

Mühazirə 1

Veb texnologiyalar müasir dünyada insanların fəaliyyətində mühüm rol oynayan vasitəyə çevrilmişdir. Artıq biz həyatımızı internetsiz təsəvvür edə bilmirik. İnternet layihələr insanların müxtəlif sahələrdə: tədrisdə, biznesdə, maketiqdə və s. vacib platforma olmur. Qısa müddət ərzində veb-texnologiyalar böyük sürətlə inkişaf edərək, global informasiya fəzasının bütün istiqamətlərini əhatə edən Ümumdünya Hörümçək Toru (World Wide Web, WWW) yaratmışdır .

Virtual mühitdə toplanan informasiyanın tez-tez dəyişdirilməsi, tipinə, formatına, strukturuna və s. parametrlərə görə müxtəlif verilənlərdən təşkil olunması veb texnologiyaların üzərinə əlavə vəzifələr qoyur. Məsələn, minlərlə veb-səhifəyə malik hər hansı bir internet-layihədə səhifələrin, səhifədəki sözlərin, hiperistinadların, mediafaylların sayı tez-tez dəyişir. Belə bir şəraitdə müasir veb texnologiyaların tədqiqatlarda təhlükəsizlik, zaman faktoru, tranzaksiya sürəti, verilənlərin tipi mühüm rol oynayır.

HTML nədir?

HTML (ing. Hypertext Markup Language) — brauzerin oxuya biləcəyi hər hansı sənədi və ya səhifəni yaratmaq üçün xüsusi hipermetn dilidir. HTML internetin fundamental baza texnologiyasıdır və veb-səhifənin növünü, funksiyasını təyin edən diskriptorlar əsasında yaradılmış dildir.

Bütün veb-qovşaqlarda toplanmış və internetə qoşulmuş kompüterlərin ekranlarında göstərilən sənədlər HTML proqram kodunda yazılırlar. HTML – səhifələrdə mətn bloklarının, təsvirlərin yerləşdirilməsinə, cədvəllərin qurulmasına, sənədin və sənəddəki mətnin rənglərinin seçilməsinə, multimediyaya elementlərinin əlavə edilməsinə, hiperistinadların və bütün bu elementlər arasında əlaqələrin yaradılmasına imkan verir.

HTML dilinin inkişaf tarixi 1989-cu ildə Oksford Universitetinin tələbəsi Tim Berners-Li tərəfindən hipermetnli sənəd sisteminin çıxarılması təklifi ilə başladı. 1990-cı ildə isə o bunu World Wide Web (Ümumdünya hörümçək toru) adlandırdı. Sistemi təşkil edən hissələrdən biri hipermetnli nişan qoymaq idi. Onun əsası 1990-cı ildə Berners–Linin hipermetnli sənədlərə baxmağa icazə verən brauzeri yaratması ilə qoyuldu. Nəhayət, 1995-ci

ildə dilin birinci versiyası – HTML 1.0 çıxdı. Yalnız 1995-ci ildə HTML 2.0 versiyasının işlənməsi başa çatdıqdan sonra, HTML dili standartlaşdı. O vaxta HTML dilinin yeni versiyasını çox brauzerlər dəstəkləyirdi.

1996-cı ildə artıq dilin HTML 3.2 (HTML 3.0 versiyasında cədvəllər əmələ gəlmişdi) versiyasını bütün brauzerlər dəstəkləyirdi. Bunun nəticəsində veb-dizayn yüksək səviyyəsinə yüksəldi. Veb-dizayn erasına başlanğıc verən yeni imkanlar ortaya çıxdı.

Dilin növbəti versiyası HTML 4.01 1999-cu ildə standartlaşdı. Bundan sonra dilin təkmilləşməsi dayandırıldı. Çünki, faktiki olaraq bundan başqa heç nə etmək olmurdu. Müasir internet istifadəçilərini daha da razı salan yeni texnologiyalar əmələ gəldi. Dinamik səhifələr yaratmağa və məlumat bazasına sorğu göndərməyə imkan verən müxtəlif proqramlaşdırma dilləri yarandı. Lakin 2004-cü ildə HTML-in yeni versiyası HTML5-in yaradılmasına başlandı və 2014-cü il 28 Oktyabrda tamamlandı və standartlaşdırıldı. Hazırda HTML dilinin davamçısı XHTML (eXtensible HyperText Markup Language – Hipermetn nişan qoyma genişlənməsi) sayılır.

MÜASİR WEB TEXNOLOGİYALAR

Hazırda bir çox veb texnologiyalar var ki, onların üzərinə düşən əsas vəzifələr kontentin təhlükəsizliyi, istifadəçilərin rahatlığı, dizayn və rahat interfeys kimi tələblərə cavab verməkdir. Hər veb texnologiya öz üslubunda yazıldığı proqram platforması üçün tələb olunan funksiyaları həyata keçirir. Hazırda bir çox veb proqramlaşdırma mövcuddur. Onlardan misal olaraq Hypertext Preprocessor (PHP), Active Server Pages (ASP.NET), Cold Fusion Markup Language (CFML), Java, Python və s. göstərmək olar.

Web proqramlaşdırmanın 2 növü var: Server tərəfdə proqramlaşdırma, Klient tərəfdə proqramlaşdırma. Server tərəfdə proqramlaşdırma dilləri kimi əsasən PHP, PERL, JAVA, C++, ASP və digərlərindən istifadə olunur. Klient tərəfdə isə JavaScript, VbScript, JScript və s. bu kimi dillərsən istifadə olunsada, bütün Web browserlər tərəfindən əsasən JavaScript problemsiz istifadə olunur.

ASP.NET. Bu texnologiya müxtəlif sahələrdə istifadə olunmaqla müştərilərin və biznes təşkilatlarının məqsədlərinə nail olmaq üçün faydalıdır.

Müasir tələblərə uyğun veb səhifələrin yaradılması üçün istifadə olunan müasir nəsil veb texnologiyadır. Microsoft şirkəti tərəfindən yaradılıb. Həmçinin server proqramlaşdırma dili kimi tanınır. ASP.NET-i Microsoftun müasir nəsil veb texnologiyası kimi adlandırmaq da olar. ASP.NET proqramlaşdırma dili aşağıdakı proqram platformalarının birləşməsi ilə həyata keçirilir

- veb dizayn dili (HTML);
- JavaScript;
- C #, .NET/VB.NET;
- Web Server (IIS /ASP.NET Development Server).

PHP. PHP proqramlaşdırma dilinin imkanları geniş və müxtəlifdir. Saysız skriptlərdən və müxtəlif komponentlərdən ibarətdir. PHP dilində kodlaşmada kodların ümumi bir stilə uyğunlaşdırılması və müxtəlif kitabxanalara problemsiz müraciəti bu texnologiyanın üstün cəhətlərindəndir. Proqramçı tərəfindən yazılan PHP kodu ümumi bir stilə gəlmədikdə avtomatik sorğu pəncərəsi açılır və proqramçıya düzgün variantlar təklif edilir. Bu xüsusiyyət proqram təminatının asan və az vaxt sərf etməklə hazırlanmasını təmin edir. Həmçinin, proqramçıya fərqli kitabxanalardan istifadə etmək imkanı da verir.

PHP veb-layihələndirmədən başqa veb tətbiqlərin (application) inkişaf etdirilməsi üçün də geniş istifadə olunur. PHP hazırlanan proqram məhsulunun testləşdirilməsi, nəşr və s. kimi

ASP.NET və PHP veb texnologiyaları arasında ən populyar olanlardır. Şəkil qalereyalarının tətbiq edilməsi bu iki texnologiya əsasında yaradılmışdır. Bu texnologiyalar arasında seçim etmək çətindir. Müəyyən təcrübəyə malik insanlar və bir çox digər amillər bu texnologiyalardan birgə istifadəyə üstünlük verirlər. Lakin əgər bu texnologiyalardan birini seçmək lazım gələrsə istifadəçinin nə istədiyi nəzərə alınmalıdır. Məsələn dəyəri, platforma mürəkkəbliyi, təhlükəsizliyi və digər xüsusiyyətlər seçim etməyə kömək ola bilər.

CSS3 and Javascript. Nəzərə almaq lazımdır ki, müasir dövrdə istifadəçilərin əksəriyyəti stolüstü kompüter vasitəsilə internetdə fəaliyyət göstərirlər. Bu gün noutbuklar, mobil telefonlar və planşetlər internet naviqasiya üçün geniş istifadə

olunur və bu cihazların siyahısı artmaqda davam edir. Bu veb tətbiqi dizaynerlərinin qarşısına ciddi tələblər qoyur, yəni artıq əvvəlki veb proqramların bəziləri bugünkü yeni texnologiyaların tələblərinə cavab vermir. Proqram internet üzərindən multimedia vasitələrindən səmərəli istifadəni təmin edir. Nəzərə almaq lazımdır ki, veb-texnologiyalar multimedia fayllarının əlverişliliyi və paylanması üçün ən yaxşı platforma olub, multimedia materiallarının müxtəlif yollarla nümayişini təmin edir.

Yuxarıda göstərilən proqram məhsulları müasir web texnologiyalarının tələblərinə müəyyən qədər cavab versələr də, onların istifadəsində müəyyən problemlər və çatışmazlıqlar mövcuddur.

MÖVCUD PROBLEMLƏR

- Veb texnologiyaların inkişafı ilə əlaqəli olaraq internetdə müxtəlif tipli verilənlərin həcmi hər an artmaqdadır. Bu mühitdə toplanan verilənlərin səmərəli saxlanması və email məsələsi ön plana keçir. Bu problemlərin öhdəsindən gəlmək üçün internet üzərində buludlar, elektron kitabxanalar, arxivlər və s. veb layihələr yaradılır. Lakin, buna baxmayaraq yaradılan yeni layihələr problemi tam həll etmək gücündə deyillər.

- Veb texnologiyalarında əsas məsələlərdən biri də təhlükəsiz veb-layihələrin yaradılmasıdır. Araşdırmalar göstərdi ki, müasir veb-texnologiyalarda informasiyanın təhlükəsizlik məsələsi tam həllini tapmamışdır. Bir çox hallarda informasiyanın təhlükəsizliyi məsələlərini qismən də olsa həll etmək üçün müxtəlif proqram platformalarından birgə istifadəyə üstünlük verilir. Əsasən də reklam xarakterli spamlarla bağlı problemlər daha aktualdır.

- Müasir veb texnologiyalar istifadəçilərin tələblərini ödəyəcək veb səhifələr yaratmaq üçün yetərli görünə də proqram platformasının mürəkkəbliyi bəzi problemlər yaradır. Belə ki, veb-layihələrdə axtarış sistemlərinin daha yaxşı təşkil olunması, semantika və verilənlərin strukturlaşdırılması problemləri mövcuddur.

- Müasir veb texnologiyalarda axtarış sistemlərinin düzgün təşkili üçün verilənlər bazasının idarəetmə sistemi səmərəli təşkil olunmalıdır ki, bu da çoxlu

vəsait tələb edir. Əksər axtarış sistemləri böyük xərclərdən qaçmaq üçün Google və Yahoo-nun bazalarından istifadə edir.

- Bir çox hallarda veb-layihələrin təhlükəsizlik məsələləri dizayn məsələlərini arxa plana atır. Lakin, nəzərə almaq lazımdır ki, internet istifadəçilərini cəlb etmək üçün dizayn məsələləri və veb-səhifələrdən rahat istifadə önəmlidir.

HTML-də teqlər

Teqlər normal mətni HTML kodundan ayırır. Brauzerinizə səhifədə nələrin göstəriləcəyini söyləyərək şəkillər, cədvəllər və əşyalar kimi bütün obyektlərə icazə verirlər. Fərqli etiketlər fərqli funksiyaları yerinə yetirəcəkdir. Səhifənizə bir brauzer vasitəsilə baxdığınız zaman teqlər özü görünmür, lakin effektləri görünür.

<html> etiketi HTML sənədinin əsasını təmsil edir. Və bütün teqlər bu teq daxilində olur. **<html>** etiketi bütün digər HTML elementləri üçün konteyner rolunu oynayır.

<head> elementi metaməlumatlar üçün bir konteynerdir və **<html>** etiketi ilə **<body>** etiketi arasında yerləşdirilir. Metadata HTML sənədinə dair məlumatdır. Metadata göstərilmir.

Aşağıdakı elementlər **<head>** elementinin içinə girə bilər:

- **<title>**
- **<style>**
- **<base>**
- **<link>**
- **<meta>**
- **<script>**
- **<noscript>**

<title> etiketi sənədin adını təyin edir. Başlıq yalnız mətn olmalıdır və brauzerin başlıq çubuğunda və ya səhifənin nişanında göstərilir. HTML sənədlərində **<title>** etiketi tələb olunur! Bir səhifə başlığının məzmunu axtarış sisteminin optimallaşdırması (SEO) üçün çox vacibdir! HTML sənədində birdən çox **<title>** elementi ola bilməz.

<body> etiketi sənədin əsas hissəsini təyin edir. <body> elementi HTML sənədinin başlıqları, abzasları, şəkillər, hiperlinklər, cədvəllər, siyahılar və s. Kimi bütün məzmununa malikdir.

Qeyd: HTML sənədində yalnız bir <body> elementi ola bilər.

<p> teqi bir paraqrafı təyin edir. Brauzerlər hər bir <p> elementindən əvvəl və sonra avtomatik olaraq bir boş sətir əlavə edirlər.

**** etiketi HTML 4-də şrift ölçüsünü və mətnin rəngini göstərmək üçün istifadə edilmişdir. Şriftin ölçüsünü dəyişmək üçün teqini SIZE= atributu ilə tətbiq etmək lazımdır. HTML dilində şriftlər üçün şərti vahidlə göstərilən yeddi (1-7) əsas ölçü göstərilib xırda, kiçik, adi, orta, böyük, iri, nəhəng! 1 2 3 4 5 6 7

FONT teqi COLOR atributuna malikdir. Yazılış qaydası belədir:

 və ya

Yadda saxlamaq lazımdır ki, bu rəng teqi teqi ilə bağlana qədər qüvvədə qalır! Susma halında şriftlərin rəngi qara olur. Əgər Siz mətnin ümumi rəngini əvvəldən dəyişmək istəyirsinizsə, onda BODY teqinin TEXT atributundan istifadə edə bilərsiniz. Yazılış qaydası belədir:

<BODY TEXT="#0000FF"> və ya <BODY BGCOLOR=Blue>

Bu zaman susma halında şriftin rəngi BODY teqinin TEXT atributunda göstərilən rəng olacaq.

