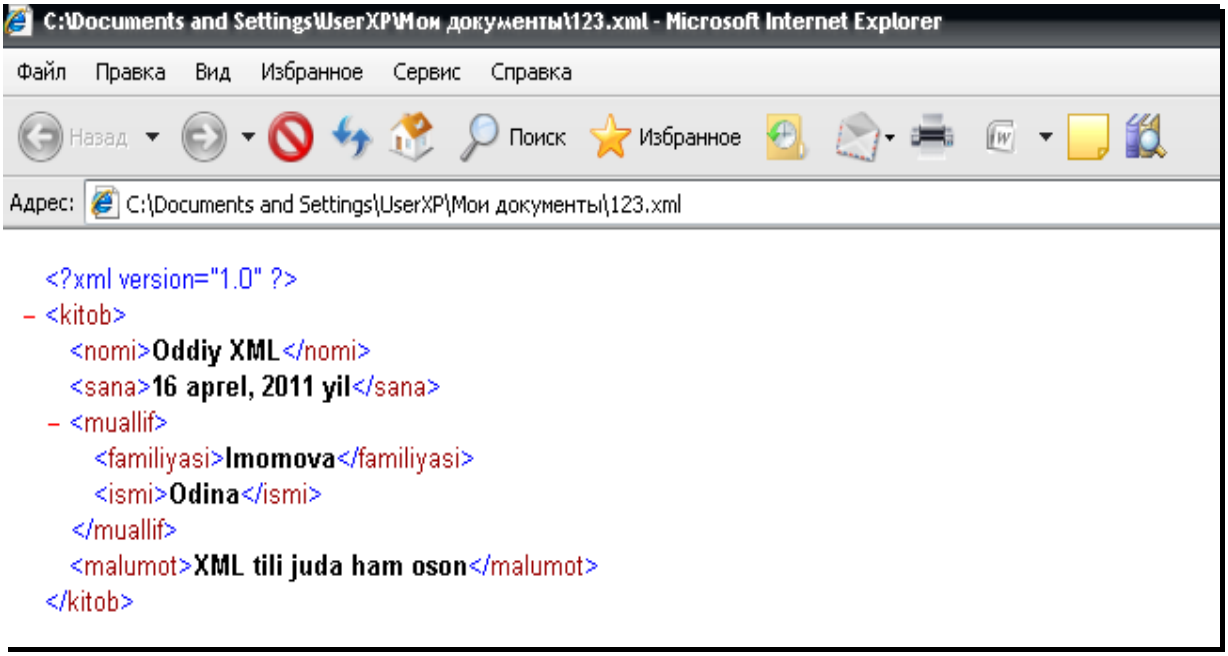
XML (eXtensible Markup Language).

XML тили ҳам HTML тилига ўхшаш тил хисобланади. HTML дан фарқли томони шундаки, XML да дастурчи ўзининг шахсий тегларини яратади ва улар орасига маълумотлар жойлаштиради. XML-теглар ҳарфлар катта кичиклигини фарқлайди. HTML теглари ҳужжатни экранда кўринишини ифодалайди. XML теглари ҳужжатдаги маълумотларни тавсифлаш учун ишлатилади. Ундан ташқари XML ёрдамида янги тегларни яратиш мумкин. XMLда маълумотлар тузилмавий ҳолда сақланади. XML асосан маълумотлар алмашинувида кўп ишлатилади, чунки XML платформадан мустақил бўлиб, ҲТТП орқали ишлаши жуда қулай.

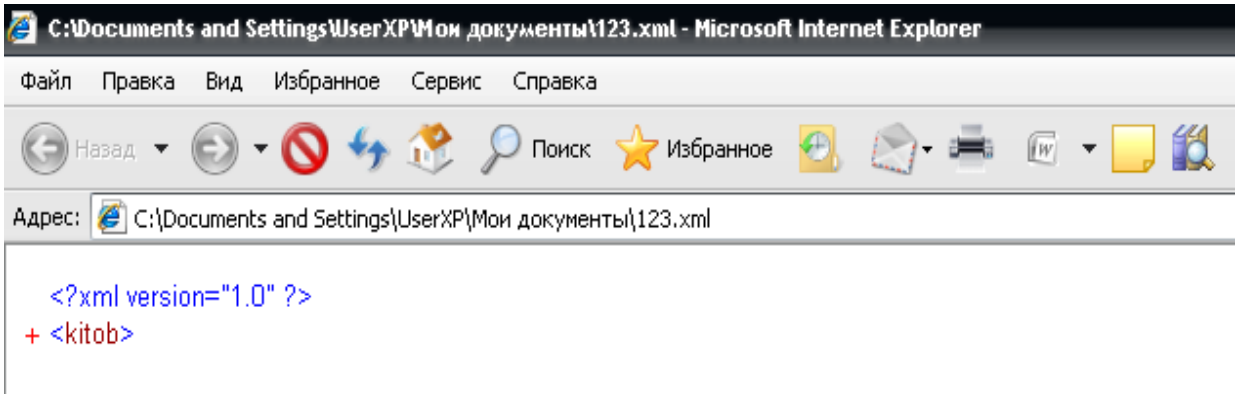
Қуйида XMLда тузилган дастур билан танишамиз:

```
<?xml version = "1.0"?>
        <kitob>
                <nomi> Oddiy XML </nomi>
                <sana> 16 aprel, 2011 yil </sana>
    <muallif>
                <familiyasi>Imomova</familiyasi>
                <ismi> Odina </ismi>
    </muallif>
    <malumot> XML tili juda ham oson </malumot>
         </kitob>
</kitob>
```

Бу дастур блокнотга ёзилади ва <дастур_номи>.XML кўринишида сақланади. Бу дастурни ишга тушириш натижасида қуйидаги ойнага эга бўламиз:



Буни қуйидагича қисқартирилади:



XHTML

XHTML тили HTML ва XML тилларининг бирлашмасини ташкил этади. XHTML тилида ёзилган ҳужжатнинг ташқи кўриниши платформага боғлиқ (Windows, Mac yoki Unix) равишда ўзгариб кетмайди. Шунга қарамай XHTML таркибида HTML дискрипторлардан фойдаланилади.

Бугунги кунда мобил алоқа воситаларидан фойдаланувчилар учун янги тил ишлаб чиқилган бўлиб, у WML (Wireless Markup Language) деб аталади; CDF (Channel Definition Format) - Мисрософт ишлаб чиққан браузерларда пуш-канал ҳосил қилишда қўлланилади.

5.Сценарийли тиллар. "Клиент-сервер" технологияси

Ҳозирда Web-саҳифанинг ривожланиши янада интерактив поғонасига чиққан. Web-сайтлар аста секинлик билан иловалар интерфейсига ўхшаб бормоқда. Буларнинг барчаси замонавий Web-дастурлаш технологияси ёрдамида амалга ошмоқда.

Web-дастурлаш технологияларини, дастурларини асосан иккита қисмга ажратиш мумкин: клиент томонидаги дастурларлаш (client-side) ва сервер томонидаги (server-side). Ушбу технологияларни тушуниш учун аввало бевосита "клиент-сервер" технологиясини тушуниш керак.

Web-саҳифанинг интерактив дастури сценарий деб аталади.

Бундай атама дастурнинг натижасига боғлиқ ҳолда вужудга келган. Унинг асосий вазифаси Web-саҳифасида фойдаланувчи ҳолатига, ҳаракатига «реаксия» беришдир.

Шу тариқа сценарийлар клиент томонида бажарилувчи ва сервер томонида бажарилувчи сценарийларга бўлинади. Клиент томонида бажарилувчи сценарийлар браузер ёрдамида бажарилади. Сервер томонида бажарилувчи сценарийлар эса Web-сервер ёрдамида бажарилади.

Клиент томонидаги сценарийлар.

Клиент томонидаги сценарийлар фойдаланувчи томонидан киритилаётган маълумотларни тўғрилигини серверга мурожаат қилмасдан текширади. Кўп ҳолларда бу сценарийлар JavaScript ва VBScript тилларида ёзилади.

JavaScript

JavaScript - бу тил Netscape ва Sun Microsystems томонидан яратилган бўлиб, Web-саҳифанинг функсионал имкониятларини орттириш мақсадида қўлланилади.

JavaScript ёрдамида одатда маълумотли ва мулоқот ойналарини чиқариш, анимацияларни кўрсатиш каби вазифаларни бажариш мумкин. Бундан ташқари, JavaScript-сценарий баъзан ўзи ишлаб турган браузер ва платформа типини аниқлаш мумкин. JavaScript-сценарийлар фойдаланувчи томонидан киритилаётган маълумотларни тўғрилигини текширишда ҳам қулай ҳисобланади.

VBScript

VBScript тили Мисрософт корпорацияси томонидан яратилган бўлиб, , Visual Basic тилининг бир қисми ҳисобланади. VBScript тили Internet Explorer ва Microsoft Internet Information Server (IIS) лар билан ишлашга мўлжалланган тилдир.

VBScript тилининг JavaScript тили билан умумий қисмлари бир нечта, жумладан у айнан Microsoft Internet Explorer билан ишлаш ва унинг қўлланиш соҳасини чеклай олиш имкониятига эга. VBScript интерпретаторли тил

ҳисобланиб, Мисрософт нинг Web-технологиялари билан ҳамкорликда ишлай олади, масалан ASP (Active Server Page) билан. Шунга қарамай VBScript клиент томонида ишловчи сценарий ҳисобланади, ASP эса сервер томонида ишлайди.

Сервер томонидаги сценарийлар.

Сервер томонида бажарилиши керак бўлган сценарийлар одатда сайт папкасининг ичидаги махсус папкага жойлаштирилади. Фойдаланувчи сўровига асосан сервер бу сценарийни бажаради. Бажарилган сценарий натижаси Web-серверга узатилади ва ундан сўнг клиентга узатилади. Сервер томонидаги сценарийларни ташкил этиш учун одатда Perl, ASP, PHP, JSP ва SSI каби тил ва технологиялардан фойдаланилади.

Perl

Perl тили Web-иловалар яратишда энг оммабоп тиллардан бири ҳисобланади. Матнларни қидириш ва таҳрирлаш, файллар билан қулай ишлай олиш қоидалари билан Perl тили Интернет нинг асосий тилларидан бири бўлиб қолди. Perl - интерпретаторли тил ҳисобланади, шу боис унда яратилган сценарийлар ишлаши учун сервер компютерда Perl-интерпретатор ўрнатилган бўлиши керак.

Бевосита Perl-коднинг интерпретация қилиниш жараёни унинг самарадорлигини пасайтиради. Бугунги кунда Perl нинг асосий ютуқларидан, унинг барча платформалар учун ишлай олиши ва унинг барча ресурслари бепул тарқатилаётганлигидир. Кўпгина Web-серверлар UNIX да ишлайди, Perl интерпретатор эса бу операцион тизимнинг бир қисми ҳисобланади.

ASP (Active Server Pages)

ASP-маълумотлар базалари ташкил этиш ва улар билан ишлаш вазифаларини бажаришда жуда мослашувчан, қулай воситадир. ASP воситалари сервер томонида ишлайди ва HTML-код ва сценарийлар каби файлларни қайта ишлайди. ASP технологияси VBScript, Java ва JavaScript тилларини қўллаб қувватлайди. ASP-код ихтиёрий HTML-ҳужжатдан, шу билан бирга бошқа ASP-ҳужжатдан чақирилиши мумкин. ASP-код жойлаштирилган Web-саҳифалар файллари кенгайтмаси .ASP бўлади.

ASP технология Windows NT ва Мисрософт ИИС Web-серверига мўлжалланган ҳисобланиб, имкониятлари ва самарадорлиги юқори бўлганлиги боис кўпгина компаниялар ўз воситаларига ASP ни қўллаб қувватлаш имкониятларини киритмоқдалар. ASP-воситаларини ишлаб чиқиш бўйича йирик компания **Чиллсофт Лидер среди независимых производителей ASP-средств - компания** Чиллсофт UNIX нинг бир қанча тури ва турли Web-серверларда ASP

ни қўллаш имкониятини киритган. Кўпгина HTML-мухаррирлар, масалан Adobe GoLive ҳам ASP ни қўллаб қувватлайди.

ASP технологияси бир нечта қулайликларни ўзида жамлаган: HTML-ҳужжатни динамик генерацилайди, формаларни қўллаб қувватлайди, маълумотлар базасига рухсатни ташкил этади ва у билан ишлай олади. ASP - дастурлаш тили ҳам, илова ҳам эмас, у интерактив Web-саҳифа ҳосил қилиш технологияси.

PHP

PHP - бу серверда қайта ишланувчи сценарийлар тилидир. ASP каби PHP кодлар ҳам бевосита HTML-ҳужжатни таркибига қўшилади. Ушбу тилнинг номи Personal Home Page Tools сўзларининг қисқартмасидан олинган. PHP да C ва Perl тилларида учраган бир қатор муаммолар ҳал этилган, бундан ташқари, PHP маълумотлар базаси билан ишлаш учун жуда қулай воситадир. Умуман олганда Perl, PHP - очиқ тизимли тиллар ҳисобланади ва уларни дастурчилар модернизациялаштира олади.

Қуйида PHP тилида ёзилган дастур кодини кўриб чиқамиз:

```
<php
    echo("<p>Hello</p>"); // izox
    echo("<p>Hello</p>"); # izox
        /*
      bu ham izox
        */
```

JSP

JSP (JavaServerPage) технологияси ўзининг функсионал имкониятларига кўра ASP га ўхшашдир. Асосий фарқи шундаки, бунда VBScript ва JavaScript билан бирга Java тили ҳам қўлланила олади. Шунга қарамай JSP Java дан олдинроқ қўлланилган ва ушбу технология мукаммал Web-иловалар яратиш учун етарли имкониятга эга.

SSI

SSI (Server Side Include) воситаси дастлаб HTML-файлни дастлаб серверда қайта ишлайди ва ундан сўнг уни клиентга узатади. Дастлабки қайта ишлаш вақтида ҳужжатга динамик генерация қилинган маълумотлар қўшилади, масалан жорий вақт ҳақидаги маълумот. Умуман олганда SSI технологияси HTML-файлнинг таркибига қўшимча қўлланмалар қўшишга мўлжалланган, HTMLнинг қисми ҳисобланади.

Хулоса

Бугунги кунда ҳаётимизнинг ҳар бир соҳасида интернет теҳнологияларидан фойдаланмоқдамиз.Интернет турли хил инсонларни ягона мақсад билан бирлашишига сабаб бўлмоқда. Интернет теҳнологияларининг муҳим элементларидан бири бўлган Web- теҳнологиялар ҳам тараққий этиб боряпди. Ҳозирда ихтиёрий инсон Web-технологияларнинг инсон ҳаётининг та’лим, коммерсия, сиёсат, кўнгил очар , … бўлакларига кириб борганлигини тасаввур эта олади ва унинг гувоҳи ва фойдаланувчисига айланмоқда. Web-дастурлаш фани асосида турли дастурлаш тиллари ёрдамида Web-сайтлар яратиш мумкин. Ҳар бир дастурлаш тилининг ўзига ҳос афзаллик ва камчиликлари бор. Ушбу қўлланмада уларни фарқли томонларини кўриб ўтдик.

Web саҳифа Интернет тармоқларида жойлашган файллар тўплами бўлиб, уларни сони соат сайин кўпайиб бормоқда. Бу файлларда маълумотларни турли хилларини: матн, график, тасвир, видео, аудио маълумотларни учратиш мумкин. Бугунги кунда Web Интернет ресурслари ичида энг оммавийси ҳисобланади. Чунки, аввалдан тайёрланган Web саҳифа орқали тегишли маълумотларни тўлдириш фойдаланувчининг қанчадан-қанча вақтини тежаш имконини беради. Шу боис математика ва информатика йўналишида таҳсил олувчи талабаларга Web технологияларни алоҳида курс сифатида ўқитила бошланди.

Бу фанни ўзлаштиришга бўлган чуқур интилиш келажакда яратилажак замонавий аҳборот теҳнологияларини савиясини кўтариши шубҳасиз.

Назорат саволлари:

1. Интернет технология деганда нимани тушунасиз?
2. Қандай разметкали тилларни биласиз?
3. Клиент-сервер технологияни қандай тушунасиз?
4. Интернет тармоғидаги Web-саҳифаларни кўришимиз учун қўлланиладиган теҳнологиялар?
5. Web-технология классификацияси?
6. Қандай теглар жуфт теглар деб аталади?
7. Сценарий нима?
8. Клиент томонида бажарилувчи сценарийлар нима ёрдамида бажарилади?
9. Сервер томонида бажарилувчи сценарийлар нима ёрдамида бажарилади?
10. Гипералоқалар нима?

2-маъруза.

HTML разметкали тилида кириш

Режа:

1. HTML кодларни ёзишда қўлланиладиган матн муҳаррирлари ҳақида маълумот;
2. HTML тили ҳақида қисқача маълумот;
3. Хужжат тузилиши;
4. Ҳужжатнинг HEAD бўлими;
5. Ҳужжатнинг BODY бўлими;
6. HTML шаблонларни яратиш;
7. HTML шаблонларни сақлаш ва синаб кўриш**;**

Калит сўзлар: Матн мухаррирлари, код мухаррирлари, WYSIWYG технологияси, HTML, HEAD, BODY

Ишдан мақсад: HTML асосий тушунчаларини ўрганиш, унинг сарлавҳа ва тана қисмига тегишли бўлган асосий теглар билан танишиш, талабаларга гиперматнли ҳужжат ташкил этиш асослари, манбалари, HTML-ҳужжат ҳақида тушунчалар

бериш ва HTML-ҳужжат ярата олиш кўникмаларини шакллантиришни ташкил этиш;

1. HTML кодларни ёзишда қўлланиладиган матн муҳаррирлари ҳақида маълумот;

Windows мухитидаги матн мухаррирлари: Notepad, TextPad, UltraEdit, EdutPlus. Кўрсатилган хамма матн мухаррирлари ёрдамида HTML кодларни ёзишда қўллаш мумкин.

Кўпгина матн мухаррирларида HTML кодларни хатто программалаштириш тилларида кодларни ёзиш жуда кулай. Баъзилари автоматик равишда операторлар, функсияларни таниш ва уларни хар-хил рангларда тасвирлаш имкониятига эга. Баъзи матн мухаррирларида HTML хужжатни web браузерда синаб кўриш тугмаси мавжуд.

HTML хужжатни яратишга мўлжалланган махсус программалар (HTML мухаррирлар) хам мавжуд: FrontPage, Adobe GoLive, Macromedia Dreamweaver, Nestcape Composer. Мухаррирлар 2 турга бўлинади:

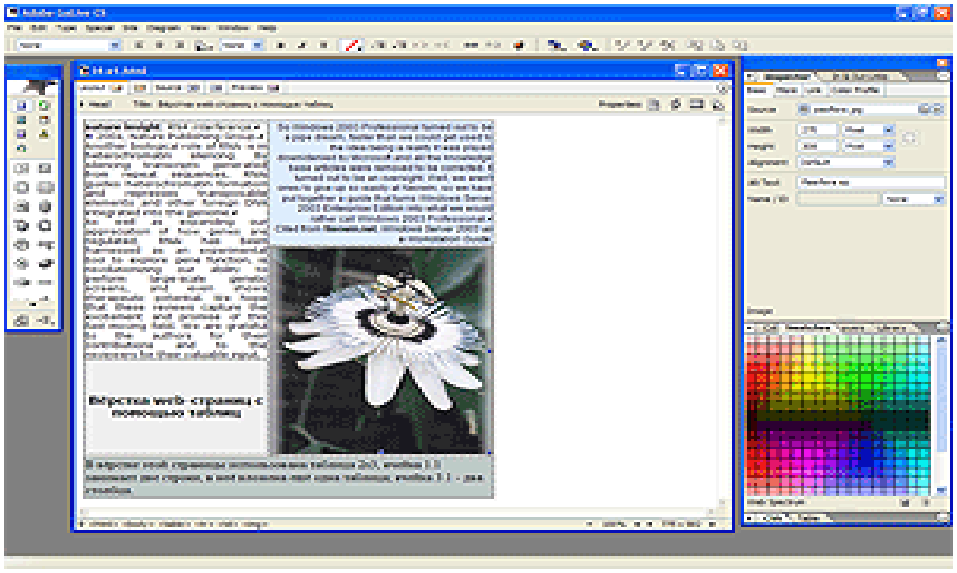
- ✓ код мухаррирлари;
- ✓ WYSIWYG технологияси (What You See Is What You Get –нимани кўрсанг ўшани оласан) асосида ишлайдиган мухаррирлари. Бу мухаррирлар ёрдамида фойдаланувчи HTML командаси ва элементларини ёзмайди, оддий матн мухаррирларидек матн ёзади, тасвирларни керакли жойга жойлаштиради ва х.к. холос.

WYSIWYG-мухаррири

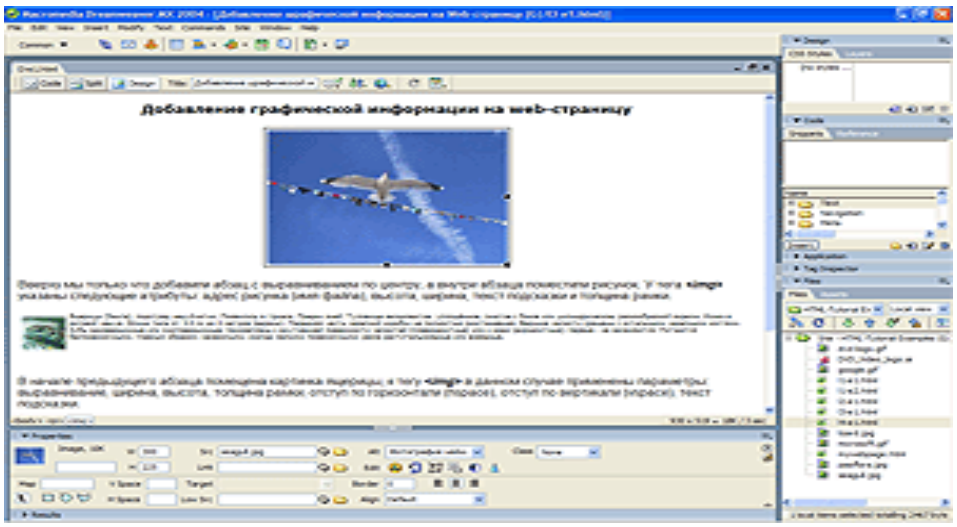
Визуал HTML мухаррирларни 2 асос бўйича этироф қилинади:

1. Adobe таркибидаги GoLive;
2. Macromedia таркибидагиDreamweaver;

Adobe GoLive CS ойнаси кўриниши қуйидагича:



Macromedia Dreamweaver MX 2004 ойнаси кўриниши қуйидагича:



2. HTML тили хақида қисқача маълумот

HTML (ing.Hypertext Markup Language — gipermatnli belgilash tili) - бу SGMLra (Standard Generalized Markup Language —стандарт умумлаштирилган белгилаш тили) асосланган ва халқаро ISO 8879 стандартига мос келувчи тил, халқаро тўрда ишлатилади.

HTML тили тахминан 1991-1992 йилларда Европа Ядровий Тадқиқотлар Марказида ишловчи британиялик мутахассис Тим Бернарс Ли томонидан ишлаб чиқилган. Дацлаб бу тил мутахассислар учун хужжат тайёрлаш воситаси сифатида

яратилган. HTML тилининг соддалиги (SGML га нисбатан) ва юқори форматлаш имкониятларининг мавжудлиги уни фойдаланувчилар орасида тез тарқалишига сабаб боълди. Бундан ташқари унда гиперматнлардан фойдаланиш мумкин эди. Тилнинг ривожланиши билан унга қоъшимча мултимедиа (расм,товуш, аниматсия ва бошқалар) имкониятлари қўшилди.

Браузерлар - махсус HTML тилида яратилган хужжатларни оъқувчи компютер дастури. Айнан броусерлар HTML тилида яратилган хужжатларни форматланган холда коъриш имкониятини беради. Хозирда энг машхур броусерлар бу Internet Explorer, Firefox, Opera ва хоказолардир.

Имкониятлари

HTML қуйидаги форматлаш имкониятларига эга:

- Матн қисмининг мантиқий ролини белгилаш (матн сарлавхаси, параграф, рўйхат ва хоказо).
- Гиперматнлар яратиш. Бу айниқса жуда қулай бўлиб ўзаро боғланган хужжат сахифалари орасида навигатсия қилишни енгиллаштиради.
- Матннинг ранги, қалинлиги ва бошқа шрифт кўрсаткичларини белгилаш.
- Махсус белгилар қўйиш. ASCII ∫, ∞, √, ½, ¼ , ¾, молия белгилари €, £, ¥, ©, ®, ™ ва хоказолар.
- Фойдаланувчи киритиши учун майдонлар яратиш.
- Мултимедиа файлларини очиш.
- Бошқа имкониятлар.

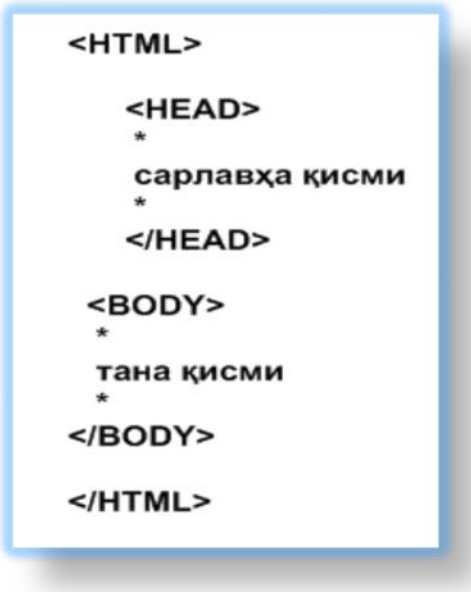
3. Ҳужжат тузилиши

HTML (Hyper Text Markup Language) – белгили тил бўлиб, яъни бу тилда ёзилган код ўз ичига маҳсус рамзларни мужассамлаштиради. Бундай рамзлар ҳужжат кўринишини фақатгина бошқариб, ўзи эса кўринмайди. HTMLда бу рамзларни тэг (тэг – ёрлик, белги) деб аталади. HTMLда ҳамма тэглар рамз-чегараловчилар (< , >) билан белгиланади. Улар орасига тэг идентификатори (номи, масалан B) ёки унинг атрибутлари ёзилади. Ягона истисно бу мураккаб чегараловчилар (<!--ва -->) ёрдамида белгиланувчи шархловчи тэглардир. Аксарият тэглар жуфти билан

ишлатилади. Очувчи тэгнинг жуфти ёпувчи тэг. Иккала жуфт тэг фақатгина ёпувчи тэг олдидан «слэш» (“/”) белгиси қўйилишини ҳисобга олмаганда, деярли бир хил ёзилади. Жуфт тэгларнинг асосий фарқи шундаки, ёпувчи тэг параметрлардан фойдаланмайди. Жуфт тэг яна контейнер деб ҳам аталади. Жуфт тэглар орасига кирувчи барча элементлар тэг контейнери таркиби дейилади. Ёпувчи тэгда зарур булмаган бир қатор тэглар мавжуд. Баъзида ёпувчи тэглар тушириб қолдирилса ҳам замонавий браузерлар аксарият ҳолларда ҳужжатни тўғри форматлайди, бироқ буни амалда қўллаш тавсия этилмайди. Масалан, расм қўйиш тэги <IMG>, кейинги қаторга ўтиш
, база шрифтини кўрсатиш <BASEFONT> ва бошқалар ўзининг </IMG>, </BR> ва ҳоказо ёпувчи жуфтларисиз ёзилиши мумкин. Нотўғри ёзилган тэгни ёки унинг параметри браузер томонидан рад килинади. (бу браузер танимайдиган тэгларга ҳам тааллуқли). Масалан, <NOFRAME> тэг-контейнери фақатгина фреймларни танийдиган браузер томонидан ҳисобга олинади. Уни танимайдиган браузер <NOFRAME> тэгини тушунмайди. Тэглар параметр ва атрибутларга эга бўлиши мумкин. Параметрлар йиғиндиси ҳар-бир тэгда индивидуалдир. Параметрлар қуйидаги коида асосида ёзилади:- Тэг номидан сўнг пробеллар билан ажратилган параметрлар келиши мумкин;- Параметрлар ихтиёрий тартибда келади;- Параметрлар ўзининг номидан кейин келувчи «қ» белгиси орқали берилувчи қийматларга эга бўлиши мумкин. - Одатда параметрлар қиймати « » - «кўштирноқ» ичида берилади. - Параметр қийматида баъзан ёзув регистри мухим. Шуни эсда тутиш лозимки, ҳамма тэглар ўзининг индивидуал параметрига эга бўлишига карамай, шундай бир қатор параметрлар мавжудки, уларни <BODY> бўлимининг барча тэгларида ишлатиш мумкин. Бу параметрлар CLASS, ID, LANG, LANGUAGE, STYLE ва TITLEлардир. HTML-ҳужжатини ёзишни бошлашда ишлатиладиган биринчи тэг бу <HTML> тэгидир. У ҳар доим ҳужжат ёзувининг бошида бўлиши лозим. Якунловчи тэг эса </HTML> шаклига эга бўлиши керак. Бу тэглар, улар орасида жойлашган ёзувнинг ҳаммаси бутун бир HTML-ҳужжатини англатиши билдиради. Аслида эса ҳужжат оддий матнли ASCII-файлидир. Бу тэгларсиз браузер ҳужжати форматини аниқлаб, таржима қила олмайди. Кўпинча бу тэг параметрга эга эмас. HTML 4.0 версиясига қадар VERSION параметри мавжуд эди. HTML 4.0да эса VERSION ўрнига <!DOSTYPE> параметри пайдо бўлди.

<HTML> ва </HTML> орасида 2 бўлимдан ташкил топиши мумкин бўлган ҳужжатнинг ўзи жойлашади. Мазкур ҳужжатнинг биринчи бўлими сарлавҳалар бўлими (<HEARD> ва </HEARD>), иккинчи бўлим эса ҳужжат тана қисмидир (<BODY> ва </BODY>), уни ҳужжат танаси ҳам деб юритамиз. Фрейм тузилиши ҳужжатлар учун <BODY> бўлимининг ўрнига <FRAMESET> бўлимидан фойдаланилади.

4. Ҳужжатнинг HEAD бўлими



HEAD бўлими сарлавҳа ҳисобланади ва у мажбурий тэг эмас, бироқ мукаммал тузилган сарлавҳа жуда ҳам фойдали бўлиши мумкин. Сарлавҳа қисмининг мақсади ҳужжатни таржима килаётган дастур учун мос ахборотни етказиб беришдан иборат. Ҳужжат номини кўрсатувчи <TITLE> тэгидан ташқари бу бўлимнинг қолган барча тэглари экранда акс эттирилмайди. Одатда <HEAD> тэги дарҳол <HTML> тэгидан кейин келади. <TITLE> тэги сарлавҳанинг тэгидир, ва ҳужжатга ном бериш учун ҳизмат килади. Ҳужжат номи <TITLE> ва </TITLE> тэглар орасидаги матн қаторидан иборат. Бу ном бараузер ойнасининг сарлавҳасида пайдо бўлади (бунда сарлавҳа номи 60 белгидан кўп бўлмаслиги лозим). Ўзгартирилмаган ҳолда бу матн ҳужжатга «закладка» (bookmark) берилганда ишлатилади. Ҳужжат номи унинг таркибини қисқача таърифлаши лозим. Бунда умумий маънога эга бўлган номлар (масалан, Homepage, Index ва бошқалар)ни ишлатмаслик лозим. Ҳужжат очилаётганда биринчи бўлиб унинг номи акс эттирилиши, сўнгра эса ҳужжат асосий таркиби кўп вақт олиб, кенгайиб кетиши мумкин бўлган форматлаш билан бирга юкланишини ҳисобга олган ҳолда, фойдаланувчи хеч булмаганда ушбу ахборот қаторини ўқий олиши учун ҳужжатнинг номи берилиши лозим.

5. Ҳужжатнинг BODY бўлими

Ушбу бўлинма ҳужжатнинг таркибий қисмини ўз ичига олади. Бўлинма <BODY> тэгидан бошланиб </BODY> тэгида тугайди. Бироқ ушбу тэглар катъий мавжуд бўлиши шарт эмас, чунки браузерлар матнга қараб ҳужжат таркибий қисмининг ибтидосини аниклаши мумкин. <BODY> тэгининг бир қатор параметрлари мавжуд бўлиб, уларнинг бирортаси ҳам мажбурий эмас.

<**BODY**> тэги параметрлари:

ALINK – фаол мурожаат (ссылка)нинг рангини белгилайди.

BACKGROUND – фондаги тасвир сифатида фойдаланилувчи тасвирнинг

URL-манзилини белгилайди.

BOTTOMMARGIN – ҳужжатнинг қуйи чегараларини пикселларда белгилайди.

BGCOLOR – ҳужжат фонининг рангларини белгилайди.

BGPROPERTIES – агар FIXED қиймати ўрнатилмаган бўлса, фон тасвири айлантирилмайди.

LEFTMARGIN – чап чегараларни пикселларда белгилайди.

LINK – хали кўриб чиқилмаган ссылканинг ранггин белгилайди.

RIGHTMARGIN – ҳужжат ўнг чегарасини пикселларда ўрнатади.

SCROOL – браузер дарчалари харакатлантириш (прокрутка) йўлакларини ўрнатади.

TEXT – матн рангини аниқлайди.

TOPMARGIN – юқори чегарасини пикселларда ўрнатади.

VLINK – ишлатилган мурожаат рангини белгилайди.

BOTTOMMARGIN, LEFTMARGIN, RIGHTMARGIN ва TOPMARGIN паметрлари матн чегараси ва дарча четлари орасидаги масофани пикселларда белгилайди. (Фақат HTML 4.0 версиясидан бошлаб IE браузерлари бу параметрларни таний олади)BGPROPERTIES параметри фақатгина битта FIXED қийматига эга. HTML даги ранглар ўн олтилик саноқ тизимида (RGB), ёки ранглар номи ёрдамида берилиши мумкин. Ранглар базаси 3 та рангга – қизил (R) , яшил (G) ва кўк (B) рангларга асосланган бўлиб, у RGB деб белгиланади. Ҳар-бир ранг учун 00 дан FF гача бўлган ўн олтилик саноқ тизимидаги қиймат берилади, бу эса 0 дан 255 гача бўлган диапазонга тўғри келади. Сўнгра бу қийматлар бир сонга бирлаштирилади ва уларнинг олдига “#” белгиси куйилади. Масалан, #800080 сиёҳрангни билдиради.

Мисоллар:

<BODY TEXT = “#000000”> ёки <BODY TEXT = black>

<BODY BGCOLOR = “#ffffff”> ёки <BODY BGCOLOR = WHITE>

<BODY LINK = “#ff0000”> ёки <BODY LINK = RED>

<BODY LINK = “#ooFFFF” ALINK = “#800080”> ёки <BODY VLINK = Aqua ALINK = PURPLE>

Ҳамма браузлар ўн олтилик саноқ тизимидаги стандарт ранларни танийди. Булар қуйидагилардир:

Black	#000000	Maroon	#800000
Silver	#C0C0C0	Red	#FF0000
Grey	#808080	Purple	#800080
White	#FFFFFF	Green	#008000
Fuchsia	#FF00FF	Navy	#000080

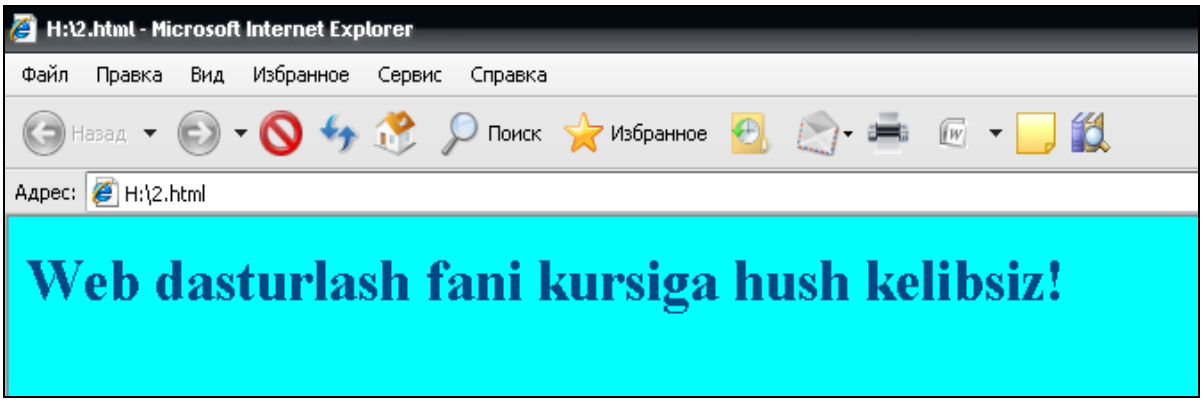
Lime	#00FF00	Blue	#0000FF
Olive	#808000	Teal	#008080
Yellow	#FFFF00	Aqua	#00FFFF

Мисол: Қуйида кўрсатилган параметрлар асосида мисол кўриб ўтамиз:

```
<html>

<BODY BGCOLOR қ AQUA  TEXT қ “#848484” LINK қ RED  VLINK қ PURPLE  ALINK қ GREEN>
```

Бу дастурни ишга тушириш асосида қуйидаги ойнага эга бўламиз:



Агар BGCOLOR параметри рангни номи ёки унинг таркибий кисмларини ўн олтилик саноқ тизимидаги кодда келтириш вазифаси ёрдамида фон ранггини чиқариш учун ишлатилса, BACKGROUND тасвир ёрдамида саҳифага фон беришда фойдаланилади. Тасвир сифатида GIF ёки JPG форматидаги график файллар ишлатилади. HTML-ҳужжат фонидаги тасвир доимо бутун саҳифани тўлдириб туради. Агар тасвир ўлчами дарча ўлчамидан кичик бўлса, у мозайка тамойилига асосан купайтирилади. Одатда фон тасвири сифатида тармоқ орқали юклаш учун унча кўп вақт кетмайдиган кичик тасвир танлаб олинади, ёки фон сифатида шаффоф рельеф логотипи тасвиридан фойдаланилади.

Мисол:

<BODY BACKGROUND = texture.gif BGCOLOR =gray>.

Саҳифа яратилишида доимо фон рангини бериш тавсия қилинади. Агар фон тасвири ҳам берилаётган бўлса, фон ва тасвир ранглари бир-бирига яқин бўлгани маъқул.

Мисол:

<BODY TEXT = BLUE LINK = RED VLINK = BLUE ALINK = PINK BACKGROUND=“”HYPERLINK "http://www.foo.com/jkorpela/HTML3.2/wave.gif" >.

Мисол:

<HTML><HEAD><TITLE> - саҳифа фонини бериш мисоли </TITLE></HEAD>

<BODY BGCOLOR = YELLOW TEXT = BLACK LINK = RED VLINK =PURPLE ALINK = GREEN>

</BODY>

</HTML>

6. HTML шаблонни яратиш

Notepad ни очамиз.HTMLфайлни яратишни бошлаймиз.

Сарлавха ёзиш учун:

<head>
</head>

teglari yoziladi.

Браузер бу теглар ўртасидаги матнни сарлавха деб тушунади ва Браузернинг энг тепа қисмига шу матнни ёзади. Энди сахифанинг танасини хосил қиламиз:

<body>
</body>

Бу теглар ўртасига хамма матн ва тасвирларни жойлаштириш керак.HTMLшаблон бўлиши учун HTML сахифанинг сарлавха ва танасини ўз ичига олган қуйидаги зарур теглар етишмаяпти:

<html>
</html>

Демак HTML шаблон қуйидаги кўринишга эга бўлди:

<html>
<head>
</head>
<body>
</body>
</html>

<head> va </head> теглари ўртасига куйидаги тегларни жойлаштириш мумкин: <title> ... </title> - хужжат номи. Масалан:

<title> web сахифа </title>

<meta /> –маълумотлар хақидаги маълумот. Яъни бу ерда асосий терминлар ёзилади. Қидирув системалари ишлаганда айнан шу терминлар бўйича қидириш ишини олиб боради, сайтларни топади. Масалан:

<meta namеқ”keywords” contentҳ”ТАТУ, Информатика, Инцитут, Факултет, Студент”/>

content 50-200 та сўзни ўз ичига олиши мумкин. namеқ(”keywords”, “autor”, “copyright”, “description”) –махфий қидиришда қўлланилади, HTTP серверга доступ бериш учун http-equip қўлланилади.

7. HTML шаблонларни сақлаш ва синаб кўриш

Веб сахифани яратишда хамма файлларни тўгри сақлаш керак. Кейин сайт яратувчиси ўзи хохлаган натижага эришганлигини текшириб кўриши керак.

Сақлаш ва синаб кўриш кетма-кетлиги:

1. File -> Save File хамма ўзгаришларни сақлаш;
2. Браузерда шу файлни очиш керак: Open -> File;
3. Натижани кўриш;
4. Агар бирор жойи тўгри ишламаса, матн мухарририга қайтиб хатоларни тўғ’рилаш керак;
5. Агар сахифа Web браузерда очиқ холда турган бўлса **Обновить** тугмасини босиб ўзгаришларни текшириш керак.

Назорат саволлари:

1. Windows мухитидаги матн мухаррирлари ҳақида тушунча беринг?
2. Матн мухаррирлари турларини айтинг?
3. HTML теги вазифасини тушунтириб беринг?
4. Теглар ҳақида маълумот беринг?
5. HTML тилида теглар қандай белгиланади?
6. HEAD теги ва унинг ичидаги теглар хақида гапиринг?
7. <TITLE> va </TITLE>теглар орасидаги сарлавҳа номини белгилаш қоидаларини кўрсатинг?
8. Параметрлар қиймати қандай кўринишда берилади?
9. BODY теги унинг параметрлари ҳақида маълумот беринг?
10. HTML шаблонларни қандай сақланади?

Фойдаланилган адабиётлар:

1. Т. Стауфер. Создание web-цранитс. Самоучител. – «Питер», Санкт-Петербург, 2003 г.
2. А. Гончарев.HTML. Самоучител. – «Питер», Санкт-Петербург, 2001 г.
3. Аллен Вайк. JavaScript. Ентсиклопедия ползователя: пер. с. анг. – «ТИД» «ДС», Киев, 2001 г.
4. webmastering –електрон ўқув қўлланма.
5. А.И. Тихонов. «Публикатсия данних в Интернет» Учебное пособие. Москва Издателцво «Мир» 2000
6. JavaScript ни урганиш буйича электрон қўлланмалар

Фойдаланилган манбалар:

1. http://uz.wikipedia.org/wiki/HTML
2. http://www.cgi.ru
3. http://www.woweb.ru
4. http://www.ru
5. http://www.vanta.ru/script/
6. http://www.vbnet.ru
7. http://www.scriptic.ru/
8. http://www.webacademy.com/
9. http://pacificwebart.com/

3-маъруза.