<h1> - <h6> teqləri HTML başlıqlarını təyin etmək üçün istifadə olunur. <h1> ən vacib başlığı təyin edir. <h6> ən az əhəmiyyətli başlığı təyin edir.

**** - teqləri arasında yazılmış mətn hissəsi qalın şrifflərlə yazılır.

**** etiketi güclü əhəmiyyətə malik mətni təyin etmək üçün istifadə olunur. İçindəki məzmun ümumiyyətlə qalın şəkildə göstərilir.

<i> və </i> Mətni (sözü, hərfi) maili (Italic) şriftlə vermək üçün istifadə olunur.

<u> və </u> teqi Mətnin (sözün, hərfin) altından xətt çəkmək üçün istifadə olunur.

**
** etiketi bir sətir fasiləsi əlavə edir. **
** etiketi boş etiketdir, yəni son etiketi yoxdur.

<hr> etiketi HTML səhifəsindəki tematik bir fasiləni təyin edir (məsələn, mövzunun dəyişməsi). **<hr>** elementi ən çox HTML səhifəsindəki məzmunu ayırmaq üçün (və ya dəyişikliyi təyin etmək üçün) istifadə olunan üfüqi bir qayda olaraq göstərilir.

<div> etiketi bir HTML sənədindəki bölməni təyin edir. **<div>** etiketi HTML elementləri üçün bir qab kimi istifadə olunur - daha sonra CSS ilə tərtib edilir və ya JavaScript ilə idarə olunur. Hər hansı bir məzmunu **<div>** etiketinin içinə qoymaq olar!

**** etiketi HTML səhifəsinə şəkil yerləşdirmək üçün istifadə olunur. Şəkillər texniki olaraq veb səhifəyə daxil edilmir; şəkillər veb səhifələrə bağlıdır. **** etiketi istinad olunan görüntü üçün saxlama sahəsi yaradır.

**** etiketində iki tələb olunan atribut vardır:

src - Görünüşə gedən yolu müəyyənləşdirir

alt - Şəkil nədənsə göstərilə bilmirsə, şəkil üçün alternativ mətn təyin edir.

<a> etiketi bir səhifədən digərinə keçid üçün istifadə olunan köprüyü təyin edir. **<a>** elementinin ən vacib xüsusiyyəti, əlaqənin təyinatını göstərən *href* atributudur.

Varsayılan olaraq, bütün brauzerlərdə bağlantılar aşağıdakı kimi görünəcəkdir:

- Dəvət olunmayan bir əlaqə altından xətt çəkilmiş və mavi rəngdədir
- Ziyarət olunan bir keçid altından xətt çəkilmiş və bənövşəyi rəngdədir
- Aktiv bir link altından xətt çəkilmiş və qırmızı rəngdədir

Şərh – HTML-sənəddəki bu mətn hissəsi brauzer tərəfdən görünmür (rəddedilir). Şərh ilk növbədə sənədin müəllifinə lazımdır. Müəllif burada öz qeydlərini (tarix, vaxt, versiya, xüsusi əlamətlər və s.) göstərə bilər. Şərh mətnində istənilən yerdə ol bilər. Yazılış forması:

<!-- Şərhin mətni --> **<!-- Burada yazılan mətn ekranda görünmür. -->**

Mühazirə 2

HTML-də başlıqlar yaratmaq

Bir veb saytın müvəffəqiyyətində başlıqlar böyük rol oynayır. Birincisi, oxucuların səhifələrinizin məzmununu skan etməsini asanlaşdırırlar. İkincisi, bəlkə də daha əhəmiyyətli, veb səhifələrinizin quruluşunu axtarış motorları ilə əlaqələndirirlər və buna görə də tez-tez məzmununuzun axtarış sisteminin nəticələrində sıralanmasına təsir göstərir..

Dediğiniz kimi, mətninizi böyük və ya qalın hala gətirmək üçün başlıq yazılarından istifadə etməmək vacibdir. Bunun üçün istifadə edilə bilən digər etikətlər də var. Bunun əvəzinə başlıq etikətləri yalnız quruluş üçün istifadə edilməlidir.

HTML dili altı səviyyə sərlövhələrlə işləməyə imkan verir. Birinci səviyyə ən əsasıdır. Ona daha çox diqqət yetirilir. Digər sərlövhələr isə məsələn, qalın şriftlə və ya çəpinə hərflə göstərilə bilər.

HTML-də birinci səviyyəli sərlövhə **<H1>** kimi işarə edilir:

Burada **n** sərlövhənin səviyyəsini bildirir, yəni 1, 2, 3, 4, 5 və ya 6. HTML-də birinci səviyyəli sərlövhə sənədin adı ilə üst-üstə düşə bilər.

<H1> və **</H1>** Birinci səviyyəli sərlövhə.

<H2> və **</H2>** İkinci səviyyəli sərlövhə.

<H3> və **</H3>** Üçüncü səviyyəli sərlövhə.

<H4> və **</H4>** Dördüncü səviyyəli sərlövhə.

<H5> və **</H5>** Beşinci səviyyəli sərlövhə.

<H6> və **</H6>** Altıncı səviyyəli sərlövhə.

Üfqi xətkəş

<HR> Üfqi xətkəş. Mətni üfqi xətlə məntiqi ayırır. Bunun üçün vacib yerdə **<HR>** teqi qoymaq lazımdır.

Yalnız **<HR>** teqini yazdıqda xətkəş səhifənin bütün enini tutacaq.

<HR WIDTH="75"> Bu halda üfqi xətt ekran səhifəsinin tam eninin 75%-ni tutacaq.

Bütöv bir paraqraf

HTML dilində sətirləri bölmək *prinsipial* mənə kəsib etmir. Bu o deməkdir ki, Siz sənədinizin sətirlərini istənilən yerdə bölə bilərsiniz. Bu onunla əlaqədardır ki, hipermətn sənəddə ardıcıl sətirlərdə gələn mətn hissələri bütöv bir mətnə çevrilir. Sizin çox sayda probel verməklə ara məsafəni böyütmək cəhdiniz heç bir nəticə verməyəcək.

AMMA! Əgər mətn hissəsi **<P>** və ya **
** teqindən sonra gəlibsə, onda bu hissə yeni sətirdən başlayacaq. Fərq yalnız ondadır ki, mətn **<P>** teqindən sonra gələndə ondan əvvəl əlavə sətir əmələ gəlir, **
** teqindən sonra gələndə isə belə boş sətir əmələ gəlmir. Dediklərimizdən belə çıxır ki, 1 **<P>** teqinin yaratdığı effekti almaq üçün 2 **
** teqi yazmaq lazımdır.

Şriftləri idarəetmə

Şriftin ölçüsünü dəyişmək üçün **** teqini **SIZE=** atributu ilə tətbiq etmək lazımdır. Burada 2 variant ola bilər:

**** və **** (****)

Birinci variantda şriftin mütlət ölçüsü verilir, ikinci halda isə şriftin cari ölçüsündən 1 vahid böyük (kiçik) ölçü götürülür. HTML dilində şriftlər üçün şərti vahidlə göstərilən yeddi (1-7) əsas ölçü götürülüb.

Axtarış sistemlərində web səhifələrin tapılması. SEO nədir? SEO baxımından üstün teqlər.

SEO ingilis dilində **Search Engine Optimization** sözünün qısaltması və öz Azərbaycan dilinə axtarış sistemi optimallaşdırma xidməti kimi tərcümə edə biləcəyimiz SEO axtarış sistemi içərisində istifadəçilərin veb səhifələrini daha asan bir şəkildə araşdırmasına, axtarış sistemi üzərində reyting kimi bəzi funksiyalarda yuxarıda çıxmasına imkan təmin edən daxili və xarici texniki tənzimləmə işləridir. Bununla birlikdə axtarış sistemi; axtarış nəticələrinin siyahısını alqoritmik, yəni sıralamaq üçün riyaziyyat bir struktur istifadə etməkdədir. Buna görə də veb proqramçısı yazdığı və meydana gətirdiyi səhifələri bu detala diqqət edərək yaratmaq vəziyyətindədir. İnternetin iqtisadi mənadakı gücünün artması nəticəsində SEO sürətlə yayılmağa başlamışdır.

SEO işində daxili və xarici işlər aparılır. Edilən bu işlər, həm saytın internet üzərində istifadəçiyə daha sürətli, daha təmiz məlumat verməsi, həm də axtarış sistemi siyahısında yuxarı sıralamalarda çıxarmaq saytın daha çox istifadə edilməsini məqsədini yaradır.

SEO baxımından üstünlük verilən teqlər

Title teqi - bu teqin kontenti **70** hərfdən çox olmamlıdır.

Başlıq teqləri -Başlıq teqləri hansı ki (<h1>)-(<h6>) kontentin təyin olunmasında köməklik edir.

Alt atributu — saytda istifadə olunan hər şəkil üçün ‘alt’ atributu vacibdir. Çünki Google şəkilləri oxumur, ancaq şəklin ‘alt’ atributunda yazılanları oxuyub tanıyır.

Strong teqi – mətninizi qalın şriftlərlə yazmaq üçün istifadə olunur. teqi ilə görünüşü eynidir.

Bəs və teqləri arasında fərq nədir? teqi əsasən sözləri vurğulamaq üçün istifadə olunur. Axtarış sistemlərində bu teq daxilində olan sözlərə də çox üstünlük verilir. Və SEO baxımından bu daha düzgündür.

 teqi - <i> teqi ilə eyni funksiyanı icra edir mətni maili şriftlə vermək üçün istifadə olunur. SEO baxımından teqindən istifadəyə üstünlük verilir.

<address> **teqi** – web sahifəmizdə ünvdan istifadə edirik sə bu teq daxilində yazılmalıdır. Axtarış sistemlərində ünvan axtarılan zaman axtarış sistemləri birinci bu teqlərin daxilinə nəzər yetirir.

<blockquote> **teqi**- qeydlər əsasən bu teq daxilində yazılır. Teqləri çox uzun sitat vermək üçün istifadə olunur. Bu teqlər arasında yazılan mətn hissəsi (bir sətirə yerləşməyən mətn) sol tərəfdən abzas qədər məsafədə yazılır.

<q> **teqi** – mətni (sözü, simvolu) dırnağ işarəsi daxilində (“”) yazmaq üçün istifadə olunan teqdir. Teq daxilində olan fraqmentin vurğulanmasını təmin edir. Bu BLOCKQUOTE elementinə oxşardır. BLOCKQUOTE elementi qeyd edilmiş mətni (sitatı) abzas qədər məsafədə yazırdısa, Q elementi qeyd edilmiş mətni abzas daxilində seçilməsi üçün istifadə olunur.

Ümumilikdə aşağıdakılar nəzərə alınmalıdır:

‘Title’ — uzunluğu 70 simvoldan az olmalıdır.

‘Heading’ teqləri — maksimum 1 ədəd h1 teqi, maksimum 4 ədəd h2 teqi və max 4 ədəd h3 teqi olmalıdır.

Daxili link sayı — 100 linki keçməməlidir.

Xarici link sayı — sayı 20 linki keçməməlidir. Lazımsız saytlara link verilsə, bunlar ‘rel’ atributunun qiyməti ‘nofollow’ ilə verilməlidir.

‘Alt’ atributu — saytda istifadə olunan hər şəkil üçün ‘alt’ atributu vacibdir. Çünki Google şəkilləri oxumur, ancaq şəklin ‘alt’ atributunda yazılanları oxuyub tanıyır.

HTML həcmi — bu həcm nə qədər az olarsa, saytın kodlaşdırılması o qədər düzgün qəbul edilmiş olar. Bir saytda html həcmi(tutumu) 32kb-ı keçməməlidir.

Domain yaşı — Saytların yaşı SEO baxımından vacib bir kriteriyadır. Saytların yaşı Google üçün təcrübə kimi qəbul olunur və vacib kriteriya kimi SEO-da yer tutur.

DIV teqi

Div html səhifəsində bir bölmə yaratmaq istədiyimiz zaman istifadə olunur. <div> etiketi blok etiketidir, beləliklə eni dəyəri vermədiyimiz halda sətri olduğu kimi əhatə edir və içərisindəki elementlərin hündürlüyünə bərabər bir hündürlüyə sahibdirlər.

Misal üçün; Bir sayt dizaynını bölmələrə ayıraq

Ən kənar hissədə konteyner adlı div yaratdıq.

```
<div id="container">
  <div id="header">

  </div>

  <div id="content">

  </div>
</div>
```

Bu konteyner divini yaratmağın məqsədi eyni zamanda bütün məzmunu birgə idarə etməkdir.

Div-lərə verdiyimiz genişlik dəyərlərinin “px”-də olması vacib deyil, faiz ilə də dəyər verə bilərik. Məsələn, brauzerin genişliyi nə olursa olsun, <div> -nin genişliyinin səhifənin 70% olmasını istəyə bilərik.

Bir <div> üçün hündürlük dəyəri versək və içindəki elementlərin hündür-lüyü div hündürlüyünü aşsa, onda daşma üçün bəzi seçimlərimiz var.

overflow : auto | hidden | scroll | visible

Auto : məzmunun hündürlüyü sabit dəyərin hündürlüyündən yüksəkdirsə, sürüşmə çubuğu, yəni şaquli və üfüqi sürüşmə çubuqları görünəcək, əks halda görünməyəcəkdir.

Hidden : bu vəziyyətdə daşan hissələr görünməz hala gətirilir.

Scrool : Avtomatikdən fərqli olaraq, sürüşmə çubuqları hər zaman görünür, heç daşma olmasa da, sürüşmə çubuqlarını passiv şəkildə görə bilərik.

Visible : daşan hissələr görünür.

Qrafikanın HTML-sənədə salınması

Təsvirin qrafik fayldan daxil edilməsi (Inline Image)

HTML-sənədə **GIF**, **JPEG** və bəzi digər formatlı qrafik fayllar daxil etmək olar. WWW – də əsasən **GIF** və **JPEG** formatlı fayllardan istifadə edilir.

Yazılışı

<IMG SRC="faylın ünvanı"

ALIGN= left | right | top | texttop | middle | absmiddle | bottom | absbottom

WIDTH=nnn HEIGHT=nnn HSPACE=nnn VSPACE=nnn ALT="alternativ mətn">

Şəklin səhifədə yerləşməsini bildirən ALIGN, WIDTH, HEIGHT, HSPACE və VSPACE atributları vacib deyil. *Alternativ mətni* bildirən ALT atributu isə vacibdir. Çünki o qrafik olmayan baruzerlərdə istifadə olunur. SRC atributu brauzerə qrafik faylın yerləşdiyi yer haqqında məlumat verir.

<audio> teqi

<audio> etiketi musiqi və ya digər səs axınları kimi sənədlərə səs məzmunu daxil etmək üçün istifadə olunur.

<audio> və </audio> etiketləri arasındakı mətn yalnız <audio> elementini dəstəkləməyən brauzerlərdə görünəcəkdir.

HTML-də üç dəstəklənən səs formatı var: MP3, WAV və OGG.

<video> teqi

<Video> etiketi, video klip və ya digər video axınları kimi bir sənəddə video məzmunu yerləşdirmək üçün istifadə olunur.

<video> və </video> etiketləri arasındakı mətn yalnız <video> elementini dəstəkləməyən brauzerlərdə görünəcəkdir.

HTML-də üç dəstəklənən video formatı var: MP4, WebM və OGG.

Brauzer qrafik faylı harada axtarmalıdır?

Brauzerin HTML-sənədə qoşulmuş şəkli tapması üçün qrafik faylın WWWfəzasında adını və yerləşdiyi yeri vermək lazımdır. HTML-sənədlə qrafik faylın bir-birinə nəzərən yerləşməsindən asılı olaraq qrafik faylın ya **mütləq** URL-ünvanı ya da HTML-sənədin faylının yerləşdiyi qovluğa nəzərən **nisbi** URL-ünvanı göstərilməlidir.

İndi 3 vəziyyəti araşdıraq:

1. Qrafik fayl HTML-sənədin yerləşdiyi eyni serverdə və eyni qovluqda yerləşir;
2. Qrafik fayl HTML-sənədin yerləşdiyi eyni serverdə, lakin digər qovluqda yerləşir;
3. Qrafik fayl və HTML-sənəd müxtəlif serverdə yerləşir.

Bu 3 halın hər birində faylın mütləq (tam) URL-ünvanını vermək olar. Mütləq URL-ünvanda protokolun adı, serverin ünvanı, portun nömrəsi (əgər susma halında qəbul olunmuş 80 sayılı portdan fərqlənsə), qovluğun adı və faylın adı göstərilir.

Lakin

birinci iki halda mütləq URL-ünvanı vermək nəinki, **izaflikdir**, həm də **məqsəduyğun**

deyil.

Birinci halda yalnız faylın adını vermək kifayətdir.

Yazılışı

Məsələn:

İkinci halda faylın adından əlavə faylın yerləşdiyi qovluğun adını vermək kifayətdir. Sənədi WWW-serverdən yükləyəndə əsas (ana) qovluq kimi serveri sazlayanda verilən **document root directory** göstərilir. Lokal diskdən yükləyəndə

əsas (ana) qovluq kimi fayl sistemindəki əsas (ana) qovluq başa düşüləcək. Qovluqların adı *UNIX* sisteminin qaydalarına uyğun verilir:

- Altqovluqlar *DOS*-dakı kimi **tərsinə cəpinə xəttlə** (\) deyil, **adi** cəpinə xəttlə (/) ayrılırlar;
- "/" ünvanın əvvəlində olanda ünvanın başlanğıcı **əsas** qovluqdan, əks halda HTML-sənədin olduğu qovluqdan hesablanır;
- bir nöqtə "." cari qovluğu, iki nöqtə ".." isə cari qovluqdan bir səviyyə yuxarı qovluğu bildirir..

Yazılışı

Yazılışı

Beləliklə, aydın olur ki, fayl HTML-sənədə nə qədər yaxındırsa, onun koordinatları haqqında bir o qədər az məlumat verilir.

Şəklin eni və uzununu (Width and Height)

WIDTH və HEIGHT atributlarının köməyi ilə şəklə piksellərlə ölçü verilir. Bunun üstünlüyü ondadır ki, brauzer HTML-sənədi yükləyəndə şəkil hələ yüklənməmiş, ölçülərini bildiyi üçün, ona ekranda yer ayırır və şəkil yüklənənə qədər mətni ekranda göstərə bilir. Bu isə əsas mətni tez oxumağa imkan yaradır. Şəklin ölçüsünü bilməyəndə, brauzer əvvəlcə şekli yükləyir, ondan sonra şəklin ölçüsünə uyğun mətni və digər elementləri ekranda yerləşdirir. Gec yüklənən səhifə həmişə əsəbləri oynadır və çox vaxt onu sonadək yükləmədən prosesi dayandırır. Çox vaxt sonralar belə səhifəyə bir daha müraciət etmirlər.

Mühazirə 3

Cədvəllərin qurulması

HTML- in ən çox istifadə olunan vasitələrindən biri də cədvəllərdir. Cədvəllərdən həm də **web** səhifənin formalaşdırılması vasitəsi kimi də istifadə olunur.

Cədvəllərdən istifadə ancaq sıralarda və sütunlarda dayanan verilənlər ilə məhdudlaşmır. Cədvəllərin tətbiqindən biri də müxtəlif tipli verilənlərin (sadə verilənlərin, təsvirlərin, başqa səhifələrin və s.) səhifədə yerləşməsini təşkil edir.

Əvvəlcə sadə cədvəllərin yaradılması üçün istifadə edilən teqləri nəzərdən keçirək. Cədvəlin təsviri sənəd bölməsi teqi olan **<BODY>**- nun daxilində olmalıdır. Bir sənəddə istənilən sayda cədvəl ola bilər və hətta bir- birinin daxilində olan cədvəllər ola bilər. Hər bir cədvəl **<TABLE>** teqi ilə başlamalı və **</TABLE>** teqi ilə qurtarmalıdır, yəni cədvəl bu teqlər cütünü daxilində təsvir edilməlidir. Hər bir cədvəl ən azı bir sətirdən, hər bir sətir isə ən azı bir xanadan ibarət olmalıdır.

Cədvəlin hər bir sətiri **<TR>** (**Table Row**) teqi ilə başlamalı və **</TR>** teqi ilə isə qurtara bilər. Sətrin hər bir xanası isə **<TD>** və **</TD>** (**Table Data**) və ya **<TH>** və **</TH>** (**Table Header**) teqləri cütləri ilə əhatə olunur. **<TH>** adətən cədvəlin başlıq, **<TD>** isə verilənlər xanaları üçün istifadə olunur. İstifadə fərqləri xananın məzmununun təsvirində susma zamanı istifadə olunan şriftlərin tipində və həm də xana daxilində verilənlərin yerləşməsindədir. **<TH>** teqi ilə verilən xanaların məzmunu **bold** (qalın) şriftlə xananın mərkəzində (**align="center"**, **valign="middle"**) təsvir edilir. **<TD>** teqi ilə təyin edilən xanalar da isə verilənlər susmaya görə sola düzləndirmə (**align="left"**) və şaquli istiqamətdə ortada (**valign="middle"**) təsvir olunurlar.

<TD> və **<TH>** teqlərindən yalnız cədvəl sətrinin təsviri zamanı istifadə edilə bilər (yəni cədvəl sətrinin təsvirindən kənarda istifadə edilə bilməz). Sonluq teqləri olan **</TR>**, **</TD>** və **</TH>** yazılmaya da bilər. Bu halda sətrin və ya xananın sonu yeni sətrin və ya xananın başlanğıcı və ya cədvəlin sonu ola bilər. Cədvəlin **</TABLE>** teqi isə buraxıla bilməz.

Cədvəldə sətirlərin sayı **<TR>** açılma teqlərinin sayı ilə, sütunların sayı isə maksimal saylı **<TD>** və ya **<TH>**- ları saxlayan sətirdəki say ilə təyin olunur. Xanaların bəzilərinə heç bir verilən olmaya da bilər. Belə xanalar **<TD>** və **</TD>** teqlər cütünü ilə təsvir oluna bilər. Əgər hər hansı bir sətrin sonundakı bir və ya bir neçə ardıcıl xanada heç bir verilən saxlanılırsa onda, onları təsvir etməmək də olar. Bu halda brauzer sətirə avtomatik olaraq tələb olunan qədər boş xana əlavə edir. Buradan belə çıxır ki, müxtəlif sətirlərində müxtəlif saylı eyni ölçülü xanaları olan cədvəl qurmaq olmaz.

Cədvəl **<CAPTION>** və **</CAPTION>** teqləri arasında verilən başlıqla verilə bilər. Cədvəlin başlığının təsviri **<TABLE>** və **</TABLE>** teqləri daxilində **</TR>**, **</TD>** və ya **</TH>** teqlərinin təsir dairəsindən başqa istənilən yerdə verilə bilər.

Susmaya görə isə başlığın mətni cədvəlin üstündə (**align="top"**) yeləşir və üfqi istiqamətdə mərkəzləşdirilir.

Sadalanmış teqlər müxtəlif saylı və ölçülü atributlara malik ola bilər. Adətən isə bir çox hallarda bu teqlər atributsuz istifadə olunurlar. Bu halda onların qiyməti susmaya görə təyin edilir.

Aşağıdakı misalı nəzərdən keçirək. Burada **<TABLE>** teqində **border** atributundan istifadə edək ki, cədvəlin xanalarının kənarına haşiyə çəkilsin (**border** atributundan tam istifadəni öyrənəcəyik). Məsələn:

```
<TABLE border>
<TR>
<TD>xana 1.1</TD>
<TD>xana 1.2</TD>
```

```

</TR>
<TR>
<TD>xana 2.1</TD>
<TD>xana 2.2</TD>
</TR>
</TABLE>

```



xana 1.1	xana 1.2
xana 2.1	xana 2.2

Cədvəlin təsvirində istifadə olunan atributları nəzərdən keçirək.

<CAPTION> elementi

Cədvəlin başlıq teqi olan <CAPTION> yalnız **top** (başlıq cədvəlin üst hissəsində) və ya **bottom** (başlıq cədvəlin alt hissəsində) qiymətlərini alan **align** atributuna malikdir. Əgər **align** atributu buraxılmışsa onda **align = "top"** qiyməti nəzərdə tutulur və cədvəlin başlığı üfüqi istiqamətdə həmişə onun mərkəzində yerləşdirilir. Cədvəlin başlığı olmayada bilər. Cədvəlin başlığı kimi sadə bir mətindən də istifadə edə bilərik. <CAPTION> və </CAPTION> teqləri arasına <BODY> bölməsində istifadə edilən ixtiyari **HTML** elementini yazı bilərik. Cədvəlin başlığının yazılmasına aid misal olaraq

```
<CAPTION align="bottom">
```

başlıq cədvəlin alt hissəsindədir

```
</CAPTION>
```

göstərə bilərik.

Internet Explorer-də <CAPTION> elementinin **align** atributu başlığın üfüqi istiqamətdə düzləndirilməsi üçün **left**, **center**, **right** qiymətlərini də ala bilər. Məsələn:

<CAPTION align="right"> yazılışı başlığın cədvəlin üst hissəsində sağa düzləndirilərək yerləşməsinə təmin edir.

Nəzərə almaq lazımdır ki, bir başlıqda **align** atributunu iki dəfə istifadə etmək olmaz. Buna görə də **top** və ya **bottom** qiymətlərini alan xüsusi əlavə şaquli düzləndirmə **valign** atributu daxil edilmişdir. Məsələn:

```
<CAPTION align="left" valign="bottom">
```

Cədvəlin aşağısında sola düzləndirilmiş başlıq

```
</CAPTION>
```

Belə yazılış başlığı cədvəlin aşağısında sola düzləndirərək yerləşdirir.

Internet Explorer brauzerində cədvəlin başlıq ilə verilməsinə aid misalı nəzərdən keçirək.

```
<TABLE border>
```

```
<CAPTION align="left" valign="bottom">
```

Başlıq cədvəlin aşağısında sola düzləndirilərək yerləşir

```
</CAPTION>
```

```
<TR>
```

```
<TD>xana 1.1</TD>
```

```
<TD>xana 1.2</TD>
```



```

</TR>
<TR>
<TD>xana 2.1</TD>
<TD>xana 2.2</TD>
</TR>
</TABLE>

```

xana 1.1	xana 1.2
xana 2.1	xana 2.2

Başlıq cədvəlin aşağısında sola düzləndirilərək yerləşir

<TABLE> teqinin atributları

<TABLE> teqi bir sıra atributlardan istifadə edə bilər. Onlardan hər hansının buraxılması mümkündür. Mümkün atributlar yığımı brauzerdən asılıdır. <TABLE> teqində **border**, **cellspacing**, **cellpadding**, **width**, **align** atributlarından istifadə etmək olar. Netscape və Internet Explorer brauzerləri sadalanan atributlardan başqa **height** və **bgscolor** atributlarından da istifadə etməyə icazə verir. Atributların mənalərini nəzərdən keçirək.

Border atributu cədvələ haşiyəni əlavə etmək üçün nəzərdə tutulub. O ədədi qiymət ala bilər.

Atributun formatı aşağıdakı kimidir:

border="x"

burada x- cədvəlin haşiyəsinin qalınlığının piksel sayını təyin edən ədədi qiymətdir. Bəzən bu qiymət hər xananın haşiyəsinə təsir etmir. Məsələn:

<TABLE border >

<TABLE border ="0">

Ədədi qiymət buraxılan zaman o minimal qiymət alır (yəni 1). Müxtəlif brauzerlərdə haşiyənin stili fərqlənə bilər.

Məsələn haşiyəsinin qalınlığı 10 piksel olan cədvəl aşağıdakı kimi olar.

1-ci sətir 1-ci xana	1-ci sətir 2-ci xana
2-ci sətir 1-ci xana	2-ci sətir 2-ci xana

Qeyd etmək lazımdır ki, **border** atributu buraxılırsa, haşiyə çəkilmir. Belə cədvələ aid misala baxaq. Bu proqram fraqmentində <TABLE> **border** atributu olmadan istifadə edilmişdir.

```

<TABLE>
<TR>
<TD>1-ci sətir 1-ci xana </TD>
<TD>1-ci sətir 2-ci xana</TD>

```

```

</TR>
<TR>
<TD>2-ci sətir 1-ci xana</TD>
<TD>2-ci sətir 2-ci xana</TD>
</TR>
</TABLE>

```

1-ci sətir 1-ci xana	1-ci sətir 2-ci xana
2-ci sətir 1-ci xana	2-ci sətir 2-ci xana

Bəzi brauzerlər **border** atributunun verilmiş ədədi qiymətini qəbul etməyə bilər, onda sıfır qəbul ediləcək.

Bir neçə misala baxaq.

```

<TABLE border >
<TABLE border ="0">
<TABLE >

```

Internet Explorer üçün ikinci və üçüncü misallar eynigüclüdür. Bu brauzer üçün **border** atributunun qiyməti susmaya görə sıfırdır.