HTML разметкали тилининг асосий теглари

Режа:

1. Ҳужжатнинг HEAD бўлими;
2. Мантиқий ва физик форматлаш;
3. HTML-ҳужжатни форматлаш;
4. HTML-ҳужжатнинг ичидаги сарлавҳа тэглари.

Калит сўзлар: HTML-ҳужжат, сарлавҳа, форматлаш теглари, HTML тилининг асосий теглари, HTML теги параметрлари.

Ишдан мақсад: Талабаларда web-саҳифа яратиш жараёни тушунчалари ҳақидаги билим ва кўникмаларни шакллантириш ва ўргатишни ташкил қилиш. Бу жараённи амалиётга татбиқ қилиш.

HTML-ҳужжатини ёзишни бошлашда ишлатиладиган биринчи тэг бу <HTML> тэгидир. У ҳар доим ҳужжат ёзувининг бошида бўлиши лозим. Якунловчи тэг эса </HTML> шаклига эга бўлиши керак. Бу тэглар, улар орасида жойлашган ёзувнинг ҳаммаси бутун бир HTML-ҳужжатини англатиши билдиради. Аслида эса ҳужжат оддий матнли ASCII-файлидир. Бу тэгларсиз браузер ҳужжати форматини аниқлаб, таржима қила олмайди. Кўпинча бу тэг параметрга эга эмас. HTML 4.0 версиясига қадар VERSION параметри мавжуд эди. HTML 4.0да эса VERSION ўрнига <!DOSTYPE> параметри пайдо бўлди.

<HTML> ва </HTML> орасида 2 бўлимдан ташкил топиши мумкин бўлган ҳужжатнинг ўзи жойлашади. Мазкур ҳужжатнинг биринчи бўлими сарлавҳалар бўлими (<HEARD> ва </HEARD>), иккинчи бўлим эса ҳужжат тана қисмидир (<BODY> ва </BODY>), уни ҳужжат танаси ҳам деб юритамиз. Фрейм тузилиши ҳужжатлар учун <BODY> бўлимининг ўрнига <FRAMESET> бўлимидан фойдаланилади.

1. Ҳужжатнинг HEAD бўлими.

BODY бўлинмасида пайдо бўлиши мумкин бўлган баъзи HTML-тэглар блок даражасидаги (block level) тэглар деб аталса, бошқалари матн даражасидаги (text level) тэглари ёки кетма-кет тэг (inline) деб аталади. Блок даражасидаги тэглар ўзида матн даражасидаги тэглар ёки блок даражасидаги бошқа тэгларни мужассамлаштириши мумкин. Блок тэглари ҳужжат тизимини таърифлайди.

```
<HTML>
   <HEAD>
   *
   сарлавҳа қисми
   *
   </HEAD>
 <BODY>
 *
 тана қисми
 *
</BODY>
</HTML>
```

Саҳифа яратилишида доимо фон рангини бериш тавсия қилинади. Агар фон тасвири ҳам берилаётган бўлса, фон ва тасвир ранглари бир-бирига яқин бўлгани маъқул.

Матн мухаррирлари билан ишлаш жараёнидан биламизки, матнларни хар-хил кўринишда ифодаланиши мумкин: қалинлаштирилган (полужирный), қўлёзма шаклида (курсив), таги

чигилган (подчеркнутый) … Бу элементларни ихтиёрий график браузерлар бир хил кўринишда ифодалайдилар.

<HEAD> бўлимида қуйидаги маълумотлар жойлашиши мумкин:

- ✓Саҳифаларни кодлаш; (Масалан, бу саҳифа Windows-1251 ((кириллица-Windows)) кодлаш усулида ёзилади)
- ✓Саҳифалар сарлавҳаси;
- ✓Жадвал усулидаги ссилкалар; (иҳтиёрий)
- ✓Скриптлар;
- ✓Калит сўзлар;
- ✓Муаллиф исми;
- ✓Яратилган файлдаги дастурнинг номи;

<HEAD> бўлимидаги аксарият маълумотлар «<meta>»тегида кодланади:

- ✓ Кодлаш: <meta http-equivқ"**content-type**" content="text/html;charsetқ***windows-1251***">
- ✓ Калит сўзлар: <meta name="**keywords**" content="Сўзларни вергул билан ёзинг!">
- ✓ Муаллиф исми: <meta name="**author**" content="Одина Имомова ">
- ✓ Дастур, ишлаб чиқилган файл: <meta name="**generator**" content="Adobe GoLive 5">

Бошқа элементлар:

- ✓ Саҳифа: <**title**>Sahifa nomi</title> Браузер ойнаси саҳифасида тасвирланади;
- ✓ Скрипт: <**script**><!—Скрипт матни //--></script>
- ✓ Боғланган файлдаги ссилка: <**link** href="http://www.w3.org">
- ✓ Жадвал усулидаги ссилка: <**link** rel="**stylesheet**" href="../usual.css">

Еслатма: Браузерда <HEAD> бўлими акс эттирилмайди.

Бу саҳифада кодни кўриш мумкин.

2. Мантиқий ва физик форматлаш

HTML-ҳужжатларда матнни форматлаш учун шартли равишда мантиқий ва физик форматлаш тэгларига тақсимласа бўлувчи тэглар яратилган. Мантиқий форматлаш тэглари фрагментнинг браузер ёрдамида экранда намойиш этилишига таъсир кўрсатмайдиган структуравий белгилашни амалга оширади. Шу сабабли бундай белгилаш мантиқий деб аталади. <CITE> тэги цитаталар ёки китоблар, мақолалар ва бошқа манбаларга ссылкаларнинг номларини белгилашда фойдаланилади. Браузерлар бундай матнни курсив (қия) шаклда чиқариб беради.

HTML тилида қуйидаги физик усулдаги таҳрирловчи теглар мавжуд:

Тег	**Тег моҳияти**
<I>...</I>	Курсив(Italic)
<B>...</B>	Қалин(Vold)
<TT>...</TT>	Телетайп
<U>...</U>	Остига чизиқ
<S>...</S>	Устига чизиқ
<BIG>...</BIG>	Шрифт ўлчамини катталаштириш
<SMALL></SMALL>	Шрифт ўлчамини кичрайтириш
_{...}	Индексга ёзиш

^{...}	Даражага ёзиш

Физик усулдаги тегларни ишлатиш қоидаси:

1. Матнни киритинг;
2. Матн олдига курсорни олиб келиб керакли тегни очувчисини ёзинг;
3. Матн охирига курсорни олиб келинг;
4. Ёпувчи тегни ёзинг.

Масалан:

 Қалинлаштирилган матн **(полужирный)**
 Қо'лёзма шаклидаги матн *(курсив)*
<tt> Харфлар оралигини кенгайтириш </tt>
<u> Таги чизилган матн (подчеркнутый) </u>
<big> КАТТАЛАШТИРИЛГАН МАТН </big>
<small> кичиклаштирилган матн </small>
C_n - C_n
$ax^2+bx+c=0$ - ax²+bx+c=0

Мантиқий усулдаги теглар:

Елемент	Вазифаси
,	тегига аналог тег.
,	тегига аналог тег.
<cite>, </cite>	Мазкур хужжат матнига цитата келтириш.

<dfn>, </dfn>	Дастур коди.
<samp>, </samp>	Дастурнинг ишлашига мисол. Олдингиси каби ишлайди.
<kbd>, </kbd>	Клавиатурадан киритиладиган матн.
<var>, </var>	Ўзгарувчи ёки миқдор.
<abbr>, </abbr>	Аббревиатура.
<acronym>, </acronym>	Акроним.

Мантиқий усуллар браузерга матнни қай тартибда экранга чоп этиш кераклигини билдиради. Мантиқий усуллар физик усуллар ишламай қоладиган холатлар учун хам ўринли бўлиши мумкин: уяли телефон интернетга уланганда <b> теги орқали қалинлаштирилган матн учраб қолса, уни ўқий олмайди. Бу холатда елементи керакли натижани бера олади.

Мисол:

<CITE> Даракчи </CITE> энг оммабоп газеталардан бири.

<EM> тэги – (Emphasis – ажратиб кўрсатиш, таъкидлаш) матннинг мухим фрагментларини ажратиб кўрсатишда фойдаланилади. Браузерлар одатда бундай матнни курсив шаклда акс эттиради.

Мисол:

Матннинг <EM> мухим сўзларини </EM> ажратиб кўрсатиш. Бу ерда мухим сўзларини деган ифода курсивда ажратиб кўрсатилади.

<DEL> тэги матнни учириб ташланган сифатида белгилайди. Бу тэг орқали белгиланган матннинг устига чизилган бўлади.

Мисол:

Бу <DEL> ўчирилган матндир </DEL> Бу ерда ушбу тэглар орасидаги «ўчирилган матндир» жумласи қуйидаги кўринишга эга бўлади: ўчирилган матндир (устига чиза олмадик)

Тэг CITE ва DATETIME параметрларига эга бўлиши мумкин. CITE - фрагемнтнинг ўчирилиб ташлаш сабабларини аниқлаштирувчи ҳужжатнинг URL-манзилини кўрсатади.

DATETIME – ўчириб ташланиш вақтини: YYYY-MM-DDThh:mm:ssTZD форматида кўрсатади. Бу формат ўчириб ташланиш йили, ойи, куни, соати, дақиқаси ва сонияларини, шунингдек соат тизими(TimeZone)ни аниқлайди.

<KBD> тэги матнни фойдаланувчи томонидан клавиатурада киритилганидек, яъни бир хил кенгликдаги (моноширинный) шрифтда акс эттиради. Мисол учун, матн мухарририни ишга тушириш учун <KBD> NOTEPAD </KBD> деб киритинг.

3. HTML-ҳужжатни форматлаш

Хар қандай матн маълум бир тузулишга эга бўлади. Одатда бундай тузулишнинг элементлари – сарлавҳалар, рўйхатлар, кичик сарлавҳалар, жадваллар, хатбоши ва бошқалар ташкил этади.

Хатбошиларга бўлиш. Хатбоши даражаси тэги.Одатда хатбошилар матнда фикр тугалланганлигини ифодалайди. Оддий матнли мухарирларда хатбоши

бошқа қаторга ўтиш белгисини киритиш (<ENTER> тугмасини босиш) орқали тузилади. Аммо HTML-ҳужжатини тузуш жараёнида бошқа қаторга ўтиш белгилари хатбошининг ҳосил бўлишига олиб келмайди. Асл ҳужжатнинг бошқа қаторга суриш белгилари эътиборга олинмаганлиги сабабли, муаллиф ҳужжати ойнасида аъло даражада кўринган матн, браузер ойнасида умуман ўқиб бўлмайдиган даражада бўлиши мумкин. Шунинг учун HTML-тилида матнларни хатбошиларга бўлиш учун <P> тэги киритилган. Бу тэгни хар бир хатбоши олдидан қўйиш керак. Ёпувчи </P> тэги бу ҳолда мажбурий эмас. Браузерлар бир неча кетма-кет жойлашган <P> тэгини бир тэгдек изоҳлайди. Одатда барузерлар хат бошиларни бир-биридан битта бўш қатор билан ажратади. <P> тэги атрибутлари:ALIGNГоризонтал текислаш атрибутининг қийматлари:LEFT, RIGHT, CENTER, JUSTIFY.

Мисол:

<HTML>

<HEAD>

<TITLE> Xatboshi tegining qo’llanilishi </TITLE>

</HEAD>

<BODY>

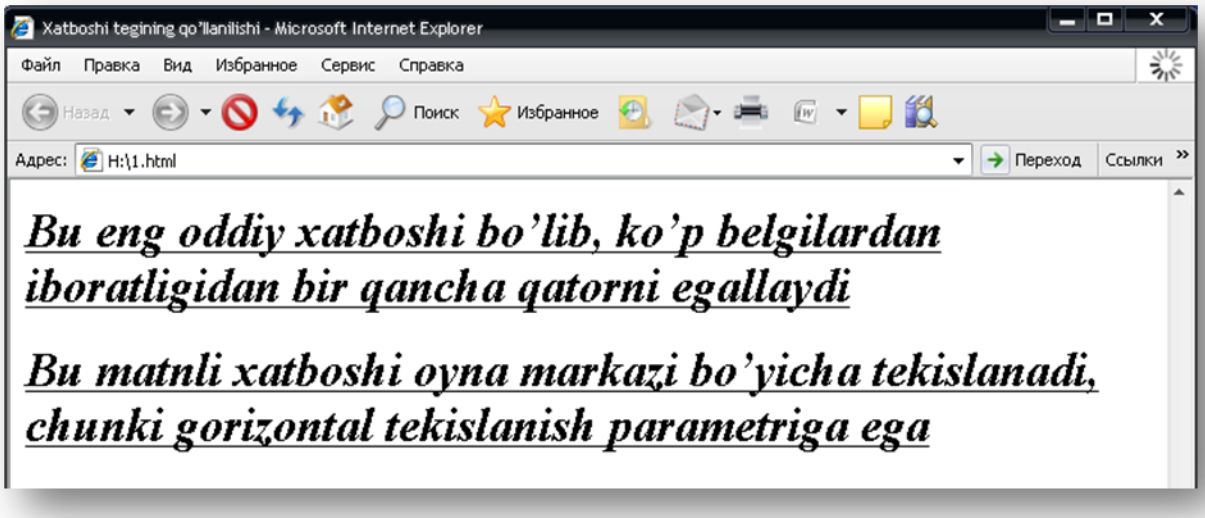
<h1> <u> <P> Bu eng oddiy xatboshi bo’lib, ko’p belgilardan iboratligidan bir qancha qatorni egallaydi </P>

<P ALIGNCENTER> Bu matnli xatboshi oyna markazi bo’yicha tekislanadi, chunki gorizontal tekislanish parametriga ega </P> </u> </h1>

</BODY>

</HTML>

Ушбу дастурни ишга тушириш натижасида қуйидаги ойна яратилади:



 тэги мажбурий бошқа қаторга суриш вазифасини бажаришга ҳизмат қилади (ёпувчи тэгга эга эмас).
 тэгининг киритилиши кейинги матн Янги қатордан бошланишини таъминлайди. Масалан, бундай усулдан маҳсус рўйхатни белгилаш тэгларини ишлатмаган ҳолда рўйхат шаклидаги тузилмаларни тузуш ёки шеърларни ёзиш каби ҳолларда фойдаланиш мумкин.

<P> тэги киритилучи хатбошини инкор этиши туфайли,
 тэги ёрдамида киритилувчи хатбошини моделлаштириш мумкин.

Масалан,

<HTML>

<HEAD>

<TITLE> бошқа қаторга сурушнинг қўлланилиши </TITLE>

</HEAD>

<BODY>

<P>Кераксиз сўз демагин

 Кўпроқ сўзламоқ учун
 Охир дакки емагин,

 деб жавоб берди.

</P>

<CITE> Комил Парпиев </CITE>

</BODY>

</HTML>

 тэгининг параметрлари:CLEAR

Параметр маънолари: LEFT, RIGHT, ALL, NONE.

<NOBR> тэги бошқа қаторга ўтишни инкор қилиш учун қўлланилади. Бу тэг орқали белгиланган матн узунлигидан қатъий назар бир қаторда жойлаштирилади.

Жуда узун қаторлар ҳосил бўлишининг олдини олиш учун қатор суриш мумкин бўлган жой кўрсатилса, бу зарурият туғилганда амалга оширилади (“юмшоқ” тарзда бошқа қаторга суруш). Бу максадда <WBR> (Word Break) тэгидан фойдаланилади.

4. HTML-ҳужжатнинг ичидаги сарлавҳа тэглари

<H1 H2 …H6>.Ҳужжатнинг алоҳида бўлимларига сарлавҳа бериш учун 6 босқичли <H1> </H1>, <H2> </H2>, <H3> </H3>, <H4> </H4>, <H5> </H5>, <H6> </H6> теглардан фойдаланилади. 1 рақамли сарлавҳа энг йириги ҳисобланади. Энг кичиги эса 6 рақамли сарлавҳадир. Сарлавҳа тэги блок даражасидаги тэгдир, шунинг учун улар ёрдамида алоҳида матн сўзларининг ўлчамларини катталаштириш учун уларни белгилай олмайди. Сарлавҳа тэгларидан фойдаланилаётганда сарлавҳадан олдин ва кейин бўш қатор колдирилади, шунинг учун матн боши тэги ва бошқа қаторга суруш керак булмайди. Сарлавҳа

тэглари атрибутлари: ALIGNТэглар параметри қийматлари:LEFT, RIGHT, CENTER, JUSTIFY (ўзгартирилмаган бўлса LEFT).

Мисол:

<HTML><HEAD><TITLE> Сарлавҳа мисоллари </TITLE></HEAD>

<BODY>

<H1> Сарлавҳа ўлчами 1 </H1>

<H2> Сарлавҳа ўлчами 2 </H2>

<H3 ALIGN=CENTER> Сарлавҳа ўлчами 3 </H3>

<H4 ALIGN=RIGHT> Сарлавҳа ўлчами 4 </H4>

<H5> Сарлавҳа ўлчами 5 </H5>

<H6> Сарлавҳа ўлчами 6 </H6>

Бу ерда ҳужжатнинг асосий матни

</BODY> </HTML>

<HR> тэги - горизонтал чизиқ. Саҳифанинг у ёки бу қисми тугалланганида визуал ажратиб кўрсатиш учун ишлатилади. Бу тэг ёпишни талаб қилмайди. Қатордан олдин ва кейин автоматик равишда бўш қатор қолдирилади.

<HR> тэги атрибутлари вазифаси:

ALIGN - LEFT, RIGHT, CENTER (ўзгартирилмаган х.)

WIDTH - чизиқ узунлиги фоиз кўрсаткичида ёки пикселда белгиланади.

SIZE - -чизиқ қалинлиги пикселларда ўрнатилади.

NOSHADE - -чизиқнинг рельефлилигини олиб ташлаш.

COLOR -- чизиқ рангги ўрнатилади.

Масалан:

<HTML><HEAD><TITLE> горизонтал чизиқ </TITLE></HEAD>

<BODY>

<P> Горизонтал чизиқ устига жойлаштирилган матн </P>

<HR ALIGN=CENTER WIDTH=70% NOSHADE>

<P> Кейин матннинг ўзи киритилади

</BODY>

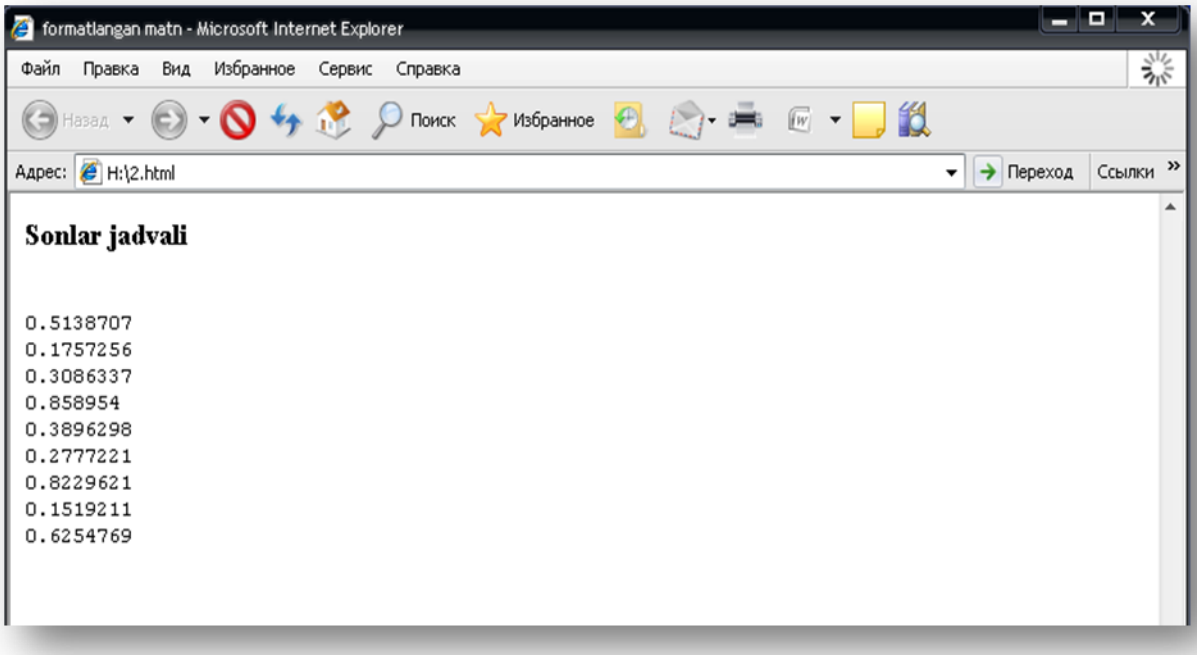
</HTML>

<PRE> тэги олдиндан форматлаштирилган матннинг қўлланилишидир. Анъанавий усулда, бошқа қаторга суриш белгилари ёрдамида бажарилган форматланган матнни киритиш учун, керакли бўш жой миқдори, табуляция рамзлари ва бошқалар учун махсус <PRE> тэг контейнери мўлжалланган. <PRE> тэги билан белгиланган матн оддий матн мухарририда қандай кўринишга эга бўлса, худди шундай шаклда акс этади. Акс эттириш учун хар доим бир хил кенгликдаги шрифт қўлланилади. Бу тэг жадвални белгиловчи махсус тэгларни ишлатмасдан қурулган жадвалларда қўлланилади. Яна бир қўлланиш ҳолати катта блокларни чиқаришдир.

Мисол:

```
<HTML><HEAD><TITLE> formatlangan matn </TITLE></HEAD>
    <BODY>
        <H3> Sonlar jadvali </H3>
            <PRE>
                                    0.5138707
                                    0.1757256
```

Дастур натижаси қуйидагича бўлади:



<PRE> тэгининг атрибути:

WIDTH - атрибут қиймати пиксел, ёки фоизларда берилиши мумкин ва форматланган матннинг максимал қатор узунлигини кўрсатади.

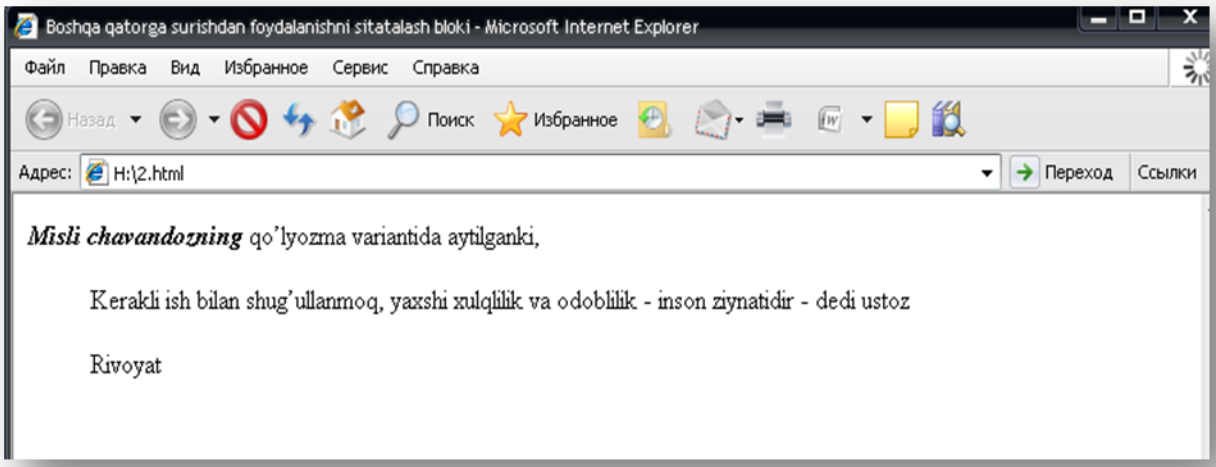
<BLOCKQUOTE> тэги асосий матндан цитаталарни ажратиб кўрсатиш учун қўлланилади. Бу тэг контейнер бўлиб, турли хил тэгларнинг форматларини аниқлайди.

<BLOCKQUOTE> тэги ёрдамида белгиланган матн акс этиш жараёнида асосий матндан бўш қаторлар билан ажратилади ва ўнг томонга кичик силжиш билан киритилади.

Мисол:

```
<HTML><HEAD>
<TITLE>Boshqa qatorga surishdan foydalanishni sitatalash bloki </TITLE></HEAD>
<BODY>
<P>
<I>
<B> Misli chavandozning </I> </B>qo’lyozma variantida aytilganki,
<BLOCKQUOTE>
```

Дастур натижаси қуйидаги кўринишда бўлади:



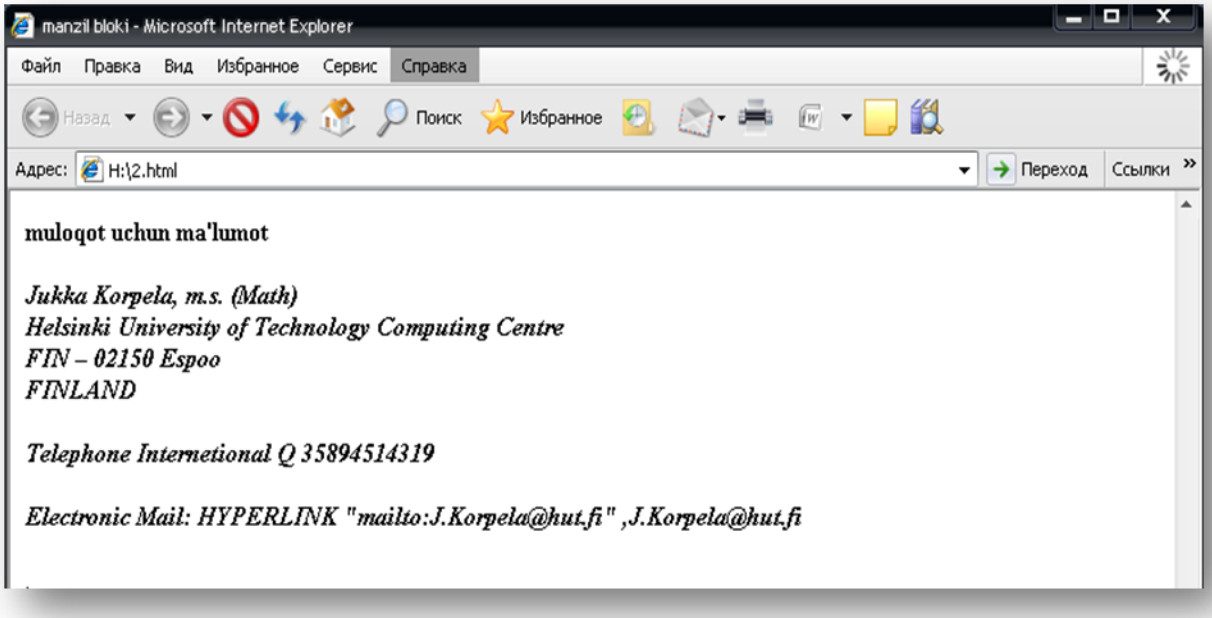
<ADDRESS> тэги ҳужжат муаллифини идентификация килиш ва муаллиф манзилини кўрсатиш учун қўлланилади. Одатда бу тэг ҳужжатнинг бошида ёки охирида жойлаштирилади. Бу тэгда тез-тез ҳужжат яратилган ва сўнгги маротаба янгиланган санаси кўрсатилади.

<ADDRESS> тэги контейнер ҳисобланади.

Мисол:

```
<HTML><HEAD> <TITLE> manzil bloki </TITLE></HEAD>
<BODY>
<h4>
<P> muloqot uchun ma'lumot </P>
```

Дастур натижаси қуйидаги кўринишда бўлади:



<! - - … - - > ҳужжатга шархлаш (коментарий) киритиш тэглари Шархлар қаторларнинг ихтиёрий сонидан ташкил топиши мумкин. Ва <!-- тэги билан бошланиб, --> тэги билан тугаши лозим. Тэг ичига киритилганларнинг барчаси саҳифани текшириш чоғида ишламайди. Одатда шархлар шахсий фойдаланиш учун мўлжалланган кузатишлар учун муаллиф томонидан ишлатилади.Шархларни ёзиш учун яна бир тэг-контейнер мавжуд бўлиб, у ҳам бўлса <COMMENT> дир.

Назорат саволлари:

1. HTML ҳужжатнинг умумий тузилиши нимадан иборат?
2. Қандай форматлаш тегларини биласиз?
3. Комментарийлардан фойдаланишни изоҳланг?
4. HTMLтеги таркибида қандай элементлар жойлаштирилиши мумкин?
5. Физик усулдаги таҳрирловчи тегларни айтинг ва мисол ёрдамида тушунтиринг?
6. Мантиқий усулдаги таҳрирловчи теглар нималардан иборат ва фикрингизни мисоллар ёрдамида тушунтиринг?

7. Физик усулдаги тегларни ишлатиш қоидаси қандай?
8. Браузерга матнни қай тартибда экранга чоп этиш кераклигини қайси усул белгилайди?
9. Қайси теглар ёпувчи тегга эга эмас?
10. Оддий матнли муҳаррирларда ҳатбошилар нимани ифодалайди?

Фойдаланилган адабиётлар:

1. Спейнауер С., Куерсиа В. Справочник Web-мастера. - К: "BHV", 1997. - 368 с.
2. Яргер Р., Риз Дж., Кинг Т. MySQL и mSQL. Базы данных для небольших предприятий и Интернета. - СПб: Символ-Плюс, 2000 - 560 с.
3. Хилайер С., Мизик Д.Программирование Active Server Pages. - М: "Русская редакция", 1999. - 296 с.
4. Холзнер С. Perl: специальный справочник. - СПб: "Питер". 2000.
5. Сҳвартс Р., Крициансен Т.Изучаем Perl. - К: "BHV", 2000. - 320 с.
6. Қосимов С.С. Ахборот теҳнологиялари. Тошкент “Алоқачи” 2006.

Фойдаланилган манбалар:

1. Intuit.ru.
2. http://pda.coolreferat.com
3. http://www.z-oleg.com
4. http://www.i2r.ru/
5. http://uz.wikipedia.org/wiki/HTML

4-маъруза. HTML тилида формалар, фреймлар ва объектлар

Режа:

1. HTML тилида формалар;
2. Форманинг берилиши-FORM елементи;
3. HTML тилида фреймлар;
4. Фреймлар орасидаги ўзаро таъсир.

Калит сўзлар: формалар (<FORM>), фреймлар (<FRAME>), объектлар, <INPUT>, <SELECT>,<TEXTAREA>

Ишдан мақсад: Web-дастурлаш фани бўйича HTML тили асосидаги формалар ва фреймлар билан ишлаш жараёнида талабаларда билим, ишлаш кўникмаси ва малакаларини шакллантириш ва бу жараённи амалга оширишда талабаларга маълумот беришни ташкил қилиш.

1. HTML тилида формалар

Формалар Веб дастурлашда фойдаланувчи томонидан киритилаётган маълумотларни тартибга солиш учун қўлланилади. Форма элементлари тўлдирилгандан кейин ундаги маълумотлар сервердаги маълумотларни қайта ишловчи дастурга юборилади. Кўп сонли жўнатилаётган маълумотлар жўнатиш тугмаси босилгандан сўнг серверда жойлашган Common Gateway Interface (CGI) ёрдамида ёки махсус файл орқали қайта ишланади. Шу тариқа фойдаланувчи Internet орқали Web-сервер билан биргаликда ишлайди.

HTML тили web-сахифалар таркибига матнли сохалар, менюлар, тугмалар каби интерфейс элементлар қўйиш имкониятини беради. Бу маълумотлар жуда оддий (електрон адрес) ёки йетарлича мураккаб бўлиши мумкин. Масалан, фойдаланувчи онлайн холатида интернет дастурлари билан ишлаши мумкин (касалхона сайти орқали врач чақириши ёки ишхонасининг сайти ёрдамида автомобилни прокатга олиши мумкин).
Електрон формаларни яратишни мақсади қуйидагича: у фойдаланувчидан информатсия сўраши ва ундан информатсия олиши мумкин. Бундай ҳолатларда маълумотларнинг бир қисмини фойдаланувчи ўзига тегишли бўлган маълумотни танлаши мумкин бўлган тайёр менюлар ташкил этиши мумкин. Сўровлар қуйидаги турларда бўлиши мумкин:

Жавоблари махсус матнли сохаларга терилиши керак бўлган саволлар (масалан, исми, фамилияси, электрон адреси, …).

Жавоблари махсус менюлардаги ёки рўйхатдаги жавоблардан танлаши керак бўлган саволлар (масалан қайси давлатга тегишли эканлиги, қайси мавзулар қизиқтириши, …).

Переключателлар орқали жавоблардан ягонасини танлашга мўлжалланган саволлар (масалан, эркак/аёл, ха/йўқ…).

Формалар HTML тилининг киритиш ойнаси, тугма, переключател каби элементлар наборларидан фойдаланган холда яратилади. Фойдаланувчи томонидан яратилаётган хар бир элемент ўзининг номига эга бўлиб, кейинчалик олдиндан аниқланган бирор ўзгарувчига берилади. Масалан, матн киритиш майдонига шаҳар номини берилса ва фойдаланувчи майдонни Жиззах деб

тўлдирса шуни билдирадики, шаҳар ўзгарувчисига Жиззах қиймати берилди. Ўзгарувчилар ва уларнинг қийматлари серверларга узатилади ва ўз навбатида сервер «скриптлар» деб аталувчи кичик программаларга мурожаат қилади. Скриптлар олинган маълумотларни қабул қилиб, ва уларни қайта ишлайди. Натижада web-сахифада фойдаланувчининг маълумоти қабул қилинганлиги тўғрисида оддийгина маълумот (масалан, “рахмат”) чиқариб қўйиш мумкин. HTML формалари билан ишлаш учун скриптлар билан ишлай олиш керак (улар CGI скриптлар дейилади ва уларни турли программалаштириш тилларида тузиш мумкин. CGI–Common Gateway Interface).

2. Форманинг берилиши —FORM элементи

FORM элементи ҳужжатни маълум бир *формага* солади ва *форма* элементлари тэгларини бошқа тэглардан ажратиб туради. <FORM> бир нечта *<INPUT>* ёки шу каби бошқа тэглар кетма кетлигидан ташкил топади. Улар <FORM> ва </FORM> тэглари орасига жойлаштирилади. Формада усулдан (method), *формага* киритилган маълумотларни қайта ишлаш учун ҳолатлар (action) мавжуд. Усул (GET ёки POST) формага киритилган маълумотлар қай тарзда серверга жўнатилиш усулини белгиласа, ҳолат эса сервердаги қайси дастурга юборилиш URL (Uniform Resource Location) манзилини ифодалайди.

<FORM METHOD=“post” ACTION= “<URL>”>

Қуйида FORM елементи параметрлари билан танишиб чиқамиз:

Форманинг бошқарув элементлари —<INPUT> теги

Бошқариш элементининг яна бир кўриниши бу – менюлардир. <Select>елементи ёрдамида хар хил менюларни хосил қилиш мумкин. Бу холатларда

фойдаланувчига бир нечта вариантлар берилиб, ундан ўзининг жавобини танлаш имконияти берилади. Name атрибути зарурий атрибут хисобланади. Size атрибути ёрдамида эса экранга бир вақтда чиқадиган вариантлар сонини бериш мумкин. Bik .Yu <select> бу контейнордир. Қийматлар <option> ёрдамида киритилиб, <select> ва </select> лар орасига жойлаштирилади. Агар фойдаланувчи менюнинг бирор-бир пунктини танласа қиймат <select> тегида номланган ўзгарувчига берилади. Bik .Yu Selected= “selected” атрибути бирор қийматни автоматик танлаш имкониятини беради (значение по умолчанию):

```
<select name="жавоб1">
<option value="first"> Биринчи</option>
<option value="monthly"> Хар ойда <</option>
<option value="weekly"> Хар хафтада </option>
<option selected="selected" value="daily"> Хар кунда </option>
```

Юқорида айтилганидек, сизе атрибути ёрдамида экранга чиқариладиган жавоблар сонини киритиш мумкин:

```
<select name="javob1" size="6">
```

Биринчи мисолда жавоблар бир қаторда ифодаланади ва сичқон тугмаси босилганда қолган жавоблар кўринади. Иккинчи мисолда сизе да неча сон кўрсатилган бўлса шунча жавоб варианти кўрсатилади ва қолган жавобларни прокруткаларни силжитиш ёрдамида кўриш мумкин.
Агар жавоблардан бир нечтасини бир вақтда танлаш зарур бўладиган бўлса <select> нинг multiple= “multiple” атрибути ёрдамга келади. Бу холатда сизе да нечта жавоблар кўрсатилган бўлса шулардан бир нечтасини танлаш имконияти бўлади.

Ушбу тэг *форманинг* қайси нуқтасига маълумот киритилишини белгилайди. У фойдаланувчи томонидан киритилаётган маълумотларни формага келтиради. Булар матн киритиш майдони, рўйхатлар, расмлар ёки тугмалар бўлиши мумкин. Майдон типи TYPE атрибути ёрдамида аниқланади.

TYPE=text атрибути

Агар фойдаланувчи унча катта бўлмаган матн киритса (бир ёки бир нечта сатр), *<INPUT>* тэгидан фойдаланади ва TYPE атрибутига text қиймати ўзлаштирилади.

Стандарт ҳолат учун бу қийматни бериш муҳим эмас. Бундан ташқари *майдонни* номлаш ва унга мурожаат қилиш учун NAME атрибути ҳам берилади.

Сизнинг исмингиз <INPUT NAME=Name SIZE=35>

Фойдаланиш мумкин бўлган яна учта қўшимча атрибутлар мавжуд. Биринчиси MAXLENGTH деб аталади, у фойдаланувчи киритаётган матн майдони максимум узунлигини белгилайди. Стандарт бўйича бу қиймат чегараланмаган. Иккинчи атрибут SIZE ҳисобланади, у эса матн майдонини кўриниб турувчи қисмини белгилайди. Стандарт бўйича унинг қиймати браузерга боғлиқ бўлади. Агар MAXLENGTH қиймати SIZE қийматидан катта бўлса, браузер маълумотни ойнага мослаштиради. Сўнга қўшимча атрибут матн майдонини бошланғич қийматини белгиловчи VALUE дир.

TYPE=checkbox атрибути

HTML *формада* мустақил белгилагич (байроқча) дан фойдаланиш учун *<INPUT>*тэгининг атрибутига TYPEқcheckbox ни ўзлаштириш керак. Формага боғлиқ равишда фойдаланувчи бир ёки бир нечта белгилагичларни белгилаши мумкин. Агар *<INPUT>* тэги атрибути билан CHECKBOX қиймати қўлланилса, у билан бирга NAME ва VALUE атрибутлари ҳам қўлланилиши керак. NAME атрибути ушбу маълумот киритиш объектининг номини ифодалайди. VALUE атрибутида ушбу *майдон*нинг қиймати кўрсатилади.

Россия<INPUT NAME="Давлат" TYPE=checkbox

VALUE="Россия">

Страны СНГ<INPUT NAME="Давлат" TYPE=checkbox

VALUE="СНГ">

Баъзи ҳолларда ушбу майдон белгиланган ҳолда қўлланилиши ҳам мумкин. Бундай ҳолларда *<INPUT>* тэгида CHECKED атрибути қўлланилиши керак.

TYPE=radio атрибути

Баъзан бир нечта қийматлар орасидан бирини танлашга тўғри келади. Бундай ҳолларда формада *<INPUT>* тэги билан бирга TYPE=radio атрибути қўлланилади. Агар *<INPUT>* тэги атрибути билан ушбу қиймати қўлланилса, у билан бирга NAME ва VALUE атрибутлари ҳам қўлланилиши керак. NAME атрибути ушбу маълумот киритиш объектининг (*тугма*) номини ифодалайди. VALUE атрибутида ушбу *майдон* нинг қиймати кўрсатилади..

Эркак жинси <INPUT NAME="Жинс" TYPE=radio

VALUE="Эркак">

Аёл жинси <INPUT NAME="Жинс" TYPE=radio

VALUE="Аёл">

TYPE=image атрибути

Форманинг таркибига қараб баъзан унда жойлашган расмнинг устига сичқончани босиш билан ундаги маълумотларни жўнатишга тўғри келиб қолади. Бунинг учун *<INPUT>* тэги TYPE=image атрибути билан қўлланилади. Фойдаланувчи расм устига сичқонча курсорини босса, айнан шу ердаги экран координаталарини браузер сақлаб қолади. Сўнг формага киритилган маълумотларни “қайта ишлайди”. Агар *<INPUT>* тэги image атрибути билан қўлланилса, у билан бирга NAME ва SRC атрибутлари ҳам қўлланилиши керак. NAME майдоннинг номини белгилайди. SRC атрибути эса расм жойлашган манбанинг URI манзилини беради. ALIGN атрибути қўшимча ҳисобланади ва у ҳам баъзан <IMG> теги билан қўлланилади.

Нуқтани танланг <INPUT TYPE=image NAME=point

SRC=image.gif>

TYPE=password атрибути

Агар *формада* пароллардан фойдаланиш керак бўлиб қолса, TYPE атрибути қийматига password (TYPEкpassword) ни ўзлаштирилади. Ушбу типдан фойдаланиш киритилаётган маълумотни ошкор бўлмаган ҳолда кўрсатишни таъмин этади. Шу сабаб, киритилган маълумот очиқ канал орқали жўнатилади ва ушбу маълумот тутиб олиниши мумкин.

Номингиз<INPUT NAME=login>Парол

<INPUT TYPE=password NAME="Сўз">

TYPE=reset атрибути

Базан фойдаланувчи *формани* тўлдириш вақтида, уларни бошдан тўлдиришга тўғри келади. Ушбу ҳолда Reset тугмаси мавжуд бўлиб, бу тугманинг босилиши формани дастлабки, кириш ҳолатиги олиб келади (формани “тозалайди”). Reset иугмасини ташкил қилиш учун *<INPUT>* тэги атрибутига TYPE=reset ўзлаштирилади. Агар *формада* reset атрибути қўлланилса, *<INPUT>* тэгига VALUE атрибутини қўшимча қилиш мумкин. Ушбу атрибут тугмадаги ёзувни ифодалайди.

<INPUT TYPE=reset VALUE="Формани тозалаш ">

TYPE=submit атрибути

HTML *форма* да фойдаланувчи маълумот киритиш жараёнини якунлаш жараёни мавжуд. Бунинг учун *<INPUT>* тэгининг атрибутига TYPE=submit қиймат ўзлаштирилади. Агар *формада <INPUT>* тэги submit атрибути билан қўлланилса, унга қўшимча равишда иккита атрибутдан фойдаланиш мумкин: NAME ва VALUE. NAME атрибути *майдоннинг* номини ифодалайди. VALUE атрибути — Submit тугмаси матнини кўрсатади.