Cellspacing atributu həm üfüqi həm də şaquli xanalar arasında məsafəni vermək üçün təyin edilib.

Atributun formatı aşağıdakı kimidir:

cellspacing="x"

burada x- atributun piksel sayını təyin edən ədədi qiymətdir, hansı ki, buraxıla bilməz. Əgər atribut buraxılırsa susmaya görə **cellspacing="2"** kimi təyin edilir. Adi nəşriyyat sistemlərində istifadə olunan cədvəllərdə xanalar arasında məsafə olmur. HTML cədvəllərində susmaya görə xanalar arasında məsafə olur. Bu əvvəlki misalda aydın görünür.

Aşağıda atributun **cellspacing="20"** qiyməti ilə bir misala baxaq.

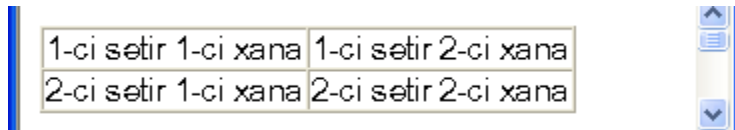
```

<TABLE border cellspacing="20">
<TR>
<TD>1-ci sətir 1-ci xana </TD>
<TD>1-ci sətir 2-ci xana</TD>
</TR>
<TR>
<TD>2-ci sətir 1-ci xana</TD>
<TD>2-ci sətir 2-ci xana</TD>
</TR>
</TABLE>

```

1-ci sətir 1-ci xana	1-ci sətir 2-ci xana
2-ci sətir 1-ci xana	2-ci sətir 2-ci xana

cellspacing="0" verilən zaman xanalar arasında məsafə olmur.



1-ci sətir 1-ci xana	1-ci sətir 2-ci xana
2-ci sətir 1-ci xana	2-ci sətir 2-ci xana

Cellpadding atributu xana daxilindəki verilənlə xananın haşiyəsi arasındakı məsafəni təyin edir. Atributun ümumi formatı **cellspacing** atributunda olduğu kimidir.

cellpadding="x"

burada x- atributun piksel sayını təyin edən ədədi qiymətdir.

Aşağıda atributun **cellpadding="20"** qiyməti ilə bir misala baxaq.

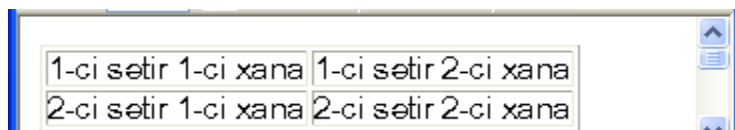
```
<TABLE border cellpadding="20">  
<TR>  
<TD>1-ci sətir 1-ci xana </TD>  
<TD>1-ci sətir 2-ci xana</TD>  
</TR>  
<TR>  
<TD>2-ci sətir 1-ci xana</TD>  
<TD>2-ci sətir 2-ci xana</TD>  
</TR>  
</TABLE>
```

Cədvəl aşağıdakı kimi olacaq.



1-ci sətir 1-ci xana	1-ci sətir 2-ci xana
2-ci sətir 1-ci xana	2-ci sətir 2-ci xana

cellpadding="0" olarsa xanadakı mətnin müəyyən hissəsi xananın haşiyəsinə toxuna bilər.



1-ci sətir 1-ci xana	1-ci sətir 2-ci xana
2-ci sətir 1-ci xana	2-ci sətir 2-ci xana

Cədvəlin təsvir olunması zamanı onların eni və hündürlüyü bir çox faktordan asılı olaraq avtomatik hesablanır. Bu vaxt cədvəli elə qurmağa çalışırlar ki, cədvəl brauzer rəncərəsinə yerləşsin.

Əksəriyyət brauzerlər mətni avtomatik formalaşdırır ki, mətnin uzunluğu pəncərənin enindən artıq olmasın. Belə halda üfüqi sürüşdürmə çubuğuna ehtiyac olmur. Analoci olaraq brauzer cədvəli elə formalaşdırır ki, cədvəlin eni pəncərənin enindən artıq olmasın.

Qeyd etmək lazımdır ki, cədvəlin eni əsas sayılır. Hündürlüklə müqayisədə birinci növbədə cədvəlin eni hesablanır. Bəzi hallarda konkret olaraq cədvəlin enini və hündürlüyünü göstərmək lazımdır. Bunun üçün **<TABLE>** teqinin **width** (cədvəlin eni) **height** (cədvəlin hündürlüyü) atributlarından istifadə olunur.

Bu atributların ümumi formatı aşağıdakı kimidir.

width="x" və ya

width="x%"

height="x" və ya

height="x%"

burada x- cədvəlin eninin və hündürlüyünün piksel sayını, x% isə bütün pəncərənin faiz ölçüsünü göstərən ədədi qiymətdir. Qeyd etmək lazımdır ki, qiyməti 100%- dan artıq da vermək olar. Məsələn:

<TABLE width="150%">

Anoloci olaraq atributlar ayrı- ayrı xanalar üçün də verilə bilər.

<TABLE> teqinin **align** atributu cədvəlin pəncərəsində üfiqi yerləşməni təyin edir. Atribut **left** (sola düzləndirmə), **right** (sağa düzləndirmə) və **center** (ortaya (mərkəzə) düzləndirmə) qiymətlərini ala bilər. Əgər **align** atributu buraxılırsa susmaya görə cədvəl sola düzləndiriləcək.

<TABLE> teqində **align** atributunun olması təkcə cədvəlin yerləşməsini təmin etmir, həm də cədvəlin yanına mətnin və ya şəkilin axmasına icazə verir. Cədvəlin hər iki tərəfinə mətnin axması mümkün deyil, yəni **align="center"** qiymətini alarsa cədvəl mərkəzə düzləndiriləcək və onun tərəflərinə mətnin axması baş verməyəcək. **Align** atributu buraxılırsa cədvəlin enindən asılı olaraq onun sağında boş sahə qoyulacaq və mətnin axması baş verməyəcək. Əgər cədvəlin yanına mətnin axması lazım deyilsə, onda onu pəncərənin mərkəzinə yerləşdirmək olar. Bunun üçün **<TABLE>** teqində **align="center"** atributundan və **<CENTER>** və **</CENTER>** və ya **<DIV>** və **</DIV>** teqlərindən istifadə etmək olar. Məsələn:

<DIV align="center">

<TABLE>

<TR>

<TD>

Cədvəl brauzer pəncərəsinin mərkəzinə düzləndirilir

</TD>

</TR>

</TABLE>

</DIV>

<CENTER>

<TABLE>

<TR>

<TD>

Cədvəl brauzer pəncərəsinin mərkəzinə düzləndirilir

</TD>

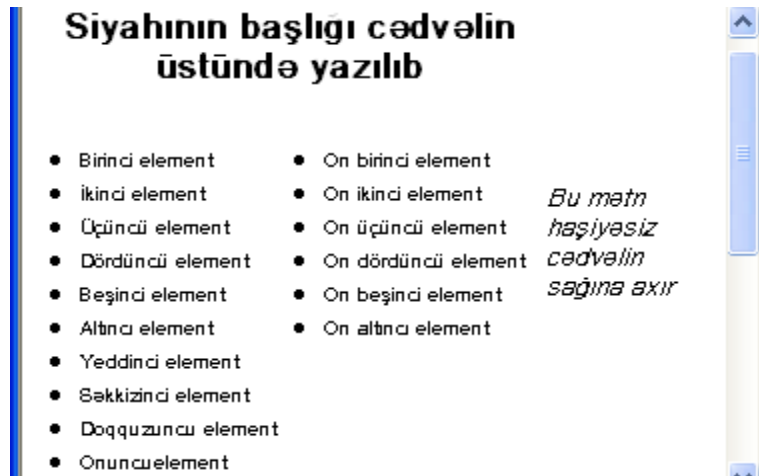
```
</TR>
</TABLE>
</CENTER>
```

```
<TABLE align="center">
<TR>
<TD>
Cədvəl brauzer pəncərəsinin mərkəzinə düzləndirilir
</TD>
</TR>
</TABLE>
```

Cədvəlin yanında mətn yazmaqla proqramm fragmentinə baxaq.

```
<TABLE align="left" width="75%">
<CAPTION>
<H3>Siyahının başlığı cədvəlin üstündə yazılıb</H3>
</CAPTION>
<UL>
<TR> <TD valign="top">
<Li>Birinci element<Li>İkinci element
<Li>Üçüncü element<Li>Dördüncü element
<Li>Beşinci element<Li>Altıncı element
<Li>Yeddinci element<Li>Səkkizinci element
<Li>Doqquzuncu element<Li>Onuncuelement
</TD> <TD valign="top">
<Li>On birinci element<Li>On ikinci element
<Li>On üçüncü element<Li>On dördüncü element
<Li>On beşinci element<Li>On altıncı element
</TD> </TR>
</UL>
</TABLE>
<BR><BR><BR><BR><BR><BR>
<I>Bu mətn haşiyəsiz cədvəlin sağına axır</I>
```

Cədvəlin təsviri aşağıdakı kimi olar



Bu sənədə izahat verək. O **align="left"** atributu ilə haşiyəsiz cədvəldən ibarətdir və cədvəlin sağına mətnin axmasına imkan verir. Cədvəl iki xanası olan bir sətirdən ibarətdir. Hər xanada müəyyən **** nömrələnməmiş siyahı var. Cədvələ siyahının bu cür daxil olunması cədvəlin bir neçə sütununa siyahının məcburi daxil olunma üsullarından biridir. Mətn cədvəlin sağına axıb. Mətnin hamısı cədvəlin sağına yerləşməyə də bilər (əgər böyükdürsə). Bu zaman mətnin ardı cədvəldən sonra yerləşəcək. Əgər brauzer pəncərəsi müəyyən vaxt kiçildilərsə, onda mətnin müəyyən hissəsi və ya mətn bütövlükdə cədvəlin altına düşə bilər.

Cədvəlin kənarındakı mətnin məcburi qırılmasını təşkil etmək üçün (məsələn, əgər növbəti mətn məntiqi olaraq cədvəllə əlaqəli deyilsə, onda o cədvəlin altında yerləşməlidir) **clear** atributunu əlavə etməklə **
** təqindən istifadə etmək lazımdır.

**
** təqinin **clear** atributu aşağıdakı qiymətləri ala bilər.

left- növbəti sətir axan obyektin ardında yaxın sətirdə sol sahədən başlayacaq

all- növbəti sətir axan obyektin ardında yaxın sətirdə istənilən sahədən başlayacaq

right- növbəti sətir axan obyektin ardında yaxın sətirdə istənilən sahədən başlayacaq

none- növbəti sətir adi halda olduğu kimi.

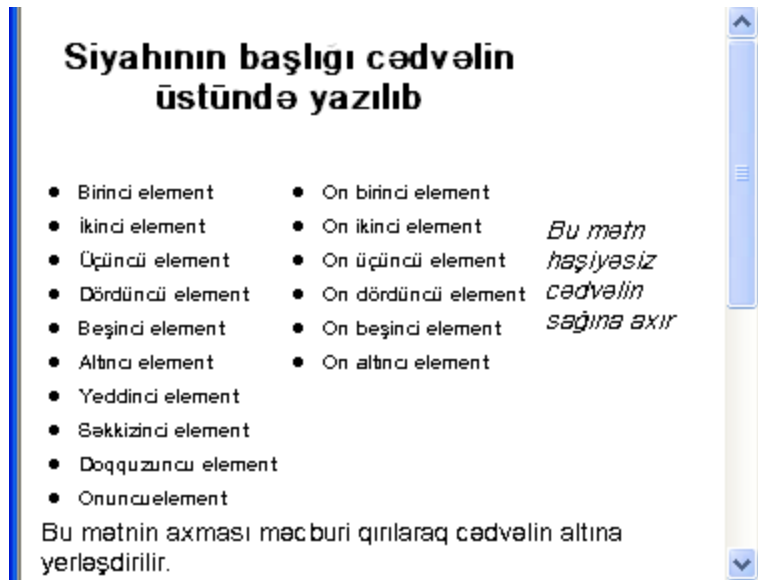
İndiki halda sürüşkən obyekt cədvəldir. Yuxarıdakı misalda

<I>Bu mətn haşiyəsiz cədvəlin sağına axır</I>

sətirdən sonra

<BR clear="left"> Bu mətnin axması məcburi qırılaraq cədvəlin altına yerləşdirilir.

sətirini yazsaq sənədin brauzer pəncərəsində təsviri aşağıdakı kimi olar.



Qeyd etmək lazımdır ki, **clear="left"** əvəzinə **clear="all"** yazmaq olar.

Bu tapşırığı **
** teqini bir neçə dəfə **clear** atributsuz yazmaqda da yerinə yetirmək olar (əvvəlki misalda mətnin bir neçə sətir aşağı sürüşdürüldüyü kimi).

Əvvəlki misala oxşar başqa bir misalı nəzərdən keçirək.

```
<TABLE>
<CAPTION>
<H3>Siyahının başlığı <BR>
cədvəlin üstündə yazılıb</H3>
</CAPTION>
<UL>
<TR>
<TD valign="top" width="40%">
<Li>Birinci element<Li>İkinci element
<Li>Üçüncü element<Li>Dördüncü element
<Li>Beşinci element<Li>Altıncı element
<Li>Yeddinci element<Li>Səkkizinci element
<Li>Doqquzuncu element<Li>Onuncuelement
</TD>
<TD valign="top" width="40%">
<Li>On birinci element<Li>On ikinci element
<Li>On üçüncü element<Li>On dördüncü element
<Li>On beşinci element<Li>On altıncı element
</TD>
<TD width="20%">
<I>Bu mətn haşiyəsiz cədvəlin sağına axmır üçüncü sütunda yerləşir</I>
</TD>
</TR>
</UL>
```

</TABLE>

Siyahının başlığı cədvəlin üstündə yazılıb		
• Birinci element	• On birinci element	<i>Bu mətn həşiyəsiz cədvəlin sağına axmır üçüncü sütunda yerləşir</i>
• İkinci element	• On ikinci element	
• Üçüncü element	• On üçüncü element	
• Dördüncü element	• On dördüncü element	
• Beşinci element	• On beşinci element	
• Altıncı element	• On altıncı element	
• Yeddinci element		
• Səkkizinci element		
• Doqquzuncu element		
• Onuncu element		

Əvvəlki iki misaldan fərqli olaraq burada mətn cədvəlin sağına axmır üçüncü xanada yerləşir. Bütövlükdə səhifə üç xanası və bir sətiri olan başlıqlı cədvəldən ibarətdir. Praktiki olaraq birinci iki xana bütün üç sənəddə eynidir. Başlıqlarda fərq var. Birinci iki sənəddə başlıq ilk iki sütunun ortasında yerləşir. Axıncı sənəddə isə başlıq bütün üç sütunun ortasında yerləşir.