<INPUT TYPE=submit VALUE="Хабарни жўнатиш "/>

TYPE=hidden атрибути

Яширин *майдон*. *INPUT* тэгини TYPE=hidden атрибути билан қўлланилиши фойдаланувчига маълум бўлмаган NAME ва VALUE атрибутларидаги қийматларни жўнатишга имкон беради.

<TEXTAREA> – кўп сатрли матн киритишни ташкил этиш тэги

Баъзан формада кўп сатрли матнларни киритиш талаб этилади. Бунинг учун ***<TEXTAREA>*** тэги ёрдамида бир неча сатрдан иборат бўлган матн майдони ташкил этиш мумкин. Ушбу тэг учта атрибут билан ишлатилади: COLS, NAME ва ROWS.

- **Атрибут COLS**

Майдоннинг устунлари (белгилар сони) сонини белгилайди.

- **Атрибут NAME**

Майдоннинг номини белгилайди.

- **Атрибут ROWS**

Майдоннинг кўринувчи сатрлари сонини белгилайди.

<TEXTAREA NAME=мавзу COLS=38 ROWS=3></TEXTAREA>

<SELECT>- формада рўйхатдан фойдаланиш тэги

Агарда *форма* мукаммал бўлса, гоҳида унда ҳаракатланувчи рўйхат ҳам қўлланилади. Бунинг учун ***SELECT*** тэгидан фойдаланилади. Рўйхат бўлимларини аниқлаш учун *<OPTION>*тэгидан фойдаланилади. *<SELECT>* тэги муҳим бўлмаган учта атрибутни қўллаб қувватлайди: MULTIPLE, NAME ва SIZE.

MULTIPLE атрибути

Бир вақтнинг ўзида бир нечта вариантни танлаш имконини беради.

NAME атрибути

Объект номини ифодалайди.

SIZE атрибути

Рўйхатни кўринувчи сатрлари сонини ифодалайди. SIZE > 1 бўлган ҳолда браузер оддий рўйхатни кўрсатади. *Формада <OPTION>* теги фақат *<SELECT>* тэглари орасида қўлланилади. Параметрлари: SELECTED ва VALUE.

SELECTED атрибути

Дастлаки ҳолатда ушбу элемент танланган эканлигини билдиради.

VALUE атрибути

Рўйхатга ўзлаштирилиши мумкин бўлган қийматни ифодалайди.

Танлаш

<SELECT NAME="Танлаш">

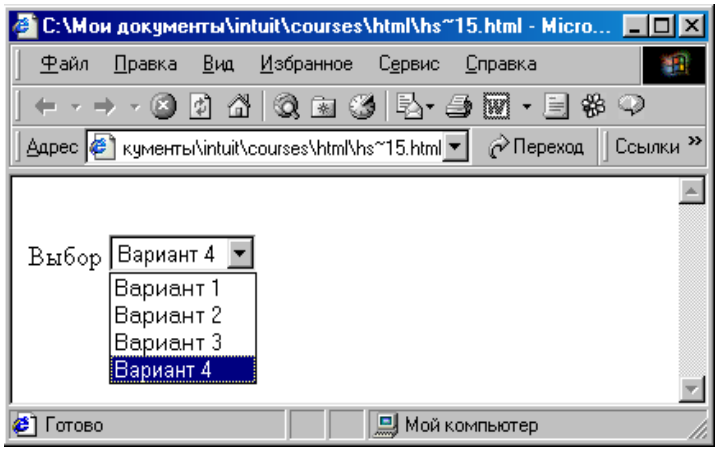
<OPTION>Вариант 1

<OPTION>Вариант 2

<OPTION VALUE="Вариант 3">Вариант 3

<OPTION SELECTED>Вариант 4

</SELECT>



Қалқиб чиқувчи меню

3. HTML тилида фреймлар

Узоқ вақтлар web-саҳифалар яратувчилари бутун сайт ҳаттоки портал саҳифалари бўйлаб ҳаракатнинг умумий сицемасини топишга ҳаракат қилдилар. Стандарт инциментарийни қўллаш учун бир ҳил менюни ҳар бир саҳифага нусхасини қўйишга тўғри келарди. HTML Frames спетсификатсиясини ривожланиши натижасида браузер ойнасини бир нечта ойналарга ажратиб, уларга бир-биридан муцақил саҳифаларни қўйиш имконияти пайдо бўлди.

Фреймлар кириб келиши билан боғлик энг асосий муаммо ҳар доимдагидек браузерларнинг ўзаро тўғри келмаслиги бўлди. Кўпгина браузерлар фреймларни «танимасди». Фреймлар спетсификатсияси HTML цандартларига 1997 йилда яратилган HTML 3.2 версиясидан бошлаб қўлланила бошланди. У вақтларда фреймлар Нецсапе фойдаланувчилари орасида оммавийлашган эди. HTML 4.0 дан бошлаб браузерлар фреймларни нормал «тушуна» бошладилар.

Фреймлар битта браузер ойнасига бир нечта турли УРЛ адресларга эга бўлган мустақил web-саҳифаларни юклаш имкониятини беради. Бу вазифани бажариш жуда оддий. Бунинг учун <body> елементи<frameset> елементи билан алмаштирилади. Бу контейнор саҳифада фреймлар ҳосил қилувчи <frame /> елементи учун мўлжалланган. Фреймлар ичида қайси фреймга чиқиши кўрсатилган алоҳида гипермурожаатлар бўлиши мумкин.

Фреймлар барузерни кузатув ойнасини ёнма-ён жойлашган бир нечта тўғри бурчакли соҳаларга бўлиш имконини беради. Мазкур бўлаклардан ҳар бирига алоҳида HTML-файл, яъни бошқалардан мустақил равишда кўздан кечирилувчи файлларни юклаш мумкин. Зарурият туғилганда фреймлар орасида ўзаро боғлиқликни ташкил этиш мумкин. Ўзаро боғлиқлик ташкил этилганда фреймлардан бирида ссылка танланса, бошқа фрейм ойнасида керакли ҳужжатнинг юкланишига олиб келади.

Гарчи HTML-ҳужжатларда фойдаланувчига ахборот акс эттирилишининг турли усуллари ҳавола этилсада, ахборотни ифодалашнинг фрейм тизими ҳам ўзининг афзалликларига эга. Қуйидаги холларда айнан фрейм тизими қўл келади:

- Бир сохада ишлаётганда бошқа бир сохага ҳужжатларни юклаш орқали бошқаришни ташкил этиш зарурати туғулганда;
- Экраннинг бошқа ҳудудларида нима бўлишидан қатъий назар экранда доимо кўриниб туриши керак бўлган ахборотни кўздан кечириш дарчасининг маълум қисмига жойлаштириш лозим бўлганда;
- Дарчанинг хар бири мустақил равишда кўриб чикилиши мумкин бўлган ёнма-ён бир неча сохаларида жойлаштириш қўлай бўлган ахборотни такдим этиш зарурати туғулганда.

Фреймлар тизимини тасвирлаш учун <FRAMESET>, <FRAME> ёки <NOFRAME> тэгларидан фойдаланилади.

<FRAMESET> тэги фреймларни белгилайди.

Фреймлардан ташкил топган Web-саҳифалар <BODY> бўлинмасига эга бўлиши мумкин эмас.

<FRAMESET> ва </FRAMESET> контейнерлари хар бир фреймни белгилаш блокини ўраб туради. Бундай контейнернинг ичида фақат <FRAME> тэглари ёки киритилган <FRAMESET> тэглари мавжуд бўлади.

<FRAMESET> тэгининг атрибутлари:

- ROWS,
- COLS.

Ушбу параметрлар қийматлари пикселларда, фоизларда ёки нисбий бирликларда берилиши мумкин. Қатор ёки устунлар сони мос рўйхатдаги қийматлар сони билан аниқланади. Масалан:

<FRAMESET ROWS =“100, 240, 140”> - учта фреймдан иборат тўпламни белгилайди. Қийматлар пикселларда берилган. Биринчи фрейм 100 пиксел, иккинчиси 240 пиксел ва ниҳоят сўнгги фрейм 140 пиксел баландликка эга.

<FRAMESET ROWS = “25%, 50%, 25%”> -

экраннинг мақбул баландлигидан юқори қаторнинг қиймати 25 фоиз, ўрта қаторники 50 фоиз, қуйи қаторники 25 фоиз эканлигини билдиради.

<FRAMESET COLS = “*, 2*, 3*”> - қийматлар нисбий бирликларда. “Юлдузча” – “*” фазони пропорционал тақсимлаш учун ишлатилади. Хар бир юлдузча бутуннинг бир қисмини билдиради. Ҳисоблаб топиш учун юлдузчалар олдидаги сонларни қўшиш ва хосил бўлган сондан касрнинг махражи сифатида фойдаланилади. Юқоридаги мисолда биринчи устун дарча умумий кенглигининг 1/6, иккинчи устун 2/6, учинчи устун 3/6 қисмини эгаллайди.

<FRAMESET COLS = “100, 25%, *, 2*”>.

<FRAME> тэги алохида файлларни белгилайди, бу тэг <FRAMESET> ва </FRAMESET> тэглари жуфтлигининг ичида жойлашиши лозим. Масалан:

<FRAMESET ROWS = “*, 2*”>

<FRAME>

</FRAME>

</FRAMESET>

<FRAMESET> тэги берилганида қанча алоҳида фреймлар белгиланган бўлса, шунча фрейм тэгларини ёзиш лозим.

<FRAME> тэги атрибутлари:

- SRC
- NAME
- MARGINWIDTH
- MARGINHEIGHT
- SCROLLING
- NORESIZE
- FRAMEBORDER=YES/NO (Фақат IE лар учун)

SRC атрибути бошидан бошлаб мазкур фреймга юкланувчи ҳужжатнинг URL-манзилини белгилайди. Одатда бундай манзил сифатида асосий ҳужжат қайси каталогда бўлса, уша ерда жойлашган HTML-файлнинг номидан фойдаланилади. Масалан:

<FRAMESET SRC=“sample.html”>

Зеро, фреймни тасвирлашда берилган HTML-файл тўлиқ HTML-ҳужжат бўлиши керак, яъни у HTML, HEAD, BODY ва бошқаларга эга бўлиши лозим. Агар фреймдан тасвирни акс эттиришда фойдаланилса, унда:

<FRAME SRC=“http://www.bhv.ru/exampl.gif”>

NAME параметри берилган фреймга ссылка сифатида ишлатиш мумкин бўлган фреймнинг номини белгилайди. Масалан:

<FRAME SRC=“sample.html” NAME=“frame1”>

frame1 деб номланган ушбу фреймга ссылка қилиниши мумкин. Масалан:

<A HREF=“other.html” TARGET=“frame1”>

frame1 </A> фреймига other.html файлини юклаш учун шу ерга сичқонча курсори босилади.

MARGINWIDTH ва MARGINHEIGHT атрибути фрейм хошия(чегара) кенглигини белгилайди.

Атрибутлар қийматлари пикселларда берилади.

Масалан:

<FRAME MARGINWIDTH =“5” MARGINHEIGHT = “7”>

Бу ерда фрейм юқори ва пастда 5 пиксел, ўнг ва чап томонларидан эса 7 пиксел чегарага эга. Ишлатилиши мумкин бўлган энг кичик қиймат 1 пикселдир.

SCROLLING атрибутидан **прокрутка** йўлакларини акс эттиришни бошқаришда фойдаланилади. Унинг синтаксиси

<FRAME SCROLLING= “YES/ NO/ AVTO”> кўринишга эга.

NORESIZE атрибути фойдаланувчи томонидан фрейм ўлчами ўзгартирилишининг олдини олишда ишлатилади. Масалан:

<FRAME NORESIZE>.

Табиийки NORESIZE атрибутининг битта фреймга нисбатан қўлланилиши бошқа фреймлар ўлчами ўзгартирилишининг ҳам олди олинишига сабаб бўлади.

Гарчи фреймлар тизими HTML 4.0да стандарт билан мустахкамланган бўлсада, <NOFRAMES> тэги фреймларни қўллаб-кувватламайдиган браузерлар ёрдамида кўздан кечиришда асқотади. Демак, фреймларга боғланмаган браузерлар учун <NOFRAMES>ва </NOFRAMES> тэглари жуфтлигидан фойдаланилади. Масалан:

<NOFRAMES> бутун HTML-ҳужжат</NOFRAMES>

Мазкур тэглар орасига жойлаштирилган барча маълумотлар фреймларни қўллаб-қувватлаш имкониятига эга бўлмаган браузерлар ёрдамида акс эттирилади. Фреймларга боғланган браузерлар эса <NOFRAMES> ва </NOFRAMES> орасидаги барча ахборотга боғлиқ емас. Юқорида келтирилган параметрлар ишлатилган мисолларни кўриб чикамиз.

Фреймларга мисол.

Экранда биз 3 қисмдан иборат саҳифани кўряпмиз. Улар ўзгартиришга эҳтиёж бўлганда керак бўлади. Бу 3 қисмда қайсики, 4-ҳужжат ичида таъминланган турли ҳужжатлар жойлашади. Барча ҳужжатлар бир-биридан муцақил ҳолда таъминланади. Саҳифанинг ўнг томонида Интернетда иҳтиёрий

калит сўзларда қидириш мумкин бўлган фомалар жойлашган, саҳифани қуйи томонида оддий сурат-ссилкаси жойлашган. Бу цруктурани барча қисми “frameset” деб номланувчи тамомила кичик ҳужжат асосида ташкил қилинган. (Инглизчадан таржима қилинганда “frameset”- “frame ташкилотчиси” деган маънони англатади)

Тўлиқ HTML коди қуйидаги кўринишда бўлади:

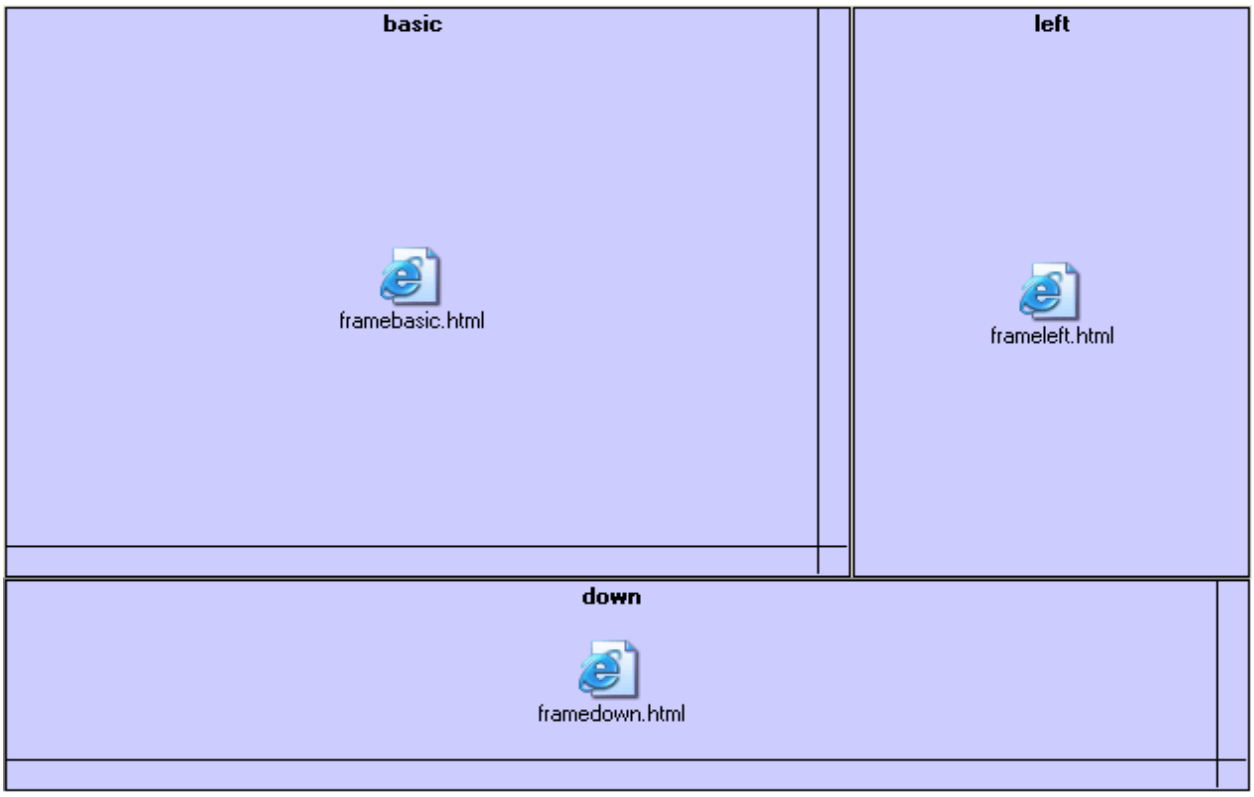
```
html>
<head>
<meta http-equiv="content-type" content="text/html;charsetқwindows-1251">
<title>HTMLда фреймлар </title>
</head>
<frameset rows="*,112" border="1" framespacing="1" frameborder="yes">
<frameset cols="*,200" frameborder="no" border="1" framespacing="1">
<frame src="framebasic.html" name="basic" scrolling="yes">
<frame src="frameleft.html" name="left" scrolling="no" noresize>
</frameset>
<frame src="framedown.html" name="down" scrolling="yes">
</frameset>
<noframes>
<body>
</body>
```

</noframes>

</html>

“Frameset” теги параметлари: “rows”параметри- сатр баландлигини, “frameborder” параметри- юқори кадр ва пацки кадр ўртасидаги чегара қалинлигини кўрсатувчи муҳим вазифаларни амалгам оширади. Қолган параметрлар мажбурий эмас. Биринчи фреймсет ўрнига асосий фрамесетдаги янгиланган фреймсет бўлади.

Агар қуйидаги расмларга яхшилаб қаралса, ҳаммаси тушунарли бўлади:



Биз ўқиётган саҳифа “basic” номли ва уни ўнг томонида жойлашган “left” деб номланувчи саҳифалар киритилган. Ушбу икки саҳифа ўз навбатида кадр пайдо қилувчи фрамесет ҳосил қилади. У ва пацки кадр “down” енг муҳим фреймсетда келади. Кадрлар иерарҳияси жадваллар иерарҳияси ўхшайди. Кадр жадвал ва фреймсет ячейкаларини таққослайди:

<Frame> тегида айланма йўллар борлиги ва кадрлар файли манзили кўрсатилиши керак.(Кўриб турибмизки, чапдаги кадрда у йўқ.)

Кадрлар орасидаги ўтиш ҳам ўзига ҳос ҳусусиятларга эга. Масалан, бир кадрдан бошқа кадрга юкланган саҳифага кўрсатиши мумкин. Бунинг учун

Target="..." теги параметридан фойдаланилади:

Ссилка матни

Бу ссилка ўнг кадрда web-саҳифа юклайди, бу ссилка эса қуйида жойлашган. Ссилка коди-1;

I ссилка

Бутун ойнада ҳамма кадрлар ўрнига саҳифалар қандай юкланади? Бунинг учун Target="_top" заҳира сифатида сақланади.

Бутун ойнага саҳифа юклаш. Бутун ойна учун код ссилкаси:

 Бутун ойнага саҳифа юклаш

Ичма-ич фреймлар. Агар экранда горизонтал ва вертикал фреймларни бирданига ҳосил қилмокчи бўлсак, ичма-ич фреймлардан фойдаланишимиз мумкин. Масалан тепада тўлиқ сатрли фрейм, унинг пацида икки уцунга ажратилган фреймларни ҳосил қилайлик. Бунинг учун икки ишни қилиш керак. Биринчидан бизга икки қаторли фрейм керак бўлади. Иккинчидан иккинчи қатор фреймни икки уцунга ажратиш керак:

```
<frameset rows="100, *">
<frame src= "banner.html" scrolling= "no" norisize ="noresize" />
<frameset cols= "25%, 75%">
<frame src= "index.html" />
```

<frame src= "viewer.html" marginwidth= "5" marginheight= "5"/>
</frameset>

4.Фреймлар орасидаги ўзаро таъсир

Фреймлар билан ишлаётганда фойдаланувчи учун қўлай бўлган ҳужжат юклаш схемасини яратиш мумкин. Фреймлар орасидаги ўзаро алоқа ҳужжатларни бошқа фреймдаги буйруқлар ёрдамида айнан танланган фреймга юклаш имконини беришидадир. Бу мақсадда <A> тэгининг TARGET атрибутидан фойдаланилади. TARGET атрибути ушбу ссылка кўрсатаётган ҳужжат юкланувчи фрейм ёки браузер ойнаси номини белгилайди. Ўзгартирилмаган ҳолда ушбу параметр йўқ бўлганда ҳужжат жорий фрейм ёки ойнада юкланади. Фрейм номи сифатида мавжуд дарча ёки фрейм номи берилиши ёки бўлмаса янги ойна очиш учун янги ном берилиши мумкин. 4 та захирадаги номлар бор. Улар:

_blank, _self, _top, _parent. Булардан ташқари “_” белгиси билан бошланувчи хар қандай номдан фойдаланиш мақсадга мувофиқ эмас.

TARGET=“_blank” – ҳужжатнинг янги ойнага юкланишини таъминлайди. Бу ойна номга эга бўлмаслиги туфайли унга бошқа ҳужжатни юклашнинг иложи бўлмайди.

TARGET=“_self” дан фойдаланилганда ҳужжат жорий фреймга юкланади.

TARGET=“_top” ҳужжатнинг бутун дарчага юкланишига сабаб бўлади.

TARGET=“_parent” ҳужжатнинг жорий фреймнинг фрейм-ота-онаси томонидан эгалланган сохасига юкланишига олиб келади.

Таркиби “A” фреймига юкланган frame_a.html файлининг биттагина test.html файлига TARGET параметрининг турли қийматларига эга 6 та ссылкаси мавжуд.

<iframe> елементи <frame /> елементи билан тўғридан-тўғри боғланган, лекин<frameset> елементига умуман алоқаси йўқ. <iframe> елементи HTML ҳужжатда ички мустақил фреймлар яратиш имкониятини беради. <iframe>

елементи иҳтиёрий саҳифада <body> нинг ичига жойлашиши мумкин. Унинг вазифаси саҳифада бошқа бир ҳужжатни<frame /> <frameset> да кўрсатгани каби кўрсатишдир.

Мисол:

<iframe src="jad.html" width="300" height="300" frameborder="0" scrolling="auto">
Мана сизга фрейм!!!
</iframe>

<iframe> елементи <frame/> елементининг frameborder, marginwidth, marginheight, scrolling каби ҳамма атрибутларини қабул қилиши мумкин. Булардан ташқари унинг width ва height атрибутлари ҳам мавжудки, улар қўйилаётган фреймнинг бўйи ва эни ўлчамларини пикселларда белгилайди. Яна <iframe> елементи align атрибутига эга бўлиб, у одатдагидек, right ва left қийматлар қабул қилади. <iframe> елементининг яна бир ҳусусияти шундаки у ўз ичига олган матнни қачонки фойдаланувчи браузери ички фреймларни қўлламасагина экранга чиқаради.

Фреймларни юқорида келтирилганлардан бошқа қўшимча имкониятлари ҳам мавжуд. Масалан, бирор фреймга саҳифани юклаш ёки фреймларни янги ойнага очиш.

Targetатрибути қуйидаги қийматлар қабул қилиши мумкин:

- /self.Мурожаат қўйилган фрейм ўзида ҳужжат очилишини таъминлайди.
- /top.Бу қийматни қўллаб, шу ойнанинг ўзида фреймларни ўчириб ҳужжатни юклаш мумкин.
- /blank.Ҳужжатни янги ойнада очилишини таъминлайди.

Назорат саволлари:

1. HTMLтилида формани вазифасини изоҳлаб беринг?
2. Фреймлар қандай ишлайди?
3. Ҳужжатда линкни фреймга боғлашни тушунтиринг?
4. Объект нима ва қанақа объектларни биласиз?
5. Форманинг қайси нуқтасига маълумот киритилишини белгиловчи тегни мисоллар ёрдамида тушунтиринг?
6. Киритилаётган матн майдони параметрлари қандай белгиланади?
7. Формада ҳаракатланувчи рўйҳат қўллаш учун қандай параметрлар керак?
8. Фреймлар ва формаларни қўллаш билан қандай имкониятларга эга бўлишимиз мумкин?
9. Frametеги атрибутларини м9соллар ёрдамида тушунтиринг?
10. <iframe> елементини қўллаш билан қандай имкониятларга эга бўлишимиз мумкин?

Фойдаланилган адабиётлар:

1. Университет портали: Intuit.ru.
2. С.С.Қосимов. Ахборот теҳнологиялари. Тошкент “Алоқачи” 2006.
3. Спейнауэр С., Куэрсиа В. Справочник Web-мастера. - К: "BHV", 1997. - 368 с.
4. Яргер Р., Риз Дж., Кинг Т. MySQL и mSQL. Базы данных для небольших предприятий и Интернета. - СПб: Символ-Плюс, 2000 - 560 с.
5. Хилайер С., Мизик Д. Программирование Active Server Pages. - М: "Русская редакция", 1999. - 296 с.
6. Xaitov F.N., Yusupov R.M., Botirov D.B., Sattarov A.R, Shukurov E.X. Web теҳнологиялар. Жиззаҳ. 2005.

Фойдаланилган манбалар:

1. Университет портали: http://www.Intuit.ru/
2. http://pda.coolreferat.com/
3. http://www.z-oleg.com/
4. http://www.i2r.ru/
5. http://www.google.ru/
6. http://uz.wikipedia.org/wiki/HTML
7. http://www.cgi.ru

5-маъруза. Web-дастурлаш фанида мижоз томонида дастурлаш ва JavaScript тилини HTML хужжатга

Режа:

1. Клиент томони (соҳаси) да дастурлаш. JavaScript га кириш
2. JavaScript нинг объектли модели тушунчаси
3. JavaScript нинг URL-схемаси
4. Синфлар иерархияси

Калит сўзлар: Мижоз томонда дастурлаш, JavaScript тили, Java тили, URL схемаси, синфлар иерарҳияси, обектлар, усуллар, ҳусусиятлар, ҳолатлар.

Ишдан мақсад: Веб-дастурлаш фани асосида ишлаш жараёнида мижоз томонида дастурлаш жараёнини ўрганиш ва JavaScript тилини HTML ҳужжатга

жойлаштириш босқичлари ва синфлар иерарҳияси ҳақида талабаларда кўникмаларни шакллантириш.

1. Клиент томони (соҳаси) да дастурлаш. JavaScript га кириш.

Web саҳифани генерация қилиш жараёнида "клиент-сервер " архитектураси билан боғлиқ равишда амалга оширилади. Саҳифалар клиент томонида ҳам сервер томонидаги каби генерация қилинади. 1995 йилда Netscape компанияси мутахассислари клиент томонидаги саҳифаларни генерация қилиш учун махсус дастурлаш тили яратишган ва уни JavaScript деб номлашган.

JavaScript – клиент томонидаги гиперматнли Web саҳифанинг сценарийларини бошқарувчи тилдир. Аниқроқ айтадиган бўлсак, JavaScript – бу фақатгина клиент томонидаги дастурлаш тили эмас. JavaScript нинг аждоди Liveware - Netscape сервери томонидаги восита ҳисобланади. Шундай қилиб, JavaScript кўпроқ клиент томонидаги сценарийларни ташкил этувчи тил сифатиди оммавийлашган.

JavaScript нинг асосий ғояси HTML саҳифаларни кўриш вақтида HTML тэг ва контейнерларнинг атрибутлари қийматларини ва хусусиятларини ўзгартиришдан иборат. Шу сабаб саҳифани қайта юклаш амалга ошмайди.

Амалиётда буни биз, саҳифа фонининг рангини ёки хужжатдаги расм хусусиятларини ўзгартиришда, янги ойна очиш ёки огоҳлантириш бериш жараёнларида яққол кузатишимиз мумкин.

JavaScript тили ECMA (European Computer Manufacturers Association – Европа компьютер ишлаб чиқариш ассоциацияси) томонидан стандартлаштирилган. Ушбу стандартлар ECMA-262 ва ISO-16262 номларини келтириб чиқарди. Бу стандартлар JavaScript 1.1 га мош тушувчи ECMAScript тилини тақдим этади. Бугунги кунда JavaScript нинг барча турлари ҳам ECMA стандартига мос тушавермайди.

JAVA бу – кўпгина замонавий программалаштириш тиллари билан рақобатлаша оладиган программалаштириш тилидир. Унинг бошқа программалаштириш тилларидан фарқи унда Интернет билан ишлашга мўлжалланган дастурлар яратишга мўлжалланган. У айниқса тармоқлараро ишлайдиган программалар яратувчилар орасида айниқса оммавийлашган. JAVA дастурларининг кўп кисмини кичкинагина тармоқлараро ишлайдиган программалардир. Уларнинг кичик ўлчамлари Интернет билан ишлашни оптималлаштиришга мўлжалланган. Яъни қанча кам хажмли маълумот узатилса, сайтни, программани ёки расмларни юклаш учун шунча кам вақт кетади. Шу тариқа апплетлар хосил бўлади. Бу программалаштитирш тилида ихтиёрий программаларни тузиш мумкин. JAVA дастури яратувчиси SUN компаниясининг сайтидан баъзи бир апплетларни олиш мумкин:

http://www.sun.com

Интернетдан кўплаб апплетларни бепул олиш мумкин:

http://www.yahoo.com/Computers_and_Internet/Programming_Languages/Java/Applets
http://java.sun.com/openstudio/
ва хоказо.

Апплетларни веб-сахифага юклаш мисоли:

<object codetype= "application/java" classid= "java:myapplet.class" standby= "Апплет юклаш" width=400 height=350> </object>

Бу ердаги myapplet.class юкланаётган апплетнинг номи ва у веб-сахифа юкланаётган каталогда жойлашган. Бундан ташқари апплетнинг тўлиқ URL адресини ифодаловчи codebase атрибутини хам ишлатиш хам мумкин:

<object codetype= "application/java" codebase= "http://www.jdpi.uz/applets/" classid= "java:myapplet.class" standby= "Апплет юклаш" width=400 height=350> </object>

Агар браузер апплетлар билан ишлай олса (хамма баузерлар хам апплетлар билан ишлаш имкониятига эга эмас) бошқа элементлар каби апплетлар экранга чиқади ва ўз вазифасини бажаради.

2. JavaScript нинг объектли модели тушунчаси

Клиент томонидаги саҳифани яратишни бошқаришда хужжатнинг объектли механизмидан фойдаланилган. Бунда ҳар бир HTML-контейнер-бу объект ҳисобланади ва қуйидаги учликни ташкил этади:

- хусусиятлар
- усуллар
- холатлар

Объектли модел саҳифалар ва браузерлар ўртасидаги боғланишсифатида кўриниши мумкин. Объектли модел – бу HTML код орқали берилган элементларни объект, усул, хусусият ва холатлар кўринишида таниш ва улар билан ишлаш демакдир. У ёрдамида биз браузерга ва фойдаланувчига мурожаат қилишимиз, хабарлар юборишимиз мумкин. Браузер бизнинг буйруқралимизни бажаради ва экранда саҳифанинг керакли қисмларини ўзгартиради.

Объектлар бир хил типли хусусиятлар, усуллар ва холатлар тўпламини бир хил типли объектлар синфларида бирлаштиради. Объектларнинг ўзлари фақат хужжатни браузер ёрдамида юклашда ёки дастурнинг натижаси сифатида намоён бўлади. Ушбу ҳолатни доимо ёдда тутиш керак.

Хусусиятлар

Кўпгина HTML-контейнерларда атрибутлар мавжуд. Масалан, якор контейнерида <A ...>...</A> HREF атрибути мавжед. Ушбу атрибут уни гипер мурожаатга айлантиради:

```
<A HREF=intuit.htm>intuit</A>
```

Агар <A ...>...</A> якор контейнерини объект сифатида кўрадиган бўлсак, HREF атрибути "якор" объектини хусусияти ҳисобланади:

```
document.links[0].href="intuit.htm";
```

Барча атрибутлар қийматларини ҳам ўзгартириб бўлавермайди. Масалан график расимларнинг ўлчамлари дастлабки берилган қиймати асосида аниқланади, яъни уларни ўзгартириб бўлмайди. Кетма кет келган барча расимлар қийматлари ўзининг дастлабки қийматигача масштабланиши мумкин. Microsoft Internet Explorer да расим ўлчамлари ўзгартирилиши мумкин.

Умумийлик учун расм хусусиятлари JavaScript да HTML-разметкада мавжуд бўлмаган объектларга бўлинади. Масалан, восита сифатида Navigator деб номланувчи объектни, ёки JavaScript даги энг асосий объектлардан – браузер ойнаси объектини олишимиз мумкин.

Усуллар

JavaScript атамаларида объект усуллари унинг хусусиятларини ўзгартирувчи функцияларни англатади. Масалан, "документ" объектида open(), write(), close() усуллар мавжуд. Ушбу усуллар мавжуд хужжатнинг қайта ишлаш ёки таркибини ўзгартириш учун хизмат қилади. Оддий мисол келтирамиз:

```
function hello()
{ id=window.open("","example","width=400, height=150");
id.focus(); id.document.open();
id.document.write("<H1>Салом!</H1>");
id.document.write("<HR><FORM>");
id.document.write("<INPUT TYPE=button VALUE='Ойнани ёпиш ");
id.document.write("onClick='window.opener.focus();window.close();'>");
id.document.close();
```

}

Ушбу мисолда open() усули хужжатга ёзиш имкониятини яратади, write() усули ушбу ёзишни амалга оширади, close() усули хужжатга ёзишни ёпади. Буларнинг барчаси оддий файлга ёзган каби амалга ошади. Агар ойнада холат сатри мавжуд бўлса (одатда хужжатнинг юкланиш даражаси берилади), хужжатга ёзиш жараёни тугалланмаган бўлса, хужжат юкланиш вақтида унда тўғри тўртбурчак шаклидаги ёзув давом этаётганлигини ифодаловчи белги “кўринади”.

Ҳолатлар

Усуллар ва хусусиятлардан ташқари объектларни ҳолатлар билан ҳам характерлаш мумкин. Шахсан, JavaScript да дастурлашда ушбу ҳолатларни қайта ишловчи воситалар мавжуд. Масалан, button типидаги объект билан onClick ҳолати амалга ошиши мумкин, яъни фойдаланувчи тугмани босиши мумкин. Ушбу атрибут қиймати сифатида дастурчи томонидан JavaScript да тузилган ҳолатни қайта ишловчи дастур кўрсатилади:

```
<INPUT TYPE=button VALUE="Нажать" onClick="window.alert('Салом!');" />
```

Ҳолатларни қайта ишлаш жараёнлари уларнинг ҳолатлари билан боғлиқ контейнерларда кўрсатилади. Масалан, BODY контейнери бутун хужжатнинг хусусиятини аниқлайди, шунинг учун бутун хужжатни ёпишни қайта ишловчи ҳолат onLoad атрибутининг қиймати сифатида BODY контейнери ичида берилади.

Изоҳ. *Қатъий айтиш мумкинки, ҳар бир браузур, Internet Explorer, Netscape Navigator ёки Opera да бўлганидек, ўзининг объектли моделига эга. Турли браузерлар объектли моделлари (ҳатто турли версиялари) бир биридан фарқланади, лекин мантиқий таркиби бир ҳилда бўлади*.

Кодни HTML-саҳифага жойлаштириш

JavaScript кодини браузерда бажарилаетганда браузердаги JavaScript интерпретатори ёрдамида амалга оширилади. JavaScript ни қўллашда тўртта функционал усулдан фойдаланиш мумкин:

- гиперматнли мурожаат (URL схема);
- ҳолатни қайта ишловчи (handler);
- подстановка (entity) (Microsoft Internet Explorer нинг 5.X ва юқори версияларида мавжуд);
- қўйиш ёки ўрнатиш (вставка).

HTML хужжат яратаётганда JavaScript нинг бир нечта усулларидан фойдаланиш мумкин.

3. JavaScript нинг URL-схемаси

URL (Uniform Resource Locator) схемаси – бу Web-технологиянинг асосий элементларидан бири ҳисобланади. Web да ҳар бир ахборот ресурси ўзининг уникал URL ига эга. URL A контейнернинг HREF атрибутида, IMG контейнернинг SRC атрибутида, FORM контейнерининг ACTION атрибутида ва бошқаларда берилади. Барча URL мулоқот протоколи турига қараб турли қисмларга бўлинади, масалан, FTP-архивга боғланиш учун ftp схема қўлланилади, Gopher-архивга боғланиш учун - gopher схемадан фойдаланилади, электрон почтани жўнатиш учун - smtp схемадан фойдаланилади. Схема тури URL нинг биринчи компонентаси орқали аниқланади: http://intuit.ru/directory/page.html

Гиперматнли тизимли дастурлаш тилининг асосий вазифаси гиперматнли ўтишларни дастурлашдир. Бу шуни англатадики, у ёки бу гиперматнли ссылканинг

босилиши гиперматнли ўтишни амалга оширувчи дастурни ишга туширади. Web-технологиада шунга ўхшаш стандарт дастурлар саҳифани юклаш дастурлари ҳисобланади. JavaScript шу стандарт дастурларни фойдаланувчи дастурига айлантиради. HTTP протокол бўйича стандарт ўтишлардан фарқланиш мақсадида JavaScript да алоҳида URL схема жорий этилган:

<A HREF="JavaScript:JavaScript_код">...</A>

<IMG SRC="JavaScript:JavaScript_код">

Масалан, “Диққат!!!” номли гиперматнли ссылка босилганда огоҳлантириш ойнасининг очилиши қуйидагича амалга оширилади:

<A HREF="JavaScript:alert('Внимание!!!');"> Диққат!!!</A>

submit типидаги тугмани босиш орқали формадаги матн объекти тўлдирилиши қуйидагича амалга оширилади:

<FORM NAME=myform METHOD=post ACTION="JavaScript:window.document.myform. myname.VALUE='Сиз Click тугмасини босдингиз';void(0);">

<TABLE BORDER=0 width=“100%”>

<TR>

<TD><INPUT NAME= “myname” value= “” /></TD>

<TD><INPUT TYPE=submit VALUE=Click></TD>

<TD><INPUT TYPE=reset VALUE=Reset></TD>

</TR>

</TABLE>

</FORM>

URL да мураккаб дастурларни жойлаштириш ва функцияларни чақириш мумкин. JavaScript нинг бу схемаси барча браузерларда ҳам ишлайвермайди, Netscape Navigator типидаги ва Internet Explorer нинг тўртинчи версиясидан бошлаб ишлайди.

Ҳолатларни қайта ишловчилар

Ҳолатни қайта ишловчи типидпги (handler) дастурлар, шу ҳолатга алоқадор контейнер атрибутида берилади. Масалан, тугма босилган вақтда click ҳолати амалга ошади:

<FORM><INPUT TYPE=button VALUE="Тугма" onClick="window.alert('tuit');"> </FORM>

Подстановкалар

Подстановкалар (entity) Web-саҳифада жуда кам учрайди. Шунга қарамай у HTML-саҳифани браузер томонида генерация қилиш қулай восита ҳисобланади. Подстановкалар HTML-контейнер атрибутининг қиймати сифатида фойдаланилади. Масалан, стандарт ҳолат бўйича форма объектлари маълумотларини жўнатиш учун адрес сифатида жорий саҳифа URL адреси кўрсатилади:

```
<script>
function myfunction(){
str = window.location.href;
return(str.length);
}</script>
<form><input value="&{window.location.href};" size="&{myfunction()};" /></form>
<script><!—бу изоҳ ...javascript-код...// --></script>
<body>Ҳужжат танаси </body>
```

Биламизки, ҳужжатнинг <head> қисмидаги матнлар браузер ойнасида кўринмайди. Шунинг учун бу қисмга ҳужжат танасида чақирилувчи ва ишлатилувчи ўзгарувчилар ва функциялар жойлаштирилади. Бу соҳада Netscape Navigator браузери Internet Explorer га қараганда бироз қатъийроқ. Агар ҳужжат танасидаги функция сарлавҳа қисмида эълон қилинмаган бўлса, ушбу функция аниқланмаганлиги ҳақида хабар беради.

Мисол: Функцияларни жойлаштириш ва фойдаланиш:

```
<html><head>
<script>
function time_scroll(){
d = new Date();
window.status = d.getHours()+":"+d.getMinutes()+":"+d.getSeconds();
setTimeout('time_scroll();',500);
}
```

```
</script>

</head>

<body onLoad= “time_scroll();”>

<h1>ҳолат сатридаги соат </h1>
```

Internet Explorer 4.0 да подстановкалар ишламайди, шу боис улардан фойдаланишда эҳтиёт бўлиш керак. Бунда аввало браузер турини билиш талаб этилади.

Ўрнатиш (SCRIPT контейнери-интерпретаторни мажбурий чақириш)

SCRIPT контейнери – бу подстановка усулининг ривожланган варианти ҳисобланади. Жумладан, SCRIPT одатда Server Side Includes, яъни сервер томонидаги ҳужжатларни генерация қилувчи ҳам деб аталади. Интерпретатор SCRIPT теглари орасидаги барча қисимни генерация қилади ва шундан сўнг яна HTML қисимга қайтади.

SCRIPT контейнери иккита асосий функцияни бажаради:

- HTML-ҳужжатга кодни жойлаштириш;
- HTML-разметкаларни браузер томонида шартли генерациялаш.

Биринчи функцияси ўзгарувчилар ва функцияларни жойлаштириш учун қўлланилади. Иккинчиси - бу хужжатни юклаш ёки қайта юклаш вақтида JavaScript код натижасини жойлаштиришдир.

HTML-хужжатга кодни жойлаштириш

Шахсан, бу ерда асосий хилма хиллик йўқ. Код сарлавҳа контейнери HEAD орасига ҳам, BODY контейнери орасига ҳам жойлаштирилиши мумкин. Сарлавҳа қисмида қўлланилишини кўриб ўтамиз.

Сарлавҳа қисмида код SCRIPT контейнери орасига жойлаштирилади:

```
<HTML>
<HEAD>
<SCRIPT>
function time_scroll()
{
d = new Date();
window.status = d.getHours()+":"+d.getMinutes()+":"+d.getSeconds();
setTimeout('time_scroll();',500);
}
</SCRIPT>
</HEAD>
<BODY onLoad=time_scroll()>
<CENTER>
<H1>Ҳолат сатридаги соат </H1>
<FORM>
<INPUT TYPE=button VALUE="Ойнани ёпиш " onClick=window.close()>
</FORM>
</CENTER>
</BODY>
```

```
</HTML>
```

Ушбу мисолда биз ҳужжат сарлавҳасида time_scroll() функциясини яратдик ва унга BODY (onLoad=time_scroll()) контейнерининг load ҳолатида мурожаат қилдик.