Cədvəl daxilində verilənlərin formatlaşdırılması

Adətən cədvəl sətrinin elanı olan <TR> teqi iki atributla tamamlanır: xananın içərisindəkiləri **align**- üfiqi və **valign**- şaquli düzləndirmək. Bu atributlar <TD> teqi vasitəsi ilə cədvəlin xanasının yaradılmasında da istifadə edilir.

<TR> teqinin **align** atributu **left**, **right** və **center** qiymətlərini də ala bilər. Məsələn:

<TR align="right">

<TR> teqinin **valign** atributu **bottom**, **top** və **middle** qiymətlərini də ala bilər. Məsələn:

<TR valign="middle">

Cədvəlin daxilindəki hər bir xanaya formatlaşdırmaq üçün müstəqil oblast kimi baxmaq olar. Mətni formatlaşdırmaq üçün istifadə edilən bütün qaydalardan xana daxilindəki mətni formatlaşdırmaq üçün istifadə etmək olar.

<TD> teqinə ətraflı baxaq. O təyinatlarına görə müxtəlif qruplarda birləşən çoxlu atributlara malikdir.

- Əsas atributlar

- Sətir və sütun xanalarını birləşdirən atributlar.

- Spesifik atributlar

Əsas atributların siyahısı aşağıdakı cədvəldə göstərilmişdir.

Atribut və qiyməti	Mənası
width="x" və ya width="x%"	Bu atribut xananın eninin piksel sayını (x) və ya cədvəlin eninin faiz ölçüsünü (x%) göstərən ədədi qiymətdir. Burada x tam

"	ədəddir.
align="x"	Bu atribut xanadakı verilələrin üfiqi istiqamətdə düzləndirmə üsulunu təyin edir. Atribut left , right və center qiymətlərini ala bilər
valign="x"	Bu atribut xanadakı verilələrin şaquli istiqamətdə düzləndirmə üsulunu təyin edir. Atributun bottom , top və middle qiymətlərini ala bilər

Aşağıda üç xanlı cədvəlin **HTML** proqramm fraqmentinə baxaq.

```
<TABLE border="1" cellspacing="0" cellpadding="0"
width="350" height="200">
```

```
<TR>
```

```
<TD width="125" align="left" valign="top">
```

Bu cədvəlin birinci xanasıdır. Onun eni 125 pikseldir. Xana daxilindəkilər üfiqi istiqamətdə sola şaquli istiqamətdə isə yuxarı düzləndirilmişdir.

```
</TD>
```

```
<TD width="125" align="right" valign="middle">
```

Bu cədvəlin ikinci xanasıdır. Onun eni 125 pikseldir. Xana daxilindəkilər üfiqi istiqamətdə sağa şaquli istiqamətdə isə ortaya düzləndirilmişdir.

```
</TD>
```

```
<TD width="100" align="center" valign="bottom">
```

Bu cədvəlin üçüncü xanasıdır. Onun eni 100 pikseldir. Xana daxilindəkilər üfiqi istiqamətdə ortaya şaquli istiqamətdə isə aşağı düzləndirilmişdir.

```
</TD>
```

```
</TR>
```

```
</TABLE>
```

Sədvəl brauzer pəncərəsində aşağıdakı kimi təsvir ediləcək.

Bu cədvəlin birinci xanasıdır. Onun eni 125 pikseldir. Xana daxilindəkilər üfiqi istiqamətdə sola şaquli istiqamətdə isə yuxarı düzləndirilmişdir.	Bu cədvəlin ikinci xanasıdır. Onun eni 125 pikseldir. Xana daxilindəkilər üfiqi istiqamətdə sağa şaquli istiqamətdə isə ortaya düzləndirilmişdir.	Bu cədvəlin üçüncü xanasıdır. Onun eni 100 pikseldir. Xana daxilindəkilər üfiqi istiqamətdə ortaya şaquli istiqamətdə isə aşağı düzləndirilmişdir.
--	---	--

Sətir və sütun xanalarını birləşdirmək üçün **</TD>** teqində uyğun olaraq **colspan** və **rowspan** atributlarından istifadə edilir.

Onların formatı aşağıdakı kimidir.

colspan="x"

burada x- sətirdəki bir xanada birləşəcək xanaların sayıdır.

rowspan="x"

burada x- sütundakı bir xanada birləşəcək xanaların sayıdır.

Əgər **colspan** atributu istifadə edilməzsə brauzer boşluqları bərpa etmək üçün öz düzəlişlərindən istifadə etməyə cəhd edir.

Birinci sətirdə dörd xanası və hər birinin eni 100 piksel, ikinci sətirdə iki xanası, birincinin eni 100, ikincinin isə eni 300 piksel olan cədvəlin **HTML** proqramm fraqmentinə baxaq.

```
<TABLE border="1" cellspacing="0" cellpadding="0"
      width="400">
```

```
<TR>
```

```
<TD width="100" align="left" valign="top">
```

Birinci sətirin birinci xanası

```
</TD>
```

```
<TD width="100" align="left" valign="top">
```

Birinci sətirin ikinci xanası

```
</TD>
```

```
<TD width="100" align="left" valign="top">
```

Birinci sətirin üçüncü xanası

```
<TD width="100" align="left" valign="top">
```

Birinci sətirin dördüncü xanası

```
</TD>
```

```
</TR>
```

```
<TR>
```

```
<TD width="100" align="left" valign="top">
```

İkinci sətirin birinci xanası

```
<TD width="300" align="left" valign="top">
```

İkinci sətirin ikinci xanası

```
</TD>
```

```
</TR>
```

```
</TABLE>
```

Məntiqi olaraq fərz etmək olardı ki, brauzer aşağıdakı sətirin birinci xanasının enini 100 piksel, ikinci xanasının enini birinci sətirdəki axırıncı üç xananın enlərinin cəminə 300 pikselə bərabər olan cədvəl əks etdirilməlidir. Amma aşağıdakı şəkildən göründüyü kimi bu belə olmadı.

Birinci sətrin birinci xanası	Birinci sətrin ikinci xanası	Birinci sətrin üçüncü xanası	Birinci sətrin dördüncü xanası
İkinci sətrin birinci xanası	İkinci sətrin ikinci xanası		

Əgər **colspan="3"** (birləşəcək xanaların sayı) atributunu istifadə etsək arzu olunan nəticəni almaq olar. Bunun üçün

```
<TD width="300" align="left" valign="top">
```

İkinci sətrin ikinci xanası

sətirlərinin əvəzinə

```
<TD width="300" align="left" valign="top" colspan="3">
```

İkinci sətrin ikinci xanası

sətirlərini yazmaq lazımdır.

Yeni **HTML** proqramına görə brauzer cədvəli aşağıdakı kimi təsvir ediləcək.

Birinci sətrin birinci xanası	Birinci sətrin ikinci xanası	Birinci sətrin üçüncü xanası	Birinci sətrin dördüncü xanası
İkinci sətrin birinci xanası	İkinci sətrin ikinci xanası		

rowspan atributu da **colspan** atributuna anoloci olaraq sütun xanalarını birləşdirmək üçün istifadə edilir.

Aşağıdakı proqram fragmentini nəzərdən keçirək.

```
<TABLE width="400" border="1" cellpadding="0"
cellpadding="0">
```

```
<TR>
```

```
<TD width="100" align="left" valign="top" rowspan="2">
```

Bu birinci və ikinci sətirlərin birinci xanasının birləşməsindən alınan xanadır

```
</TD>
```

```
<TD width="100" align="left" valign="top">
```

Birinci sətrin ikinci xanası

```
</TD>
```

```
<TD width="100" align="left" valign="top">
```

Birinci sətrin üçüncü xanası

```
<TD width="100" align="left" valign="top">
```

Birinci sətrin dördüncü xanası

```
</TD>
```

```
</TR>
```

```
<TR>
```

```
<TD width="100" align="left" valign="top">
```

İkinci sətrin ikinci xanası

```

<TD width="100" align="left" valign="top">
İkinci sətrin üçüncü xanası
<TD width="100" align="left" valign="top">
İkinci sətrin dördüncü xanası
</TD>
</TR>
</TABLE>

```

Cədvəlin brauzer pəncərəsində təsviri aşağıdakı kimidir.

Bu birinci və ikinci sətirlərin birinci xanasının birləşməsindən alınan xanadır	Birinci sətirin ikinci xanası	Birinci sətirin üçüncü xanası	Birinci sətirin dördüncü xanası
	İkinci sətirin ikinci xanası	İkinci sətirin üçüncü xanası	İkinci sətirin dördüncü xanası

Hər iki **rowspan** və **colspan** atributundan birgə istifadə etməklə başqa bir misala baxaq.

```

<TABLE border="1">
<TR>
<TD colspan="2" rowspan="2">
İlk iki sətir və iki sütunun xanalarının birləşməsindən alınan xana
<TD colspan="2">
Birinci sətirin 3-cü və 4-cü xanalarının birləşməsindən alınan xana
</TR>
<TR>
<TD>Xana 2.3
<TD>Xana 2.4
</TR>
<TR>
<TD>Xana 3.1
<TD>Xana 3.2
<TD>Xana 3.3
<TD>Xana 3.4
</TR>
</TABLE>

```

İlk iki sətir və iki sütunun xanalarının birləşməsindən alınan xana		Birinci sətirin 3-cü və 4-cü xanalarının birləşməsindən alınan xana	
		Xana 2.3	Xana 2.4
Xana 3.1	Xana 3.2	Xana 3.3	Xana 3.4

Xananın bir sıra əlavə atributları (spesfik atributlar) mövcuddur. Onların adları və mənalari aşağıdakı cədvəldə göstərilmişdir.

Atribut və qiymət	Mənası
bgcolor="x"	Bu atribut susmaya görə təyin edilmiş fon rəngindən fərqli fon rəngini seçməyə imkan verir. Burada x- rəngin adı və ya 16- lıq say sistemində kodudur.
Background="URL"	Bir neçə brauzer o cümlədən Netscape və Internet Explorer xanaya fon şəkli yerləşdirməyə imkan verir. Burada <i>URL</i> - təsvir faylının mütləq və ya nisbi URL -dir.
height="x"	Bu atribut xananın hündürlüyünü təyin etməyə imkan verir.

Əgər <TABLE>, <TR> və <TD> teqlərinin hər hansı birində və ya hamısında **BGCOLOR** istifadə edilərsə <TD> teqindəki **BGCOLOR** <TR>-də olan **BGCOLOR**-u, <TR>-də olan **BGCOLOR** <TABLE> -də olan **BGCOLOR**-u ləğv edir.

Bu deyilənləri aşağıdakı misala tətbiqində ətraflı görmək olar.

```
<TABLE BORDER=3 BGCOLOR="#FF6633">
```

```
<TR BGCOLOR="#009900">
```

```
<TD BGCOLOR="#9999FF">XANA 1.1</TD>
```

```
<TD>XANA 1.2</TD>
```

```
<TD>XANA 1.3</TD>
```

```
</TR>
```

```
<TR>
```

```
<TD>XANA 2.1</TD>
```

```
<TD>XANA 2.2</TD>
```

```
<TD>XANA 2.3</TD>
```

```
</TR>
```

```
</TABLE>
```

XANA 1.1	XANA 1.2	XANA 1.3
XANA 2.1	XANA 2.2	XANA 2.3

Aşağıdakı sadə bir misalı nəzərdən keçirək. Tutaq ki, kompüter komplektində aşağıdakı siyahıda göstərilən qurğular var.

Kompüter komplektindəki qurğular:

- Sistem bloku
- Monitor
- Klaviatura

- Multimedia
- Siçan

Yaradılmış siyahının qarşısında kompüter komplektinin şəkilini göstərək.

Bunun üçün cədvəldən istifadə edək. Şəkili birinci xananın sağ tərəfinə sıxıb siyahını ikinci xanaya yerləşdirib cədvəlin sərhəd xəttini götürək.

```
<TABLE BORDER=0 WIDTH=100%>
```

```
<TR>
```

```
<TD WIDTH=40% ALIGN=RIGHT>
```

```
<IMG SRC=" computer.jpg " WIDTH=130
```

```
HEIGHT=80></TR>
```

```
<TD WIDTH=60%>
```

```
Kompüter komplektindəki <BR>
```

```
qurğuların siyahısı:
```

```
<UL>
```

```
<LI>Sistem bloku
```

```
<LI>Monitor
```

```
<LI>Klaviatura
```

```
<LI>Multimedia
```

```
<LI>Siçan
```

```
</UL>
```

```
</TD>
```

```
</TR>
```

```
</TABLE>
```

Mühazirə 4

Formalar

Formaların internet səhifəyə necə əlavə edilməsini nəzərdən keçirək. Formalar **<FORM>** teqi vasitəsi ilə internet səhifəyə əlavə edilir. **<FORM>** teqi vasitəsi ilə internet səhifəyə sifariş forması, abzas, əks əlaqə olaraq başqa əlavələr və sairə əlavə edə bilərsiniz. Aşağıdakı misala baxaq.

<FORM> Formanın başlanğıcı

<INPUT> informasiyani bir və ya bir necə üsulla daxil edir

<INPUT>..... istifadəçinin istədiyi qədər oblastlar daxil edə bilər.

<FORMS> Formanın sonu

Bundan sonra biz **brauzer**– ə göstərməliyik ki, verilənləri hara göndərək, bunun üçün nə tələb olunur və göndərmə hansı üsulla yerinə yetirilməlidir. İki əsas üsul var:

1. Verilənləri **JGİ** senarisinə e'mal etmək üçün göndərmək

2. Siz verilənləri elektron poçt vasitəsi ilə özünüə göndərə bilərsiniz.

Birinci halda **JGİ** senarisinin müəllifi göndərilənlər haqqında sizə məlumat verməlidir.

İkinci halda forma **mailto <FORM>** teqində tə'yin olunmuş atributa malik olmalıdır.

Fikir verin **Misrosoft İnternet Explorer 3.0 mailto** formanı tanıyır. Nə vaxt ki, informasiyanı ötürməyə cəhd etsəniz yeni poçt pəncərəsi meydana gələcək. **Explorer JGİ** ssenarisinə ötürülən verilənlər formasını tanıyır.