Қуйидаги функцияни яратиш ва чақириш орқали алоҳида ойна яратиш мумкин:

```
function sel()
{
id = window.open("","example","width=500,height=200,status,menu");
id.focus();
id.document.open();
id.document.write("<HTML><HEAD>");
id.document.write("<BODY>");
id.document.write("<CENTER>");
id.document.write("<H1>Change text into child window.</H1>");
id.document.write("<FORM NAME=f>");
id.document.write("<INPUT TYPE=text NAME=t SIZE=20 MAXLENGTH=20 VALUE='This is the test'>");
id.document.write("<INPUT TYPE=button VALUE='Close the window' onClick=window.close()></FORM>");
id.document.write("</CENTER>");
id.document.write("</BODY></HTML>");
```

```
id.document.close();
}
<INPUT TYPE=button VALUE="Ҳолат сатрини ўзгартириш"
onClick="id.defaultStatus='Салом'; id.focus();">
```

4. Синфлар иерархияси

Объектга-мўлжалланган дастурлаш тили объектлар дарахтидан ташкил топади. JavaScript да бу иерархик дарахт Window объектидан бошланади, яъни ҳар бир объект у ёки бу ойнада ёзилади. Ихтиёрий объектга ёки объект хусусиятига мурожаат қилиш учун ундан юқорида турганг объект орқали мурожаат қилиш керак бўлади:

Умуман айтганда, JavaScript классик объектли тил ҳисобланмайди (уни соддалаштирилган объектли тил ҳам дейиш мумкин). Унда меросийлик ва наслдорлик мавжуд эмас. Дастурчи function оператори ёрдамида ўзининг класини, синфини объектини яратиши мумкин, аммо уларни яратишда одатда стандарт объектлардан ҳам фойдаланади. Бу шуни англатадики, JavaScript-дастурнинг амал қилиш соҳаси жорий саҳифа чегарасидан чиқиб кетмайди.

Баъзан JavaScript нинг турли объектларида бир ҳил номли ҳусусиятлар бўлади. Бу ҳолда дастурчи қайси объект хусусиятига мурожаат қилаётганини аниқ кўрсатиши керак. Масалан, Window ва Document ларда location ҳусусияти мавжуд. Фақат, Window учун бу Location синфи объекти, Document – URL да кўрсатилиб юкланаётган ҳужжатни адресини ифодалайди.

Таъкидлаш керакки, кўпгина объектларда объект хусусиятларини оддий қийматга ўзгартирувчи стандарт усуллар мавжуд бўлади. Масалан, стандарт ҳолда барча объектлар учун белгиларни сатрга айлантирувчи усул мавжуд: toString().

Назорат саволлари:

1. Мижоз томонда дастурлаш деганда нимани тушунасиз?
2. JavaScript тили ҳақида маълумот беринг?
3. Гиперматнли Web-саҳифанинг сценарийлари қандай бошқарилади?
4. JavaScript тили объекти нима ва қанақа объектларни биласиз?
5. JavaScript тили функсиясини HTML ҳужжатда эълон қилинг.
6. JavaScript тили синфлари иерархиясини нима?
7. URL схемаси қандай элемент ҳисобланади?
8. Қийматларини ўзгартириб бўлмайдиган атрибутларга мисол келтиринг?

Фойдаланилган адабиётлар:

7. Т. Стауфер. Создание web-цранитс. Самоучител. – «Питер», Санкт-Петербург, 2003 г.
8. А. Гончарев. HTML. Самоучител. – «Питер», Санкт-Петербург, 2001 г.
9. Аллен Вайк. JavaScript. Энтсиклопедия ползователя: пер. с. анг. – «ТИД» «ДС», Киев, 2001 г.
10. webmastering –електрон ўқув қўлланма.
11. А.И. Тихонов. «Публикатсия данних в Интернет» Учебное пособие. Москва Издателцво «Мир» 2000

12. JavaScript ни ургапиш бўйича электрон қўлланмалар

Фойдаланилган манбалар:

10. http://uz.wikipedia.org/wiki/HTML
11. http://javascripts.boom.ru
12. http://www.vanta.ru/script/
13. http://www.vbnet.ru
14. http://www.scriptic.ru/
15. http://www.webacademy.com/
16. http://pacificwebart.com/
17. http://www.yahoo.com/Computers_and_Internet/Programming_Languages/ J ava

6-ma'ruza. JavaScriptда маълумотлар типлари ва ўзгарувчилар билан ишлаш.

Режа:

1. JavaScript да ўзгарувчилар;
2. JavaScript да маълумотлар типи;
3. JavaScript тили операторлари;
4. JavaScript тилида функция.

***Калит сўзлар**:* ўзгарувчилар, маълумотлар типи, операторлар, функциялар.

Ишдан мақсад: JavaScript тилида маълумотлар типлари ваўзгарувчилар ҳақида тушунчага эга бўлиш ва талабаларда маълумотлар ва ўзгарувчилар билан ишлаш жараёни кўникмаларини ташкил қилиш.

1. JavaScript да ўзгарувчилар

JavaScript тилида ўзгарувчиларни ишлатиш мумкин ва уларни номлари билан адреслаш мумкин. Ўзгарувчилар глобалли ва локалли бўлиши мумккин. Глобалли ўзгарувчилар сценарийнинг хохлаган жойида рухсати бўлиши мумкин. Локалли ўзгарувчиларнинг харакати эса эълон қилинган ўзгарувчилар ичидаги функциялар билан чегараланган. Basic дастурлаш тили сингари JavaScript сценарийсини яратаётган вақтда аввалдан эълон қилинмаган ўзгарувчиларни ишлатиш мумкин.

Ўзгарувчилар эълони

Java Script да хамма ўзгарувчилар var калит сўзи орқали эълон қилинади ва қуйидагича кўрсатилган:

```
var MyHelloMsg;
```

Ўзгарувчи типи ўзлаштириладики қачонки, унга бирор бир қиймат ўзлаштирилса, қуйида аввалдан эълон қилинмаган матнли қатор ўзгарувчига ёзилмоқда:

```
MyMsg = "Салом!";
```

MyMsg ўзгарувчи номи ўзлаштирилгандан сўнг рухсат берилади.

Ўзгарувчи номини танлаганда, ыуйидаги оддий ыоидаларни ушлаб ыҳйиш керак:

Ўзгарувчи номи харфлардан ёки "_", "$" белгилардан бошланиш керак ва фақат харфлардан, сонлардан ва "_", "$" белгилардан иборат бўлиши керак;

Ўзгарувчилар номи JavaScript нинг захираланган калит сўзлари билар мос келмаслиги керак.

Қуйида JavaScript нинг захираланган калит сўзлар келтирилган:

break	**case**	**catch**	**class**	**continue**	**const**
debugger	**default**	**delete**	**do**	**else**	**enum**
export	**extends**	**false**	**finally**	**for**	**function**
if	**import**	**in**	**new**	**null**	**return**
super	**switch**	**this**	**throw**	**true**	**try**
typeof	**var**	**void**	**while**	**with**	

Бу сўзлар орасида JavaScript тилида ва унинг ривожланишида ўзлаштириш режалаштирилмоқда.

Ўзгарувчининг қийматини ўзлаштириш

"=" ўзлаштириш оператори ёрдамида ўзгарувчилар қиймати ўзлаштирилади. Мисол қилиб ўуйидаги ўзгарувчи келтирилган ва унда матнли қатор ёзилган:

```
var MyHelloMsg;

MyHelloMsg = "Hello, world!";
```

MyHelloMsg сонли ўзгарувчини дастурнинг хохлаган жойида ўзлаштириш мумкин, мисол учун:

```
MyHelloMsg = 4;
```

Бу оператор бажарилгандан сўнг ўзгарувчи типи ўзгаради, шунингдек интерпретация жараёнида браузер хеч қандай огохлантирувчи хабарларни юбормайди.

Ўзгарувчини махсус null қиймати орқали ўзлаштириш мумкин:

```
MyHelloMsg = null;
```

Бундай ўзлаштириш хеч қандай типда ўзгарувчини белгиламайди.

2. JavaScript да маълумотлар типи

JavaScript тилида бир нечта маэлумотлар типи мавжуж. Булар сонлар, матнли қаторлар, мантиқий маэлумотлар, объектлар, аниқланмаган типли маэлумотлар, хамда махсус тип null.

Сонлар

JavaScript тили хар хил форматдаги сонларни ишлатишга рухсат беради, булар бутун сонлар, сузувчи нуқтали ўнли форматдаги сонлар ва илмий нотация сонлар. Бутун сонлар 8, 10, 16 асосида берилиши мумкин. мисол учун:

25 10 асосидаги бутун сон

0137 8 асосидаги бутун сон

0xFF 16 асосидаги бутун сон

386.7 Сузувчи ўнли нуқтали сон

25e5

или 25E5 Илмий нотациядаги сон, 2500000 га тенг.

Айрим холларда "сон бўлмаган" арифметик функциялар келиб чиқиши мумки. JavaScript да айтилганидек NaN (Not a Number). "Сон бўлмаган" – бу хеч қандай сонга лойиқ бўлмаган махсус қиймат. Бу сонлар устида операция бажарилаётган вақтда, ва натижа сон кўринишида тақдим этилмаган холларда пайдо бўлади. "Сон бўлмаган" қийматга тўғри келишини isNaN функцияси ёрдамида текшириш мумкин.

Матнли қатор

Матнли қатор – бу бир ёки қўштирноқ кетма кетлик белгиси, мисол учун:

"Hello, world!"

""

"12345"

'Бу матнли қатор'

"" қатори –бўшдир. Қуйидаги 2 та ўзлаштириш эквивалент эмаслигини аниқлаймиз:

MyStr=""

MyStr1=null

Биринчи холда MyStr ўзгарувчисида матнли қатор сақланмоқда (бўш бўлса хам), иккинчисида эса хеч нарса.

Мантиқий маълумотлар

Мантиқий маълумотлар фақат 2 та қийматни, яъни True ва False ни ўз ичига олади. Бу қийматлар 0 ва 1 сонлар билан боғлиқ эмас. Бу қийматларнинг асосий

образи солиштириш операцияси бажарилаётган вақтга қаратилган, хамда шартли операциялар ишлатилганда хам.

Аниқланмаган типли маълумотлар

Агар ўзгарувчи эълон қилинган бўлса, аммо унга хали қиймат ўзлаштирилмаган бўлса, у холда у аниқланмаган типга бўлади. Мисол учун қуйидаги қаторда аниқланмаган типга эга бўлган MyVariable ўзгарувчиси эълон қилинган: ***var MyVariable;***

Агарда бу ўзгарувчини null қиймати билан ўзгартирсак, у холда ўзгарувчи типи ўзгаради ва null қийматга эга бўлган ўзгарувчига айланади: ***MyVariable = null;***

Маълумотлар типини ўзгартириш

Агарда ифодаларда хар хил типли ўзгарувчилар учраб қолса, JavaScript интерпретатори автоматик холда сонли маэлумотларни матнли қаторларга ўзгартириши мумкин. Тескари айлантиришни (қаторни-сонга) махсус функциялар ёрдамида, яъни parseInt ва parseFloat функциялари ёрдамида ўзгартириш мумкин. Буни қуйидаги мисол орқали кўриш мумкин:

Мисол

```
<HTML>
<HEAD><TITLE>Type conversion sample</TITLE></HEAD>
<BODY>
<H1>Type conversion sample</H1>
```

```
<TABLE>
<SCRIPT LANGUAGE="JavaScript">
var MyTextBuf = "";
MyTextBuf = 4 + " – тўрт сони" + "<BR>";
MyBuf2 = (parseInt("2") + 2) + "  - тўрт сони" + "<BR>";
document.write(MyTextBuf + MyBuf2);
</SCRIPT>
</TABLE></BODY></HTML>
```

Бу ерда биз ***MyTextBuf*** ўзгарувчисини эълон қилдик ва уни бўш қатор билан инициализация қилдик. Қуйида биз бу қаторда 4 сонни суммасини ва 2 та матнли қаторни ўзлаштирдик.

MyTextBuf = 4 + " – тўрт сони" + "
";

Бу ифода хисобланаётган вақтда 4 қиймат автоматик холда матнли қаторга ўзгаради. Кейинги йиғинди HTML хужжатларида ишлатиладиган   ажралмайдиган пробел белгисига ахамият бериш лозим. Агарда уни оддий пробелга алмаштирсак, қатор конкатенациясидан кейин бу пробел йўқолади. Кейинги қаторда parseInt функцияси ёрдамида матнли қатор “2” сонли қаторга ўзгаради, натижада 2 сони қўшилади, ундан кейин хар иккала матнли қаторларда конкатенация бажарилади:

MyBuf2 = (parseInt("2")+2)+" – тўрт сони"+ "
";

Натижада MyBuf2 ўзгарувчиси MyTextBuf.ўзгарувчиси каби қаторга эга бўлади.

3. JavaScript тили операторлари

Унар оператори

Унар оператори белгининг ўзгариши учун тўлдириш операциясини бажаришда, инкраментда хамда декрементда ишлатилади:

- тескари холатда белгининг ўзгариши

! Қушимча. Мантиқий ўзгарувчиларнинг қийматини реверсирования қилиш учун ишлатилади.

++ Ўзгарувчи қийматини ошириш. Ўзгарувчи префикси ёки унинг суффикси бўлиб қўлланиши мумкин.

-- Ўзгарувчи қийматини камайтириш. Ўзгарувчи префикси ёки унинг суффикси бўлиб қўлланиши мумкин.

Унар операторини ишлатишга доир мисоллар:

```
i=0;      // i тенг 0 даги ўзгарувчининг бошланғич қиймати

i++;      // i тенг 1 даги қиймат

--i;      // i тенг 0 даги қиймати

var j=3;  // j тенг 3 даги ўзгарувчининг қиймати

i = -j;   // i тенг -3 даги ўзгарувчининг қиймати

var fYes = true; // fYes тенг true даги ўзгарувчининг қиймати

testFlag(!fYes); // testFlag функциясига false қиймати узатилмоқда
```

Бинар оператори

Бинар оператори 2 та операндни бирлаштиради. JavaScript тилида бинар операторлари айириш, бўлиш, қўшиш, кўпайтириш хамда бўлинмани қолдиғини хисоблаш учун ишлатилади (кўрилади):

- Айириш

+ Қўшиш

* Кўпайтириш

/ Бўлиш

% Бўлинмани қолдиғини хисоблаш

Бу операторлар С тилида ишлатилганидек JavaScript да хам худди шундай ишлатилади, мисол учун:

```
i=0;        // i тенг 0 даги ўзгарувчининг қиймати
i = i + 1;  // i тенг 1 даги қиймат
var j=9;    // j тенг 9 даги ўзгарувчининг қиймати
i = j / 2;  // i тенг 4 даги ўзгарувчининг қиймати
k = j % 2;  // i тенг 1 даги ўзгарувчининг қиймати
```

Алохида битлар билан ишлаш оператори

Сценарияларда шундай операторлар ишлатиладики, улар алохида битлар билан ишлаш операторлари хисобланади, улар қуйидагилар: ВА, ЁКИ, МАНТИҚИЙ ИНКОР, ЭМАС:

& ВА

| ЁКИ

^ МАНТИҚИЙ ИНКОР

~ ЭМАС

Силжувчи операторлари

JavaScript да силжиш операцисини бажариш учун 3 та оператор курилган:

>> Силжиш ўнг томонга

<< Силжиш чап томонга

>>> Бўшатиладиган разрядларни ноллар билан тўлдириб ўнгга силжиш

Мунособат операторлари

Мунособат операторлари ўзгарувчиларнинг қийматини солиштириш учун ишлатилади. Бу операторлар солиштириш натижаларига боғлиқлик true ёки false мантиқий қийматларни қайтаради ва шартли операторларда асосий бўлиб ишлатилади. True қийматини қайтарадиган JavaScript тилининг мунособат операторлари кўрсатилган:

> Чап операнд ўнг операнддан катта

>= Чап операнд ўнг операнддан катта ёки тенг

< Чап операнд ўнг операнддан кичик

<= Чап операнд ўнг операнддан кичик ёки тенг

== Чап операнд ўнг операндга тенг

!= Чап операнд ўнг операндга тенг эмас

Мантиқий операторлар

|| ЁКИ оператори. True қиймат қайтаради, қачонки операндлардан бири true бўлса.

&& ВА оператори. True қиймат қайтаради, қачонки икки операнд true бўлса

Ўзлаштириш оператори

Ўзлаштириш оператори ўзгарувчиларнинг қийматини ўзлаштириш учун ишлатилади. JavaScript тилида ва C дастурлаш тилидаги каби бу оператор бошқа операторлар билан комбинациясига рухсат этилади. Қуйида ўзлаштириш операторини бошқа операторлар билан комбинацияси берилган:

= Оддий ўзлаштириш

+= Сонли қийматни катталаштириш ёки қаторларни қўшилиши

-= Сонли қийматни кичиклаштириш

*= Кўпайтириш

/= Бўлиш

%= Бўлишдан қолган қолдиқни хисоблаш

>>= Ўнгга силжиш

>>>= Бўшатиладиган разрядларни ноллар билан тўлдириб ўнгга силжиш

<<= Чапга силжиш

|= ЁКИ

&= ВА

^= МАНТИҚИЙ ИНКОР

С тили билан таниш бўлмаганлар учун ўзлаштириш операторини бошқа операторлар билан биргаликда ишлатилиши қийинроқ ва ғайриоддий туйилиш мумкин, лекин аслида сценарийни осонлаштирадибошланғич текстни соддалаштиради.

Масалан сонли ўзгарувчилар қийматини ошириш учун += оператори ишлатилади. Аввал бу вазифани ечимини += операторини ишлатмаган холатда кўриб чиқамиз. Қуйида nCounter ўзгарувчиси эълон қилинди ва унга бошланғич 1 қиймати ўзлаштирилди, сўнг бу қиймат 5 га оширилди:

```
var nCounter = 1;

nCounter = nCounter + 5;
```

Энди буни += оператори ёрдамида бажарамиз:

```
var nCounter = 1;

nCounter += 5;
```

Кўриниб турибдики 2-усул 1-усулга нисбатан қисқа.

Ўзгарувчи қийматини 3 разрядга ўнгга силжитиш учун >>= рператоридан фойдаланиш мумкин ва у қуйидаги матнда кўрсатилган:

```
nCounter >>= 3;
```

Натижа эса қуйидаги матнда кўрсатилганидек бўлади:

```
nCounter = nCounter >> 3;
```

4. JavaScript тилида функциялар

Бошланғич матн бўлагини функция кўринишида ёзиш мумкин ва уларни JavaScript сценарийсининг турли жойларидан мурожаат қилиш мумкин. Одатда функциялар HTML документини сарлавха бўлимида аниқланади. Функциялар чақирилишидан аввал эълон қилиниши керак ва барча функция эълони HTML документ сарлавхасида жойлаштирилган бўлиши керак.

Функциянинг умумий эълони қуйида келтирилган:

```
function имя([параметр 1] [,параметр 2] [...,параметр N])
{ . . .
  Функция матни қаторлари
  . . .
  [return қиймат]
}
```

Барча параметрлар функцияга қийматига берилади. Шунинг учун функция унга параметр сифатида бериладиган ўзгарувчилар қийматини ўзгартира олмайди.

Return калит сўзи ёрдамида функция қиймати қайтарилади.

Назорат саволлари:

1. Ўзгарувчилар нима ва уларни вазифаси нималардан иборат?
2. Ўзгарувчи типини тушунтириб беринг?
3. Қандай JavaScript тили операторларини биласиз?
4. JavaScript тили функсияси нима ва унинг вазифаси?
5. JavScript тили ёрдамида қандай маълумотлар билан ишлаш мумкин?
6. Ўзгарувчи номини танлаганда, қандай қоидаларга амал қилинади?
7. Бинар ва унар операторлари орасидаги фарқни тушунтиринг?
8. Унар операторларини ишлатишга доир мисоллар келтиринг?
9. Бинар операторларини ишлатишга доир мисоллар келтиринг?
10. Функциялар қайси бўлимда аниқланиши керак?

Фойдаланилган адабиётлар:

1. Спейнауер С., Куерсиа В. Справочник Web-мастера. - К: "БЩВ", 1997. - 368 с.
2. Яргер Р., Риз Дж., Кинг Т. МйС+Л и мС+Л. Базы данных для небольших предприятий и Интернета. - СПб: Символ-Плюс, 2000 - 560 с.
3. Хилайер С., Мизик Д. Программирование Астиве Сервер Пагес. - М: "Русская редакция", 1999. - 296 с.
4. Холзнер С. Перл: специальный справочник. - СПб: "Питер". 2000.
5. Швартс Р., Крициансен Т.Изучаем Перл. - К: "БҲВ", 2000. - 320 с.
6. Қосимов С.С. Ахборот теҳнологиялари. Тошкент “Алоқачи” 2006.
7. Хаитов Ф.Н., Юсупов Р.М., Ботиров Д.Б., Саттаров А.Р, Шукуров Э.Х.
8. Web теҳнологиялар. Жиззаҳ. 2005.

Фойдаланилган манбалар:

1. Intuit.ru.
2. http://pda.coolreferat.com
3. http://www.z-oleg.com
4. http://www.i2r.ru/
5.http://uz.wikipedia.org/wiki/HTML
6. http://www.cgi.ru

7.http://pacificwebart.com/

7-маъруза. Жараёнларни бошқариш элементлари.

Режа:

1. JavaScript тилида шарт операторлари;
2. Бошқарув ўтказувчи операторлар;
3. Цикллар;
4. JavaScript тилида Switch оператори.

Калит сўзлар: шартли ва бошқарув ўтказувчи операторлар, цикл операторлари, switch оператори.

Ишдан масад: JavaScript тилида дастурлаш жараёнида шартли ва бошқарув ўтказувчи операторлар, цикл операторлари, Switch оператори билан ишлаш кўникма ва билимига эга бўлиш.

1. JavaScript тилида шарт операторлари

JavaScript тилида бир неча шарт операторлари бор. Ҳозир улардан баъзилари билан танишиб чиқамиз.

JavaScript тилида if оператори

Бу оператор **JavaScript** дастурлаш тилидаги муҳим операторлардан биридир. У шартга боғлиқ равишда код фрагментини бажаришга мўлжалланган. *if* операторининг структурасини қуйидагича ифодалаш мумкин:

1- вариант. if оператори синтаксиси:

```
if(шарт)
{
Амаллар кетма кетлиги
}
```

2- вариант. if оператори синтаксиси:

```
if(шарт)
{

Амаллар кетма кетлиги 1

}

else

{

Амаллар кетма кетлиги 2

}
```

Бу ерда **шарт** JavaScript тилидаги мантикий еки ихтиёрий шартдир. Агарда шарт мантикий типдаги узгарувчи булса, қиймати рост (True) бўлса, у ҳолда *Амаллар кетма кетлиги1* бажарилади. Акс ҳолда *Амаллар кетма кетлиги2* бажарилади.

Мисол. if шарт оператори

```
function checkData()
```

```
{
if (document.form1.threeChar.value.length==3)
{return true;
}
else
{  alert('3 ракамини киритинг');
return false;
}
}
```

JavaScript тилида if оператори образ кабилардан фойдаланади. If буйруғи бўйича стандарт операторга мисол келтирамиз:

```
if (a == 1} window.alert("l топилди!");
```

if оператори бир неча ҳаракат операторларидан ҳам фойдаланади:

```
1: if (a == 1) {
2: window.alert("l топилди!");
3: a = 0;
4; }
```

Бу операторлар блоки а ўзгарувчи устида текширув ўтказади ва агар у 1 га тенг бўлса, у ҳолда ҳабар акс эттирилади ва а ўзгарувчи янги қиймат 0 га аниқланади.

JavaScript тилидаги шарт операторларида қуйидагилардан фойдаланилади:

- == (тенг)
- != (тенг эмас)
- < (кичик)
- > (катта)
- <= (кичик ёки тенг)
- => (катта ёки тенг)

JavaScript тилида else оператори

Биз юқорида фақат *if* операторининг асосий қисминигина кўрдик. Бу операторнинг бир нечта кенгайган шакли мавжуд. *else* оператори *if* операторида текширилаётган ифода нотўғри бўлган ҳолатдагина кенгайтиради ҳамда бу ҳолатда янги шартда бирор амал бажаради.

else оператори ёрдамида кенгайтирилган *if* операторининг структурасини қуйидагича ифодалаш мумкин:

if (шартли ифода) *бажариладиган_блок1*
else *бажариладиган_блок2*

Бу if...else конструкцияси қуйидагича интерпретация қилиниши мумкин: агар шарт бажарилса (яъни ифода=true), у ҳолда *бажариладиган_блок1* даги амаллар бажарилади, акс ҳолда *бажариладиган_блок2*даги амаллар бажарилади. *else* операторидан фойдаланиш мажбурий эмас.

Қуйида else операторига доир мисол кўриб ўтамиз:

```
1: if (a == 1) {

2: alert("l топилди");

3: a = 0;

4: }
```

```
5: else {
6: alert("Нотўғри белги: " = a);
1: }
```

JavaScript тилида elseif оператори

if шарт операторининг яна бир кенгайган шакли – бу *elseif* операторининг қўлланилишидир. *elseif* – бу *else* ҳамда *if* операторларининг комбинациясидир. У худди *else* оператори каби *if* операторида шарт бажарилмаган ҳолда кенгайтиради. Бироқ *else* операторидан фарқи бир-бирига зид амалларни фақат агарда ***elseif*** шарт рост бўлгандагина бажаради. *else* ҳамда *elseif* операторлари ёрдамида кенгайтирилган *if* операторининг структурасини қуйидагича ифодалаш мумкин:

if (ифода1) *бажариладиган_блок1*
elseif (ифода2) *бажариладиган_блок2*
else бажариладиган_блокN

elseif операторлари битта *if*-блокида бир неча марта учраши мумкин. *elseif* тасдиғи фақат олдинда турган *if*-шартлари ҳамда *elseif*-шартлари False қийматни, берилган *elseif*-шарти эса True қийматни қайтаргандагина бажарилади.

Мисол. elseif оператори

```
fanction makeMinutes() {
   var minstring="";
   var now = new Date();
   var min = Date.getMinutes();
   if (min<10) {
     minstring+=":0"+min;}
   elseif(min>10){
     minstring+=":"+min;}
   return minstring;
}
```

2. Бошқарув ўтказувчи операторлар

Баъзида цикл ёки унинг алоҳида итерация ишини тезда тўхтатишга тўғри келади. Бунинг учун ***break*** ҳамда ***continue*** операторлари керак бўлади.

JavaScript тилида break оператори

Break оператори мавжуд циклни амалга оширишни тугаллайди, *for, while, do while* ёки *switch break* структурани бошқарувчи, цикл ёки шартни текширишни тугаллаш кераклигини билдирувчи, унинг таркибига кирувчи рақамли аргумент билан қўлланилади.

Мисол. Break оператори

```
for (i = 0; i < a.length; i++) {
  if (a[i] = theValue)
    break;
}
```

Ушбу скриптда a массив элементи theValue узгарувчи кийматига тенг булганда цикл тўхтатилади.

JavaScript тилида continue бошқарув ўтказувчи оператори

Баъзан цикл ишини бутунлай тўхтатиш лозим бўлмайди, фақатгина унинг янги итерациясини бошлаш керак. *Continue* оператори исталган циклни амалга ошириш блокидан кейинги инструкцияларни ўтказиб юбориш ва янги доира билан амалга оширишни давом эттириш имконини беради. *continue* ни унинг таркибида бошқарилувчи конструкциялар ишини якунлаш кераклигини кўрсатувчи рақамли аргумент тарзида ишлатиш мумкин.

Олдинги параграфда берилган мисолдаги *break* операторини *continue* га алмаштирамиз. Бундан ташқари тўрт цикли миқдорини камайтирамиз.

Мисол:

```
i = 0;

n = 0;
```

```
while (i < 5) {
  i++;
  if (i == 3)
    continue;
  n += i;
}
```

Бу скриптда агарда i узгарувчи i=3 кийматга эга булса, у холда n=(1,3,7,12) кийматларга эга булади.

Мисол:

```
checkiandj :
  while (i<4) {
    document.write(i + "<BR>");
    i+=1;
    checkj :
      while (j>4) {
        document.write(j + "<BR>");
        j-=1;
        if ((j%2)==0)
          continue checkj;
        document.write(j + " is odd.<BR>");
      }
    document.write("i = " + i + "<br>");
```

```
    document.write("j = " + j + "<br>");
  }
```

Ушбу циклда агарда continue операторн топилса, checkj цикли тухтатилади ва checkj нинг кейинги циклига утилади. Хар сафар continue топилганда checkj итерацияни бошидан бошлайди, токи унинг шарти false булмагунича. checkj шарти false булганда, checkiandj нинг колган операторлари бажарилади ва бу цикл хам checkiandj шарти false булгунига кадар давом этади.

3. Цикллар

JavaScript тилида шартга боғлиқ равишда қайтариладиган амаллардан иборат бир нечта конструкциялар мавжуд. Бу *while*, *do..while*, *foreach* ҳамда *for* цикллардир. Уларни батафсил кўриб чиқамиз.

JavaScript тилидаWhile цикли

Структураси:

while (ифода) { *бажариладиган_блок* }
ёки
while (ифода): *бажариладиган_блок* endwhile;

while – бу оддий цикл. У ифода қиймати True (бу ерда худди *if* оператори каби ифода мантиқий типга ўзлаштирилади) бўлгунича *бажариладиган_блок*даги буйруқларни бажаришга буюради. Ифода қиймати ҳар цикл бошланганда текшириб борилади, агарда унинг қиймати *бажариладиган_блок* бажарилиш жараёнида ўзгарган тақдирда ҳам итерация тугамагунча (яъни *бажариладиган_блок*даги барча буйруқлар бажарилмагунча) цикл тўтатилмайди.

Мисол: while оператори

```
i = 0
n = 0
while (i<5) {
 i ++;
if (i == 3)
    continue n + = i
  }
```

Мисол: while оператори

```
n = 0;
x = 0;
while( n < 3 ) {
   n ++;
   x += n;
}
```

JavaScript тилида do... while цикли

do..while цикли *while* циклга ўхшайди, аммо фарқли томони шундаки, ифоданинг ростлигига цикл бошида эмас, балки охирида текширилади. Қулай томони шундаки, *бажариладиган_блок do..while* цикли ичида ҳеч бўлмаганда бир марта бажарилади.

Структураси:

do { *do..while* цикли } while (ифода);

Мисол. do..while оператори

```
do {
   i+=1;
   document.write(i);
} while (i<5);
```

JavaScript тилида for цикли

Бу JavaScript тили цикл операторларидан бири. Улар С дастурлаш тилидаги цикллар кабидир.

Структураси:

for (ифода1; ифода2; ифода3) { *бажариладиган_блок* }
ёки
for (ифода1; ифода2; ифода3): *бажариладиган_блок* endfor;

Бу ерда кўриниб турибдики шар учта ифодадан ташкил топади. Биринчи *ифода*1 ифода цикл бошида шартсиз бажарилади. Ҳар бир итерациянинг бошланишида *ифода*2 бажарилади. Агар у True қийматни қабул қилса, у ҳолда цикл ўз ишини давом эттиради ва *бажариладиган_блок*даги барча буйруқларни бажаради. Агар *ифода*2 False қийматни қабул қилса, у ҳолда цикл тўхтатилади. Ҳар бир итерация (яъни *бажариладиган_блок*даги барча буйруқларни бажарилишидан кейин) охирида *ифода*3 бажарилади.

Ҳар бир 1-,2- ва 3-ифодалар бўш бўлиши мумкин. Агар *ифода*2 бўш бўлса, бу циклни чексиз бажарилишини билдиради. Бу унчалик бефойда эмас, чунки циклни *break* оператори ёрдамида тўхтатса бўлади.

Мисол: Формада танланган элементлар сонини экранга чикариш

```
<SCRIPT>
function howMany(selectObject) {
  var numberSelected=0;
  for (var i=0; i < selectObject.options.length; i++) {
    if (selectObject.options[i].selected==true)
      numberSelected++;
  }
  return numberSelected;
}

</SCRIPT>
<FORM NAME="selectForm">
<P><B>Choose some music types, then click the button below:</B>
<BR><SELECT NAME="musicTypes" MULTIPLE>
<OPTION SELECTED> R&B
<OPTION> Jazz
<OPTION> Blues
<OPTION> New Age
<OPTION> Classical
<OPTION> Opera
```

```
</SELECT>
<P><INPUT TYPE="button" VALUE="How many are selected?"
onClick="alert('Number of options selected:'+howMany(document.selectForm.
musicTypes))">
</FORM>
```

Агарда ушбу оператор ичидаги барча учала ифода ҳам тушириб қолдирилса, у ҳолда счётчик **var i** ўзгарувчини бошланғич қиймати берилмайди ва ҳар бир цикл охирида у ўзгармайди. Бу барча буйруқларни алоҳида буйруқлар кўринишида ёки циклдан аввал *бажариладиган_блок* ичида ёзса ҳам бўлади:

```
var i=0; // счётчикни бошланғич қийматини берамиз
for ( ; ; ){
   if (i>=10) break;
    // агар i катта ёки тенг 10 бўлса, у ҳолда цикл ишини тўхтатамиз.
   if (i % 2 == 0) print $i;
    // агар сон жуфт бўлса, уни экранга чиқарамиз.
i++; // счётчик қийматини биттага оширамиз.
}
```

for цикли конструкциясидаги учинчи ифодада вергулдан кейин яна бир нечта оддий буйруқларни ҳам ёзса бўлади. Масалан, агар биз оддийгина барча сонларни экранга чиқармоқчи бўлсак, дастурни қуйидагича ёзса бўлади:

```
for (i=0; i<10; document.write(i), i++)
/* Агарда бажариладиган_блок буйруқлардан ташкил топмаган
ёки битта буйруқдан ташкил топган бўлса,
фигурали қавсга олинган қисмни
ташлаб кетса бўлади.*/
```

4. JavaScript тилида Switch оператори

Яна бир шартни текшириб турли амалларга боғлиқ равишда иш кўрсатадиган конструкция бу – *switch* операторидир. Бу операторни узбек тилига таржима қилинганда “йўналишни ўзгартиргич” маъносини беради ҳамда бу операторнинг вазифаси ҳам шунга ўхшашдир.

Switch оператори бир нечта асосий элементлардан ташкил топган:

- Switch бошланиш оператори.Бу оператор қавс ичида киритилган белгиларни таққослайди.
- Фигурали қавслар {}. Ifоператоридаги қавслар каби вазифани бажаради.
- Бир ёки бир неча case оператори.

- Break калит сўзи. Case операторидан чиқиш ҳаракатини аниқлаш учун ишлатилади.

Ўзгарувчини қандай қийматни қабул қилишига боғлиқ равишда у йўналишни ўзгартириб турли блоклардаги амалларни бажаради. *switch* оператори *if...elseif...else* ёки *if* оператори мажмуига жуда ўхшаш бўлади. *switch* операторининг структурасини қуйидагича ифодалаш мумкин:

```
switch (ифода ёки ўзгарувчи){
case қиймат1:
   амаллар_блоки1
break;
case қиймат2:
   амаллар_блоки2
break;
...
default:
   амаллар_блоки_автоматик_тарзда
}
```

if операторидан фарқли томони бу ерда ифодалар мантиқий тип қабул қилмай, балки фақат *case* калит сўзидан кейинги қийматларни (*қиймат1*, *қиймат2* ва ҳ.к.) таққослайди холос. Агар ифода қиймати қандайдир вариант билан устма-уст тушса, икки нуқтадан кейинги *break* операторигача бўлган *амаллар_блоки*даги амалларни бажаради. Агарда ифода қиймати берилган вариантлардан ҳеч бирига устма-уст тушмаса, *default* калит сўзидан кейинги автоматик тарзда бажариладиган блок (*амаллар_блоки_автоматик_тарзда*) бажарилади. *switch* операторидаги ифода фақат бир марта ҳисобланади, *elseif* операторида эса ҳар бир текширишда ҳисобланади, шунинг учун агарда ифода етарли даражада мураккаб бўлса, у ҳолда *switch* оператори тезроқ ишлайди.

Мисол 1: *switch* оператори:

```
function getName(){
var names = array("Name1"," Name2"," Name3");
var selected="";
switch (names[0]){
case "Name1":  selected="Name1 is selected";
break;
case "Name2":  selected="Name2 is selected";
break;
case "Name3":  selected="Name3 is selected";
```

```
break;
default: selected="Default select"+$names[0];
}
return selected;
}
```

Юкоридаги мисолдан куриниб турибдики, getName() функцияси ишга туширилганда, names массивидаги биринчи элемент текширилади. Бу холда массив 1-элементи Name1 булгани учун бизга selected="Name1 is selected"; кийматни кайтаради.

Мисол 2: *switch* оператори:

```
var change = prompt("Харакатни танланг:\n1 – Мошина сотиб олиш\n2 – Мошина сотиш \n3 – Мошина алмаштириш ");
switch (change) {
 case "1": {
  document.write("Сиз мошина сотиб олишни хохлайсиз ");
  break;
 }
 case "2": {
  document.write("Сиз мошина сотишни хохлайсиз ");
  break;
 }
 case "3": {
  document.write("Сиз мошина алмаштиришни хохлайсиз ");
  break;
 }
 default: {
  document.write("Сиз нотугри команда киритдингиз ");
  break;
 }
}
```

switch операторининг констркуцияси учун худди *if* оператори каби альтернатив синтаксиси мавжуд. Бу ерда *switch* операторидаги очиладиган фигурали қавс икки нуқтага ўзгартирилади, ёпиладигани эса мос равишда *endswitch*; калит сўзига ўзгартирилади.

Назорат саволлари:

1. JavaScript тилида шарт операторларининг ишлаши ва вазифасини тушунтириб беринг.

2.Бошқарув ўтказувчи операторлар нима ва уларнинг вазифаси нималарда намоён бўлади?

3.Қанақа цикл операторларини биласиз ва улар қандай ишлайди?

4. Switch оператори ишлаш принципини тушунтириб беринг?

5.Турли шарт операторларининг бир-биридан фарқли томонларини келтиринг?

6.Қай ҳолатда қайси шарт операторидан фойдаланиш қулай деб ўйлайсиз?

7.Ҳар бир операторни тегишли мисоллар ёрдамида тушунтиринг?

8. Switch операторидан фойдаланиш қандай афзалликларга эга?

9. JavaScript тилидаги операторлар ва цикларнинг бошқа дастурлаш тилларидаги операторлардан фарқли жиҳатлари борми?

10.Бошқарув ўтказувчи операторларнинг ишлаш принципини кўрсатинг?

Фойдаланилган адабиётлар:

13.Т. Стауфер. Создание web-странитс. Самоучител. – «Питер», Санкт-Петербург, 2003 г.

14.А. Гончарев. HTML. Самоучител. – «Питер», Санкт-Петербург, 2001 г.

15.Аллен Вайк. JavaScript. Энциклопедия ползователя: пер. с. анг. – «ТИД» «ДС», Киев, 2001 г.

16.webmastering –электрон ўқув қўлланма.

17.А.И. Тихонов. «Публикация данних в Интернет» Учебное пособие. Москва Издателцво «Мир» 2000.

18.JavaScriptни ўрганиш бўйича электрон қўлланмалар.

Фойдаланилган манбалар:

Интернет web-сайтлари:

1. http://www.cgi.ru
2. http://www.woweb.ru
3. http://www.ru
4. http://www.vanta.ru/script/
5. http://www.vbnet.ru
6. http://www.scriptic.ru/
7. http://www.webacademy.com/
8. http://pacificwebart.com/
9. http://www.google.ru

8-маъруза. JavaScript тилида функция ва усуллар тушунчаси. Объектлар ва катталиклар. JavaScriptда ҳужжатнинг объектли модели. Ҳусусиятлар ва

Режа:

1. JavaScript тилида функциялар;
2. Global класси;
3. Math класси;
4. Date класси;
5. JavaScript тилидаги объектлар.

***Калит сўзлар**:* JavaScript тилидаги функциялар, Global, Math, Date синфлари, объектлар, катталиклар, синфлар, усуллар, объектли модел.

Ишдан мақсад: Талабаларда JavaScript тилидаги функция ва усуллар, объектлар ва катталиклар тушунчаси ҳақида кўникма ва билим ҳосил қилиш, дастурлаш жараёнида улардан фойдаланишни ўргатиш, ҳужжатнинг объектли модели ва ҳусусиятлар ҳақида билимга эга бўлиш.

1. JavaScript тилида функциялар

JavaScript сценарийли тили объектга-мўлжалланган тилдир. JavaScript объектлари хусусиятлар ва усуллар тўпламини ифодалайди. Объект хусусияти – бу, объектга боғлиқ бўлган маълумотлардир, усуллар эса - объект маълумотларини қайта ишловчи функциялардир. JavaScript сценарийда хусусиятларни адреслаш уларнинг номлари билан ёки уларнинг номерлари билан амалга ошиши мумкин. Кейинги вариант бўйича, ҳар бир хусусият массивнинг бир элементи сифатида олинади ва улар ўзларининг уникал номерларига эга бўладилар.

JavaScript тилида C ва Java дастурлаш тилларидаги каби процедура ва функциялар мавжуд бўлиб, улар қуйидагича эълон килинади:

- function калит сўзи;
- функция номи;
- вергул ва кавс билан ажратилган аргументлар рўйхати;
- фигурали кавс ичига олинган функция танаси.
-

function myFunction(arg1, arg2, ...)
{
...
Операторлар кетма-кетлиги
...
}

Бу ерда myFunction – функция номи, arg1, arg2 – параметрлар.

Мисол:

```
function Factorial(n) {
if((n<0)||(round(n)!=n)) {
alert("Factorial функцияси ушбу аргументда аниқланмади "+n);
return NaN;
} else {
result=(n*Factorial(n-1));
return result;
} }
```

Функцияда return калит сўзи орқали қиймат қайтарилмаслиги ҳам мумкин.

Мисол:

```
function Greeting(s) {
document.write("Hello,"+s+"!");
return ;
}
```

Функцияни чақириш аниқ параметрлар билан чақирилади:

Мисол:

Factorial(3); - бу функция натижаси 6 га тенг,

Greeting("world"); - бу функция экранга "Hello, world!" стрини чиқаради.

Ҳар бир функция, масалан, **myFunction** функцияси myFunction номли объект хисобланади, агарда аргументлар arguments номи билан берилса, унга мурожаат қуйидагича:

myFunction.arguments[i], бу ерда **i** — аргумента номери (рақамлаш 0 дан бошланади).