<FORM METHOD=POST ACTION= "mailto: xxx@xxx.xxx" ENCTYPE=
"aplication/x-www-Form-urlencoded">

</FORMS>

Bu sətir çox vacibdir. Ancaq **mailto**-dan sonra öz **email** ünvanınızı yazın. Qalanları necə yazılıbsa elə də qalmalıdır. **FORM, METHOD, POST** və **ACTION** sozlərinin böyük və ya kiçik hərflərlə yazılmasının fərqi yoxdur, ancaq bunlar arasında boşluq qoyulması vacibdir.

<INPUT >

Bu element bayraq, çevirici (radio düymə), daxilətmə sahəsi və sairə kimi formanın müxtəlif hissələrini yaratmağa imkan verir. Element sonuncu teq tələb etmir və bütün parametrlər atributların köməyi ilə verilir.

Elementin görünüşünü **TYPE** atributu təyin edir.

type="text"- daxilətmə sahəsi yaradır, hansı ki, istənilən mətni avtomatik yerləşdirmək olar. **value** atributu istifadə olunur;

type="password"- parolun daxil edilməsi üçün sahənin yaradılması. Parol daxil edilərkən bütün simvollar ulduzlarla göstəriləcək;

type="checkbox"- bayrağın yaradılması;

type="radio"- bir çeviricini (radio düymənin) yaradır. Bir qrup çevirici yaratmaq üçün **INPUT** elementini bir neçə dəfə istifadə etmək lazımdır.

type="button"- ixtiyari təyinatlı düymə yaradır;

type="submit"- düymənin yaradılması, hansı ki, üzərində siçanın sol düyməsini vurmaqla formaya informasiyanın daxil edilməsini təsdiqləyir. **Value** atributu düymə üzərindəki yazını təyin etmək üçün istifadə olunur;

type="reset"- bu da düymədir, formaya daxil edilmiş verilənləri ləğv edir;

type="image"- şəkilli düymə yaratmaq üçün istifadə olunur. Qrafik faylı göstərmək üçün **src** atributundan istifadə olunur.

type="file"- formaya birləşdirmək üçün faylı seçmə vasitələri. İstifadəçiyə təklif olunur ki, daxiletmə sahəsinə faylın adını yazsın. Bundan əlavə brauzer avtomatik daxiletmə sahəsi yanında görünüş (**obzor**) düyməsi yaradır, hansı ki, faylları seçmək üçün dialoq rəncərəsi açır;

Qalan atributlar elementin xüsusiyyətlərini təyin etmək üçün lazımdır. Bunlardan əksəriyyəti verilənləri emal etmək üçün məcburi sayılır.

name atributu təsdiq etmə və təmizləmə düymələrindən başqa bütün **INPUT** elementində olmalıdır. Bu atributun qiyməti formada sahənin adını təyin edir, yəni bu sahəyə daxil etmək üçün verilənlər blokunu. Serverin proqramı bu ada əsasən lazımi verilənləri seçə bilər.

maxlength atributu iki üsulla istifadə edilə bilər. Birincisi o daxiletmə sahəsinə yazıla bilən sətirin maksimal uzunluğunu təyin edir. İkincisi onun köməyi ilə formaya birləşəcək faylın ölçüsünü məhdudlaşdırmaq olar.

İndi vaxta qənaət etmək üçün yalnız **form** teqi içərsində olanları yazacağıq.

<INPUT TYPE=TEXT >

Hər daxiletməyə ad (**NAME**) tələb olunur. Məsələn hər hansı ünvanı daxil etmək üçün

<INPUT TYPE=TEXT NAME ="ADDRESS">

yaza bilərik. Bu sahəyə daxil edilən mətn **ADDRESS** dəyişənin qiyməti olacaq

Bu sahənin qiymətini **VALUE** atributu ilə **INPUT** -da verə bilərik. Məsələn:

<INPUT TYPE=TEXT NAME ="ADDRESS" VALUE= "BAKU 15">

Əgər istifadəçi bu sahəni dəyişməsə **ADDRESS** dəyişəni avtomatik olaraq **Baku 15** qiymətini alacaq.

Qeyd: dırnaq işarəsini lazım olan yerdə qoymaq lazımdır.

Mətn Blokunun ölçüsünü **SIZE=N** atributu ilə verə bilərik.

Burada **N** daxil ediləcək simvolların sayıdır. Məsələn:

<INPUT TYPE=TEXT NAME ="ADDRESS"

VALUE= "BAKU 15" SIZE=15>

Susmaya görə **SIZE=20** götürülür.

VALUE= "BAKU 15" parametrlərini silək.

<INPUT TYPE=TEXT NAME ="ADDRESS" SIZE=30>

Biz istifadəçinin daxil edəcəyi simvolların sayını da təyin edə bilərik. Bu **MAXLENGTH=N** ilə təyin edilir. Məsələn:

<INPUT TYPE=TEXT NAME ="ADDRESS" SIZE=30 MAXLENGTH=10>

Bu çox vacib elementlərdən biridir.

TYPE=PASSWORD-da **TYPE=TEXT**-ə çox oxşardır. Fərq yalnız bu sahəyə daxil olan simvolun * kimi əks olunmasıdır.

Brauzer sahəyə daxil olmuş veriləni göstərməyəcək lakin sizə real verilənləri göndərəcək.

<INPUT TYPE=PASSWORD>

Yaddan çıxarmayın ki, hər bir <INPUT> **NAME**-ə malik olmalıdır. Məsələn:

<INPUT TYPE=PASSWORD NAME="USER"

PASSWORD”>

Burada da **SIZE**, **VALUE** və **MAXLENGTH** atributları istifadə edilir. Teq brauzerə məlumat verir ki, teq içərisindəki atribut nə etmək haqqında məlumat verir.

Radio düymə və bayraqlar

Bu düymələr verilən variantların seçilməsi üçün istifadə edilir. Radio düymə istifadəçiyə bir neçə variantdan yalnız birini seçməyə imkan verir. İstifadəçi bayraqlardan isə birini və ya bir neçəsini seçə bilər.

Əvvəlcə 1 radio düymə düzəldək.

<INPUT TYPE=RADIO NAME = “Yaxşı dost”>



İndi 2 dənə də əlavə edək.

<INPUT TYPE=RADIO NAME = “Yaxşı dost”>

<INPUT TYPE=RADIO NAME = “Yaxşı dost”>



<INPUT TYPE=RADIO NAME = “Yaxşı dost”>

İndi 1 sətirin sonuna yeni sətirə keçid teqi qoyaq.

<INPUT TYPE=RADIO NAME = “Yaxşı dost”>

<INPUT TYPE=RADIO NAME = “Yaxşı dost”>

<INPUT TYPE=RADIO NAME = “Yaxşı dost”><P>



Fikir verin hər bir sahə eyni ada malikdir.

Hər bir radio düyməyə **VALUE** dəyişənini mənimsədək.

<INPUT TYPE=RADIO NAME = “Yaxşı dost” VALUE =”İsmayıl”>

<INPUT TYPE=RADIO NAME = “Yaxşı dost” VALUE =”Mamed”>

<INPUT TYPE=RADIO NAME = “Yaxşı dost” VALUE =”Arif”><P>



İndi hər bir düyməni adlandırmaq.

<INPUT TYPE=RADIO NAME = “Yaxşı dost” VALUE =”İsmayıl”> İsmayıl

<INPUT TYPE=RADIO NAME = “Yaxşı dost” VALUE =”Mamed”> Mamed

<INPUT TYPE=RADIO NAME = “Yaxşı dost” VALUE =”Arif”> Arif <P>



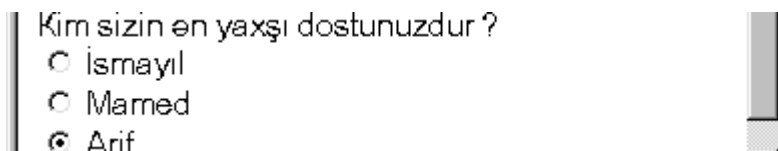
Siz bu nişanları başqa **HTML** teqləri ilə də istifadə edə bilərsiniz. Siz radio düymələrlə işləməyə hazırsınız. Siz düymələrə son görkəm verə bilərsiniz, düymələrdən əvvəl hər hansı qeyd yazı bilərsiniz və əgər istəsəniz düymələrdən birini susmaya görə seçə bilərsiniz.

Kim sizin ən yaxşı dostunuzdur ?

<INPUT TYPE=RADIO NAME = “Yaxşı dost” VALUE =”İsmayıl”> İsmayıl

<INPUT TYPE=RADIO NAME = “Yaxşı dost” VALUE =”Mamed”> Mamed

<INPUT TYPE=RADIO NAME = “Yaxşı dost” VALUE =”Arif” CHECKED > Arif <P>



İstifadəçi yalnız 1 qeydi seçə bilər. Cari vəziyyətdə əgər istifadəçi düyməni dəyişməyə susmaya görə **"Yaxşı dost"** =Arif götürüləcək.

Bayraq sahələrinin düzəldilməsi

Bayraq sahələrinin düzəldilməsi radio düymə sahələrinin düzəldilməsinə çox oxşardır. Məsələn;

```
<INPUT TYPE= CHECKBOX NAME="İsmayıl">
```

İki belə sahə də əlavə edin və hər 1 sahənin adını yazın və əgər istəyirsinizsə növbəti sətirə keçid təqini yazın.

```
<INPUT TYPE= CHECKBOX NAME="İsmayıl"><BR>
```

```
<INPUT TYPE= CHECKBOX NAME="Mamed"><BR>
```

```
<INPUT TYPE= CHECKBOX NAME="Arif"><P>
```

Hər bir bayrağa **VALUE** dəyişəni mənimsədilir.

```
<INPUT TYPE= CHECKBOX NAME="İsmayıl" VALUE ="YES"><BR>
```

```
<INPUT TYPE= CHECKBOX NAME="Mamed" VALUE ="YES"><BR>
```

```
<INPUT TYPE= CHECKBOX NAME="Arif" VALUE ="YES"><P>
```

Fikir verin bayraqlar üçün ad (**NAME**) dəyişdikdə dəyişən **VALUE** olduğu kimi qalır, radiodüymələrdə isə dəyişən **VALUE** dəyişəndə həmin vaxt ad **NAME** də dəyişirdi.

```
<INPUT TYPE= CHECKBOX NAME="İsmayıl" VALUE ="YES"> İsmayıl  
<BR>
```

```
<INPUT TYPE= CHECKBOX NAME="Mamed" VALUE ="YES"> Mamed  
<BR>
```

```
<INPUT TYPE= CHECKBOX NAME="Arif" VALUE ="YES"> Arif <P>
```

Əgər istəyirsinizsə bayraqların yuxarı tərəfinə hər hansı mə'lumatı yazı bilərsiniz və onlardan hər hansını susmaya görə seçə bilərsiniz.

Aşağıdakılardan hansı sizin dostunuzdur ?


```
<INPUT TYPE= CHECKBOX NAME="İsmayıl" VALUE ="YES" CHECKED > İsmayıl  
<BR>
```

```
<INPUT TYPE= CHECKBOX NAME="Mamed" VALUE ="YES"> Mamed <BR>
```

```
<INPUT TYPE= CHECKBOX NAME="Arif" VALUE ="YES" CHECKED > Arif <P>
```

İstifadəçi bir, bir neçə və hətta bütün sahələri seçə bilər ya da heç birini seçməyə bilər.

Yuxarıdakı misalda əgər istifadəçi heç bir dəyişiklik etməzsə susmaya görə

İsmayıl=YES

Arif=YES qiymət alır.

Əgər istifadəçi heç bir sahəni seçməzsə heç nə əldə edə bilməyəcək.

Aşağıdakılardan hansı sizin dostunuzdur ?

☒ İsmayıl

☐ Mamed

☒ Arif

<LABEL></ LABEL>

Bu element formanın başqa elementləri üçün istifadə olunur. Məsələn, bu elementin köməyi ilə yazını daxil etmə sahəsinə birləşdirmək olar.

<label> sizin ünvan: <input type="text" id="ünvan"></label>

Əgər elementi və başqa element ayrılıqda yerləşirsə onda, **for** atributundan istifadə etmək lazımdır. Bu atributun qiyməti uyğun elementdəki **id** atributunun qiyməti ilə uyğun gəlməlidir.

<label for="ünvan">sizin ünvan:</label>

Hər bir **LABEL** elementi üçün bir forma elementi yaradılır.

LABEL elementindən istifadə etməklə bir forma yaradaq.


```
<form action="mailto:ad@server.domen" method="post">
```

| | <tr> |
 adi: | || | |
 familiyasi: | || <tr> |
 <label for="telefon">telefonu:</label> | || <tr> |
 cinsi: |☐K ☐Q <td align="bottom"> |

adi:	Abbasov
familiyasi:	Adil
telefonu:	345-33-76
cinsi:	<input checked="" type="radio"/> K <input type="radio"/> Q
<input type="button" value="gonder"/> <input type="button" value="temizle"/>	

Bu misalda cədvəl forma elementlərini düzləndirmək üçün istifadə olunur. Bunu yoxlamaq üçün formanı doldurub öz elektron poçtunuza göndərə bilərsiniz. Bunun üçün proqramda real

elektron poçt ünvanınızı yazmalısınız. Bundan sonra **göndər** düyməsinə basın. O **isxodəhie** poçt proqramı qovluğunda yerləşəcək. Bir neçə dəqiqədən sonra öz elektron poçtunuzu yoxlaya bilərsiniz.

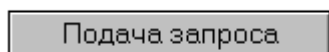
INPUT elementindən istifadə etməklə bir neçə düymənin düzəldilməsinə baxaq.

<INPUT TYPE=file name="myfile">



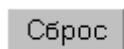
Bu düymə vasitəsi ilə istifadəçilər sərt diskdəki fayllardan istifadə edə bilər.

<INPUT TYPE=submit>



Bu düymə vasitəsi ilə istifadəçi sorğunu göndərə bilər.

<INPUT TYPE=reset>



Bu düymə vasitəsi ilə istifadəçi formanı təmizləyə bilər.

INPUT elementinə **value** atributunu əlavə etməklə düymələrin adını dəyişmək olar.

<SELECT></SELECT>

Bu teqlər vasitəsi ilə açılan siyahı yaratmaq olar. Siz **<SELECT>** teqini **<INPUT>** teqinin yerində yazın. Bu teqdə bağlanan teqdir.

<SELECT>

</SELECT>

İndi ad vermək lazımdır.

<SELECT NAME="YAXSI DOST">

</SELECT>

Sonra isə bir nəşə ad daxil edin.

<SELECT NAME="YAXSI DOST">

<OPTION> ISMAYIL

<OPTION> AZER

<OPTION> AKIF

</SELECT>



Hər **<OPTION>**- a **VALUE** verək və **SELECTED** atributundan istifadə siyahıdan hər hansı birini susmaya görə təyin edə bilərik.