Функция эълонида аниқ параметрлар формал параметрларга тенг еки кўп сонда бўлиши лозим. Бунда функция ишга туширилганда жунатилаетган аргументлар миқдори myFunction.arguments.length майдони ёрдамида аниқланади ва ушбу майдондаги қийматни қайта ўзлаштиришни динамик ўзгартириш мумкин.

Мисол:

Экранга HTML форматидаги рўйхатни чиқариш.

Бу ерда (ListType) нинг биринчи аргументи тартибланмаган рўйхат учун "o" еки "O", тартибланмаган рўйхат учун "u" еки "U" булиши мумкин.

```
function myList(ListType)

{

document.write("<"+ListType+"L");

for(var i=1; i < myList.arguments.length; i=i+1) {

document.write("<LI>"+myList.arguments[i]);

}

document.write("</"+ListType+"L>");

}
```

HTML ҳужжатида функцияга мурожаат қуйидагича:

```
<script>

myList("o", "матн", 2, "3")

</script>
```

Натижа:
матн

2

3

2. Global класси

Ушбу класс JavaScript нинг функционал қисми бўлиб, бу класс бир объектда бир нечта усул ва хоссаларни бирлаштириш вазифасини бажаради. Усулга мурожаат қилинганда объект кўрсатилмайди, аниқроғи бу усул конструкторга эга бўлмайди. Бундай хосса ва усулларга қуйидагиларни келтириш мумкин:

Хосса	**Мазмуни**
Nan	NaN (Not A Number)
Infinity	Number.POSITIVE_INFINITY қийматни ўз ичига олади

Усул	**Мазмуни**
escape	Қаторни барча платформаларга мос ҳолда тасвирлаш
eval	JavaScript тили функцияси еки усулларини узатиш
isFinite	Аргументнинг охирги рақамлилигини аниқлаш
isNan	Аргументнинг рақам ёки рақам эмаслигини аниқлаш
parseFloat	Қаторни кўчиб юрувчи нуқтали сон кўринишида тасвирлаш
parseInt	Қаторни бутун сонга айлантириш

unescape	Escape функцияси натижасини қайтариш

eval(s) функцияси - s қаторни JavaScript операторлари кетма-кетлиги кўринишида тасвирлаш.

getClass(Jobj) функцияси – JavaObject типидаги аргумент учун JavaClass объектини қайтаради.

Мисол:

var myJavaRClass=new java.awtRectangle()

var myJavaRClass=getClass(myJavaRect)

getClass() Java-методи билан адаштирманг:

var myJavaRCObject=myJavaRect.getClass() - бу java.awt.Rectangle класининг Java тилидаги реализация ҳолати.

isNaN(x) функцияси – x “Not a Number”, яъни сон эмаслигини текшириш.

parseFloat(s) функцияси – Float типидаги s рақамни аниқлаш. Агар сон топилмаса у ҳолда NaN (“Not a Number”) қиймати қайтарилади.

parseInt(s) – Integer типи учун юқоридаги ҳолат.

eval(s) функцияси

eval(s) функцияси – JavaScript нинг ички функцияси ҳисобланади. Ушбу функция бир ёки бир нечта JavaScript операторларидан иборат бўлган s сатрни аргумент томонидан узатилган кодни бажаради. Бунда s сатридаги операторлар нуқтали вергул ёрдамида ажратилади. Бу функция нафақат операторни бажариш,

балки бирор амалларни ҳисоблаш имконини ҳам беради. Бунда у кодда келтирилган амал хисобининг охирги қийматини қайтаради.

isNaN(x) функцияси

Бу функция x аргументнинг “сон эмас” лигини текширади. Натижа NaN қийматга эга эмаслигини, яъни мумкин булмаган сон (масалан, нолни нолга бўлиш натижаси) ни текширади. Ушбу функция JavaScript да литерал кўринишда NaN қийматни бериш мумкин эмаслиги учун муҳимдир. Бундан ташқари parseFloat(s) ва parseInt(s) функциялар натижаларини текшириш (мумкин бўлган сон эканлигини) ва арифметик хатолар мавжудлиги, масалан, нол сонига бўлиш мавжудлигини текширади.

parseFloat(s) функцияси

s сатрини синтактик анализ қилиш ва дастлаб ракамни қайтариш (сатрни рақамна айлантиради). parseFloat(s) да s сатрида рухсат этилмаган рақам элементлари (масалан, белгилар, рақам, ўнли вергуллар, даража кўрсаткичи ва ҳоказо) мавжуд булса анализ тўхтатилади ва қиймат қайтарилади. Агарда s сатрда сон билан бошланмаса, у ҳолда parseFloat(s) функция NaN қийматни қайтаради.

parseInt(s) функцияси

Бу функция сатрни бутун сонга айлантиради. parseInt(s) функциядаги s сатрда ҳисоблаш тизимида кўрсатилмаган қийматларга эга бўлганда синтактик анализ тўхтатилади ва қиймат қайтарилади. Одатда, parseFloat ва parseInt функциялар s сатр сон билан бошланмаганда NaN қиймат қайтаради.

parseInt(s,n) ҳолатида n асос иҳсобланиб, агарда n=10 бўлса, parseInt(s) функция сатрдаги 10 лик санок системасидаги сонларни текширади. n=8 бўлса, 8

лик соноқ тизимидаги сонлар мавжудлигини (бунда n 0 дан 7 гача бўлган сонлар қийматига эга бўлиши мумкин). n=16 бўлса, 16 лик соноқ тизимидаги сонлар мавжудлигини (бунда 0 дан 9 гача бўлган сонлар ва A дан F гача бўлган ҳарфлар қийматига эга бўлинади). Агарда n=0 бўлса ёки қиймат берилмаса, у ҳолда parseInt(s) функция сатрнинг ўзидан асосни аниқлайди. Бу ҳолатда агарда сатр 0x билан бошланса, унда функция сатрнинг қолган қисмини 16 лик саноқ тизимидаги сон сифатида анализ қилади, агарда сатр 0 дан бошланса, сатр 8 лик саноқ тизимидаги қиймат сифатида анализ қилинади.

3. Math класси

Math – константалар ва методлардан иборат классдир. Улар объект учун одатдагидек мурожаат қилинади:

Math.константа

Math.функция(i..)

Math класси константалари

E – e сони (натурал лагорифм асосли)

LN10 — 10 ли натурал лагорифм (ln10 сони)

LN2 — 2 ли натурал лагорифм (ln2 сони)

LOG10E — 10 асосли e лагорифм (log10e сони)

LOG2E — 2 асосли e лагорифм (log2e сони)

PI — p константаси ("пи" сони)

SQRT1_2 — 2 нинг тескари квадрат илдизи ($1/\sqrt{2}$)

SQRT2 — 2 нинг квадрат илдизи ($\sqrt{2}$)

Math класси методлари

abs(x) (x-сон еки ифода) – абсолют қийматни ҳисоблаш;

acos(x) (x бу ерда [-1.0;1.0] радиан интервалдаги сон еки ифода) – арккосинусни ҳисоблаш. Қайтариладиган қиймат 0 дан p радиан оралиғида бўлади.

asin(x) (x бу ерда [-1.0;1.0] радиан интервалдаги сон еки ифода) – арксинусни ҳисоблаш. Қайтариладиган қиймат –p/2 дан p/2 радиан оралиғида бўлади.

atan (x) (x – сон еки ифода) – арктангенсни радианларда ҳисоблаш. Қайтариладиган қиймат –p/2 дан p/2 радиан оралиғида бўлади.

atan2(x,y)(x,y — тўғри бурчакли кординаата системаси кординаата нуқталари) – қутб кординатасида (x,y) нуқталар бурчагини ҳисоблайди. Қиймати 0 дан 2p радиан оралиғида бўлади.

ceil(x) (x — сон ёки сонли ифода) – сонни бутун сонга йўналтирилган ҳолда яхлитлаш. Манфий сонлар 0 сони йўналишига караб яхлитланади.

cos(x) (x – радиандаги бурчак) – косинусни ҳисоблаш, қайтариладиган қиймат -1.0 дан 1.0 радиан оралиғида бўлади.

sin(x) (x – радиандаги бурчак) – косинусни ҳисоблаш, қайтариладиган қиймат -1.0 дан 1.0 радиан оралиғида бўлади.

Exp(x) (x — сон ёки сонли ифода) — e экспонентсини ҳисоблаш.

Floor(x) (x — сон ёки сонли ифода) – сонни бутун қисмига йўналтириб яхлитлаш, масалан, floor(-1,1) тенг (-2); floor(1,1) тенг 1.

Log(x) (x — мусбат сон ёки ифода) – натурал лагорифмни ҳисоблаш.

max(a,b) (a,b — сон ёки ифода) – икки қийматдан каттасини қайтаради.

min(a,b) (a,b — сон ёки ифода) – икки қийматдан кичигини қайтаради.

pow(x,y) — x ни ҳисоблаш (биринчи аргументни даражага кўтариш).

random — 0 дан 1 гача интервалдаги тасодифий сонларни ҳисоблаш.

round — сонни бутун қисмига қараб яхлитлаш (масалан, round(15.5) натижаси 16 ни беради, round(-15.5) дает -15).

Math.round(x) (x — сон ёки ифода)

Math.sin(x) (x — радианда берилган бурчак)

Math.sqrt(x) (x — 0 га тенг ёки катта бўлган сон ёки ифода)

tan —тангенсни ҳисоблаш.

Math.tan(x) (x — радианда берилган бурчак)

4. Date класси

Date() методи аргументсиз берилганда қиймати жорий сана ва вақтга эга Date объекти яратилади. Date() методида янги объект учун аргументи сифатида сана ва зарур ҳолларда вақт кўрсатилади. Date методи JavaScript тили объекти ҳисобланиб, HTML тилида ҳеч қандай аналогга эга эмас. Кўп ҳолларда Date объекти методлари унинг экземпляти ёрдамида чақирилади, масалан:

d=new Date(); // бугунги сана ва вақтни олиш

system.write("Today is: "+d.toLocalString()); // ва уни тасвирлаш

Date объектини яратишнинг юқоридаги синтактикасида кўрсатилгани бўйича, сана ва вақт ҳудудий вакт бўйича берилади. Агарда тузилаётган дастур фойдаланувчи жойлашган часовой поясга боғлиқ бўлмаган ҳолда ишлаши зарур бўлса, у ҳолда Гринвич (GMT) еки универсал координация вақти (UTC) бўйича санани кўрсатиш керак бўлади.

Date объектини яратишда қуйидаги 5 та синтактик вариантдан фойдаланиш мумкин. 3-5 вариантларда вақт ҳудудий тарзда интерпретация қилинади (Гринвич да эмас):

1. new Date();
2. new Date(миллисекунд) – бу ерда миллисекунд жорий сана билан 01.01.1970 сана яримкуни орасидаги сон;
3. new Date(сана сатри) – бунда сана сатри = ой номи, дд, гг [чч:мм[:сс]])
4. new Date(йил, ой, кун) – бунда, йил 2011; ой 0-11; кун 1-31;
5. new Date(йил, ой, кун , соат, минут, секунд) – 24 соатликтизимда.

Date класси методлари

getDate() - Date объектининг 1 дан 31 гача оралиқдаги қийматини беради;

getDay() - Date объектининг 0 [якшанба] дан 6 [шанба] гача оралиқдаги хафта кунлари беради;

getHours() - Date объектининг 0 [ярим тун] дан 23 гача оралиқдаги соат қийматини беради;

getMinutes() - Date объектининг 0 дан 59 гача оралиқдаги минут қийматини беради;

getSeconds() - Date объектининг 0 дан 59 гача оралиқдаги секунд қийматини беради;

getMonth() - Date объектининг 0 [январ] дан 11 [декабр] гача оралиқдаги ойларни беради;

getTime() - Date объекти вақт кўрсаткичининг миллисекундлардаги қийматини беради;

getYear() - Date объекти вақт кўрсаткичининг йиллар майдони қийматини беради; бунда 2011 йил 11 кўринишида берилади;

parse() – сананинг сатр кўринишидаги ҳолатини синтактик анализ қилади ва натижани миллисекунд форматида беради;

setDate() - Date объекти вақт кўрсаткичини ўрнатади;

дата.setDate(ой сони) //ой сони 1-31 оралиқда.

toLocalString() – жорий ҳудудий вақт майдонини асосида Date форматини матнли (String) кўринишга келтиради;

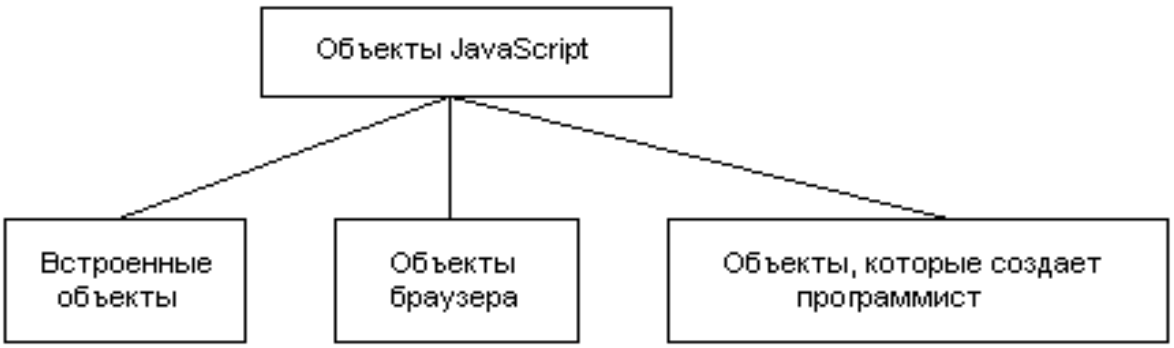
UTC() – сана ва вақтнинг рақамли кўринишини миллисекундли форматга айлантиради.

5. JavaScript тилидаги объектлар

JavaScript тилида уч турдаги объектлар мавжуд:

1. стандарт объектлар;
2. браузер объектлари;
3. дастурчи томонидан яратилувчи объектлар.

Уларнинг ҳар бири ўзларининг таснифи ва хусусиятларига эга.



Стандарт объектлар

Қуйида JavaScript да қўлланилувчи стандарт объектлар, хусусиятлар ва усуллар келтирилган. Уларни ишлатишда олдиндан эълон қилиш талаб этилмайди.

Объект	Таснифи
Array	Массив
Boolean	Мантиқий маълумотлар
Date	Календарли вақт
Function	Функция
Global	Глобал усуллар
Math	Математик константа ва функциялар
Number	Сон
Object	Объект
String	Сатр

Стандарт объектлар билан қандай ишлаш мумкин? Анча оддий. Объектни реализация қилувчи дастур ёзилади ва унинг хусусият ва усулларига мурожаат қилинади. Мисол сифатида жорий вақтни кўрсатувчи HTML хужжатни кўрамиз.

```
<HTML>  <HEAD>   <TITLE>Жорий кун ва вақт </TITLE>  </HEAD>
 <BODY BGCOLOR=WHITE>
   <H1> Жорий кун ва вақт </H1>
   <SCRIPT LANGUAGE="JavaScript">
     var dt;
     var MyDate="";
```

```
    dt = new Date();
    MyDate = "Date: " + dt.getDate() + "."  + dt.getMonth() + "." + dt.getYear();
    document.write(MyDate);
    document.write("<BR>");
    document.write("Time: " + dt.getHours()
     + ":" + dt.getMinutes() + ":" + dt.getSeconds());
  </SCRIPT>  </BODY></HTML>
```

Бу ерда JavaScript сценарий new калит сўзи ёрдамида Data объектини яратади. Бунда Date конструктори параметрларсиз келтирилади:

```
var dt;
dt = new Date();
MyDate = "Date: " + dt.getDate() + "."
 + dt.getMonth() + "." + dt.getYear();
```

getDate, getMonth ва getYear усуллар ёрдамида жорий сана олинади. Ушбу усуллар dt объекти учун чақирилади.

Матн сатри эса HTML хужжатга write усули ёрдамида босмага чиқарилади. Бу усул document объектининг усули ҳисобланади:

```
document.write(MyDate);
```

Date объекти жорий вақтни ҳам ўз ичига олади. Бу маълумотлар getHours, getMinutes ва getSeconds (соат, минут ва секунд) усуллари ёрдамида кўрилади:

```
document.write("Time: " + dt.getHours()
 + ":" + dt.getMinutes() + ":" + dt.getSeconds());
```

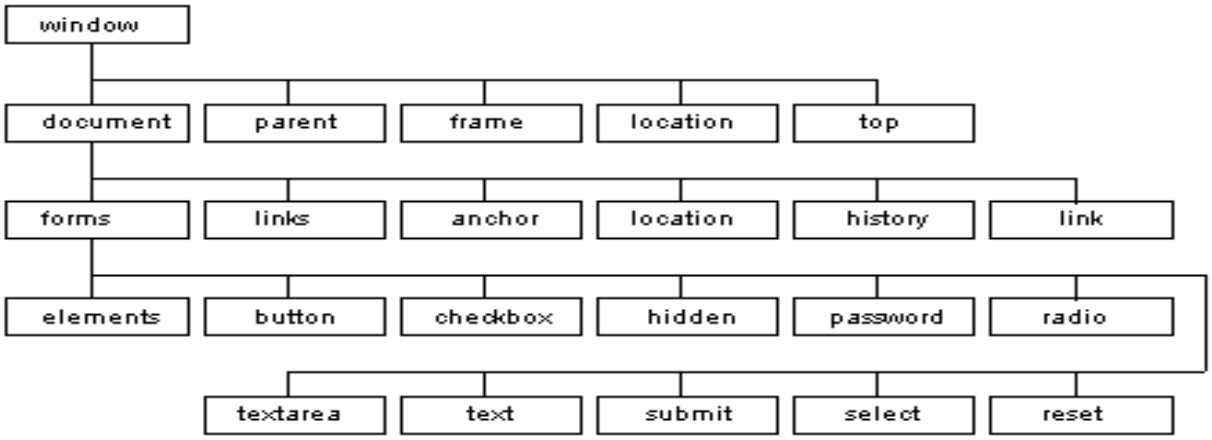
Браузер объектлари

JavaScript сценарий нуқтаи назари бўйича объектлар иерархик дарахт кўринишда ташкил этилади.

Браузер объектлари фойдаланувчи учун яратилган, браузер ойнасида жойлашган объектлар ҳисобланади. JavaScript сценарида браузер объектлари, хусусият ва усулларидан фойдаланиб бир класс асосида бошқа класс яратиб бўлмайди.

Браузер объектлари иерархияси

Қуйидаги расмда объектлар дарахти иерархияси келтирилган.



window объекти бу иерархиянинг илдизи ҳисобланади. Қачонки HTML хужжат юкланса унмнг ичида **document**, **parent**, **frame**, **location** ва **top** бошқа объектлар ҳосил бўлади.

Объектлар билан боғлиқ ҳолатлар

Браузернинг ҳар бир объекти билан аниқ бир ҳолатлар тўпламидан ташкил топади.

Масалан, **window** объекти **onLoad** ва **onUnload** ҳолатлари билан боғлиқ ҳолда ишлайди. Биринчи ҳолат браузер ойнани юклаб бўлгач ишга тушади. Иккинчиси эса браузер ойнани ёпиш вақтида ишга тушади.

Дастурчи томонидан яратилувчи объектлар

Аввало myRecord номли класс яратамиз. Ҳозирча унда усуллар мавжуд эмас, уларни кейинчалик қўшамиз. Бу класс қуйидагича яратилади:

```
function myRecord(name, family, phone, address)  {
  this.name    = name;
  this.family  = family;
  this.phone   = phone;
  this.address = address;
  this.secure  = false;
}
```

Яратилаётган объектни хусусиятларини кўрсатиш учун махсус this калит сўзидан фойдаланилади. Бу калит сўз объектнинг хусусиятларига бўлган мурожаатини кўрсатади.

Келтирилган классдан қандай фойдаланиш мумкин? Яратилган класс асосида исталган сондаги объектлар яратиш мумкин. Қуйида берилган myRecord классида иккита rec1 ва rec2 объектлари яратилган:

```
var rec1;
```

```
var rec2;
rec1 = new myRecord("Иван", "Иванов",
  "000-322-223", "А. Темур кўча, д. 225, кв. 226");
rec2 = new myRecord("Петр", "Петров",
  "001-223-3334", "Бобур кўча, д. 552, кв. 662");
rec2.secure = true;
```

Объектлар new оператори ёрдамида яратилади.

Назорат саволлари:

1. JavaScript тилининг қанақа функцияларини биласиз?
2. Global класси ва унинг операторларини санаб ўтинг?
3. Math класси ва унинг операторларини айтинг?
4. Date класси ва унинг операторларини айтинг?
5. Қандай турдаги объектлар мавжуд?
6. Браузер объектлари иерарҳиясини чизма ёрдамида тушунтириб беринг?
7. Турли класслардан фойдаланишнинг аҳамиятли томонларини айтиб ўтинг?

8.Турли класслар орасидаги фарқли жиҳатларни кўрсатинг?

Фойдаланилган адабиётлар:

7. Университет портали: Intuit.ru.
8. С.С.Қосимов. Ахборот теҳнологиялари. Тошкент “Алоқачи” 2006.
9. Спейнауэр С., Куэрсиа В. Справочник Web-мастера. - К: "БҲВ", 1997. - 368 с.
10.Яргер Р., Риз Дж., Кинг Т. MySQL и mSQL. Базы данных для небольших предприятий и Интернета. - СПб: Символ-Плюс, 2000 - 560 с.
11. Хилайер С., Мизик Д. Программирование Active Server Pages. - М: "Русская редакция", 1999. - 296 с.
12.Хаитов Ф.Н., Юсупов Р.М., Ботиров Д.Б., Саттаров А.Р, Шукуров Э.Х. Web теҳнологиялар. Жиззаҳ. 2005.

Фойдаланилган манбалар:

8. Universitet portali: http://www.Intuit.ru/
9. http://pda.coolreferat.com/
10.http://www.z-oleg.com/
11.http://www.i2r.ru/
12.http://www.google.ru/

9-маъруза. PHP дастурлаш тили. PHPга кириш. Сервер томонидан дастурлаш. PHPни ўрнатиш ва тестлаш.

Режа:

1. PHP дастурлаш тилига кириш.
2. PHP дастурлаш тили асослари;
3. PHP дастурлаш тили имкониятлари.
4. Дастурий воситани созлаш ва ўрнатиш.

Калит сўзлар: PHP дастурлаш тили, изоҳлар, ўзгарувчилар, константалар, ташқи ўзгарувчилар, сервер томонда дастурлаш, денвер пакети, Денвер Дистрибутиви.

Ишдан мақсад: PHP дастурлаш тили билан танишиш ва ишлаш жараёнини ўрганиш. PHPни ўрганиш ва маълумотлар типларидан фойдаланиш. Ўзгарувчилар ва амаллар билан ишлай олишни ўргатиш.

1. PHP дастурлаш тилига кириш

Ҳозирги кунда интернет кенг оммалашгани сабабли, замон тараққиётини веб-технологиясиз тассаввур этиш мумкин эмас. Веб технологияларига талаб ошган

сари Web-дастурлаш тилларини билиш ҳар бир дастурчи учун муҳим вазифа саналмоқда. Шуларни инобатга олган ҳолда замонавий веб-дастурлаш тилларидан бири ҳисобланган, содда, ўрганишга қулай, барча маълумотлар базаси билан ишлай оладиган PHP ҳақида батавсилроқ тўхталишга ният қилдик. Келгусида бу тил ўзбек тилида ёритилиб борилади ҳамда мутахассис ва ўрганувчилар учун форум ташкил қилинади.

PHP тарихи.

Кўпгина бошқа дастурлаш тилларидан фарқли равишда, PHP қандайдир ташкилот ёки кучли дастурчи томонидан яратилган эмас. Уни оддий фойдаланувчи Расмус Лердорф 1994 йили ўзининг бош саҳифасини интерактив услубда кўрсатиш учун яратган. Унга Personal Home Page (PHP – шахсий бош саҳифа) деб ном берган.

1995 йили Расмус PHPни ўзининг HTML формалари билан ишлайдиган бошқа дастур билан умумлаштириб PHP/FI Version 2 (”Form Interpretator“) ҳосил қилди. 1997 йилга бориб PHP дан фойдаланувчи сайтлар 50 мингдан ошди. Шундан сўнг веб технология усталари PHP ғояси асосида мукаммал тил яратишга Зива Сураски ва Энди Гутманс асосчилигида киришилди. PHPни самарали деб ҳисобланмагани учун деярли нолдан бошлаб, мавжуд C ва Перл тилларидан ибрат олиб PHP3 талқинини яратилди. 1999 йилга келиб PHP асосида қурилган сайтлар миллиондан ошиб кетди. 2000 йилда эса Zend Technologies ширкати янги кўпгина функцияларни қўшган ҳолда PHP4 шарҳловчисини яратди.

PHP – веб техноогия тили. PHPни ўрганиш учун аввал HTML ва дастурлаш тилидан ҳабардор бўлиш талаб қилинади. HTML/CSS ва JavaScript ларни мукаммал билгувчилар учун PHPни ўрганиш мураккаблик туғдирмайди. PHPнинг вазифаси HTML файлини яратиб бериш. JavaScript ёрдамида бажариладиган кўпгина операцияларни PHP орқали ҳам амалга ошириш мумкин, аммо эътибор қилиш лозимки, PHP – серверда; JavaScript – клиент томонда бажарилади. PHPда ёзилган код сервернинг ўзида бажарилиб, клиентга HTML шаклида етиб боради. Бу ҳавфсизлик жаҳатдан анча мақсадга мувофиқ. JavaScript ёрдамида код ёзиш,

маълумот узатиш ва қабул қилишни бироз тезлаштирса-да, кодни клиент кўриш имкониятига эга бўлади. Барибир ҳар иккисини бошқаси боса олмайдиган ўз ўрни бор, равшанки бу ўрин PHPда муҳимроқ ва каттароқ.

2. PHP тили асослари

PHP дастурлари

PHP дастурлари икки усулда бажарилиши мумкин: Web-сервер томонидан ссенарий иловаси ва консол дастури сифатида.

Бизнинг мақсадимиз web иловаларни дастурлаш бўлгани учун асосан биринчи усулни кўрамиз.

PHP одатда Интернет билан боғлиқ дастурлар яратиш учун ишлатилади. Лекин PHP дан команда сатрлар интерпретатори, асосан *nix тизимларда фойдаланиш мумкин. Охиргиси CORBA ва COM интерфейслар ҳамда PHP-GTK кенгайтмаси ёрдамида мумкин. Бу холда қуйидаги масалаларни эчиш мумкин:

- Интерактив команда қаторлари ёрдамида иловалар яратиш;
- Кросс-платформали GUI иловаларни PHP-GTK библиотекаси ёрдамида яратиш;
- Windows ва Linux учун баъзи масалаларни автоматизатсия қилиш.

Серверга браузернинг мурожжат қилишида ёрдамида PHP-ссенарийлари бажарилишини кўриб чиқамиз. Аввал бороузер .PHP кенгайтмали саҳифани сўрайди, сўнгра web-сервер дастурни PHP машинадан ўтказади ва натижани HTML-код шаклида қайтаради. Агар стандарт HTML саҳифани олиб, кенгайтмасини .PHP га ўзгартирилса ва PHP машинадан ўтказилса, фойдаланувчига ўзгартирмасдан қайтаради. Буф файлга PHP командани қўшиш учун, PHP командалани махсус теглар ичига олиш керак. Бу тегларнинг 4 хил шакли мавжуд бўлиб, ихтиёрийсидан фойдаланиш мумкин:

1. **XML қайта ишлаш инцруктсияси:**
2. <**?php**
3. ...

?>

4. **SGML қайта ишлаш инцруктсияси:**
5. <**?**

```
6.      ...
?>
7.          HTML ссенарийлари қайта ишлаш инструктсияси:
8.  <script language = "php">
9.      ...
10.             </script>
11.         ASP услубидаги инструктсия:
12.             <%
13.                 ...
14.             %>
```

Биз XML ёки SGML услубига риоя қиламиз.

Хусусан бирор блок ичида PHPдан чиқиш мумкин, фақат кейинчалик яна унинг ичига кириб кодни тугатиш шарти билан, қуйидаги конструктсия мумкин:

```
<?
  if(5<3){
  echo("<p>Hello, world!<p>");
?>
  <p>Hello!</p>
  // бу қатор PHP коди сифатида қаралмайди
  // ва код блоки бажарилаётган бўлса чиқарилади
<?
    echo("<p>Hello, world!<p>");
  }
?>
```

PHP да echo командаси web – саҳифаларда учрайдиган ҳар қандай маълумотни(матн, HTML ажратувчи символи, сон) чиқариш учун қўлланади. Унинг маъноси мисолда кўрсатилган.

Изохлар

PHP тилида изохларни жойлаш учун бир неча усуллар мавжуд. Энг соддаси иккилик слеш(//)дан фойдаланиш, шундан сўнг PHP сатрлар охиригача ёзилганни ўтказиб юборади. Бундан ташқари S (/*…*/) услубидаги кўп қаторли изохлардан фойдаланиш мумкин. Бир қаторли изохлар учун (#) символдан фойдаланиш қулай.(UNIX script тилларидаги изох).

```
<php
  echo("<p>Hello</p>"); // изох
  echo("<p>Hello</p>"); # изох
  /*
    бу ҳам изох
  */
?>
```

Шуни эсдан чиқармаслик лозимки PHP услуби изохлари фақат PHP чегаранишлари орасида таъсир қилади. Агар PHP бу изохлар символларини чегаранишлари ташқарисида учратса, уларни бошқа матнга ўхшаб, HTML-саҳифага жойлаштиради.

Масалан:

```
<php
  echo("<p>Hello</p>"); // нормал изох
?>
//бу изох браузерда кўринади.
<!—HTML изохи.
```

Бу изох HTML кодда кўринади, браузерда эмас -->

Изохларни фақат оператор охирига эмас, қуйидагича жойлаш ҳам мумкин:

```
<?
  $a = "Hello, world";
  echo strstr($a,"H");
  // бу функцияни кейинчалик қараб чиқамиз
?>
```

Ўзгарувчилар ва константалар

PHP да ўзгарувчилар доллар **($)** белгисидан бошланади. Бу символдан ихтиёрий сондаги ҳарф, рақам ва остига чизиқ символлари келиши мумкин, лекин биринчи символ албатта ҳарф бўлиши керак. Шуни эсда тутиш керакки, PHPда ўзгарувчиларнинг номлари калит сўзлардан фарқли регистрга боғлиқдир.

PHP да ўзгарувчиларни таърифлаганда ошкора типини кўрсатиш шарт эмас ва дастур давомида итта ўзгарувчи ҳар хил типларга эга бўлиши мумкин.

Ўзгарувчи унга қиймат берилганда инициализатсия қилинади ва дастур бажарилгунча мавжуд бўлади. Яъни web-саҳифа холида то сўров тугамагунча.

Ташқи ўзгарувчилар

Клиент сўрови веб-сервер томонидан тахлил қилиниб, PHP машинага узатилгандан сўнг, у сўровга тегишли маълумотларни ўз ичига олган ва бажариш давомида мурожаат қилиш мумкин бўлган бир неча ўзгарувчиларни яратади. Олдин PHP сизни тизимингиз атроф мухит ўзгарувчиларини олади ва шу номдаги ва шу қийматдаги PHP ссенарийси атрофидаги ўзгарувчиларни яратади, токи серведаги ссенарийларга клиент тизими хусусиятлари билан ишлаш мумкин бўлсин. Бу ўзгарувчилар **$HTTP_ENV_VARS** ассотсиатив массивга жойлаштирилади.

Табиийки **$HTTP_ENV_VARS** массиви ўзгарувчилари тизимга боғлиқдир (чунки улар аслида атроф мухит ўзгарувчиларидир). Атроф мухитўзгарувчилари қийматларини сизни машинангиз учун env (Unix) ёки set (Windows) командаси ёрдамида кўришингиз мумкин.

Сўнгра PHP у GET-ўзгарувчиларнинг гурухини яратади. Улар сўров сатрини тахлил қилишда яратилади. Сўров сатри **$QUERY_STRING** ўзгарувчида сақланади ва сўралган URL даги "?" символдан кейинги информациядан иборат. PHP сўров сатрини **&** символлари бўйича алохида элементларга ажратади, ва ҳар бир элементда "=" белгисини қидиради. Агар "=" белгиси топилган бўлса, тенглик чап томонидаги символлардан иборат ўзгарувчи яратади. Қуйидаги формани кўрамиз:

```
<form action = "http://localhost/PHP/test.php" method="get">
  HDD: <input type="text" name="HDD"/><br>
  CDROM: <input type="text" name="CDROM"/><br>
<input type="submit"/>
```

Агар сиз бу формада HDD қаторда "Махтор", CDROM қаторда "Nec" терсангиз, қуйидаги сўров шаклини хосил қилади:

http://localhost/PHP/test.php?HDD=Махтор &CDROM=Nec

Бизнинг мисолимизда PHP қуйидаги ўзгарувчиларни яратади: **$HDD** = "Махтор" va **$CDROM** = "Nec".

Сиз ўзингизни скриптингиздаги (бизда – test.php)бу ўзгарувчилар билан оддий ўзгарувчилар билан ишлагандек ишлашингиз мумкин. Бизнинг мисолимизда улар экранга чиқарилади:

```
<?
  echo("<p>HDD is $HDD</p>");
  echo("<p>CDROM is $CDROM</p>");
?>
```

Агар саҳифа сўрови POST усули ёрдамида бажарилса, POST-ўзгарувчиларнинг гурухи яратилиб, интерпретатсия қилинади ва **$HTTP_POST_VARS** массивга жойлаштирилади.

Константалар

Константалар PHP да **define**() функцияси ёрдамида эълон қилинади:

define(CONSTANT, value)

Бу функция биринчи параметри – константа номи, иккинчиси – унинг қиймати. Константадан фойдаланилганда номи бўйича илова қилинади:

```
<?
  define(CONSTANT1,15);
  define(CONSTANT2,"\x20"); // пробел коди
  define(CONSTANT3,"Hello");
  echo(CONSTANT1);
  echo(CONSTANT2);
  echo(CONSTANT3);
?>
```

Одатга кўра константалар номлари юқори регистр ҳарфлари билан ёзилади. Бу фақат одат бўлса ҳам унга риоя қилишни маслахат берамиз, чунки яхши одатларга риоя қилмайдиган дастурчилардан ёмон дастурчилар чиқади. Константалар аниқланганлигини **defined**() функцияси ёрдамида текшириш мумкин:

```
<?
  define(CONSTANT,"Hello");
  if(defined("CONSTANT"))
  {
    echo("<p>CONSTANT is defined</p>");
  }
?>
```

3. PHP дастурлаш тили имкониятлари

«PHP да ҳар қандай дастур бажарса бўлади», – деган эди унинг яратувчиси. Биринчи навбатда PHP тили сервер томонидан бажариладиган скриптлар яратиш учун фойдаланилади ва айнан шунинг учун у яратилган. PHP тили ихтиёрий CGI-скриптлари масалаларини ечишга ва бундан ташқари html формали маълумотларни қайта ишлашга ҳамда динамик равишда html саҳифаларни ишлаб чиқишга қодир. Бироқ PHP тили фойдаланиладиган бошқа соҳалар ҳам мавжуд. Бу соҳаларни биз учта асосий қисмга бўламиз:

Биринчи соҳа – биз юқорида айтиб ўтганимиздек, сервер томонидан бажариладиган иловалар (скриптлар) яратиш. PHP тили бундай турдаги скриптларни яратиш учун жуда кенг қўлланилади. Бундай иш кўрсатиш учун PHP-парсер (яъни php-скриптларни қайта ишловчи) ва скриптларни қайта ишловчи web-сервер, скриптларни натижасини кўриш учун браузер ва албатта php-кодини ёзиш учун қандай бўлса ҳам матн муҳаррири керак бўлади. PHP-парсер CGI-дастурлар кўринишида ёки сервер модуллари кўринишида тарқалган. Уни ва web-серверни компьютеримизга қандай ўрнатамиз, биз бу ҳақида кейинроқ кўриб ўтамиз.

Иккинчи соҳа – буйруқлар сатрида бажариладиган скриптларни яратиш. Яъни PHP тили ёрдамида бирор-бир компьютерда браузер ва web-серверлардан мустақил равишда ўзи бажариладиган скриптларни ҳам яратиш мумкин. Бу ишларни бажариш учун ҳеч бўлмаганда PHP-парсер (бу ҳолатда биз уни буйруқлар сатри интерпретатори (CLI, command line interpreter) деб атаймиз) талаб этилади. Бундай ишлаш услуби турли масалаларни режалаштириш ёрдамида бажарилиши учун керак бўлган скриптлар ёки оддий матнни қайта ишлаш учун керак бўлган масалага ўхшаш ишлайди.

Ва ниҳоят охирги учинчи соҳа – бу мижоз томонидан бажариладиган GUI-иловаларни (график интерфейс) яратиш. Бу соҳа PHP тилини эндигина ўрганаётган фойдаланувчилар учун унча муҳим бўлмаган соҳадир. Бироқ агарда сиз PHP тилини чуқур ўрганган бўлсангиз, бу соҳа сиз учун анча муҳимдир. PHP тилини бу соҳага қўллаш учун php кенгайтмали махсус ёрдамчи – PHP-GTK талаб этилади.

Шундай қилиб, PHP тилини қўлланилиш соҳалари кенг ва турличадир. Юқоридаги масалаларни еча оладиган бошқа турлича дастурлаш тиллари ҳам мавжуд, унда нима учун PHP тилини ўрганишимиз керак? У тил бизга нима беради? Биринчидан, PHP тили ўрганиш учун жуда қулай. PHP тилини синтаксиси асосий қоидалари ва ишлаш принципи билан етарлича танишиб чиқиб ўзингизни шахсий дастурингизни тузиб кўриб, сўнгра уни бошқа дастурлаш тилларида тузилган вариантлари билан солиштирсангиз бунга гувоҳи бўласиз.

Иккинчидан, PHP тили барча бизга маълум платформаларда, барча операцион тизимларда ҳамда турлича серверларда эркин ишлай олади. Бу хусусият жуда муҳим. Масалан, кимдир Windows операцион тизимдан Linux операцион тизимга ёки IIS сервердан Apache серверга ўтмоқчи бўлса PHP тилини ўрганиши шарт.

PHP дастурлаш тилида дастурлашнинг иккита ҳаммабоп парадигмалари ишлатилади, булар процедурали ва объектли дастурлаш. PHP4 дастурлаш тили процедурали дастурлашни бутунлай қўллаб қувватлайди, бироқ объектли стилдаги дастурларни ҳам қўлласа бўлади. PHP5 дастурлаш тилининг биринчи тестлаш версиясида PHP4 дастурлаш тилида учрайдиган объектга йўналтирилган дастурлаш моделларининг камчиликлари тўлдирилган. Шундай қилиб, ҳозирда таниш бўлиб улгурган ишлаш принципини танлаш керак.

Агарда PHP тилини ҳозирги имкониятлари тўғрисида гаплашадиган бўлсак, у ҳолда биз PHP тилини биринчи версиясидан анча йироқлашиб кетган бўламиз. PHP дастурлаш тили ёрдамида тасвирлар, PDF-файллар, флэш-роликлар яратиш мумкин; ҳозирги вақтдаги замонавий маълумотлар базасини қўллаб қувватлайди; ихтиёрий матнли файл форматлари билан, ҳамда XML ва файллар тизими билан ишлайдиган функциялар ҳам қўшилган. PHP тили турли сервислар ўртасидаги протоколларнинг ўзаро алоқасини қўллаб қувватлайди. Буларга мисол тариқасида папкаларга киришни бошқариш протоколи LDAP, тармоқ қурилмалари билан ишлайдиган протокол SNMP, маълумотларни узатиш протоколлари IMAP, NNTP ҳамда POP3, гиперматнларни узатиш протоколи HTTP ва бошқаларни олиш мумкин.

PHP дастурлаш тилини турли дастурлаш тиллари ўртасидаги ўзаро алоқасига диққатни қаратсак, бунга Java дастурлаш тилини айтиб ўтиш керакки, Java дастурлаш тили объектларини PHP тили ўз объектлари сифатида қарайди. Объектларга мурожаат сифатида CORBA кенгайтмасидан фойдаланилади.

Матнли ахборотлар билан ишлаш учун PHP тили ўзига Perl дастурлаш тилидаги тартибланган ифодалар билан ишлай оладиган механизмларни (катта бўлмаган ўзгаришларсиз) ва UNIX-тизимини мерос қилиб олади. XML-ҳужжатларини қайта ишлаш учун стандарт сифатида DOM ва SAX, XSLT-трансформацияси учун API дан фойдаланиши мумкин.

Электрон тижорат иловаларини яратиш учун бир қатор тўловни амалга оширадиган Cybercash, CyberMUT, VeriSign Payflow Pro ҳамда CCVS каби фойдали функциялар мавжуд.

4. Дастурий воситани созлаш ва ўрнатиш

Юқорида PHP тили имкониятларини, қўлланилиш соҳаларини муҳокама қилдик ва тарихини ўргандик. Энди дастурий воситани ўрнатишга керак бўлган ускуналар мажмуини кўриб ўтсак. Модомики, асосий курснинг амалиёти сифатида биз қуйидаги масалаларни кўриб чиқамиз: клиент-сервер технологияси сифатида ишланадиган масалалар, мос равишда скриптлар яратилишида қўлланилиши, серверларни қайта ишлаш. Булар учун бизга web-сервер ҳамда PHP тили интерпретатори керак бўлади. Web-сервер сифатида, масалан, web-мутахассислар

ўртасида машҳур бўлган Apache серверни оламиз. Дастур натижасини кўриш учун web-браузер керак бўлади, бунга мисол Internet Explorer.

Денвер Дистрибутиви

Биз юқорида Linux ва Windows платформалари учун PHP дастурий воситасини созлаш ва ўрнатиш билан етарлича танишмиз. PHP дастурий воситаси ва уни ишлаши учун керак бўладиган компоненталарни ўрганишни хоҳламайдиганлар учун PHP дастурининг тайёр PHP тилини тўлдирадиган дистрибутивлари мавжуд. Бундай дистрибутивлар ичида кенг тарқалгани - Денвер (http://dklab.ru/chicken/web/). Уни ўрнатишни ўрганиш учун web-мутахассислар сайтларига мурожаат қилиш керак. Денверни ўрнатиш жуда оддий ҳамда унга ҳеч қандай билим талаб этилмаслигини айтиб ўтиш керак. Бу дистрибутивни PHP тилини эндигина ўрганаётган ёш дастурчилар учун тавсия этамиз. Жиддий масалаларни ҳал этиш учун эса PHP дастурлаш тилини тўлиқ ўрнатиш ва созлаш керак бўлади.