<SELECT NAME="YAXSI DOST">

<OPTION VALUE="ISMAYIL"> ISMAYIL

<OPTION VALUE="AZER" SELECTED >AZER

```

<OPTION VALUE="AKIF">
AKIF
</SELECT>

```



Sürüşdürmə düymələrindən istifadə etməklə siyahının yaradılması açılan siyahının yaradılmasına oxşardır. Övvəlcə siyahıya bir neçə ad da əlavə edək.

```

<SELECT NAME="YAXSI DOST">
<OPTION VALUE="ISMAYIL"> ISMAYIL
<OPTION VALUE="AZER">AZER
<OPTION VALUE="AKIF"> AKIF
<OPTION VALUE="ANAR"> ANAR
<OPTION VALUE="RAUF">RAUF
<OPTION VALUE="RASIM"> RASIM
</SELECT>

```



İndi **<SELECT>** teqinə **size** əlavə edək ki, neçə nəfərin adını göstərsin.

```

<SELECT NAME="YAXSI DOST" SIZE=3>
<OPTION VALUE="ISMAYIL"> ISMAYIL
<OPTION VALUE="AZER">AZER
<OPTION VALUE="AKIF"> AKIF
<OPTION VALUE="ANAR"> ANAR
<OPTION VALUE="RAUF">RAUF
<OPTION VALUE="RASIM">

```



```

RASIM
</SELECT>

```

Açılan pəncərədə neçə parametrin görünməsini **SIZE** atributu təşkil edir. Burada da siyahıdan hər hansı birini susmaya görə təyin edə bilərik.

```

<SELECT NAME="YAXSI DOST" SIZE=3>
<OPTION VALUE="ISMAYIL">

```

```

ISMAYIL

```

```

<OPTION VALUE="AZER">AZER
<OPTION VALUE="AKIF"> AKIF
<OPTION VALUE="ANAR"> ANAR
<OPTION VALUE="RAUF">RAUF
<OPTION VALUE="RASIM" SELECTED> RASIM
</SELECT>

```



<TEXTAREA></TEXTAREA>

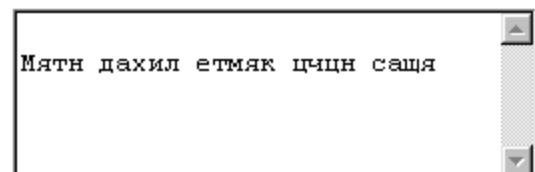
Bu elementin köməyi ilə mətni daxil etmək və mətnə baxmaq üçün sahə yaratmaq olar. Bu sahəni aşağıdakı kimi yaratmaq olar.

```

<h2> TEXTAREA elementi
<TEXTAREA name="text001" rows=5 cols=30>
Mətn daxil etmək üçün sahə
</TEXTAREA >

```

TEXTAREA elementi



</h2>

Sahənin ölçüsünü **rows** (sətrlərin sayı) və **cols** (sütunların sayı) atributu vasitəsi ilə verilir.

SELECT və **TEXTAREA** elementləri tək-cə forma daxilində deyil, səhifədə sərbəst də istifadə edilə bilər.

<BUTTON ></BUTTON>

Bu element də **INPUT** elementi kimi düymə yaratmağa imkan verir. Bu sonuncu təqə malik olması ilə **INPUT** - dan fərqlənir. Deməli bu element müəyyən qədər çətin ola bilər. Ümumi yazılış aşağıdakı kimidir.

```
<BUTTON name="Ad" value="submit" type="submit">
```

```
Mətn <IMG src="Fayl.gif" alt="Komentariya"></button>
```

type atributu aşağıdakı qiymətləri ala bilər.

- **button**- bu düymə vurulduqda həmin düməni idarə edən proqram işə düşür;
- **submit**- düymə təsdiq edir ki, forma dolub;
- **reset**- düymə formanı təmizləyir.

Mühazirə 5

Siyahılar

Siyahılar (**list**) **HTML**-ə mətn redaktorlarının təsiri altında daxil edilib.

Siyahılar adi mətndən onunla fərqlənir ki, istifadəçi obyektlərin nömrələnməsini özü aparmır, çünki bu funksiyanı proqram özü yerinə yetirir. Əgər siyahıya yeni obyektlər əlavə edilərsə və ya siyahıdan obyektlər ləğv edilirsə onda, nömrələnmə avtomatik olaraq korreksiya olunur. Siyahıların nömrələnməməsi halları baş verdiyi zaman hər bir obyektin qarşısında proqram markerlər qoyur: dairəciklər, düzbucaqlılar, romblar və ya başqa təsvirlər.

Nəticədə siyahı yaxşı oxunan görünüş alır. Siyahıların yaradılması təqini şərti olaraq iki qrupa bölmək olar: biri siyahının ümumi görünüşünü, ikincisi isə onun daxili strukturunu təyin edir. Siyahılarda standart atributlardan da istifadə etmək olar. Bir neçə görünüşlü siyahılar mövcuddur. Onların vasitəsi ilə nömrələnmiş və nömrələnməmiş siyahı yaratmaq olar. Məsələn:

Nömrələnmiş	Nömrələnməmiş
1. Böyük	• Qırmızı
2. Kiçik	• Göy
3. Uzun	• Köhnə
4. Qısa	• Təzə

Nömrələnməmiş siyahı ()

UL (Unordered List) ən sadə nömrələnməmiş siyahıdır. teqləri siyahının əvvəlində və sonunda yeni sətirə keçidi təşkil edir ki, bununla da siyahını sənədin əsas məzmunundan ayırır.

 (**List İtem**) teqi siyahının hər bir obyektinin qarşısına yazılmalıdır. Ümumiyyətlə vardır amma onu yazmamaq da olar.

Onun ümumi formatı aşağıdakı kimidir.

 obyekt 1

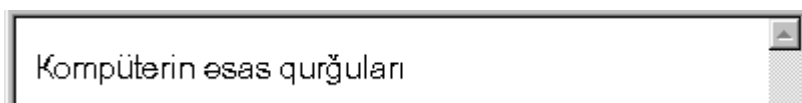
 obyekt 2

Məsələn: kiçik bir siyahı düzəldək və bunu mərhələlərlə göstərək.

<BODY BGCOLOR="#FFFFFF">

Kompüterin əsas qurğuları

</BODY>



Fikir versəniz görərsiniz ki, biz hələ siyahı qurmağa başlamamışıq. Bu hələ ki, başlıqdır. İndi isə proqrama teqlərini əlavə edək və siyahının hər bir obyektlərinin qarşısına teqini yazmaqla siyahını təşkil edək.

<BODY BGCOLOR="#FFFFFF">

Kompüterin əsas qurğuları

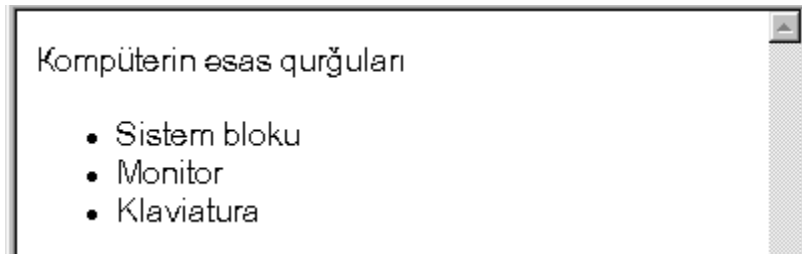
Sistem bloku

Monitor

Klaviatura

</BODY>

Nəticədə ekranda aşağıdakını alırıq.



UL elementində **type** atributlarından istifadə edə bilərik. **type** atributu markerin görünüşünü təyin edir və **square**, **circle**, **disk** qiymətlərini ala bilər.

Markerlər aşağıdakı kimi təyin edilə bilərlər.

- **type= "disk"**- olduqda markerin görünüşü rənglənmiş dairə şəklində olacaq.
- **type= "circle"**- olduqda markerin görünüşü çevrə şəklində olacaq.
- **type= "square"**- olduqda markerin görünüşü kvadrat şəklində olacaq.

Siyahının ayrı- ayrı obyektlərinin markerlərinin görünüşünü **type** atributunu **** elementində istifadə etməklə dəyişmək olar. Əgər **type** atributu aşkar verilməzsə, susmaya görə **type= "disk"** kimi təyin edilir.

Markerin konkret görünüşü istifadə edilən **brauzerdən** asılıdır. Məsələn:

<BODY BGCOLOR="#FFFFFF">

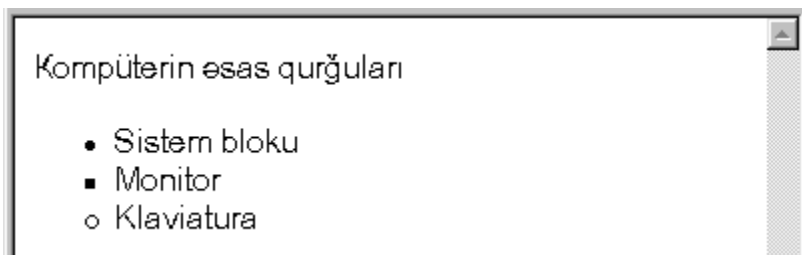
Kompüterin əsas qurğuları

Sistem bloku

<LI type="square">Monitor

<LI type="circle">Klaviatura

</BODY>



Marker kimi qrafiki təsvirlərdən istifadə etmək olar. Əgər siyahılarda marker kimi qrafiki təsvirlərdən istifadə edilsə **** təqindən istifadə etməmək olar. Bunun üçün siyahının hər bir

elementinin qarşısında arzuladığınız qrafiki təsviri qoymaq və siyahının elementlərini bir-birindən ayırmaq lazımdır. (Məsələn, **
** və ya **<P>** teqləri ilə).

```
<BODY BGCOLOR="#FFFFFF">
```

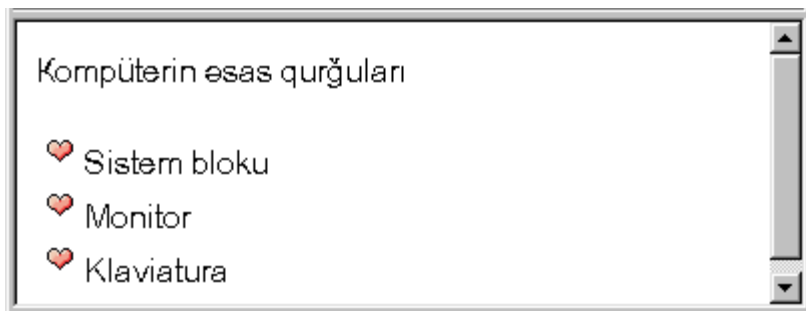
```
Kompüterin əsas qurğuları<p>
```

```
<IMG SRC="urek.gif">Sistem bloku<br>
```

```
<IMG SRC="urek.gif">Monitor<br>
```

```
<IMG SRC="urek.gif">Klaviatura
```

```
</BODY>
```



Nömrələnmiş siyahı (**** ********)

OL (Ordered List) nömrələnmiş siyahısını yaratmaq üçün istifadə edilir. **OL** elementinin strukturu **UL** elementinin strukturu kimidir.

Yuxarıdakı misalda **** teqini **** teqi ilə əvəz edək.

```
<BODY BGCOLOR="#FFFFFF">
```

```
Kompüterin əsas qurğuları
```

```
<OL>
```

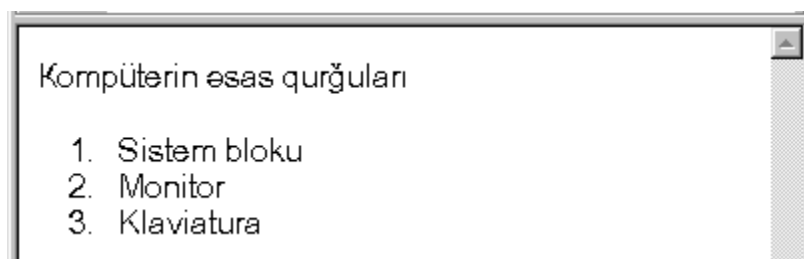
```
<LI>Sistem bloku
```

```
<LI>Monitor
```

```
<LI>Klaviatura
```

```
</OL>
```

```
</BODY>
```



Nömrələrin görünüşünü dəyişmək üçün **OL** elementində **type** atributundan istifadə etmək olar.

type atributu aşağıdakı qiymətləri ala bilər.

Atribut	Nömrələrin görünüşü
type="1"	1, 2, 3, 4,...
type="I"	I, II, III, IV, ...
type="i"	i, ii, iii, iv, ...
type="a"	a, b, c, d,...
type="A"	A, B, C, D,...

Siyahıda **start** atributundan istifadə edərək nömrələnmənin başlanğıcını təyin etmək olar. **Start** atributunun qiyməti kimi nömrələnmənin görünüşündən asılı olmayaraq həmişə natural ədəd götürülür. Məsələn:

```
<OL type="A" start="3">
```

Belə nömrələnmiş siyahı **C**-dən başlayır.

LI elementində **type** atributundan istifadə etməklə nömrələnmənin formasını cari sətir üçün dəyişmək olar.

```
<BODY BGCOLOR="#FFFFFF">
```

Kompüterin əsas qurğuları

```
<OL start="3">
```

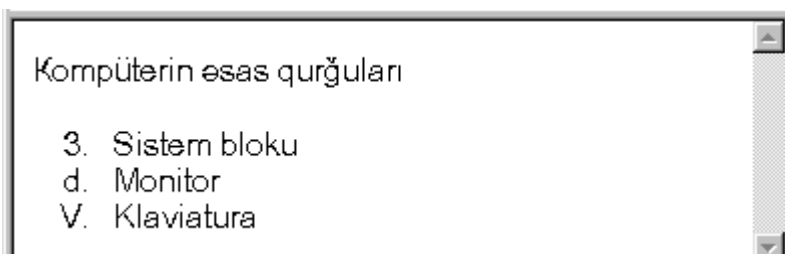
```
<LI>Sistem bloku
```

```
<LI type="a">Monitor
```

```
<LI type="I">Klaviatura
```

```
</OL>
```

```
</BODY>
```



LI elementinin **Value** atributu siyahının verilən elementinin nömrəsini dəyişməyə imkan verir. Bu halda növbəti elementlərin də nömrələnməsi dəyişəcək. Məsələn:

```
<BODY BGCOLOR="#FFFFFF">
```

Kompüterin əsas qurğuları

```
<OL>
```

```
<LI>Sistem bloku
```

```
<LI>Monitor
```

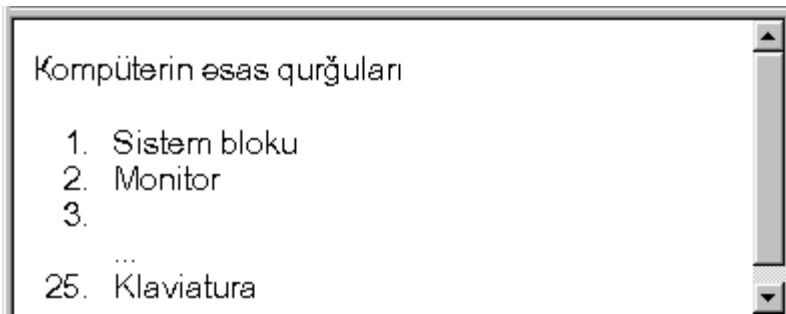
```
<LI><br>
```

...

```
<LI value="25">Klaviatura
```

```
</OL>
```

</BODY>



Tə'yinetmə siyahısı (<DL><DT><DD></DL>)

Tə'yinetmə siyahısı və ya xüsusi terminlərin təyini lüğəti siyahının xüsusi formasıdır. Siyahıların digər tiplərindən fərqli olaraq hər bir təyin etmə siyahısı iki hissədən təşkil edilir. Siyahı elementinin birinci hissəsində təyin edilən termin, ikinci hissəsində isə lüğət cümləsi formasında mətn, yəni terminin mənası yazılır.