PHP дастурлаш тили, сервер томонда дастурлаш, денвер пакети

Назорат саволлари:

1. Қанақа дастурлаш тилларини биласиз?
2. PHP қандай дастурлаш тили ҳисобланади?
3. Бу тилни вужудга келишига сабабчи бўлган эҳтиёжлар нимадан иборат деб ўйлайсиз?
4. PHP дастурлаш тили ҳақида умумий маълумот беринг?
5. PHP дастурлаш тилининг асосий тушунчаларини келтиринг?
6. PHP дастурлаш тили имкониятлари ҳақида гапиринг.
7. Сервер томонда дастурлаш деганда нимани тушунасиз?
8. PHP ни ишга тушириш, Денвер пакети ва ундан фойдаланиш ҳақида маълумот беринг?

9. Бошқа дастурлаш тиллари билан PHP дастурлаш тилининг тилининг фарқини тушунтиринг?

10. PHP тилининг қандай афзалликлари бор?

Фойдаланилган адабиётлар:

1. Холзнер С. Перл: специалний справочник. - СПб: "Питер". 2000. - 496 с.
2. Шварс Р., Кристиансен Т. Изучаем Перл. - К: "БҲВ", 2000. - 320 с.
3. Ратшиллер Т., Геркен Т. PHP4: разработка Web-приложений.- СПб:Питер, 2001. - 384 с.
4. Томсон Л., Веллинг Л. Разработка Web-приложений на PHP и MySQL. - К.: "DiaSoft", 2001. - 672 с.
5. Основи современних компютерних технологий. Ред. Хомченко А.Д.
6. Савелев А.Я., Сазонов Б.А., Лукянов Б.А. Персоналний компютер для всех. Хранение и обработка информатсии. Т.1 М.: Висшая школа, 1991.
7. Брябрин В.М. Программное обеспечение персоналних ЭВМ. М.: Наука, 1990.
8. Фролов А.В., Фролов Г.В. Глобалнийе сети компютеров. Практическое введение в Интернет, Э-Маил, FTP, WWW и HTML. М.:Диалог-МИФИ, 1996.

Фойдаланилган манбалар:

1. Intuit.ru.
2. http://pda.coolreferat.com
3. http://www.google.ru
4. http://www.z-oleg.com
5. http://www.i2r.ru/

10-маъруза. PHP дастурлаш тилининг тузилиши. Маълумотлар типлари ва ифодалар билан ишлаш.

Режа:

1. PHP дастурлаш тилининг асосий тузилиши;
2. PHP дастурлаш тилидаги асосий синтаксислар;
3. Маълумотлар типлари билан ишлаш;
4. Альтернатив синтаксислар.

Калит сўзлар: PHP код синтаксиси, маълумот типлари, асосий синтаксислар, альтернатив синтаксислар, ифодалар, жараёнларни бошқариш.

Ишдан мақсад: PHP дастурлаш тили билан танишиш ва ишлаш жараёнини ўрганиш. PHPни ўрганиш ва ва бу тилнинг асосий тузилиши ҳақида маълумотга эга бўлиш. Маълумотлар типлари билан ишлаш жараёни билан танишиб чиқиш. Бу тилни Web теҳнология яратиш жараёнига татбиқ қилиш.

1. PHP дастурлаш тилининг асосий тузилиши

Кўп ҳолларда *PHP* тилини интерпретатори ишлаётганлигини текшириб кўриш учун тузиладиган дастур энг содда дастур деб аталади. Ҳозир биз *PHP* тилидаги ушбу дастурни чуқур ўрганамиз ҳамда уни бошқа дастурлаш тиллари Си, Perl ва JavaScript лардан фарқли томонини текширамиз. Ушбу **мисолни** кўрамиз:

```
<html>
  <head>
  <title> Мисол </title>
  </head>
  <body>
  <?php
  echo "<p> Салом, бу мен – PHP скрипт! </p>";
  ?>
</body>
</html>
```

Бу *PHP* дастурлаш тилининг махсус кодли теглари ёрдамида тузилган содда html-файлдир.

Юқорида айтиб ўтганимиздек, *PHP* дастурлаш тили Си ва Perl дастурлаш тилига ўхшаш. Бироқ келтирилган дастур Си ва Perl дастурлаш тилидаги дастурдан анча катта фарқ қилади. Бу ерда HTML саҳифага чиқариш учун бир қатор махсус буйруқларни ёзиш шарт эмас. Бевосита *PHP*-код асосида қурилган бирор вазифани бажарадиган HTML-*скрипт* ёзилади (бизни мисолда экранда чиқарилган матн). *PHP* дастурлаш тилининг Си ва Perl дастурлаш тилларидан камчилиги шуки, мураккаб скриптларни *PHP* дастурлаш тили анча секин бажаради.

PHP-скриптлар – бу серверда бажариладиган ва қайта ишланадиган дастурлардир. Бу скриптларни JavaScript типидаги скриптлар билан таққослаш мумкин эмас, чунки JavaScript тилидаги скриптларда ёзилган буйруқлар фақат клиент компьютеридагина бажарилади. Клиент компьютерида ва сервер компьютерида бажариладиган скриптларнинг фарқи нимада? Агарда скрипт серверда қайта ишланса, мижоз компьютерига фақатгина натижа юборилади. Масалан, агарда серверда скрипт бажарилаётган бўлса, юқорида келтирилганга ўхшаб мижоз HTML-саҳифа кўринишдаги натижани олади:

```
<html>
   <head>
   <title> Мисол </title>
   </head>
   <body>
   <p> Салом, бу мен – PHP скрипт! </p>
   </body>
```

</html>

Бу ҳолатда мижоз қандай код бажарилаётганини билмайди. Ўз серверингизни HTML-файлларни *PHP* процессори қайта ишлайдиган қилиб созлаб олишингиз ҳам мумкин. Яъни клиентлар оддий HTML-файлни қабул қилдими ёки скрипт натижасини кўрдими буни била олмайди. Агарда скрипт клиент компьютерида қайта ишланса (масалан, JavaScript тилидаги дастур), у ҳолда клиент скрипт кодидан иборат HTML-саҳифани кўради.

Биз юқорида айтиб ўтгандикки, *PHP-скриптлар HTML*-код ичида ёзилади. Қандай қилиб деган савол туғилади. Бунинг бир нечта усуллари мавжуд. Булардан бири биринчи мисолда келтирилганидек, *<?php* теги билан бошланиб *?>* теги билан тугаган синтаксис. Бундай кўринишдаги махсус теглар HTML ва *PHP* режимидагина ишлатилади. Бу синтаксис *PHP* тилини *XML* ҳужжатлари билан биргаликда ишлайдиган дастурларида жуда маъқул кўрилади (масалан, XHTML тилида ёзилган дастурларда). Бироқ базан қуйидаги альтернатив вариантдан фойдаланса ҳам бўлади(echo "Some text" буйруғи «Some text» матнини экранга чиқаради.):

```
1.  <? echo "Бу PHP тилида
2.          оддий қайта ишлашнинг
3.          инструкцияси"; ?>
4.
5.  <script language="php">
6.   echo "Бир нечта редакторлар
7.         (FrontPage) қуйидагича
8.         қабул қилишади";
9.  </script>
10.
11. <% echo " ASP технологиясидаги тегдан
12.         ҳам фойдаланса бўлади"; %>
```

Бу келтирилган усуллардан биринчиси ҳар доим ҳам бажарилавермайди. Ундан фойдаланиш учун қисқа тегларни ишлатиш керак, ёки *PHP*3 учун *short_tags*() функцияни ишлатиш керак, ёки *PHP* тилининг конфигурацион файлига short_open_tag буйруқни ўрнатиш керак, ёки *PHP* дастурлаш тилида *enable-short-tags* параметр билан компиляция қилиш керак. Агарда php.ini-dist буйруққа юқоридагилар автоматик қўшилган бўлса, у ҳолда қисқа теглардан фойдаланиш тавсия этилмайди. Иккинчи усул худди ўрнига қўйишга ўхшайди, масалан, JavaScript кодлари ва унинг учун мос html теглар. Шунинг учун ундан ҳар доим фойдаланиш мумкин, лекин бу ноқулайлиги учун камдан-кам ишлатилади. Учинчи усулдан фақат ASP технологиясидаги теглар *asp_tags* конфигурациясида ишлатилгандагина фойдаланилади.

PHP дастурлаш тили файлни қайта ишлаётганда у оддий матнни *PHP* код интерпретация қилиши керак бўлган махсус тегларни учратмагунча қайтариб беради. Интерпретатор ҳақида гапирганда у топилган барча кодни ёпиладиган теггача бажаради, сўнг яна оддий матн қайтарилади. Бу механизм *PHP*-кодни HTML саҳифага айлантиради, яъни барча *PHP* теглардан ташқари барча матнларни ўзгаришсиз сақлайди ва ичкаридагиларни эса интерпретациялайди. Яна шуни айтиш керакки, php-файл *CGI*-скриптга ўхшамайди. php-файл бажарилиши шарт эмас, ёки яна қандайдир белгиланади.

php-файлни серверда қайта ишлаш учун жўнатишда сервер томонидан браузер сатрида бу файлни йўлини кўрсатиш шарт. *PHP* скриптлар www орқали киришга рухсат этилган жойда жойлашиши шарт. Агарда php-файл локал компьютерда мавжуд бўлса, у ҳолда уни буйруқлар сатри интерпретатори ёрдамида қайта ишлаш мумкин.

Шундай қилиб, биз *PHP* дастурлаш тили ҳақида маълумотга эга бўлдик, у қандай дунёга келган ва тарқалган, уни қандай ва қаерда фойдаланилишини ўргандик, дастурий воситани ўрнатдик ҳамда уни ишлаши учун барча созлашларни бажардик ва *php*-дастур нималардан ташкил топишини англадик. Кейинги бўлимларда биз *PHP* дастурлаш тилининг асосий синтаксисларини кўриб чиқамиз ҳамда бир қанча амалий масалаларни ҳал этамиз.

2. PHP дастурлаш тилидаги асосий синтаксислар

Инструкцияни бир нечта қисмга бўлиб кўриб чиқамиз, яъни комментарийлар яратиш, ўзгарувчилар, ўзгармаслар ва маълумот типлари, операторларга.

Биз энди *PHP* дастурлаш тилининг асосий синтаксис элементларини ўрганишга ўтамиз. Мисол сифатида электрон мактуб тайёрлаш масаласини кўриб ўтайлик. Унинг маъноси қуйидагидан иборат.

Фараз қиламизки, сизда қандайдир эълон ва эълонни жўнатишингиз керак бўлган бир нечта одамлар мавжуд бўлсин. Бунинг учун сиз эълонни ичида ўзгарадиган (қабул қилувчи билан боғлиқ бўлмаган) бир нечта параметрлари мундарижаси билан тайёрлайсиз.

Биринчи навбатда PHP дастурлаш тили синтаксисига нисбатан нималарни билиш керак. Бу HTML-код ичига ўрнатилган ва PHP дастурлаш тилидаги коддир, уни интерпретатор фарқлай билади. Аввалги бўлимларда булар ҳақида айтиб ўтгандик. Ҳаммасини қайтариб ўтмаймиз, фақат биз кўп ҳолларда мисолларда <?php ?> вариант ўрнига қисқартирилган <? ?> теглардан фойдаланишни айтиб ўтамиз.

Инструкцияларни ажратилиши

PHP дастурлаш тилидаги дастур(ихтиёрий дастурлаш тилидаги) – бу буйруқлар (инструкциялар) тўпламидир. Дастурни қайта ишлаш учун бир буйруқни бошқа буйруқдан фарқини билиш керак. Бунинг учун махсус символлар – ажратгичлардан фойдаланилади. PHP дастурлаш тилида инструкцияларни худди Си ёки Perl дастурлаш тиллари каби ажратилади, яъни ҳар бир ифода нуқтали вергул (“;”) билан тугайди.

«?>» ёпиладиган тег ҳам инструкцияни тугашини англатади, шунинг учун ундан олдин нуқтали вергул қўйилмайди. Масалан, қуйидаги икки фрагментлар эквивалентдир:

```
<?php
echo "Hello, world!"; // буйруқлар охирида нуқтали вергул қўйиш шарт
?>
<?php
echo "Hello, world!" ?>
<!-- "?>" борлиги учун
   нуқтали вергул ташлаб кетилди -->
```

Комментарийлар

Кўп ҳолларда дастур тузганда кодни тушунарли бўлиши учун унга қандайдир изоҳ-***комментарийлар*** қўйиш керак бўлиб қолади. Бу ҳолат катта ҳажмдаги дастурлар яратганда ҳамда агарда битта дастур устида бир нечта дастурчи ишлаётганда жуда муҳим. Комментарийлар дастурнинг коди тушунарли бўлиши учун ёзилади. Бундан ташқари масалани қисмларга ажратиб ҳал қилинганда ишнинг камчилиги бор жойида кейинчалик эсдан чиқмаслиги учун комментария ёзиб қўйилади. Барча дастурлаш тилларида дастур ичига комментария қўшиш имконияти мавжуд. *PHP* дастурлаш тили бир қанча кўринишдаги комментарийларни қўллаб қувватлайди: Си, C++ дастурлаш тиллари стилидаги ҳамда Unix қобиғидаги комментарийлар. // ва # белгилар бир сатрли

комментарийларни англатса, /* ва */ белгилар эса мос равишда кўп сатрли комментарийларнинг бошланиш ва тугашини англатади.

Мисол:PHP дастурлаш тилида комментарийнинг қўлланилиши

```
<?php
echo "Мени исмим Алишер";
  // Бу бир сатрли комментарий
  // C++ дастурлаш тили стилидаги
echo "Мени фамилиям Болиев";
/* Бу кўп сатрли комментарий. Бу ерга бир қанча сатр ёзиш мумкин. Дастур бажарилиш жараёнида бу ердаги барча ёзувлар (комментарийланган), ўқилмайди. */
echo "Мен PHP дастурлаш тилини INTUIT.ru дан ўрганяпман";
  # Бу комментарий
  # Unix қобиғидаги комментарий.
?>
```

Ўзгарувчилар, ўзгармаслар ва операторлар

Ҳар бир дастурлаш тилида муҳим элементлардан бири бу *ўзгарувчилар*, *ўзгармаслар* ва улар қўлланиладиган *операторлар*дир. PHP дастурлаш тили бу элементларни қандай белгилаши ва қайта ишлашини кўриб чиқамиз.

Ўзгарувчилар

PHP дастурлаш тилида *ўзгарувчилар* олдига доллар белгиси (“$”) қўйиб эълон қилинади, масалан, $my_var.

Ўзгарувчилар номлари регистрларни фарқлайди, яъни $my_var ҳамда бош ҳарфли $My_var ўзгарувчилари турли хил ўзгарувчилардир.

PHP дастурлаш тилида ўзгарувчилар номи қолган дастурлаш тиллари қоидалари каби эълон қилинади: ўзгарувчи номи лотин алфавити билан бошланиши ва ундан кейин ҳарфлар ёки тагига чизилган белги ёки рақамлар бўлиши мумкин.

PHP4 дастурлаш тилида булардан ташқари ўзгарувчига қиймат ўзлаштиришнинг яна бир усули мавжуд: ссылка бўйича *ўзлаштириш*. Ссылка бўйича ўзгарувчига қиймат ўзлаштириш учун уни номи бўлиши шарт, яъни у қандайдир ўзгарувчини тақдим этиши керак. Бир ўзгарувчи қийматини бошқа ўзгарувчига Ссылка бўйича *ўзлаштириш* учун биринчи ўзгарувчи олдига амперсанд & белгиси қўйиш шарт.Бунга юқоридаги мисолни кўриб чиқамиз, фақат first ўзгарувчи second ўзгарувчига ссылка бўйича ўзлаштирилади:

Мисол. Ссылкалар бўйича ўзлаштириш.

```
<?php
$first = ' Text '; // $first ўзгарувчига
                   // ' Text ' қиймат ўзлаштирилди
$second = &$first;
/* $second.орқали $first ўзгарувчига ссылка қиламиз
   Энди бу ўзгарувчилар қийматлари
    ҳар доим тенгдир */
// $first ўзгарувчи қийматини
// ' New text ' қийматга ўзгартирамиз
$first = ' New text ';
echo "first номли ўзгарувчи қиймати $first га тенг <br>";
// $second ўзгарувчи қийматини экранга чиқарамиз
echo "second номли ўзгарувчи қиймати " . "$second га тенг";
?>
```

Бу скриптни натижаси эса қуйидагича бўлади:

- first номли ўзгарувчи қиймати New text га тенг.
- second номли ўзгарувчи қиймати New text га тенг.

Яъни $first ўзгарувчи қиймати ўрнига $second ўзгарувчи қиймати ўзлаштирилди.

Ўзгармаслар

Скрипт бажарилиш жараёнида ўзгармайдиган қийматли катталикларни сақлаш учун ***ўзгармаслар***дан фойдаланилади. Бундай катталиклар математик ўзгармаслар, пароллар, файлларнинг йўллари ва бошқалар бўлиши мумкин. Ўзгармасларнинг ўзгарувчилардан асосий фарқи шуки, уларни фақат бир мартагина ўзлаштирилади ва уни қийматини эълон қилингандан кейин бекор қилиб бўлмайди. Бундан ташқари ўзгармаслар олдида доллар белгиси қйилмайди ҳамда уни оддий қиймат ўзлаштириш каби қараш мумкин эмас. Ўзгармаслар қандай аниқланади? Бунинг учун махсус define() функцияси мавжуд, унинг синтаксиси қуйидагичадир:

define("Ўзгармас номи", "Ўзгармас қиймати", [регистрга_сезгирлиги_кичик])

Ўзгармаслар номи регистрга сегирлиги катта. Ҳар бир ўзгармасларда уни ўзгартириш мумкин, яъни *регистрга_сезгирлиги_кичик* аргументни қиймати

сифатида True ыиймати кўрсатилади. Ўзгармаслар номи ҳар доим катта регистр билан ёзишга келишиб олинган.

Ўзгармасни қийматини билиш учун уни номини кўрсатиш керак. Ўзгарувчидан фарқи ўзгармас номи олдига $ белги қўйилмайди. Бундан ташқари ўзгармасни қийматини билиш учун константа номи билан параметр сифатида constant() функциясидан фойдаланиш мумкин.

Мисол. PHP дастурлаш тилида ўзгармаслар.

```
<?php
// ўзгармасни аниқлаймиз PASSWORD
define("PASSWORD","qwerty");
// регистрланмаган PI ўзгармасни қийматини аниқлаймиз 3.14
define("PI","3.14", True);
// PASSWORD ўзгармас қийматини оламиз,  яъни qwerty
echo (PASSWORD);
// бу ҳам qwerty ни чиқаради
echo constant("PASSWORD");
echo (password);
/* password ни чиқаради ва биз регистрланган ўзгармас PASSWORD ни кутгандик.*/
echo pi;
// 3.14 ни чиқаради, чунки ўзгармас PI регистрланмаган ва аниқланган.
?>
```

Дастурчи томонидан ўзгарувчилардан ташқари юқорида айтиб ўтганимиздек *PHP* дастурлаш тилида мавжуд ўзгармаслар ҳам интерпретатор томонидан аниқланади. Масалан, __FILE__ ўзгармас дастур бажарилиш жараёнида файл номини (ва файл йўлини), __FUNCTION__ функция номидан ташкил топади, __CLASS__ - синф номи, PHP_VERSION – *PHP* дастурлаш тили интерпретатори версиясини ўзида сақлайди. Бундай ўзгармасларнинг барча рўйхатини *PHP* дастурлаш тили учун мўлжалланган қўлланмалардан топиш мумкин.

Амаллар

Ўзгарувчилар, ўзгармаслар ва ифодалар устида турли ҳисоблашларни бажарадиган бу ***амаллар***дир. Биз ҳали бу ифодалар ҳақида тўхтаб ўтганимиз йўқ. Ифодалар қийматини ушбу амаллар ёрдамида аниқланади. Ўзгарувчилар ва ўзгармаслар – бу ифодаларнинг асосий ва жуда содда шаклидир. Шундай

ифодаларни кўпайтириши мумкин бўлган амаллар тўплами мавжуд. Уларни қуйида тўлиқроқ муҳокама қиламиз:

9.1-жадвал. Арифметик амаллар.

Белгиланиши	Номланиши	Мисол
+	Қўшиш	$a + $b
-	Айириш	$a - $b
*	Кўпайтириш	$a * $b
/	Бўлиш	$a / $b
%	Бўлишдаги қолдиқ	$a % $b

9.2-жадвал. Сатрли амаллар.

Белгиланиши	Номланиши	Мисол
.	*Конкатенация* (*сатрларни қўшиш*)	$c = $a . $b (бу $c сатр $a ва $b сатрлардан иборат)

9.3-жадвал. Ўзлаштириш амаллари.

Белгиланиши	Номланиши	Изох	Мисол

=	Ўзлаштириш	*Операторлан* ўнг томонда турган ўзгарувчилар устида бажарилган амаллардан ҳосил бўлган натижа қиймати ўзлаштирилади.	$a = ($b = 4) +5; ($a 9 га тенг, $b 4 га тенг)
+=		Қисқартириш. Ўзгарувчига сон қўшилади ва кейин натижа ўзлаштирилади.	$a += 5; ($a = $a + 5 ифодага эквивалент;)
.=		Ўзлаштириш ва конкатенация амаллари комбинациясини қисқартирилган шакли(даставвал сатрлар қўшилади, сўнгра ҳосил бўлган сатр ўзгарувчига ўзлашади).	$b = "Ҳаммага "; $b .= "салом"; ($b = $b . "салом" ифодага эквивалент;) Натижаси: $b="Ҳаммага салом"

9.4-жадвал. Матиқий амаллар.

Белгиланиши	**Номланиши**	**Изох**	**Мисол**
and	ВА	$a ва $b рост (True)	$a and $b $a && $b $a or $b $a || $b $a
&&	ВА		
or	ЁКИ	$a ёки $b ўзгарувчилардан ҳеч бўлмаганда биттаси рост бўлса (иккаласи ҳам рост бўлишиш мумкин).	
||	ЁКИ		
xor	Инверсия ЁКИ	Ўзгарувчилардан биттаси рост бўлса. Агарда иккаласи ҳам рост бўлса инверсияланади.	

!	Инверсия (NOT)	Агарда $a=True, у ҳолда !$a=False ва акс ҳолда тескариси бўлади.	xor $b ! $a

9.5-жадвал. Таққослаш амаллари.

Белгиланиши	Номланиши	Изох	Мисол
==	Тенглик	Ўзгарувчилар қийматлари тенг	$a == $b
===	Эквивалентлик	Ўзгарувчилар қийматлари ва типлари тенг	$a === $b
!=	Тенгсизлик	Ўзгарувчилар қийматлари тенг эмас	$a != $b
<>	Тенгсизлик		$a <> $b
!==	Ноэквивалентлик	Ўзгарувчилар эквивалент эмас	$a !== $b
<	Кичик		$a < $b
>	Катта		$a > $b
<=	Кичик ёки тенг		$a <= $b
>=	Катта ёки тенг		$a >= $b

9.6-жадвал. Инкремент ва декремент амаллари.

Белгиланишти	Номланиши	Изох	Мисол
++$a	Пре-*инкремент*	$a қиймати бирга оширилади ва $a қиймати қайтарилади	<? $a=4; echo "4 бўлиши шарт:" .$a++; echo "6 бўлиши шарт:" .++$a; ?>
$a++	Пост-*инкремент*	$a қиймати қайтарилади ва сўнгра $a қиймати бирга оширилади	
--$a	Пре-*декремент*	$a қиймати бирга камайтирилади ва $a қиймати қайтарилади	
$a--	Пост-*декремент*	$a қиймати қайтарилади ва сўнгра $a қиймати бирга камайтирилади	

3. Маълумотлар типлари билан ишлаш

PHP дастурлаш тили саккизта содда *маълумот типлари*ни қўллаб қувватлайди:

Тўрттаси скаляр *типлар*:

- *boolean (мантиқий)*;
- *integer (бутун)*;
- *float (нуқтаси силжийдиган)*;
- *string (сатрли)*.

Иккитаси аралиш *типлар*:

- *array (массив)*;
- *object (объект)*.

Иккитаси махсус *типлар*:

- *resource (ресурс)*;
- *NULL*.

PHP дастурлаш тилида ўзгарувчилар типлари ошкора эълон қилинмайди. Кўпинча ўзгарувчи қўлланилган контексдан, яъни ўзгарувчига ўзлаштирилган қиймат итпидан мустақил равишдаги дастур бажарилиш жараёнидан интерпретатор ўзи бу ишни бажаради. Қуйида юқорида санаб ўтилган *маълумотлар типлари*ни бирма-бир кўриб чиқамиз.

Boolean типи(Буль ёки мантиқий тип)

Бу содда тип қийматни рост эканлигини ифодалайди, яъни ўзгарувчи фақат иккита қиймат қабул қилади – рост TRUE ёки ёлғон FALSE.

Мантиқий типларни аниқлаш учун TRUE ёки FALSE калит сўзларидан фойдаланамиз. Бу иккала типлар регистрланмаган.

Мисол. Мантиқий тип.

```
<?php
$test = True;
?>
```

Мантиқий типлар турли *бошқариладиган конструкциялар*да (цикллар, шартлар ва шунга ўхшаш, булар ҳақида кейинроқ айтиб ўтамиз) қўлланилади. Бир қанча амаллар (масалан, тенглик амали) ҳам мантиқий тип қабул қилиши мумкин, яъни фақат икки қиймат рост ёки ёлғон қийматни қабул қилади. Улар *бошқариладиган конструкциялар*да шартларни текшириш учун қўлланилади. Масалан, шартли конструкторда амаллар ёки ўзгарувчилар қиймати ҳақиқийлигини текширади ва натижадан қатъий назар шу ёки бошқа амалларни бажарилишини текширади. Бу ерда шарт рост ёки ёлғон бўлиши мумкин, чунки *мантиқий тип амаллари* ва *ўзгарувчилар* кўрсатилган.

Мисол. Мантиқий типларнинг қўлланилиши.

```
<?php
// '==' амал тенгликка текширади мантиқий қийматни қайтаради
if ($know == False) { // агар $know қиймат false бўлса
echo "PHP дастурлаш тилини ўрган!";
}
if (!$know) { // худди юқоридагидек $know қиймати false бўлади
echo " PHP дастурлаш тилини ўрган!";
}
```

/* == амал $action ўзгарувчи қиймати билан "PHP дастурлаш тилини ўрганиш!" сатрни устма-уст тушишини текширади. Агар устма-уст тушса true қийматни қайтаради, бошқа ҳолда false ни қайтаради. Агар true ни қайтарса фигурали қавс ичидаги амаллар бажарилади. */

```
if ($action == " PHP дастурлаш тилини ўрганиш ")
{ echo "Ўрганишни бошладим";}
?>
```

Integer (бутун) типи

Бу тип бутун сонлар тўпламидан Z = {..., -2, -1, 0, 1, 2, ...} бирини қайтаради. Бутун сонлар хоҳишга қараб олдига «-» ёки «+» белгиларни қўйиб саноқ системасини ўнлик, ўн олтилик ёки саккизлик тизимларида кўрсатилган бўлиши мумкин.

Агар сиз саккилик саноқ системасидан фойдаланаётган бўлсангиз, олдиндан 0 (ноль) рақамини кўрсатишингиз керак. Ўн олтилик саноқ системасида эса рақамлар олдига 0x белгини қўйиш шарт.

```
<?php
# ўнлик рақам
$a = 1234;
# манфий сон
$a = -123;
# саккизлик сон (ўнлик системасидаги
# 83 сонга эквивалент)
$a = 0123;
# ўн олтилик сон (ўнлик системасидаги
# 26 сонга эквивалент)
$a = 0x1A;
?>
```

*Бутун сон*ни ўлчами платформага боғлиқ, лекин қоидага кўра максимал қиймати икки миллиард (бу ишорали 32 битли қиймат) атрофида бўлади. Ишорасиз *бутун сон*ни *PHP* дастурлаш тили қўллаб қувватламайди.

Агар сиз *бутун сон* чегарасидан ташқари бирор қиймат берсангиз интерпретатор бу сонни *қўзғалувчан вергулли сон*га ўзгартиради. Худди шундай *бутун сон* чегарасидан ташқари чиқиб кетадиган бирор амал бажарсангиз ҳам бу сонни *қўзғалувчан вергулли сон*га ўзгартирилади.

PHP дастурлаш тилида бутун сонларни бўлиш амали мавжуд эмас. 1/2 ифода қиймати *қўзғалувчан вергулли сон* 0.5 га тенг. Сиз натижангизни бутун типга стандарт қоида асосида ёки round() функциясидан фойдаланган тақдирда ўзгартиришингиз мумкин. Ўзгарувчини аниқ бир типга ўзгартириш учун унинг олдига қавс ичида керакли типни ёзиш керак бўлади. Масалан, $a=0.5 ўзгарувчини

бутун типга ўзгартириш учун (integer)(0.5) ёки (integer) $a кўринишда ёки қисқартирилган (int)(0.5) кўринишда ёзиш керак бўлади. Бундай ошкора янги типга ўтиш имконияти барча *маълумотлар типлари* учун ўринли бўлади (албатта, ҳар доим ҳам қийматни бир типдан бошқасига олиб ўтиш шарт эмас). Биз келтирилган барча типларни чуқур ўрганишимиз шарт эмас, чунки *PHP* дастурлаш тили контексдан мустақил равишда ўзи бу ишларни бажаради.

Float (қўзғалувчан вергулли сон) типи

*Қўзғалувчан вергулли сон*лар (улар икки карра аниқлик ёки ҳақиқий сонлардир) қуйидаги синтаксислар ёрдамида аниқланиши мумкин:

```
<?php
$a = 1.234;
$b = 1.2e3;
$c = 7E-10;
?>
```

*Қўзғалувчан вергулли сон*ни ўлчами ҳам платформага боғлиқ, лекин қоидага кўра максимал қиймати ~1.8e308 аниқлик билан 14 хонали рақам атрофида бўлади.

Resource (ресурслар) типи

Ресурс – бу ташқи ресурсга (масалан, маълумотлар базаси билан боғланиш) ссылка орқали боғланган махсус ўзгарувчидир. Ресурслар махсус функциялар (масалан, mysql_connect(), pdf_new() ва шунга ўхшашлар) ёрдамида яратилади ва фойдаланилади.

Null типи

Махсус *NULL* қиймати *ўзгарувчи*ни қийматга эга эмаслиги ҳақида огоҳлантиради.

Ўзгарувчи NULL қиймат қабул қилади, агарда:

- унга *ўзгармас NULL* ($var = NULL) ўзлаштирилган бўлса;
- унга ҳеч қандай қиймат берилмаган бўлса;
- у *unset*()функция ёрдамида тозаланган бўлса.

*NULL тип*ли фақат битта қиймати мавжуд – регистрга сезгирлиги кичик NULL калит сўзидир.

Масаланинг ечилиши

Энди бўлимнинг бошида қўйилган масалага қайтсак. У турли сабаблар бўйича ҳар хил одамларга тузилган мактубни жўнатишдан иборат эди. Бу масалани ҳал этиш учун ўрганилган воситалардан – *ўзгарувчилар*, *амаллар*, *ўзгармаслар*, *сатрлар* ва *массивлар*дан фойдаланишга ҳаракат қиламиз. Кўрсатилган мактуб қабул қилувчига боғлиқ равишда мурожаат ва ҳолати ўзгаради, шунинг учун табиий равишда бу катталикни *ўзгарувчи* деб белгилаймиз. Бундан ташқари ҳодисалар ва одамлар кўп, шунинг учун *массив ўзгарувчи типи*дан фойдаланиш қулай. Мактуб матни ҳар доим ўзгармас, шунинг учун уни *ўзгармас* деб бериш мақсадга мувофиқдир. Жуда узун ва қўпол сатрларни ёзмаслик учун сатрлар *конкатенация*(қўшиш) амалидан фойдаланамиз. Шундай қилиб, қуйидагига эга бўламиз:

```
<?
// бизнинг ёзувимиз
// ўзгармас бўлсин.
define("SIGN","Ҳурмат билан, Азамат");
// одамлар ва ҳодисалар массивини берамиз
$names = array("Иван Иванович",
               "Петр Петрович",
               "Семен Семенович");
$events = array(
   "f" => "очиқ эшиклар куни",
   "o" => "кўргазманинг очилиши",
   "p" => "битирувчилар бали");

// таклифнома матнини тузамиз.
$str = "Ҳурматли, $names[0]";
$str .= "<br> Сизни таклиф этамиз ".
       $events["f"];
$str .= "<br>" . SIGN;
echo $str; // матнни экранга чиқарамиз.
?>
```

Шундай қилиб, бу бўлимда биз *PHP* дастурлаш тилининг асосий синтаксиси билан танишиб чиқдик, турли типдаги ўзгарувчилар, ўгармаслар ва амаллар билан ишлашни, *PHP* дастурлаш тилидаги мавжуд типларини ўргандик. *Массивлар* ва *сатрлар* маълумот типлари ҳақида гап кетганда уларни чуқур ва қисмларга ажратиб ўргандик. Бу конструкциялар фойдаланишга қулай ва соддадир. Булар ҳақида кенг маълумотлар кейинги бўлимларда келтирилган. Масаланинг ечилиши бор билимларга асосланган ҳолда содда ечилган, шунинг учун ечим амалиётда

қўллашга жуда яқин келмайди. Кейинги бўлимларда бу камчиликларни тўғрилаймиз ва электрон мактубни умумий шаблонини яратамиз.

4. Альтернатив синтаксислар

PHP дастурлаш тили ўзининг бир нечта *if*, *while*, *for*, *foreach* ҳамда *switch* бошқариладиган структуралари учун альтернатив синтаксисни тақдим этади. Ҳар бир ҳолатда очиладиган қавс икки нуқтага (:), ёпиладигани эса мос равишда endif;, endwhile; ва ҳоказоларга ўзгартирилади.

Масалан, *if* шарт оператори синтаксисини қуйидагича тфодалаш мумкин:

if (ифода) : *бажариладиган_блок* endif;

Маъноси ўзгармасдан қолади: агар *if* шарт оператори думалоқ қавси ичидаги шарт рост бўлса, икки нуқтадан «:» то endif; буйруғигача барча код бажарилади. Бундай синтаксисдан фойдаланиш html-код ичида қурилган php-код учун қулайдир.

Мисол. Альтернатив синтаксисдан фойдаланиш.

```
<?php
$names = array("Карим","Салим","Содиқ");
if ($names[0]=="Карим"):
?>
Салом, Карим!
<?php endif ?>
```

Агарда *else* ҳамда *elseif* конструкцияларидан фойдаланилса, у ҳолда ҳам альтернатив синтаксисдан фойдаланса бўлади:

```
<?php
if ($a == 5):
    print "a ўзгарувчи 5 га тенг";
    print "...";
elseif ($a == 6):
    print "a ўзгарувчи 6 га тенг ";
    print "!!!";
else:
    print "a ўзгарувчи на 5 га ва на 6 га тенг ";
endif;
?>
```

Назорат саволлари:

1. PHP кодни тузилишини тушунтириб беринг.

2. PHP да қандай маълумот типларидан фойдаланилади?

3. PHP да ифодалар қандай эълон қилинади?

4. Альтернатив синтаксислар деганда нимани тушунасиз?

5. Бошқа дастурлаш тиллари билан PHP дастурлаш тилининг фарқини тушунтиринг?

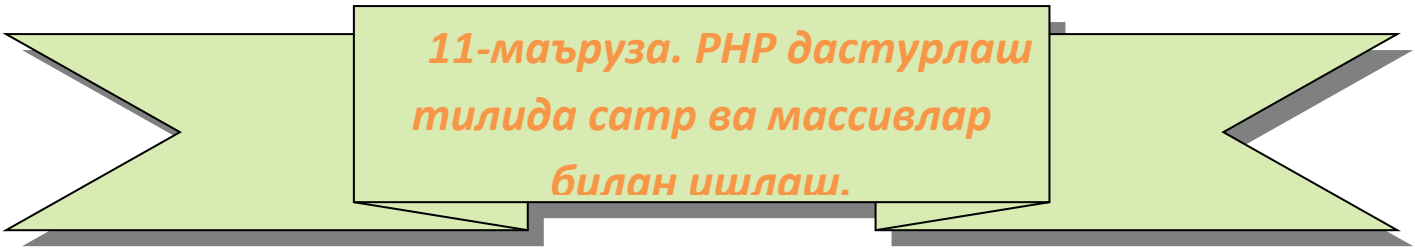
6. PHP тилининг қандай афзалликлари бор?

Фойдаланилган адабиётлар:

9. Холзнер С. Перл: специалний справочник. - СПб: "Питер". 2000. - 496 с.
10. Шварс Р., Кристиансен Т. Изучаем Перл. - К: "БҲВ", 2000. - 320 с.
11. Ратшиллер Т., Геркен Т. PHP4: разработка Web-приложений.- СПб:Питер, 2001. - 384 с.
12. Томсон Л., Веллинг Л. Разработка Web-приложений на PHP и MySQL. - К.: "DiaSoft", 2001. - 672 с.
13. Основи современних компютерних технологий. Ред. Хомченко А.Д.
14. Савелев А.Я., Сазонов Б.А., Лукянов Б.А. Персоналний компютер для всех. Хранение и обработка информатсии. Т.1 М.: Висшая школа, 1991.
15. Брябрин В.М. Программное обеспечение персоналних ЭВМ. М.: Наука, 1990.
16. Фролов А.В., Фролов Г.В. Глобалнийе сети компютеров. Практическое введение в Интернет, Э-Маил, FTP, WWW и HTML. М.:Диалог-МИФИ, 1996.

Фойдаланилган манбалар:

6. Intuit.ru.
7. http://pda.coolreferat.com
8. http://www.google.ru
9. http://www.z-oleg.com
10. http://www.i2r.ru/



Режа:

1. PHP дастурлаш тилида сатр типи;

2. PHP дастурлаш тилида массивлар типи;

3.Квадрат қавс синтаксиси ёрдамида массивни аниқлаш;

4.Массивлар инициализатсияси;

5. Массивларни кўриб чиқиш учун foreach цикли;

6. Кўп ўлчовли массивлар;

7.Массивларни тартиблаш функциялари.

Калит сўзлар: PHP да сатр типи, массив типи, кўп ўлчовли массивлар, квадрат қавс синтаксиси, массив инициализасияси, foreach цикли, heredoc синтаксиси, тартиблаш функциялари.

Ишдан мақсад: PHP дастурлаш тили билан танишиш ва ишлаш жараёнини ўрганиш. PHP дастурлаш тили билан ишлаш жараёнида сатрлар ва массивлар билан ишлаш кўникмаларини ҳосил қилиш.

1. PHP дастурлаш тилида string (сатр) типи

Сатр – бу белгилар тўпламидир. *PHP* дастурлаш тилида белги бу бир байт ва 256 та турли белгилар мавжуд. *PHP* дастурлаш тили Unicode типидаги белгиларни қабул қилмайди. *PHP* дастурлаш тилида амалда сатрларга чегирма мавжуд эмас, шунинг учун сатрларни ишлатганда унинг аниқ узунлиги ҳақида ўйлаш шарт эмас.

PHP дастурлаш тилида сатрлар учта турли хил усулларда аниқланади:

- *битталик қўштирноқлар* ёрдамида (‘’);
- *қўштирноқлар* ёрдамида (“”);
- *heredoc-синтаксиси* ёрдамида.

Биттали тирноқлар

Сатрларнинг аниқлашнинг оддий усули – у «'» *биттали қўштирноқлар* ичида ёзилади. Агарда сатр ичида ҳам биттали тирноқ ишлатишга тўғри келиб қолса, биттали тирноқдан олдин «\» белгини қўйиш, яъни уни экранлаш шарт. Агарда «\» белги биттали тирноқдан олдин ёки сатрнинг охирида бўлса, у ҳолда белгини иккилантириш керак, яъни «\\'».

Агарда биттали тирноқ ичидаги сатр ичида ихтиёрий белгидан олдин («\» ва «'» лардан фарқли равишда) тескари слэш «\» белгиси учраса, у ҳолда уни оддий белги деб қараб барча белгиларни ўз ҳолича экранга чиқаради. Шунинг учун

тескари слэш «\» белгисини сатр охирида ёпиладиган қўштирноқдан аввал турганини экранлаш шарт.

PHP дастурлаш тилида тескари слэш «\» белгиси билан ифодаланадиган бир қатор белгилар мажмуи мавжуд. Уларни ***кетма-кетликни бошқарувчилар*** деб аталади ҳамда улар махсус вазифаларни бажаради. Улар ҳақида кейинроқ тўхталиб ўтамиз. Ўзгарувчилар ва кетма-кетликни бошқарувчилар битталик қўштирноқлар сатри ичида учрашса, улар ўртасидаги фарқ *кетма-кетликни бошқарувчилар*ни *қайта ишланмайди*.

```
<?php
echo 'Сатрлар мажмуи';
// Экранга чиқаради: ' белгини чиқариш учун ундан олдин \ белги қўйилади.
echo ' Белгини \' чиқариш учун ундан олдин'
    ' \\ белгини қўйиш керак';
// Экранга чиқаради: Сиз шуни ўчирмоқчимисиз C:\*.*?
echo ' Сиз шуни ўчирмоқчимисиз C:\\*.*?';
// Экранга чиқаради: Буни қўйманг: \n
// янги қаторга
echo ' Буни қўйманг: \n янги қаторга ';
// Экранга чиқаради: ўзгарувчи $expand ҳам
// $either қўйилмайди
echo 'ўзгарувчи $expand ҳам $either' .
   'қўйилмайди';
?>
```

2. PHP дастурлаш тилида array (массив) типи

PHP дастурлаш тилида ***массив*** типи тартибланган карталарга ўхшайди ва қийматини калитга ўзлаштирадиган типдир. Бу тип бир неча йўналишларда оптималлаштирилади, шунинг учун сиз уни хусусий *массив*, рўйхат (вектор), хеш-жадвали (картани амалга ошириш учун ишлатилади), стэк, навбат ва бошқалар сифатида фойдаланишингиз мумкин. Модомики, *PHP* дастурлаш тилида бир массивни қийматини бошқасига ўзлаштириш учун дарахтлардан фойдаланасиз.

*Массивлар*ни *array*() конструкцияси ёрдамида аниқланади ёки элементларига қиймат бериш билан аниқланади.

array() конструкцияси ёрдамида аниқлаш

```
array ([key] => value,
     [key1] => value1, ... )
```

PHP дастурлаш тилининг ***array***() конструкцияси вергул билан ажратилган жуфт параметрлар *калит* => ***қиймат*** билан ажратилган. => белги мос равишда

қиймат ва унинг калити ўртасида алоқа ўрнатади. Калит бутун сон бўлиши мумкин, унинг қиймати эса *PHP* дастурлаш тилидаги ихтиёрий типни қабул қилиши мумкин. Калит рақамини биз кўпинча индекс деб атаймиз. *PHP* дастурлаш тилида индекслаш нолдан бошланади. Массив элементининг қийматини олиш учун массив номи ва квадрат қавс ичида унинг калити кўрсатилиши керак. Агар массив калити стандарт бутун сон бўлса, у ҳолда унинг қийматини бутун сон деб қараса бўлади, акс ҳолда у сатр деб қаралади. Шунинг учун $a["1"] ёзув $a[1] ёзувга тенг кучли, $a["-1"] ёзув эса $a[-1] ёзувга тенг кучли.

Мисол. PHP дастурлаш тилида массивлар.

```
<?php
$books = array ("php" =>
            "PHP users guide",
            12 => true);
echo $books["php"];
//экранга чиқаради: "PHP users guide"
echo $books[12];     //экранга чиқаради: 1
?>
```

Агарда элемент учун калит берилмаган бўлса, у ҳолда калит сифатида калитнинг максимал қийматига бир қўшиб ҳисобланади. Агарда қиймати мавжуд калит кўрсатилган бўлса, у ҳолда шу калит қийматини экранга чиқаради. PHP 4.3.0 дастурлаш тили версиясидан бошлаб калитнинг максимал қиймати манфий сон деб қаралса, у ҳолда массивнинг кейинги калити ноль (0) бўлади.

Мисол. PHP дастурлаш тилида массивлар.

```
<?php
// $arr ҳамда $arr1 массивлар эквивалентдир.
$arr = array(5 => 43, 32, 56, "b" => 12);
$arr1 = array(5 => 43, 6 => 32,
          7 => 56, "b" => 12);
?>
```

Агарда TRUE ёки FALSE калит сифатида қўлланилса, у ҳолда унинг қиймати мос равишда *integer* типининг бир ва нолига ўзлаштирилади. Агар *NULL* дан фойдаланилса, у ҳолда калит ўрнига бўш сатр ҳосил бўлади. Бу бўш сатрни калит сифатида фойдаланса бўлади, аммо уни қўштирноққа олиш керак бўлади. Бу усул

бўш квадрат қавсни ишлатиш каби эмас. Массивлар ёки объектлар калити сифатида фойдаланиш мумкин ҳам эмас.

3.Квадрат қавс синтаксиси ёрдамида массивни аниқлаш

Массивга қиймат бериш орқали массив яратиш мумкин. Биз юқорида айтиб ўтганимиздек, массив элементи қийматига эга бўлиш учун квадрат қавс ичига унинг калити кўрсатилиши керак, масалан, $book["php"]. Агарда янги калит ва янги қиймат кўрсатсангиз қуйидагича бўлади: $book["new_key"]="new_value" ҳамда массивга янги элемент қўшилади. Агарда калитни кўрсатмай фақат қийматни ўзлаштирсак, яъни $book[]="new_value", у ҳолда массивга янги элемент қўшилади ва уни калити мавжуд максимал қийматга бир қўшилади. Агарда биз қиймат берган *массив* яратилмаган бўлса, у ҳолда биз қиймат бергандан кейин у *яратилади*.

```
<?
$books["key"]= value; // key калити билан value қиймат $books массивига қўшилади
$books[] = value1; /* 13-калит билан value1 қиймати массивга қўшилади, чунки
бизда калитнинг максимал қиймати 12 эди. */
?>
```

Массивнинг аниқ бир элементини ўзгартириш учун унинг шу калити билан янги қийматга ўзлаштириш керак. Массив элементи калитини ўзгартириш мумкин эмас, фақат ўчириш (калит ва элементи жуфтлигини) ва янги қўшиш мумкин холос. Массив ***элементини ўчириш*** учун ***unset***() функциясидан фойдаланиш керак.

```
<?php
$books = array ("php" =>"PHP users guide",12 => true);
$books[] = "Book about Perl"; /* 13-калит(индекс) билан янги элемент қўшилди,
 бу қуйидагига эквивалент $books[13] = "Book about Perl";
$books["lisp"] = 123456; /* Бу массивга янги "lisp" калитли 123456 қиймали янги
 элемент қўшиш*/
unset($books[12]); // Бу 12-калитли элементни массивдан ўчириш
unset ($books); // массивни бутунлай ўчириш
?>
```

Бўш квадрат қавсдан фойдаланганда калитнинг максимал қиймати массивда мавжуд охирги қайта индексланган калитлар орасидан қидирилади. Массивни ***array_values***() функцияси ёрдамида ***қайта индекслаш*** мумкин.

Мисол. Массивни қайта индекслаймиз.

```
<?php
$arr =
 array ("a","b","c");
/* "a", "b" ва "c" қийматли массивни яратамиз. Бу ерда калит кўрсатилмаган бироқ мос равишда улар 0,1,2 бўлади. */
print_r($arr); // массивни экранга чиқарамиз (калити ва қийматини)
unset($arr[0]);
unset($arr[1]);
unset($arr[2]);
// массивдан ҳамма элементини ўчирамиз
print_r($arr);  // массивни экранга чиқарамиз (калити ва қийматини)
$arr[] = "aa";   // массивга янги элемент қўшамиз. Уни индекси(калити) 3 бўлади, 0 эмас.
print_r($arr);
$arr =
array_values($arr); // массивни қайта индекслаймиз.
$arr[] = "bb"; // бу элементни калити 1 бўлади.
print_r($arr);
?>
```

Бу скриптнинг натижаси қуйидагича бўлади:

```
Array ( [0] => a [1] => b [2] => c )
Array ( )
Array ( [3] => aa )
Array ( [0] => aa [1] => bb )
```

4. Массивлар инициализатсияси

PHP да массивларни инициализатсия қилишнинг 2 усули мавжуд. Биринчиси массив элементларига қиймат беришдан иборат:

```
<?
```

```
  $car[] = "passenger car";
  $car[] = "land-rover";
  echo($car[1]); // чиқаради"land-rover"
?>
```

Массив индексини очиқ кўрсатиш мумкин:

```
<?
  $car[0] = "passenger car";
  $car[1] = "land-rover";
  echo($car[1]); // чиқаради"land-rover"
?>
```

Агар массив элементларини эълон қилишда ошкора индексацияли ва индексациясиз ўзгарувчилар аралашиб келса индекси берилмаган элементга ишлатилган индекслар ичида энг каттасидан кейин келувчи рухсат берилган индексни беради. Масалан агар биз яратган массив элементлдар индекслари 10, 20 ва 30 бўлса ва индекс кўрсатмасдан янги элемент яратсак, унинг индекси автоматик равишда 31 бўлади:

```
<?
  $car[10] = "passenger car";
  $car[20] = "land-rover";
  $car[30] = "station-wagon";
  $car[] = "victoria";
```

```
  echo($car[31]);
?>
```

Алтернатив усул **array**() конструктсиясидан фойдаланишдан иборат:

```
<?
  $car = array("passenger car","land-rover");
  echo($car[1]); // чиқаради "land-rover"
?>
```

Индексларни ошкора кўрсатиш учун => оператор қўлланади:

```
<?
  $car = array("passenger car", 5 => "land-rover",
    "station-wagon","victoria");
  echo($car[0]); echo("<br>"); // чиқаради "passenger car"
  echo($car[5]); echo("<br>"); // чиқаради "land-rover"
  echo($car[6]); echo("<br>"); // чиқаради "station-wagon"
  echo($car[7]); // чиқаради "victoria"
?>
```

Массив индекслари сатрлар ҳам бўлиши мумкин:

```
<?
  $car = array("pc" => "passenger car", "lr" => "land-rover");
  echo($car["lr"]); echo("<br>"); // чиқаради "land-rover"
  echo($car["pc"]);  // чиқаради "passenger car"
?>
```

5. Массивларни кўриб чиқиш учун foreach цикли

PHP4 да массив элементларини кўриб чиқиш учун **foreach** операторидан фойдаланиш мумкин. Бу оператор синтаксиси:

```
foreach (array as [$key =>] $value)
{
  statements;
}
```

Бу цикл маъноси содда: ҳар бир элемент кўрилганда унинг индекси **$key** ўзгарувчига, қиймати бўлса **$value** ўзгарувчига жойлаштирилади. Бу икки ўзгарувчиларнинг номлари ихтиёрийдир.

Мисол:

```
<?
  $car = array("passenger car", "land-rover",
```

```
    "station-wagon","victoria");
  foreach($car as $index => $val)
  {
    echo("$index -> $val <br>");
  }
?>
```

Синтаксисдан кўриниб турибдики, **$key** ўзгарувчидан фойдаланиш шарт эмас, шунинг учун ташлаб юборилиши мумкин:

```
<?
  echo(
    "available cars: <br> <ul>"
  );
  $car = array("passenger car", "land-rover",
    "station-wagon","victoria");
  foreach($car as $val)
  {
    echo("<li>$val</li>\n");
  }
  echo("</ul>");
?>
```

6. Кўп ўлчовли массивлар

Кўп ўлчовли массивларни кўриб чиқиш учун ички жойлашган **array**()концруктсиясидан фойдаланилади. Кўп ўлчовли массивларни ўқиб чиқиш жойланган цикллар ёрдамида амалгам оширилади. Қуйидаги скриптда кўп ўлчовли массив яратиш ва кўриб чиқиш кўрсатилган.

Мисол:

```
<?
  $ship = array(
    "Passenger ship" => array("Yacht","Liner","Ferry"),
    "War ship" => array("Battle-wagon","Submarine","Cruiser"),
    "Freight ship" => array("Tank vessel","Dry-cargo ship","Container
    cargo ship")
  );
  foreach($ship as $key => $type)
  {
    echo(
    "<h2>$key</h2>\n"."<ul>\n");
    foreach($type as $ship)
    {
      echo("\t<li>$ship</li>\n");
    }
```

```
  }
  echo("</ul>\n");
?>
```

Бу скрипт бажарилиш натижаси:

Passenger ship

Yacht

Liner

Ferry

War ship

Battle-wagon

Submarine

Cruiser

Freight ship

Tank vessel

Dry-cargo ship

Container cargo ship

Энди PHP да мавжуд массивлар билан ишлаш функцияларини кўрамиз. Биз массивларни тартиблаш функциялардан бошлаймиз. Лекин аввал мисолларимизда кўп фойдаланадиган учта функцияни кўриб чиқамиз.

1. Функция count()

Синтаксис:

int **count(**mixed **var)**

Бу функция аргумент сифатида массивни қабул қилиб, ундаги элементлар сонини қайтаради.

2. Функция in_array()
Синтаксис:

boolean **in_array(**mixed **needle**, array **haystack** [, bool **strict**]**)**

Бу функция **haystack** массивда **needle** қийматни қидиради ва агар у мавжуд бўлса **true** қайтаради, акс холда **false** қайтаради.

3. Функция reset()
Синтаксис:

mixed **reset(**array **array)**

Функция **reset**() массив кўрсаткичини биринчи элементга ўрнатади ва массив биринчи элементи қийматини қайтаради.

Кичкина изох. PHP да ҳар бир массив жорий элементга кўрсаткичга эга. Қуйидаги **foreach** каби концруктсиялар билан ишлашда кўрсаткич ҳақида ўйлаш керак эмас, чунки **foreach** уни массив бошига ўрнатади. Лекин бошқа массивлар функциялари масалан **prev**(), **next**(), массив кўрсаткичларини суради, бу эса **array_walk**(),каби қайта ишлашни кўрсаткични турган жойидан бошловчи функциялар учун катта аҳамиятга эга.

Энди тартиблаш билан шуғулланамиз.

7. Массивларни тартиблаш функциялари

Қуйидаги тартиблаш функциялари мавжуд:

- **sort()**

Массивни ўсиш бўйича тартиблаш функцияси.

Синтаксис:

void **sort(**array **array** [, int **sort_flags**]**)**

Функция **array** массивини ўсиш бўйича тартиблайди. Мажбурий бўлмаган элемент **sort_flags** элементлар қандай тартибланиши кераклигини кўрсатади(тартиблаш байроқларини белгилайди). Аргументнинг мумкин бўлган қийматлари қуйидагилар:

- ✓ SORT_REGULAR – элементлар нормал солдиштиради нормалное сравнение элементов (элементларни "борича" солиштиради);
- ✓ SORT_NUMERIC – элементларни сонлар сифатида солиштиради;
- ✓ SORT_STRING - элементларни сатрлар сифатида алмаштиради;

Умуман олганда бу функция рўйхатларни тартиблаш учун мўлжалланган. Рўйхат деганда калитлари нулдан бошланган ва бўшликларга эга бўлмаган массив тушунилади. Функция **sort**() ихтиёрий массивни рўйхат деб қарайди.

Мисол:

```
<?
  $arr = array("2", "1", "4", "3","5");
  sort($arr);
```

```
  for($i=0; $i < count($arr); $i++)
  {
    echo ("$i:$arr[$i]  ");
  }
  // чиқаради "0:1 1:2 2:3 3:4 4:5"
?>
```

Натижа:

0:1 1:2 2:3 3:4 4:5

Агар сиз сатрларни тартилаётган бўлсангиз, мисол учун массив қуйидаги кўринишга эга бўлса

```
  array("one", "two", "abs", "three", "uic", "for","five");
```

Бу ажойиб функция қуйидаги натижани қайтаради:

Натижа:

0:abs 1:five 2:for 3:one 4:three 5:two 6:uic

Яъни сатрларни у алфа-бета тартибда, соддароқ айтганда биринчи ҳарфлари алфавитда келиши бўйича тартиблайди.

- **rsort()**

Массивларни камайиш бўйича тартиблаш.

Синтаксис:

void **rsort(**array **arr** [, int **sort_flags**]**)**

Шунга ўхшаш **sort**() функцияси фақат камайиш бўйича тартиблайди. Олдинги **sort**() функцияси учун кўрилган скриптни оламиз, фақат сорт (**$arr**) ўрнига **rsort** (**$arr**) қўямиз.

Натижа:

0:5 1:4 2:3 3:2 4:1

- **asort()**

Ассоциатив массивни ўсиш бўйича тартиблаш.

Синтаксис:

void **asort(**array **arr** [, int **sort_flags**]**)**

Функция **asort**() берилган **arr** массивни шундай тартиблайдики унинг қийматлари алфавит тартибида (агар сатр бўлса) ёки ўсиш тартибда (сонлар учун) тартибда жойлашади. Бу функциянинг **sort**()функциясидан мухим фарқи шундаки **asort**() функцияси қўлланилганда калитлар ва уларга мос қийматлар орасида боғликлик сақланади, **sort**() функциясида бўлса бу боғлиқлик узилади.

Мисол:

```
<?
  $arr = array("a" =>"one","b" => "two","c" => "three","d" => "four");
  asort($arr);
```

```
  foreach($arr as $key => $val)
  {
    echo (" $key => $val ");
  }
?>
```

Натижа:

d => four a => one c => three b => two

Кўриниб турибдики "калит-қиймат" боғланишлари сақланиб қолган.

- **arsort()**

Ассотсиатив массивларни камайиш бўйича тартиблаш.

Синтаксис:

void **arsort(**array **arr** [, int **sort_flags**]**)**

Бу функция **asort**() функциясига ўхшаш, фақат у масивни ўсиш бўйича эмас камайиш бўйича тартиблайди.

- **ksort()**

Массивларни калит ўсиши бўйича тартиблаш.

Синтаксис:

int **ksort(**array **arr** [, int **sort_flags**]**)**

Бу функцияда тартиблаш қийматлар бўйича эмас, балки калитлар бўйича ўсиш тартибида амалга оширилади.

```
<?
  $arr = array("a" =>"one","b" => "two","c" => "three","d" => "four");
  ksort($arr);
  foreach($arr as $key => $val)
  {
    echo (" $key => $val ");
  }
?>
```

Натижа:

a => one b => two c => three d => four

- **krsort()**

Индекслар камайиши бўйича массивларни тартиблаш.

Синтаксис:

int **krsort(**array **arr** [, int **sort_flags**]**)**

Худди **ksort**() функцияга ўхшаш, фақат массивни калитлар бўйича тескари тартибда (камайиш бўйича) тартиблайди.

- **array_reverse()**

Массив элементларини тескари жойлаштириш.

Синтаксис:

array **array_reverse(**array **arr** [, bool **preserve_keys**]**)**

Функция **array_reverse**() элементлари параметрда берилган **arr** массиви элементларига нисбатан тескари жойлаштирилган массивни қайтаради. Калитлар ва қийматлар орасидаги боғланиш сақланиб қолади. Агар мажбурий бўлмаган параметр **preserve_keys** га **true** берилса, калитлар ҳам тескари тартибда жойлашади.

Мисол:

```
<?
  $arr = array ("php", 4.0, array ("green", "red"));
  $result = array_reverse ($arr);
  echo "Массив: <br>";
  foreach($result as $key => $val)
  {
    echo ("$key => $val <br>");
  }
  echo("<br>");
  echo "Тартибланган массив: <br>";
  $result_keed = array_reverse ($arr, false);
  foreach($result_keed as $key => $val)
```

```
    {
      echo ("$key => $val<br> ");
    }
  ?>
```

Биринчи холда:

Натижа:

Массив:

0 =>Array
1 =>4
2 =>php

Тартибланган массив:

0 =>Array
1 =>4
2 =>php

Агар иккинчи параметрга **true** қиймат берилса:

Натижа:

Массив:

0 =>Array
1 =>4
2 =>php

Тартибланган массив:

2 =>Array

1 =>4
0 =>php

- **shuffle()**

Массив элементларини тасодифий жойлаштириш.

Синтаксис:

void **shuffle(**array **arr)**

shuffle() функцияси **arr** массиви элементларини тасодифий аралаштиради.

- **natsort()**

Табиий тартиблашни бажаради.

Синтаксис:

void **natsort(**array **arr)**

Бундай тартиблашни сатрларни тартиблашда учратган эдик. Табиий тартиблаш деб элементлар тушунарли тартибда жойлашга айтилади.

Мисол:

```
<?
  $array1 = $array2 = array("pict10.gif", "pict2.gif", "pict20.gif", "pict1.gif");
  echo ("оддий тартиблаш:"); echo ("<br>");
  sort($array1);
  print_r($array1);
```

```
  echo ("<br>"); echo ("табиий тартиблаш:"); echo ("<br>");
  natsort($array2);
  print_r($array2);
?>
```

Натижа:

оддий тартиблаш:

```
Array ( [0] => pict1.gif [1] => pict10.gif [2] => pict2.gif [3] => pict20.gif )
ецецвенная сортировка:
Array ( [3] => pict1.gif [1] => pict2.gif [0] => pict10.gif [2] => pict20.gif )
```

Назорат саволлари:

1. PHP дастурлаш тилида қандай типларни биласиз?
2. PHP дастурлаш тилида сатр типини тушунтириб беринг?
3. PHP дастурлаш тилида сатрлар қандай аниқланади?
4. PHP да массивлар тушунтириб беринг?
5. Массив типи қандай аниқланади?
6. Массивларн иинитсиализатсия қилишнинг қандай усулларини биласиз?

7.Қандай тартиблаш функцияларини биласиз?

8.Турли тартиблаш функцияларини ишлашга доир мисоллар билан тушунтиринг?

9. Кўп ўлчовли массивлар ҳақида маълумот беринг?

10. Foreach циклининг ишлаш моҳиятини тушунтиринг?

Фойдаланилган адабиётлар:

17. Холзнер С. Перл: специалний справочник. - СПб: "Питер". 2000. - 496 с.
18. Шварс Р., Кристиансен Т. Изучаем Перл. - К: "БҲВ", 2000. - 320 с.
19. Ратшиллер Т., Геркен Т. PHP4: разработка Web-приложений.- СПб:Питер, 2001. - 384 с.
20. Томсон Л., Веллинг Л. Разработка Web-приложений на PHP и MySQL. - К.: "DiaSoft", 2001. - 672 с.
21. Основи современних компютерних технологий. Ред. Хомченко А.Д.

22. Савелев А.Я., Сазонов Б.А., Лукянов Б.А. Персоналний компютер для всех. Хранение и обработка информатсии. Т.1 М.: Висшая школа, 1991.
23. Брябрин В.М. Программное обеспечение персоналних ЭВМ. М.: Наука, 1990.
24. Фролов А.В., Фролов Г.В. Глобалнийе сети компютеров. Практическое введение в Интернет, Э-Маил, FTP, WWW и HTML. М.:Диалог-МИФИ, 1996.

Фойдаланилган манбалар:

11. Intuit.ru.
12. http://pda.coolreferat.com
13. http://www.google.ru
14. http://www.z-oleg.com
15. http://www.i2r.ru/

12-маъруза. Функциялар. Класслар ва объектлар. Хатоликлар билан ишлаш. PHP функциялар.

Режа:

1. Функциялар;

2. Функцияларнинг аргументлари;

3. Ўзгарувчан узунлик аргументлари рўйхатлари;

4. Функциялар ичида ўзгарувчилардан фойдаланиш;

5. Функциянинг ўзгарувчилари;

6. Ички жойлашган (ичма-ич) функциялар;

7. Синфлар ва объектлар;

8. Ўзгарувчиларни инициаллаштириш;

9. Объектлар;

Калит сўзлар: Функсиялар, аргументлар, рўйҳатлар, ўзгарувчилар, ичма-ич функсиялар, класслар ва объектлар, хатоликлар, ўзгарувчиларни инициаллаштириш.

Ишдан мақсад: Талабаларга функциялар, класслар ва объектлар, функцияларнинг аргументлари ҳақида маълумот бериш ва улар билан ишлаш кўникмаларини ўргатиш.

1. Функциялар

Функциялар нима учун керак? Бу саволга жавоб бериш учун, функция ўзи нима эканлигини тушуниб олиш лозим бўлади. Дастурлашда, худди математикадаги каби, унга боғлиқ кўпгина аргументларнинг унинг кўпгина маъноларида акс этишидир. Демак, функция аргументнинг ҳар бир маънолари жамланмаси учун унинг бажарган иши натижаси сифатида қандайдир маъно

қайтаради. Функциялар нима учун керак, буни мисоллар билан ойдинлаштиришга ҳаракат қиламиз. Дастурлашдаги функцияга классик мисол – бу соннинг факториал аҳамиятини ҳисоблаб берувчи функция. Демак, биз унга сон берамиз, у эса бизга унинг факториалини қайтаради. Бунда биз факториалини олишни хоҳлаган ҳар бир сон учун айнан бир хил кодни қайтаравермаймиз – бу сонга тенг бўлган аргументли функцияни чақиришнинг ўзи кифоя қилади.

Натурал сон факториалини ҳисоблаш функцияси

Мисол:

```
<?php
function fact($n){
if ($n==0) return 1;
else return $fact = $n * fact($n-1);
}
echo fact(3);
// echo (3*2) деб ёзиш мумкин эди; лекин сон катта бўлса,
echo fact(50);
// функциядан фойдаланиш қулайроқ, echo (50*49*48*...*3*2) деб ёзгандан;
?>
```

Шу йўл билан биз бирон-бир маълумотга боғлиқлик зарурияти туғилган амални бажарганимизда, бу ҳолда ҳам биз айнан шундай амалларни бажаришимиз оширишимиз лозим бўлади, фақат бошқа бошланғич маълумотлардан фойдаланамиз, функциялар механизмидан фойдаланиш–

функция танаси кўринишидаги амаллар блокини тахт қилиш, ўзгарувчан маълумотларни эса – унинг параметрлари сифатида фойдаланиш қулайроқ бўлади.

Функция топшириғи (эълони) умумий тарзда қандай бўлишини кўрамиз. Функция қуйидаги синтаксис ёрдамида аниқланади:

```
function  Функция_номи (1-параметр, 2-параметр,  ... N-параметр) {
  Амаллар блоки
  return "функцияга айланувчи маъно";
}
```

Агар php-дастурда тўғридан-тўғри ёзилса, ҳеч нарсани ишлаб бўлмайди. Биринчидан, функция номи функция параметрлари номлари (1-параметр, 2-параметр ва б.) PHP да номланиш қоидаларига мувофиқ келиши керак (унда яхшиси кириллча символларни ҳам ишлатмаган маъқул). Функция номлари регистрга нисбатан сезувчан бўлади. Иккинчидан, функция параметрлари – тилнинг ўзгарувчан қисмлари, шунинг учун уларнинг ҳар бирининг номлари олдидан $ белгиси туриши лозим бўлади. Параметрлар рўйхатида ҳеч қандай кўп нуқталарни қўйиш мумкин эмас. Учинчидан, амаллар блоки сўзи билан бирга функция танасида исталган тўғри PHP-код мавжуд бўлиши керак (параметрларга мувофиқ бўлиши мажбурий эмас). Ва ниҳоят, return калит сўзидан сўнг тартибли php-ифода келиши лозим (маънога эга бўлган қандайдир символлар). Бундан ташқари, функцияда қайтарилувчи маъно каби параметрлар бўлмаслиги ҳам мумкин. Функцияни тўғри эълон қилишга мисол – юқорида келтирилган факториални ҳисоблаш функцияси.

Функция чақириш қандай амалга ошади? Функция номи ва юмалоқ қавслар ичида унинг параметрлари маънолари рўйхати кўрсатилади, агар шундайлари мавжуд бўлса:

Мисол:

```
<?php
  Функция_номи ("1-параметр_учун_маъно", "2-параметр_учун_маъно ",...);
/* Функцияни чақиришга мисол – функцияни чақириш факториални ҳисоблаш юқорида бор, 3 сони факториалини ҳисоблаш учун у ерда биз fact(3) деб ёзганмиз;  у ерда fact – чақирилувчи функция номи,  а 3 –$n номли унинг параметри маъноси */
?>
```

Функцияни қачон чақириш мумкин? Бу ғалати савол бўлиб туюлиши мумкин. Функцияни уни аниқлангандан кейин чақириш мумкин, яъни function f_name(){...} блокидан пастда исталган дастур қаторида. PHP3 да бу айнан шундай. Лекин PHP4 да бундай талаб йўқ. Ҳамма гап интерпретатор олинган кодни қандай қайта ишлашида. Биргина истисно шартли равишда аниқланадиган функциядан ташкил топади (шартли операторлар ёки бошқа функциялар ичида). Функция шу тарзда аниқланган тақдирда, уни аниқлаш уни чақиришдан олдин бажарилади.

Мисол. Шартли функция ичида функцияни аниқлаш

```
<?
$make = true;
/* бу ерда Make_event() ни чақириш мумкин эмас; Чунки у ҳали мавжуд эмас, лекин Save_info() ни чақириш мумкин */
Save_info("Собир","Содиқов", "Мен PHP курсини танладим");
if ($make){
// Make_event() функциясини аниқлаш
```

```
function Make_event(){
echo "<p> Python<br> ни ўрганмоқчиман";
}
}
// энди Make_event() ни чақириш мумкин
Make_event();
// Save_info функциясини аниқланади
function Save_info($first, $last, $message){
echo "<br>$message<br>";
echo "Исм: ". $first . " ". $last . "<br>";
}
Save_info("Мурод","Ёқубов", "Мен Lisp ни танладим");
// Save_info ни бу ерда ҳам чақириш мумкин
?>
```

Агар функция дастур ичида аниқланган бўлса, уни кейин қайта аниқлаш ёки ўчириб ташлаш мумкин эмас. Функция номларига регистр таъсир қилмаслигига қарамасдан, яхшиси функцияни аниқлаш пайтида берилган ном билан чақириш мумкин бўлади.

Мисол. Функция ичидаги функцияни аниқлаш

```
<?php
/* маълумотларни сақлаш, яъни DataSave() функциясини чақириш мумкин эмас. Унинг тўғрилиги текширилмасдан олдин, яъни DataCheck() функцияси чақирилмасдан олдин бу мумкин эмас.*/
```

```
DataCheck();

DataSave();

function DataCheck(){

// маълумотлар тўғрилигини текшириш

  function DataSave(){

  // маълумотларни сақлаймиз

} } ?>
```

Функция аргументлари, уларнинг маънолари ва ишлатилишини батафсил кўриб чиқамиз.

2. Функцияларнинг аргументлари

Ҳар бир функцияда, аввал айтганимиздай, аргументлар рўйхати бўлиши мумкин. Бу аргументлар ёрдамида функцияга ҳар хил маълумотлар берилади (масалан, факториали ҳисобланиши керак бўлгансон маъноси). Ҳар бир аргумент ўзгарувчи ва константага эга бўлади.

Аргументлар ёрдамида маълумотлар функцияга уч хил турли усуллар билан ўтказилиши мумкин. Бу аргументларни маъносига кўра (ўзгармас ҳолатда фойдаланилади), иловаларга кўра ва ўзгармас ҳолатда аргументларга маъно беришга кўра ўтказиш . Бу усулларни атрофлича кўриб чиқамиз.

Аргумент функцияга маъносига кўра ўтказилса, функция ичидаги аргумент маъносининг ўзгариши унинг функция ташқарисидаги маъносига таъсир қилмайди. Функцияга унинг аргументларини ўзгартиришга йўл қўйиш учун уларни ҳаволаларга кўра ўтказиш керак. Бунинг учун аргумент номи олдидан функцияни аниқлашда ампенсанд “&” белгисини ёзиш керак.

Мисол. Аргументларни ҳаволасига кўра ўтказиш

```
<?php
// қўшимча қилиши мумкин бўлган функцияни ёзамиз checked сўзи қаторига
function add_label(&$data_str){
  $data_str .= "checked";
}
$str = "<input type=radio name=article ";
  // бундай қатор мавжуд бўлсин
echo $str .="><br>";
// форма элементини келтиради – белгиланмаган радио кнопкасини
add_label($str);
    // функцияни чақирамиз
echo $str .="><br>";
    // бу энди белгиланган  радио кнопкани келтиради
?>
```

Функцияда тинч ҳолатда фойдаланилаётган аргументлар маъносини аниқлаш мумкин. Айни пайтдаги маънонинг ўзи константа ифода бўлиши, ўзгартириш ва синф вакили ёки бошқа функция чақируви бўлмаслиги лозим.

Бизда информацион хабар тузувчи функция, унга берилган параметр маъносига мувофиқ тарзда ўзгарувчи имзо бор. Агар параметр маъноси берилмаган бўлса, “Ташкилий қўмита” имзосидан фойдаланилади.

Мисол. Тинч ҳолатдаги аргумент маъноси

```
<?php
function Message($sign="Таш.қўмита."){
// бу ерда параметр sign айни пайтда “Таш.қўмита” маъносига эга
  echo "Кейинги йиғилиш эртага бўлиб ўтади.<br>";
  echo "$sign<br>";
}
Message();
   // Параметрсиз функцияни чақирамиз. Бу ҳолда имзо – Бу Ташкилий қўмита
Message("Ҳурмат билан Камолиддин");
   // Бу ҳолда имзо "Ҳурмат билан Камолиддин." бўлади
?>
```

Бу скрипт ишининг натижаси қуйидагича:

Кейинги йиғилиш эртага бўлиб ўтади.

Ташкилий қўмита.

Кейинги йиғилиш эртага бўлиб ўтади.

Ҳурмат билан Камолиддин.

Агар функциянинг бир неча параметрлари бўлса, тинч ҳолатда маъно берилувчи бу аргументлар функция аниқланишида бошқа барча аргументлардан

кейин ёзилиши керак. Акс ҳолда, агар бу аргументлар функцияни чиқариш пайтида кўздан қочирилса хато юзага келиши эҳтимоли бор.

Масалан, биз каталогга мақола тавсифини киритмоқчимиз. Фойдаланувчи мақолага унинг номланиши, муаллифи ва қисқа тавсиф каби характеристикаларни келтириши лозим бўлади. Агар Фойдаланувчи мақола муаллифи исмини киритмади, у Мурод Ёқубов деб олайлик.

```
<?php
function Add_article($title, $description, $author="Мурод Ёқубов"){
echo "Мақолани каталогга киритамиз: $title,";
echo "муаллиф $author";
echo "<br>Қисқа тавсиф: ";
echo "$description <hr>";
}

Add_article("Информатика ва биз","Бу мақола информатикага оид ...", "Зайниддин Саидов");
Add_article("Характерлар ким", "Бу мақола характерлар ҳақида ...");
?>
```

Скрипт иши натижаси сифатида қуйидагиларни оламиз

Каталогга мақола киритамиз:

Информатика ва биз,

Муаллиф Мурод Ёқубов.

Қисқа тавсиф:

Бу мақола информатикага оид...

Каталогга мақола киритамиз:

Характерлар ким,

Муаллиф Одил Зияев.

Қисқа тавсиф:

Бу мақола характерлар ҳақида...

Агар биз қуйидагича ёзсак:

```
<?php
function Add_article($author="Одил Зияев", $title, $description){
// ... аввалги мисолдаги каби амал
}
Add_article("Характерлар ким", "Бу мақола характерлар ҳақида...");
?>
```

Натижа қуйидагича бўлади:

```
Warning: Missing argument 3 for add_article() in c:\users\nina\tasks\func\def_bad.php on line 2
```

3. Ўзгарувчан узунлик аргументлари рўйхатлари

PHP4 да аргументларнинг ўзгарувчан сони билан функция тузиш мумкин. Яъни биз уни неча аргументлар билан чақирилишини билмасдан, функция тузамиз. Бу каби функция ёзиш учун ҳеч қандай махсус синтаксис керак бўлмайди. Ҳаммаси унинг ичига ўрнатилган функциялар func_num_args(), func_get_arg(), func_get_args() ёрдами билан қилинади.

func_num_args() функцияси аргументлар сонини қайтаради. Бу функция фақат фойдаланувчи функциясини аниқлаш мобайнида фойдаланиши мумкин. Агар у функциядан ташқарида пайдо бўлса, интерпретатор огоҳлантириш беради.

Мисол. func_num_args() функциясидан фойдаланиш

```
<?php
function DataCheck() {
  $n = func_num_args();
  echo "Функция аргументлари сони $n";
}
DataCheck();
// қаторни келтиради "0 функция аргументлари сони"
DataCheck(1,2,3);
// қаторни келтиради "3-функция аргументлари сони"
?>
```

func_get_arg функцияси (аргумент_рақами тўлалигича) аргументни ўзгаришлар рўйхатидан аргументлар функциясига қайтаради, унинг тартиб рақами func_get_arg параметри билан берилади. Функция аргументлари нолдан бошлаб ҳисобланади. **func_num_args**() каби бу функция фақат бирон-бир функцияни аниқлашда фойдаланилади.

Аргумент рақами функцияга ўзгарган аргументлар сонидан ортиб кетиши мумкин эмас. Акс ҳолда огоҳлантириш умумлаштирилади ва func_num_args() функциясига False қиймат қайтади.

Маълумотларни текшириш учун функцияга унинг аргументларини тузамиз. Агар функциянинг биринчи аргументи – бутун сон, иккинчиси – қатор бўлса, текшириш муваффақиятли ўтди, деб ҳисоблаймиз.

Мисол. Маълумотлар типини, унинг аргументларини текшириш

```
<?
function DataCheck(){
  $check =true;
  $n = func_num_args();
/* функцияга ўзгарган аргументлар сонини текширамиз, биринчи ўзгарган аргумент бутун сонми-йўқми */
  if ($n>=1) if (!is_int(func_get_arg(0)))
     $check = false;
/* текширамиз, иккинчи ўзгарган аргумент қаторми-йўқми */
  if ($n>=2)
    if (!is_string(func_get_arg(1)))
```

```
        $check = false;
    return $check;
    }
    if (DataCheck(123,"text"))
      echo "Текширув тўғри ўтди<br>";
    else echo "маълумотлар шартларни қондирмайди <br>";
    if (DataCheck(324))
    echo "Текширув тўғри ўтди<br>";
    else echo "маълумотлар шартларни қондирмайди <br>";
    ?>
```

Дастур натижаси қуйидагича бўлади:

Маълумотлар шартларни қониқтирмайди. Текширув тўғри ўтди.

func_get_args() фунцияси аргументлар рўйхатидан ташкил топган массив қайтаради. Массивнинг ҳар бир элементи аргументга, функция ўзгаришига тўғри келади. Агар функция фойдаланувчи функцияси аниқлигидан ташқарида фойдаланилса огоҳлантириш умумлаштирилади.

Аввалги мисолни кўчирамиз, бу функциядан фойдаланамиз. Функцияни ҳаракатлантирувчи жуфт аргумент бутун сон эканлигини текширамиз:

Мисол:

```
<?
```

```
function DataCheck(){
  $check =true;
  $n = func_num_args();
  // функцияга ўзгарган аргументлар сони
  $args = func_get_args();
  // функция аргументлари массиви
  for ($i=0;$i<$n;$i++){
    $v = $args[$i];
    if ($i % 2 == 0){
      if (!is_int($v)) $check = false;
// текширамиз, жуфт аргумент бутунми-йўқми
}
}
return $check;
}
if (DataCheck(array("text", 324)))
   echo "Текширув тўғри ўтди<br>";
else echo "Маълумотлар шартларни қониқтирмайди <br>";
?>
```

Бундан, func_num_args(), func_get_arg() ва func_get_args() функция комбинацияси функциялар ўзгарувчан аргументлар рўйхатига эга бўла олиши учун фойдаланилади. Бу функциялар фақат PHP4 га киритилган. PHP3 да шундай натижага эришиш учун, аргумент сифатида массив функциясидан фойдаланиш

мумкин бўлади. **Масалан**, ҳар бир тоқ функциялар параметри бутун сонлигини текширувчи скриптни қуйидагича ёзиш мумкин:

```
<?
function DataCheck($params){
  $check =true;
$n = count($params);
// функцияга ўзгарган аргументлар сони
  for ($i=0;$i<$n;$i++){
    $v = $params[$i];
    if ($i % 2 !== 0){
// текширамиз, тоқ аргумент бутунми-йўқми
if (!is_int($v)) $check = false;
    }
  }
return $check;
}
if (DataCheck("text", 324))
    echo "Текширув тўғри ўтди<br>";
else echo "Маълумотлар шартларни қониқтирмайди<br>";
?>
```

4. Функциялар ичида ўзгарувчилардан фойдаланиш

1. Глобал ўзгарувчилар

2. Статистик ўзгарувчилар

3. Қайтарилувчан маънолар

4. Ҳаволани қайтариш

Глобал ўзгарувчилар

Функциялар ичида ундан ташқарида берилган ўзгарувчилардан фойдаланиш учун, бу ўзгарувчиларни глобал деб эълон қилиш керак. Бунинг учун функция танасида унинг номларини global калит сўзидан кейин келтириш лозим бўлади:

```
global $var1, $var2;
```

```
<?
$a=1;
function Test_g(){
global $a;
  $a = $a*2;
  echo ' $a=',$a функция ишида натижа;
}
echo 'функциядан ташқарида $a=',$a,', ';
Test_g();
echo "<br>";
echo функциядан ташқарида $a=',$a,', ';
Test_g();
```

?>

Мисол. Глобал ўзгарувчилар

Бу скрипт ишидан қуйидаги натижаларни оламиз:

$a=2 функциядан ташқарида, $=2 функция ишида натижа

$a=2 функциядан ташқарида, $=4 функция ишида натижа

Ўзгарувчи глобал деб эълон қилинганда, аниқ глобал ўзгарувчи учун ҳавола тузилади. Бунинг учун бундай ёзув қуйидагига эквивалент (GLOBALS массиви мавжуд кўриниш соҳаларига мувофиқ барча глобал ўзгарувчиларни ўз ичига олади):

$var1 = & $GLOBALS["var1"];

$var2 = & $GLOBALS["var2"];

Бундан келиб чиқадики, $var1 ўзгарувчини ўчириш $_GLOBALS["var1"] глобал ўзгарувчиини ўчириб ташламайди.

Статистик ўзгарувчилар

Ўзгарувчилардан фақат функция ичида фойдаланиш учун бунда унинг маъносини сақлаган ҳолда ва функциядан чиққандан сўнг, бу ўзгарувчиларни статистик деб эълон қилиш керак. Статистик ўзгарувчилар фақат функциялар ичида кўринади ва дастурни юклаш функция доирасидан ташқарига чиқса ўз маъносини йўқотмайди. Бу ўзгарувчиларни эълон қилиш static калит сўзи ёрдамида амалга оширилади:

```
static $var1, $var2;
```

Ҳар қандай маъно статистик ўзгарувчи сифатида талқин қилиниши мумкин, фақат ҳавола эмас.

Мисол. Статистик ўзгарувчилардан фойдаланиш

```
<?
function Test_s(){
static $a = 1;
// ифода ёки ҳаволани ўзлаштириб бўлмайди
  $a = $a*2;
  echo $a;
}
Test_s(); // 2 чиқади
echo $a;  // ҳеч нарса чиқмайди, зеро $a фақат функция ичида кириш йўлаги бор
Test_s(); // $a=2 функция ичида, шунинг учун функция иши натижаси 4 сони бўлади
?>
```

Қайтарилувчан маънолар

Юқорида мисол қилиб келтирилган барча функциялар бирор-бир амал бажаришган. Бундай ҳоллардан ташқари, ҳар қандай функция ўз иши натижаси сифатида қандайдир қиймат қайтаради. Бу return тасдиғи ёрдамида қилинади.

Қайтарилувчан қиймат ҳар қандай турда, шу жумладан, рўйхат ва объектлар бўлиши мумкин. Интерпретатор функция танасида return командасига учраганда, у дарҳол уни бажаришни тўхтатади ва функция чақирилган қаторга ўтиб кетади.

Масалан, инсон ёшини қайтарувчи функция тузамиз. Агар инсон вафот этмаган бўлса, ёш жорий йилга мувофиқ ҳисобланади.

```
<?php
/* агар иккинчи параметр true каби ҳисоблаб чиқилса, у вафот этган санадай кўриб чиқилади, */
function Age($birth, $is_dead){
  if ($is_dead) return $is_dead-$birth;
  else return date("Y")-$birth;
}
echo Age(1971, false); // выведет 33
echo Age(1971, 2001); // выведет 30
?>
```

Бу мисолда return функциясидан фойдаланмаса ҳам бўлади, шунчаки уни чиқариш функциясини echo га алмаштирилади. Аксинча, агар биз функция бирор-бир қиймат қайтарадиган қилсак (бу мисолда инсон ёши), биз дастурда ўзгарувчини бу функция қийматини исталган ўзгарувчига ўзлаштиришимиз мумкин.

```
$my_age = Age(1981, 2004);
```

Функция иши натижасида фақат битта қиймат қайтарилиши мумкин. Бир неча қийматни қийматлар рўйхати қайтарилган тақдирда олиш мумкин (бир ўлчамли массив). Биз инсон ёшини кунигача аниқликда олмоқчимиз, деб ҳисоблайлик.

```
<?php
function Full_age($b_day, $b_month, $b_year)
{
	$y = date("Y");
	$m = intval(date("m"));
	$d = intval(date("d"));
	$b_month = intval($b_month);
	$b_day = intval($b_day);
	$b_year = intval($b_year);

	$day = ($b_day > $d ? 30 - $b_day + $d : $d - $b_day);
	$tmpMonth = ($b_day > $d ?  -1 : 0);
	$month = ($b_month > $m + $tmpMonth
	 ? $b_month + $tmpMonth - $m : $m+$tmpMonth - $b_month);
	$tmpYear  = ($b_month >  $m + $tmpMonth ?  -1 : 0);
	if ($b_year > $y + $tmpYear)
	{
		$year = 0; $month = 0; $day = 0;
	}
	else
	{
		$year = $y + $tmpYear - $b_year;
```

```
        }
        return array ($day,$month,$year);
    }
    $age = Full_age("29","06","1986");
    echo "Сиз $age[2] ёш, $age[1] ойлар ва $age[0] кунлар";
    ?>
```

Функция бир неча қийматларни уларни дастурда қайта ишлаш учун қайтарганда, бир амал билан маънони бирданига бир неча ўзгарувчиларни ўзлаштиришга имкон берувчи list() тил конструкциясидан фойдаланиш қулай бўлади. Масалан, юқоридаги мисолда функцияни, унинг қийматига ўзгартириш киритмай қайта ишлаш қуйидагича бўлиши мумкин:

```
<?
// Full_age() функция киритиш
list($day,$month,$year) = Full_age("07",
   "08","1974");
echo "Сизнинг ёшингиз $year, $month ой ва
   $day кун";
?>
```

list() конструкциясини умуман ўзгарувчини ўзлаштириш учун исталган массив элементи қийматидан фойдаланиш мумкин.

Мисол. list() дан фойдаланиш

```
<?
$arr = array("first","second");
list($a,$b) = $arr;
  // ўзгарувчи $a ўзлаштирилади, биринчи массив қиймати, $b – иккинчи
echo $a," ",$b;
// «first second» қатори келтирилади
?>
```

Ҳаволани қайтариш

Функция ўз иши натижасида шунингдек ҳаволани бирор-бир ўзгарувчига қайтариши мумкин. Бу функцияни қандай ўзгарувчи ҳаволага ўзлаштириш кераклигини аниқлаш учун фойдаланилади. Функциядан ҳавола олиш учун, эълон олдидан амперсанд (&) белгисини ёзиш керак бўлади ва ҳар сафар функция чақируви пайтида унинг номи олдидан ҳам амперсанд (&) ёзиш керак бўлади. Кўпинча функция ҳаволани бирор-бир глобал ўзгарувчига (ёки унинг қисмини – ҳаволани глобал массив элементига), ҳаволани статистик ўзгарувчига (ёки унинг қисмини) ёки ҳаволани аргументлардан бирига қайтаради, агар у ҳавола бўйича берилган бўлса.

Мисол. Ҳаволани қайтариш

```
<?
$a = 3; $b = 2;
function & ref($par){
```

```
global $a, $b;
  if ($par % 2 == 0) return $b;
  else return $a;
}
$var =& ref(4);
echo $var, " и ", $b, "<br>"; // 2 ва 2 келтирилади
$b = 10;
echo $var, " и ", $b, "<br>";  // 10 ва10 келтирилади
?>
```

Ҳавола синтаксисидан фойдаланишда бизнинг мисолдаги $var ўзгарувчи ўзгарувчининг $b қиймати $ref қайтарилган функциясига кўчирилмайди, бу ўзгарувчига ҳавола тузилади. Демак, энди $var ва $b тенг кучли ўзгарувчилар ва улар бир пайтда ўзгартирилади.

5. Функциянинг ўзгарувчилари

PHP функциялар ўзгарувчиларига кўмаклашади. Бу дегани, агар ўзгарувчи номи оддий қавслар билан тугаса, PHP шу каби номли функцияни қидиради ва уни бажаришга ҳаракат қилади.

Мисол. Функциялар ўзгарувчиларидан фойдаланиш

```
<?
/* Иккита оддий функция тузамиз: Add_sign – қаторга имзо қўшади ва Show_text –матн қаторини чиқариб беради*/
```

```
function Add_sign($string,
  $sign="Ҳурмат билан, Мурод"){
  echo $string ." ".$sign;
}
function Show_text(){
 echo "Хабарни почтадан жўнатиш<br>";
}
$func = "Show_text"; // маънога эга ўзгарувчи тузамиз, у функция номига тенг Show_text
$func();  // у  Show_text функцияни чақиради
$func = "Add_sign";  // маънога эга ўзгарувчи тузамиз, у функция номига тенг Add_sign
$func("Ҳаммага салом <br>");
// бу функцияни чақиради Add_sign "Ҳаммага салом" параметрли
?>
```

Бу мисолда Show text функция шунчаки матн қаторини чиқаради. Агар echo махсус функцияси мавжуд бўлса, нега бунинг учун алоҳида функция тузиш керак, дейиш мумкин. Гап шундаки, echo(), print(), unset(), include() каби функциялардан функциялар ўзгарувчилари сифатида фойдаланиб бўлмайди. Яъни биз ёзсак:

```
<?
$func = "echo ";
$func("TEXT");
?>
```

Интерпретатор хатони кўрсатади:

Fatal error: Call to undefined function:

echo() in c:\users\nina\tasks\func\var_f.php on line 2

Шунинг учун юқорида келтириб ўтилган исталган функциялардан ўзгарувчилар функцияси сифатида фойдаланиш учун юқоридаги мисолдаги йўлни тутдик.

6. Ички жойлашган (ичма-ич) функциялар

Фойдаланувчи томонидан аниқланадиган функциялар ҳақида гапирганда ички жойлашган функциялар ҳақида гап кетмаслиги мумкин эмас. Юқорида биз echo(), print(), date(), include() каби ички жойлашган функциялар билан танишдик. Бундан ташқари date() функциядан бошқа барча функциялар PHP дастурлаш тили конструкциясига эга. Улар PHP дастурлаш тили ядросига жойлашган бўлиб, ҳеч қандай модуллар ва қўшимча ўзгартиришлар талаб этмайди. Аммо шундай функциялар мавжудки, уларга турли файл библиотекалари ва мос равишда модулларни юкламасдан иложи йўқ. Масалан, MySql маълумотлар базаси билан ишлайдиган функциялардан фойдаланиш учун шундай кенгайтмали файлларни қўллаб қувватлайдиган компонеталари керак. Охирги вақтларда бу функциялардан фойдаланиш учун қўшимча компоненталар керак эмас, чунки уларнинг барчаси ҳозирда PHP дастурлаш тили ядросига киритилган.

7. Синфлар ва объектлар

Объектга йўналтирилган дастурлашнинг асосий тушунчалари – синфлар ҳамада объектлардир. Бу тушунчаларни қуйидагича тушуниш мумкин: объект – бу дастурда қўлланиладиган тушунча ёки бирор физик предмет ҳақида маълумот

берадиган структураланган ўзгарувчидир, синфлар эса бу объектларнинг тавсифи ва улар устида бажариладиган ҳаракатлардир.

PHP дастурлаш тилида синфлар қуйидаги синтаксис ёрдамида аниқланади:

```
class Синф_номи{
  var $хусусият_номи;
    /* хусусиятлар рўйхати */
  function метод_номи( ){
   /* усулларнинг танаси */
  }
  /*усуллар рўйхати*/
}
```

Синф объектлари хусусиятлари номи var калит сўзи ёрдамида эълон қилинади, берилган синф объектларига қўлланилган усуллар функция сифатида ишлатилади. Синф танаси ичида this калит сўзи ёрдамида тақдим қилинаётган жорий синфга мурожаатни амалга ошириш мумкин.

Масалан, биз мақола категориясини тасвирловчи синф тузишимиз керак. Ҳар бир мақоланинг номи, муаллифи ва қисқа мазмуни каби хусусиятлари бор. Биз мақола билан қандай амал бажармоқчимиз? Биз санаб ўтилган хусусиятларга маъно беришимиз, мақолани браузерда кўрсатишимиз керак бўлади. Шунда бу синфнинг ифодаланиши қуйидагича ҳолатда бўлади:

```
<?
class Articles { // Мақола синфини тузамиз
```

```
  var $title;
  var $author;
  var $description;
// мақола атрибути маъносини ўзлаштирувчи усул
  function make_article($t, $a, $d){
    $this->title = $t;
    $this->author = $a;
    $this->description = $d;
  }
//синф  нусхасини  ифодалаш  учун усул
  function show_article(){
    $art = $this->title . "<br>" .
          $this->description .
      "<br>Муаллиф: " . $this->author;
    echo $art;
} } ?>
```

Шундай қилиб “мақола” туридаги физик объектларни тасвирлаш учун биз уч ўзгартувчидан ташкил топган, мақола характеристикасини ўзида жамлаган Articles номли синф ва муайян мақола тузиш ва уни тасвирлаш учун иккита функция туздик.

Маълумки, PHP билан ишлаш даврий ҳолатда HTML режимида юкланиши мумкин. Бу ҳолда дастур бир неча коднинг бўлаклари(блоклар)дан ташкил топади. Синфни ифодалаш php-коднинг ҳар хил блоклари бўйича ва қолаверса ҳар хил файллар бўйича тарқатилмаслиги керак. Яъни қуйидагича ёзсак:

```
<?php
class Articles { // Синфни тасвирлашнинг боши
  var $title;
?>
<?php   // синфни тасвирлашнинг давоми
   function show_article(){  // усулнинг таркиби
   }
} // синфни тасвирлашнинг якуни
?>
```

бунда дастур тартибли ишлайди.

Синф номи масаласида айрим нарсаларни эътиборда тутиш керак. Синфнинг номи PHP тилидаги объектлар номланиши қоидаларига жавоб бериши лозим, лекин бир қатор номлар борки, техник мутахассислар томонидан ўз мақсади учун захира қилинади. Биринчи навбатда бу номлар “_” қуйи чизиқдан бошланувчилардир. Синфлар ва функциялар тузиш учун бу каби номларни ишлатмаслик керак. Бундан ташқари stdClass номи захира қилинган, зеро у PHP сурилгичи ичида ишлатилади.

8. Ўзгарувчиларни инициаллаштириш

Баъзан айрим синф атрибутларига маънони синф иштирокчисини тузиш биланоқ ўзлаштириш керак бўлади. Биз мақола синфини тузганимизда, синф атрибутлари (хусусиятлари) маъноларини ўзлаштириш учун махсус функция make_article() дан фойдаландик. Умуман олганда, биз тўғри йўл тутмадик, чунки

“велосипед ихтироси” билан шуғулландик. Синф атрибутларининг бошланғич маъноларини бериш учун махсус иккита стандарт усул мавжуд. PHP4да маънони var оператори ёки конструктор функцияси ёрдамида инициаллаштириш мумкин. var ёрдамида фақат константа маъноларни инициаллаштириш мумкин. Константа бўлмаган маъноларни бериш учун объект синфдан ажраб чиққанда ўз-ўзидан ишга тушувчи конструктор фунциясидан фойдаланилади. Конструктор-функция у ифодаланган бутун синфга мос келувчи номга эга бўлиши керак.

Мисол. “мақола” деб номланган объектни тузишда унинг хусусиятларини қуйидагича белгилаш мумкин: муаллифлар – “Камолов” сатрига тенг, номланиш ва қисқа мазмун - $_POST глобал массиви элементларига мос, мақола нашри – мазкур санада.

```
<?
class Articles { // мақола синфини тузиш
  var $title;
  var $author = "Камолов";
  var $description;
  var $published;       // синф атрибути маъносини ўзлаштирувчи усул
 function Articles(){
  $this->title = $_POST["title"];
  $this->description = $_POST["description"];
  $this ->published = date("Y-m-d");
} } ?>
```

PHP3 ва PHP4 да конструкторлар ҳар хил ишлашини ҳисобга олиш керак. Функция PHP3 да, агар у синфники каби номга эга бўлса, консртукторга айланади, PHP4 да эса – агар у ифодаланган синфники каби номга эга бўлса шундай бўлади. Бир синф бошқасини кенгайтирганда ва хусусиятларнинг ва база синфлар усулларининг эргашишида усуллар орасидаги фарқ кўриниб турибди. Лекин биз бу ҳақда бироз кейинроқ гапирамиз. PHP5да синф конструктори _construct деб номланади. Бундан ташқари, PHPда деструкторлар – объектни йўқ қилишда ўз-ўзидан ишга тушувчи функциялар пайдо бўлди. PHP5 да функция-деструктор destruct деб номланиши керак бўлади.

9. Объектлар

Object (объектлар) типи

Объектлар – объектга йўналтирилган дастурлашдан кириб келган *маълумот типи*дир. Объектга йўналтирилган дастурлаш тамойилига кўра, синф – аниқ хоссаларга эга ва улар билан ишлайдиган методли объектлар тўплами. Объект эса мос равишда синф нусхасидир. Масалан, дастурчилар – бу дастурни тузувчи, компьютер адабиётларини ўрганадиган одамлар синфи ва бундан ташқари ҳамма одамлар қатори исм ва фамилияси мавжуд. Энди агарда бирор аниқ дастурчи – Азамат Бобоевни олсак, у ҳолда уни шу хоссага эга бўлган дастурчи синфини объекти сифатида қараш мумкин ва у ҳам дастур тузади, ҳамда исми мавжуд ва бошқалар.

PHP дастурлаш тилида *объект методи*га мурожаат -> *амали*дан фойдаланилади. Объектни инициализация қилишда *объектни ўзгарувчан нусхаси*ни яратадиган new ифодасидан фойдаланилади.

```
<?php     // одам синфини яратамиз.
class Person
{       // PHP дастурлаш тилини ўрганадиган одам методи
   function know_php()  {
          echo "Энди мен PHP дастурлаш тилини биламан!";
       }
}
$bob = new Person; // одам синфини объектини яратамиз.
$bob -> know_php(); // уни PHP тилига ўргатамиз.
?>
```

Синф – бу объект типидаги маълумотларнинг бир туридаги ифодаланишидир. Синфлар реал ўзгарувчилар учун шаблон вазифасини ўтайди. Керакли типдаги ўзгартувчи new оператори ёрдамида синфдан тузилади. Объектни тузиб, биз барча усулларни қўллашимиз ва барча синф ифодасида кўрсатиб ўтилган хусусиятларни олишимиз мумкин бўлади. Бунинг учун қуйидагича синтаксисдан фойдаланилади: $объект_номи->хусусият ёки _номи$объект_номи->усулнинг_номланиши(аргументлар рўйхати). Хусусиятлар ёки усулар номлари олдидан $ белгиси қўйилмайди.

Мисол: Объект усуллари ва хусусиятларига эркин кириш (доступ)

```
<?php
$art = new Articles;     // объект тузамиз $art
echo ($art ->title);     // объекга номланиш берамиз $art
$another_art = new Articles;     // объект тузамиз $another_art
$another_art->show_article();  // объектнинг браузердаги ифодаси учун усулни чақирамиз
?>
```

Синфнинг ҳар бир объекти айнан бир хил хусусиятлар ва усулларга эга бўлади. Демак, $art объектда ва $another_art объектда title, description, author хусусиятлари ва Articles(), show_article() усуллари мавжуд. Лекин булар икки хил объектлар. Объектни файллар системасидаги директория деб ҳисоблаймиз, унинг характеристикаси эса – бу директориядаги файллар сингари бўлсин. Аниқки, ҳар бир директорияда бир хил файллар ётиши мумкин, лекин шундай бўлса-да, улар ҳар хил директорияларда сақланаётгани учун ҳар хил ҳисобланиши мумкин. Худди шунингдек, хусусиятлар ва усуллар ҳам, агар улар турли объектларга қўлланиладиган бўлса, ҳар хил ҳисобланади. Юқори босқичдаги директориядан

керакли файлни олиш учун бу файлга йўлни батафсил ёзиб чиқамиз. Синфлар билан ишлаш мобайнида биз чақиришни истаган функциянинг номини тўлиқ ёзишимиз керак бўлади. PHP даги юқори босқич директорияларига глобал ўзгарувчиларнинг бўш ўрни бўлади, йўл эса -> тақсимловчиси ёрдамида кўрсатилади. Шу тарзда $art->title ва $another_art->title номлари икки хил турли ўзгарувчиларни англатади. PHP да ўзгарувчи ном олдидан фақат битта доллар белгисига эга бўлади, шунинг учун $art->$title кўринишида ёзиш мумкин эмас. Бу конструкция $art объектининг title хусусиятига мурожаат сифатида кўриб чиқилмайди, $title ўзгартувчи кўринишида берилган номли хусусият сифатида кўрилади (масалан, $art->"").

Мисол: Хусусиятлар маъносини ўрнатиш

```
<?php
$art->title = " Internet га кириш"; // объект хусусияти маъносини шундай ўрнатиш мумкин
$art->$title = "Internet га кириш"; // объект хусусияти маъносини бундай ўрнатиб бўлмайди
$property = "title";
$art->$property = "Internet га кириш"; // объект хусусияти маъносини шундай ўрнатиш мумкин
?>
```

Синфни тузиб, бу синфнинг объекти қандай номга эга бўлишини била олмаймиз, қолаверса объектлар жуда кўп бўлиши ва уларнинг барчаси ҳар хил номга эга бўлиши мумкин. Синфни юзага чиқариш ичида объектга қандай муносабатда бўлишни билмаймиз. Синф юзага чиқиши ичида функциялар ва ўзгарувчиларга эркин кириш учун, $thid ўриндош ўзгарувчисидан фойдаланиш керак. Масалан, $this->title шундай синф объектининг title ини қайтаради. Баъзан

бу ўзгарувчини “менинг хусусий мулким” (хусусиятга муносабат тариқасида) деб ўқишни таклиф қилинади.

Назорат саволлари:

1. PHP да функцияларни тушунтириб беринг.
2. PHP да функциялар аргументлари қандай эълон қилинади?
3. PHP да ўзгарувчилар эълонини тушунтиринг.
4. Ичма-ич функцияларни нима?
5. PHP да синфлар ва объектлар нима?
6. Функцияни чақириш қандай амалга оширилади?
7. PHP дастурлаш тилида функсиялар нима учун керак?
8. Функсиялар ичида қандай турдаги ўзгарувчилардан фойдаланиш мумкин?
9. PHP дастурлаш тилида класслар ва объектлар қандай эълон қилинади?
10. Синфнинг объектлари қандай ҳусусиятларга эга бўлиши мумкин?
11. PHP дастурлаш тили конструктсиясига эга бўлмаган функцияни айтинг ва у ҳақида маълумот беринг?

Фойдаланилган адабиётлар:

25. Холзнер С. Перл: специалний справочник. - СПб: "Питер". 2000. - 496 с.
26. Шварс Р., Кристиансен Т. Изучаем Перл. - К: "БҲВ", 2000. - 320 с.
27. Ратшиллер Т., Геркен Т. PHP4: разработка Web-приложений.- СПб:Питер, 2001. - 384 с.
28. Томсон Л., Веллинг Л. Разработка Web-приложений на PHP и MySQL. - К.: "DiaSoft", 2001. - 672 с.
29. Основи современних компютерних технологий. Ред. Хомченко А.Д.

30. Савелев А.Я., Сазонов Б.А., Лукянов Б.А. Персоналний компютер для всех. Хранение и обработка информатсии. Т.1 М.: Висшая школа, 1991.
31. Брябрин В.М. Программное обеспечение персоналних ЭВМ. М.: Наука, 1990.
32. Фролов А.В., Фролов Г.В. Глобалнийе сети компютеров. Практическое введение в Интернет, Э-Маил, FTP, WWW и HTML. М.:Диалог-МИФИ, 1996.

Фойдаланилган манбалар:

18. http://uz.wikipedia.org/wiki/HTML
19. http://javascripts.boom.ru
20. http://www.vanta.ru/script/
21. http://www.vbnet.ru
22. http://www.scriptic.ru/
23. http://www.webacademy.com/
24. http://pacificwebart.com/
25. http://www.yahoo.com/Computers_and_Internet/Programming_Languages/Java

13-маъруза. PHP дастурлаш тилида маълумотлар базалари билан ишлаш. MySQL маълумотлар базаси.

Режа:

1. Маълумотлар базаси ҳақида тушунча;
2. Маълумотлар базаси интерфейси;
3. Маълумотлар базаси билан боғланиш;

4. Маълумотлар базаси устида амаллар.

Калит сўзлар: Маълумотлар базаси, MySql дастури, PHP да MySql билан боғланиш, файллар, файллар билан ишлаш, МБ амаллари.

Ишдан мақсад: PHP дастурлаш тилида маълумотлар базаси ва файллар билан ишлаш билим ва малакасига эга бўлишни ўрганиш ва уни амалиётга татқиқ қилиш.

1. Маълумотлар базаси ҳақида тушунча

Ушбу бўлим PHP ва MySql МББС ўртасидаги ҳамкорлик усуллари билан танишишга мўлжалланган. Асосий эътибор маълумотлар базаси билан боғланишни ўрнатиш, сўровлар жўнатиш функциялари ва жавобларни (mysql_connect, mysql_query, mysql_result, mysql_num_rows, mysql_close) қайта ишлашга қаратилади. Мисол сифатида виртуал тарих музейи маъмурияти учун web-интерфейс тузиш масаласини кўрайлик. PHP дистрибутивида MySQL маълумотлар базаси билан ишлаш учун мўлжалланган функциялар мавжуд. Бунда бу функцияларнинг MySql даги баъзи бир маълумотлар базасини тасвирлаш ва тўлдириш мақсадида web-интерфейсларни тузиш имконини берувчи функциялар билан танишамиз. Маълумотлар базасига маълумотларни қўшиш учун web-интерфейс билан ишлашда бу маълумотларни шунчаки html-формага киритиш ва уларни серверга жўнатиш керак бўлади.

Намойиш этишда бу интерфейсни виртуал музей экспонатлари ҳақидаги маълумотлар сақланадиган Artifacts жадваллари учун тузамиз. Artifacts коллекциясидаги ҳар бир экспонат қуйидаги характеристика ёрдамида тасвирланишини эслатиб ўтамиз:

ном (title);
муаллиф (author);
ифода (description);
ўриндош ном (alternative);
тасвир (photo).

Номланиш ва ўриндош номланиш узунасига 255 белгидан кам сатр (яъни VARCHAR(255)), тасвирлаш – матнли майдон (TEXT турига мансуб) ҳисобланади, “муаллиф” ва “тасвир” майдонларида эса Persons коллекциясидан муаллифнинг идентификаторлари ва Images коллекциясидан экспонат тасвирларига мувофиқ мавжуд бўлади.

2. Маълумотлар базаси интерфейси

Маълумотлар базасидаги мавжуд жадвал структурасини (яъни унинг майдонлари жамланмасини) html-формада тасвирлаш учун қуйидаги таркибий топшириқларни режалаштириш мумкин:

- МБ билан уланишни ўрнатиш;
- МБ ишини танлаш;
- Жадвал майдонлари рўйхатини олиш;
- html-формада майдонларни тасвирлаш.

Бундан кейин формага киритилган маълумотларни маълумотлар базасига киритиш мумкин.

3. Маълумотлар базаси билан боғланиш (MySQL дастури мисолида)

Алоқа ўрнатиш

Маълумотлар базаси билан алоқа ўрнатиш учун **mysql_connect** функциясидан фойдаланилади.

mysql_connect синтаксиси

mysql_connect ресурси (“сервер қатори”, “username”, “password”)

Бу функция MySQL сервери билан алоқа ўрнатади ва бу алоқага кўрсаткич қайтаради ёки муваффақиятсиз чиққанда FALSE кўрсатади. Одатда қуйидаги параметрлар қиймати эълон қилинади:

server = 'localhost:3306'

username = сервер жараёни эгасидан фойдаланувчи исми

password = бўш парол

Сервер билан уланиш, агар у бунгача mysql_close() ёрдамида ёпилмаган бўлса, скриптни амалга ошириш тугалланишида база билан алоқа ёпилади.

Мисол:

```
<?
$conn = mysql_connect("localhost", "nina","123") or die("Уланишни амалга ошириб бўлмайди: ". mysql_error());
echo "Уланиш амалга ошди";
mysql_close($conn);
?>
```

mysql_connect амали

shell>mysql -u nina -p123 буйруғи билан тенг кучли.

4. Маълумотлар базаси устида амаллар

Маълумотлар базаларини танлаш

MySQL да маълумотлар базасини танлаш use буйруғи ёрдамида амалга оширилади:

mysql>use book;

PHP да бунинг учун mysql_select_db функцияси мавжуд.

mysql_select_db: синтаксиси

мантиқий mysql_select_db (database_name қатори);

Бу функция TRUE қийматни маълумотлар базасини муваффақиятли танланганда қайтаради ва FALSE ни эса – аксинча бўлганда.

Мисол: Book маълумотлар базасини танлаш

```
<?
$conn = mysql_connect("localhost","user","123") or die("Уланишни амалга ошириб бўлмайди: ". mysql_error());
echo "Уланиш амалга ошди";
mysql_select_db("book");
?>
```

Жадвал майдонлари рўйхатини олиш

PHP да маълумотлар базаси билан боғланилгандан сўнг, ундаги жадваллар рўйхатини олиш мумкин. Бу функция - mysql_list_fields.

mysql_list_fields синтаксиси

mysql_list_fields (database_name қатори, table_name қатори)

mysql_field_name функцияси сўров амалга оширилиши натижасида олинган майдон номини қайтаради. mysql_field_len функцияси майдон узунлигини қайтаради. mysql_field_type функцияси майдон типини қайтаради, mysql_field_flags функцияси эса пробел билан ёзилган майдон байроқлари рўйхатини қайтаради. Майдон типлари int, real, string, blob ва ҳ. бўлиши мумкин. Байроқлар not_null, primary_key, unique_key, blob, auto_increment ва ҳ. бўлиши мумкин.

Бу барча буйруқлар синтаксиси бир хил:

mysql_field_name (result қатори, бутун field_offset) ресурси;
mysql_field_type (result қатори, бутун field_offset) ресурси;
mysql_field_flags (result қатори, бутун field_offset) ресурси;
mysql_field_len (result қатори, бутун field_offset)

Бу ерда result – бу сўров натижаси идентификатори (масалан, mysql_list_fields ёки mysql_query функциялар билан жўнатилган сўров), field_offset эса – натижадаги майдоннинг тартиб рақами.

mysql_num_rows(result ресурси) буйруғи result нинг кўпгина натижалари қаторлари миқдорини қайтаради.

Мисол: Artifacts (экспонатлар коллекцияси) жадвали майдонлари рўйхатини олиш.

```
<?
$conn = mysql_connect("localhost","user","123") or die("Алоқа ўрнатиб бўлмайди: ". mysql_error());
echo "Алоқа ўрнатилди";
mysql_select_db("book");
$list_f = mysql_list_fields ("book","Artifacts",$conn);
$n = mysql_num_fields($list_f);
for($i=0;$i<$n; $i++){
  $type = mysql_field_type($list_f, $i);
  $name_f = mysql_field_name($list_f,$i);
  $len = mysql_field_len($list_f, $i);
  $flags_str =  mysql_field_flags ($list_f, $i);
echo "<br>Майдон номи: ". $name_f;
echo "<br>Майдон тури: ". $type;
echo "<br>Майдон узунлиги: ". $len;
echo "<br>Майдон байроқлари қатори: " . $flags_str . "<hr>";
}
?>
```

Натижа сифатида тахминан қуйидагиларни олиш мумкин (албатта, жадвалда иккита майдон бўлганда):

- Майдон номи: id
- Майдон тури: int
- Майдон узунлиги: 11
- Майдон байроқлари қатори:
- not_null primary_key auto_increment
- Майдон номи: title
- Майдон тури: string
- Майдон узунлиги: 255
- Майдон байроқлари қатори:

HTML-формада майдонлар рўйхатининг акс этиши

Майдон ҳақидаги маълумотни html-форма элементида акс эттирамиз. BLOB туридаги элементларни textarea га ўтказамиз (TEXT турида биз тузган description майдони BLOB типига эга), рақамлар ва қаторларни <input type=text> матнли қаторларида акс эттирамиз, автоинкремент белгисига эга элементни эса умуман акс эттирмаймиз, чунки унинг маъноси ўз-ўзидан ўрнатилади.

Бунинг учун explode функциясидан фойдаланилади:

explode: синтаксиси

explode массиви (separator қатори, string қатори , int limit)

Бу функция string қаторини separator тақсимлагичи ёрдамида қисмларга бўлади ва олинган қаторлар массивини қайтаради.

Бизнинг ҳолатда тақсимлагич сифатида пробел “ ” ни олиш кера, бўлиш учун бошланғич қатор сифатида эса – майдон байроқлари қаторини.

Мисол. Artifacts жадвалига маълумот киритиш учун форма (index.php)

```
<?php
  $dblocation = "localhost";
  $dbname = "book";
  $dbuser = "root";
  $dbpasswd = "";
$dbcnx = @mysql_connect($dblocation,$dbuser,$dbpasswd);
@mysql_select_db($dbname,$dbcnx);
if(isset($_POST['save_hide'])) {
$title=($_POST['title']);
$query = "INSERT INTO Artifacts (title) VALUES ('$title')";
 if(@mysql_query($query))
  {
   echo "<HTML><HEAD><META HTTP-EQUIV='Refresh' CONTENT='0;
URL=index.php'></HEAD></HTML>";
  } else {  print mysql_error();  error("Маълумотни базага езишда хатолик");}
} else {
?>
<TABLE width="100%" align=center border=0><TR><TD>
<form method="POST" name="save" action="" enctype="multipart/form-data" >
<input type=hidden name=save_hide value=save_hide>
<TABLE width="100%" align=center border=0><TR>
<TD>Экспонат номини киритинг </TD>
<TD><INPUT class="input" size=“35” name=“title” value=""></TD>
</TR></TABLE>
```

```
<input type="submit" value="Saqlash" class="batton" />
</form></TD></TR></TABLE>
<? } ?>
```

Маълумотлар базасига маълумотлар ёзиш

Маълумки, маълумотларни жадвалга ёзиш учун SQL тилидаги INSERT буйруғи ишлатилади:

```
mysql> INSERT INTO Artifacts SET title='Eksponat nomi';
```

PHP скриптда бундай буйруқдан фойдаланиш учун mysql_query() функцияси мавжуд.

mysql_query синтаксиси

mysql_query ресурси (query қатори)

mysql_query() SQL-сўровни MySQL маълумотлар базасининг маълумотлар базасига жўнатади. Агар очиқ алоқа бўлмаса, функция параметрсиз mysql_connect() функциясига ўхшаш ҳолда МББТ билан боғланишга уринади.

Сўров натижаси буферланади.

Назорат саволлари:

1. Маълумотлар базаси деганда нимани тушунасиз?
2. Қанака маълумотлар базаси дастурларини биласиз ва уларни имкониятлари?
3. PHP да MySql билан боғланиш функциясини тушунтириб беринг?
4. PHP да MySql сўровларини амалга оширишга мисол келтиринг?
5. Турли маълумотлар базаси дастурларида ишлаш жараёнида ҳар бирига тегишли бўлган камчилик ва афзалликларни айтинг?
6. Web-сервердаги маълумот билан ишлаш учун қандай усуллар мавжуд ва уларнинг ишлаш моҳиятини тушунтиринг?

Фойдаланилган адабиётлар:

33. Холзнер С. Перл: специалний справочник. - СПб: "Питер". 2000. - 496 с.
34. Шварс Р., Кристиансен Т. Изучаем Перл. - К: "БҲВ", 2000. - 320 с.

35. Ратшиллер Т., Геркен Т. PHP4: разработка Web-приложений.- СПб:Питер, 2001. - 384 с.
36. Томсон Л., Веллинг Л. Разработка Web-приложений на PHP и MySQL. - К.: "DiaSoft", 2001. - 672 с.
37. Основи современних компютерних технологий. Ред. Хомченко А.Д.
38. Савелев А.Я., Сазонов Б.А., Лукянов Б.А. Персоналний компютер для всех. Хранение и обработка информатсии. Т.1 М.: Висшая школа, 1991.
39. Брябрин В.М. Программное обеспечение персоналних ЭВМ. М.: Наука, 1990.
40. Фролов А.В., Фролов Г.В. Глобалнийе сети компютеров. Практическое введение в Интернет, Э-Маил, FTP, WWW и HTML. М.:Диалог-МИФИ, 1996.

Фойдаланилган манбалар:

26. http://uz.wikipedia.org/wiki/HTML
27. http://javascripts.boom.ru
28. http://www.vanta.ru/script/
29. http://www.vbnet.ru
30. http://www.scriptic.ru/
31. http://www.webacademy.com/
32. http://pacificwebart.com/
33. http://www.yahoo.com/Computers_and_Internet/Programming_Languages/Java

14-маъруза.Cookie, seans, FTP ва e-mail технологиялари.

Режа:

1. Cookie технологиясини ўрнатиш;
2. Cookie технологиясини ўқиш;
3. FTP технологияси;

4. E-mail технологияси.

Калит сўзлар: Cookie, сеанс, FTP, E-mail технологиялари, FTP сервер, FTP клиент, аноним FTP, FTP Proxy, FTP host, FTP сайт.

Ишдан мақсад: Ишлаш жараёнида Cookie, seans, FTP, E-mail теҳнологиялари билан ишлаш кўникмаларига эга бўлиш ва талабаларга бу хақида билим бериш.

1. Cookie технологиясини ўрнатиш

Cookie — клиент компьютерида сақланувчи ва у ҳар сафар серверга мурожаат қилаётганда web-серверга юбориладиган матн сатридир. Шу тариқа маълумотлар турли скриптлар аро формалар ёки URL адресларсиз узатилиши мумкин. Cookie қийматини ўрнатиш учун қуйидаги синтаксисдан иборат бўлган setcookie функциясидан фойдаланилади:

bool setcookie (string name [, string value [, int expire [, string
path [, string domain [, bool securej]]]])

Функция клиент компьютерида сақланувчи cookie ларни тасниф этади. Қуйида ушбу функция параметрларининг таснифи кетирилган:

• name. Cookie нинг номи.
• value. Cookie нинг қиймати.
• expire. Cookie нинг амал қилиш муддати. Агар уберилса, у тугагач cookie ўчириб ташланади. Агар кўрсатилмаса, cookie браузер ойнаси ёпилгандан сўнг ўчириб ташланади
• path. Сервердаги cookie га рухсат этилган адрес.
• domain. cookie га рухсат этилган домен.
• secure. HTTPS протокол орқали боғланишда cookie нинг ҳавфсизлик белгиси. Стандарт ҳолатда cookie HTTPS учун ҳам HTTP даги каби ишлайди.

Кўриниб турибдики, cookie HTTP-сўровнинг қисми ҳисобланади ва у браузерга юборилади. Шу сабаб HTML-код формалаштиришдан олдин унинг қиймати ўрнатилиши керак. Бу шуни англатадики, setcookie функциясининг қиймати HTML-тегдан олдин ва echo операторигача ўрнатилади.

Агар бу қоида сақланмаса setcookie функциясини чақириш FALSE қиймат қайтаради ва бу cookie формировка қилинишида хатолик бўлганлигини англатади. cookie нинг тўғри формировка қилиниши функциянинг TRUE қиймат қайтаришига олиб келади. Бу cookie клиентга қабул қилинди дегани эмас албатта. Чунки браузерни созлашда cookie ўчириб ташлаш ёки серверга жўнатмаслик параметрлари ўрнатилган бўлиши мумкин.

2. Cookie технологиясини ўқиш

Cookie қиймати ўрнатилганидан сўнг, саҳифа қайта юкланмагунича дарҳол ундаги скрипт учун cookie га рухсат мавжуд бўлмайди. Чунки cookie фойдаланувчи компьютерида сақланади ва браузер орқали web-серверга жўнатилади. Бундан ташқари, cookie қачонки унинг домени сервер домени билан мос тушгандагина жўнатилади.

cookie га рухсатни олиш учун махсус **$_COOKIE** суперглобал массивидан фойдаланилади. Массивнинг қиймати сифатида олдин ишлатилган cookie номи олинади. Массив одатда скрипт юкланаётган вақтда **$_GET**, **$_POST** ва **$_REQUEST** массивлари билан автоматик тарзда тўлдирилади

cookie дан фойдаланишдан олдин унинг қиймати ўрнатилганлигига ишонч ҳосил қилинг. Бунинг учун isset функциясидан фойдаланиш жуда қулай. Қуйидаги мисолда message номли cookie нинг қиймати текширилиши ва кўрсатилиши келтирилган.

Мисол:

```
<HTML>
<HEAD> <TITLE> Cookie нинг қийматини ўқиш </TTTLE> </HEAD>
<BODY>
<CENTER>
<H1> Cookie нинг қийматини ўқиш </Hl>
Cookie нинг қиймати:
<?php
if (isset ($_COOKIE ['message']))
{
echo Cookie нинг қиймати:' . $_COOKIE ['message' ] ;
}
else
{
echo 'Cookie ўрнатилмаган' ;
}
?>
</CENTER;.
<BODY>
</HTML>
```

Cookie массивларда ҳам ташкил этилган бўлиши мумкин. Масалан қуйида учта cookie ўрнатилган:

```
setcookie ("cookie[one]" , "Бугун");
setcookie ("cookie[two] ", "Ҳаёт");
```

```
setcookie ("cookie[three] ", "гўзал!") ;
```

Натижада $_COOKIE['cookie'] массив қийматлари қуйидаги тарзда босмага чиқарилиши мумкин:

```
if (isset ($_COOKIE ['cookie'])) {
foreach ($_COOKIE'['cookie'] as $data)
{
echo "$data <BR>";
…
```

3. FTP технологияси

Қуйида FTP нинг баъзи атамаларини кўриб чиқамиз:

1. **FTP**

File Transfer Protocol (файлларни алмашиш протоколи); тармоқ орқали (хусусан Интернет) файлларни узатиш протоколи. Буни бир компьютердан бошкасига файл нусхасини кўчириш каби тушуниш мумкин.

2. **FTP Server**

Бу файлни жунатилиши сўровини кутувчи компьютер ёки сервер.

3. **FTP Client**

Бу FTP серверга сўров жўнатувчи компьютер. Сўров текшируви ёки тасдиқланишидан сўнг FTP клиент компьютери маълумотларни серверга юклаши ёки сервердан юклаб олиши мумкин.

4. **Аноним FTP**

Ушбу серверга FTP клиент компьютери орқали авторизациядан ўтмасдан боғланиши мумкин. Бундай имкониятни бир нечта веб сайтларда кўришимиз мумкин, қайсики рўйхатдан ўтмасдан маълум файлларни юклаб олиш имкониятини беради.

5. **FTP Host**

Ушбу хизмат FTP сайт сифатида, яъни сайтда рўйхатдан ўтиб, сайтда келтирилган маълумотлар файлларини юклаб олиш имкониятини берувчи компьютер. Бу фойдаланувчилар сони чекланган пуллик хизмат ҳисобланади.

6. **FTP сайт**

FTP хост компьютери тасарруфидаги фойдаланувчи логин пароли талаб қилинувчи веб саҳифа. Хост серверда бир нечта веб-саҳифа жойланиши мумкин, бунда ҳар бир сайт учун алоҳида фойдаланувчи авторизацияси мавжуд.

7. **FTP Proxy**

FTP Proxy бу шунақа сервер компьютерки, бунда сўров FTP серверга жўнатилади, ушбу сўров прокси орқали ўтиб, кейин зарур манзилга йўналтирилади.

4. E-mail технологияси

Бу хизмат интернет тизими орқали ўзаро почта хабарлари алмашинуви усулидир. Бундай тизим оралиқ сақланиш методи асосида йўлга қўйилган. Замонавий интернет протоколлар хабарлар жўнатиш иконияти бўйича юқори даражага кўтарилди. Почта жўнатишнинг энг кенг тарқалган протоколи SMTP (Simple Mail Transport Protocol) ҳисобланади.

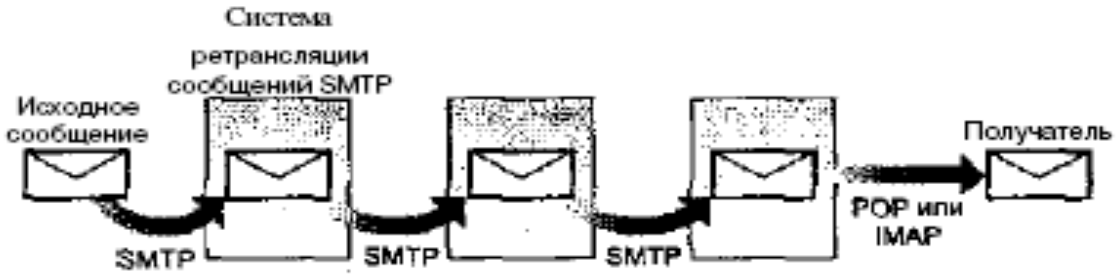


Рис. 22.2. Для пересылки и получения почты используются различные протоколы

Почтани қабул қилиш почта протоколи POP (Post Office Protocol), ёки Интернет хабарларини қабул қилувчи протокол IMAP (Internet Message Access Protocol) дан фойдаланилади.

Назорат саволлари:

1. Cookie технологияси нима?
2. Сеанс технологияси нима ва унинг моҳиятини тушунтиринг?
3. FTP протокол ва унинг вазифаси нимадан иборат?
4. E-mail tehnologiyasi ва унинг вазифаси ҳақида гапиринг?
5. FTP ҳост ҳизмати ишлаш моҳиятини тушунтиринг?
6. FTP технологиясида мавжуд бўлган яна қандай ҳизматларни биласиз?

Фойдаланилган адабиётлар:

41. Холзнер С. Перл: специалний справочник. - СПб: "Питер". 2000. - 496 с.
42. Шварц Р., Крициансен Т. Изучаем Перл. - К: "БҲВ", 2000. - 320 s.
43. Ратшиллер Т., Геркен Т. ПҲП4: разработка Web-приложений.- СПб:Питер, 2001. - 384 с.
44. Томсон Л., Веллинг Л. Разработка Web-приложений на PHP и MySQL. - К.: "ДиаСофт", 2001. - 672 с.
45. Основи современних компютерних технологий. Ред. Хомченко А.Д.
46. Савелев А.Я., Сазонов Б.А., Лукянов Б.А. Персоналний компютер для всех. Хранение и обработка информатсии. Т.1 М.: Висшая школа, 1991.
47. Брябрин В.М. Программное обеспечение персоналних ЭВМ. М.: Наука, 1990.
48. Фролов А.В., Фролов Г.В. Глобалнийе сети компютеров. Практическое введение в Интернет, E-mail, FTP , WWW и HTML . М.: Диалог-МИФИ, 1996.

Фойдаланилган манбалар:

16. Intuit.ru.
17. http://pda.coolreferat.com
18. http://www.google.ru
19. http://www.z-oleg.com
20. http://www.i2r.ru/