Tə'yinetmə siyahısı <DL> (**Definition List**) </DL> elementləri arasında verilir. <DT> və </DT> (**Definition Term**) elementləri vasitəsilə təyin olunan termin qeyd olunur. <DD> və </DD> (**Definition Description**) elementi vasitəsilə isə abzas təyinatı ilə verilir. **DD** və **DT** elementləri **DL** elementinin daxilində verilməlidir. Beləliklə, tə'yinetmə siyahısının ümumi yazılışı aşağıdakı kimidir:

<DL>

<DT> *Termin* </DT>

<DD> *Terminin təyinatı* </DD>

ixtiyari sayda DT və DD elementləri

</DL>

Mətnə <DT> elementindən sonra blok səviyyəli elementlər olan abzas elementi <P> və ya başlıq elementi <H> gələ bilməz. Təyin edilən terminin mətni bir sətirdə yerləşməlidir. Terminin təyinatını verən mətn isə terminin təyindən sonra növbəti sətirdən sağa çəkilmə (*otstup*) ilə (bəzi brauzerlərdə isə bir sətir buraxma ilə) başlanır. Məsələn:

<BODY BGCOLOR="#FFFFFF">

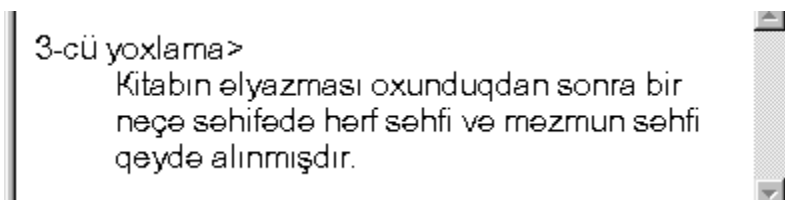
<DL>

<DT> 3-cü yoxlama>

<DD> Kitabın əlyazması oxunduqdan sonra bir neçə səhifədə hərflər və məzmun səhfi qeydə alınmışdır.

</DL>

</BODY>



Bəzi vaxt başlığı qalın yazırlar. Bu da bir qədər oxunaqlı görsənir.

```
<BODY BGCOLOR="#FFFFFF">
```

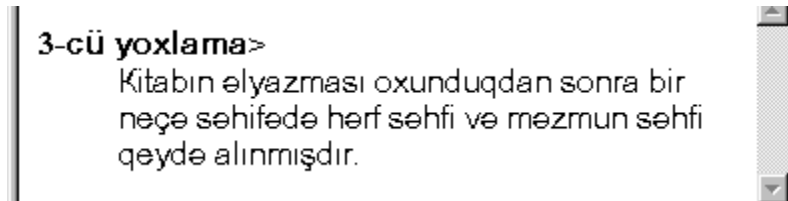
```
<DL>
```

```
<DT> <B> 3-cü yoxlama> </B>
```

<DD> Kitabın əlyazması oxunduqdan sonra bir neçə səhifədə hərf səhfi və məzmun səhfi qeydə alınmışdır.

```
</DL>
```

```
</BODY>
```



DD elementindən sonra gələn informasiyada blok səviyyəli elementlər yerləşə bilər. Buradan çıxır ki, bir- birinin daxilinə qoyulmuş təyinetmə siyahıları ola bilər.

Təyinetmə siyahısında nömrələnmiş və nömrələnməmiş siyahıdan istifadə etməklə aşağıdakı misalı nəzərdən keçirək.

```
<HTML>
```

```
<HEAD>
```

```
<TITLE> Mürəkkəb siyahı</title>
```

```
</head>
```

```
<BODY BGCOLOR="#FFFFFF">
```

```
<CENTER>Yoxlamanın nəticələri</CENTER>
```

```
<DL>
```

```
<DT>Birinci yoxlamada aşağıdakı səhvlər qeydə alındı
```

```
<DD>
```

```
<UL>
```

```
<LI>Məzmun səhvləri
```

```
<LI>Hərf səhvləri
```

```
<LI>İşarə səhvləri
```

```
</UL>
```

```
<DT>İkinci yoxlamada aşağıdakı səhvlər qeydə alındı
```

```
<DD>
```

```
<OL>
```

```
<LI>Hərf səhvləri
```

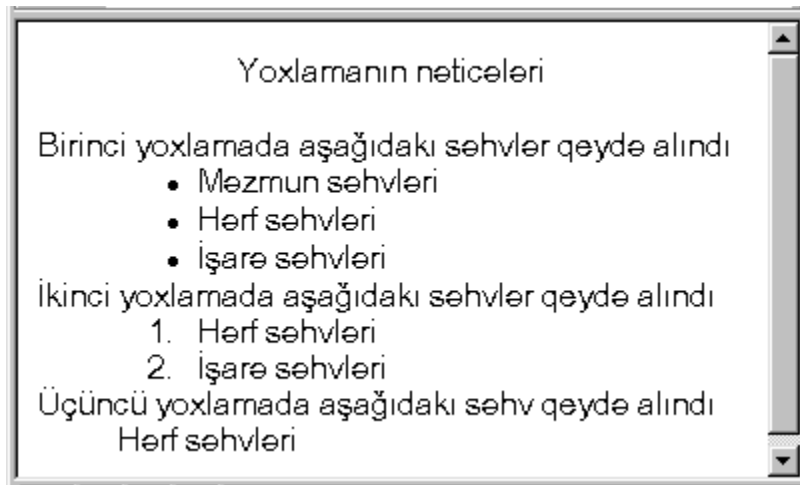
```
<LI>İşarə səhvləri
```

```
</OL>
```

```
<DT>Üçüncü yoxlamada aşağıdakı səhv qeydə alındı
```

```
<DD>Hərf səhvləri
```

</DL>
</BODY>
</html>



CSS-İN KÖMƏYİ İLƏ NƏ ETMƏK OLAR?

CSS HTML-sənədlərin təsvirini müəyyən edən stillərin dilidir. Məsələn, CSS şriftlərlə, rənglə , sahələrlə, sətirlərlə, hündürlüklə, enlə, fon təsvirləriylə, elementlərin yerləşdirilməsiylə və çox başqa şeylərlə işləyir. HTML web-saytların tərtib edilməsi üçün istifadə olunur. Amma səhifədə istəkləri yerinə yetirilməsi üçün CSS daha çox imkanlar yaradır. CSS, bu gün, bütün brauzerlərlə dəstəklənir.

CSS və HTML-in arasında fərq nədir?

HTML səhifənin tərkibinin strukturlaşdırılması üçün istifadə olunur. CSS bu strukturlaşmış tərkibin formatlaşdırılması üçün istifadə olunur.

World Wide Web-i ixtira ediləndə, HTML dili yalnız strukturlaşmış mətnin təsviri üçün istifadə olunurdu. Müəllif yalnız mətni qeyd edə bilirdi: **<h1>** və **<p>** teqindən istifadə edərək bu " başlıqdır" və ya "bu -" paraqrafdır. Web inkişafı müddətində dizaynerlər onlayn sənədlərinin formatlaşdırılmasının imkanlarını axtarmağa başladılar. İstehlakçıların artmış tələblərinə təmin etmək üçün, istehsalçıları yeni HTML-teqləri ixtira etdi Məsələn, HTML-in digər teqlərindən fərqli olaraq səhifənin strukturunu deyil zahiri görünüşündə dəyişiklik edən **** teqi

Bu həmçinin ona gətirib çıxardı ki, strukturlaşdırmanın orijinal teqlərindən olan **<table>** teqi, daha çox mətnin strukturlaşdırılmasının yerinə səhifələrin dizaynı üçün tətbiq edilməyə başladı.Və bəzi teqlər məsələn blink bəzi brauzerlər tərəfindən tanınmırdı.Və səhifənin açılması üçün istifadəçiyə istifadə etməli olduğu brauzer bildirilirdi

CSS bütün brauzerlərin dəstəklədiyi vahid dizayn formasının verilməsi vasitəsi ilə bu problemləri həll etdi Eyni zamanda sənədin tərkibi ilə təqdim edilməsinin ayrılması oldu ki, buda işi əhəmiyyətli dərəcədə sadələşdirdi.

CSS HANSI ÜSTÜNLÜKLƏRİ TƏQDİM EDİR?

CSS yaranması web-dizayn dünyasında inqilab oldu. CSS-in konkret üstünlükləri aşağıdakılardır:

- bir stillərin cədvəlinin(css) köməyi ilə bir çox sənədin təsvirinin idarə olunması;
- səhifələrin zahiri görünüşü üzərində daha dəqiq nəzarət;
- müxtəlif məlumat daşıyıcıları üçün müxtəlif təqdim etmələr ;
- dizaynın mürəkkəb və hazırlanmış texnikası.

CSS necə işləyir?

Tutaq ki bizə HTML səhifəsində arxa fonun rənginin dəyişməsi tələb olunur. Bunun üçün biz

```
<body bgcolor="#FF0000">
```

yazmamız lazımdır. CSS –də bunu etmək üçün isə

```
body {background-color: #FF0000;}
```

yazırıq. Nümunədən göründüyü kimi sintaksislər demək olar ki bir birinə çox yaxındır. Aşağıdakı şəkildə CSS –in fundamental modeli verilmişdir.



CSS KODLARI HARDA YERLƏŞDİRMƏK LAZIMDIR?

HTML sənədinə CSS-in qoşulmasının 3 yolu var. Biz daha çox 3-cü modelin üzərində dayanacağıq.

1. Inline model – style atributu vasitəsi ilə HTML sənəddə teqlərə stillər verilir. Məsələn

```
<html>
<head>
<title>Example</title>
</head>
<body style="background-color: #FF0000;">
<p>This is a red page</p>
</body>
</html>
```

2. Internal model – style teqinin köməyi ilə HTML sənədə daxil edilir. Məsələn

```
<html>
<head>
<title>Example</title>
<style type="text/css">
body {background-color: #FF0000;}
</style>
</head>
<body>
<p>This is a red page</p>
</body>
</html>
```

3. External model – Bu model məsləhət görülən modeldir. Bu halda CSS sənədləri kənarda hazırlanaraq link olaraq HTML sənədinə qoşulur .

Stillərin xarici cədvəli bu .css genişlənməsiylə olan mətn fayllarıdır. Siz web-serverinizə və ya sərt diskə(HDD) başqa fayllar kimi stillərin cədvəlini yerləşdirə bilərsiniz.

Məsələn, tutaq ki, stillərin sizin cədvəli **style.css** adlanır və *style* qovluğundadır. Bu belə təsvir etmək olar:

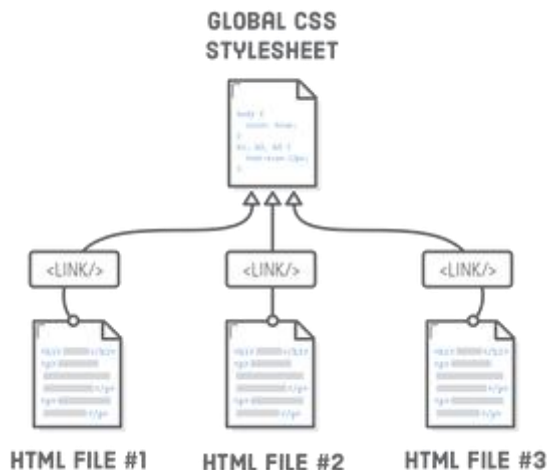


HTML sənədində hazır CSS sənədinə istinad vermək üçün **head** təqinə aşağıdakı kod hissəsini artırmaq lazımdır.

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

Burada əsas diqqət olunmalı məsələ **href** atributunun dəyəridir. Bu istinad (link) brauzerə göstərir ki, o HTML-faylın təsviri üçün CSS-faylındakı qaydalardan istifadə etməlidir.

Ən əhəmiyyətlisi burada odur ki, bir neçə HTML-sənəd bir stillərin cədvəlinə(CSS) istinad edilə bilər. Başqa sözlə, bir CSS-fayl bir çox HTML-sənədin təsvirini idarə etməsi üçün istifadə etmək olar.



Əgər siz, məsələn, 100 səhifədən ibarət web-saytın fonun rəngini dəyişdirmək istəyirsinizsə stillərin cədvəli sizi 100 HTML-sənədini əl ilə dəyişdirmək ehtiyacından qurtaracaq. CSS-dan istifadə edərək, bu dəyişiklikləri sadəcə stillərin mərkəzi cədvəlində bir kodu dəyişdirib bir neçə saniyə ərzində etmək olar.

CSS-də rənglər və fon?

Ön planın rəngi : ‘color’ xüsusiyyəti

Color xüsusiyyəti elementin ön planının rəngini təsvir edir. Məsələn, təsəvvür edin ki, biz sənədin bütün başlıqlarını tünd qırmızı etmək istəyirik. Başlıqlar HTML-də məsələn **<h1>**teqi ilə nümayiş edilir. Aşağıda göstərilən kodda **<h1>** elementinin rəngi qırmızı təyin edilir.

```
h1 {  
color: #ff0000;  
}
```

Rəngləri

- onaltılıq qiymətlərlə göstərmək olar, məsələn nümunədə bu **#ff0000** olaraq təyin edilib,
- və ya rəngləri ingiliscə adlarını istifadə edərək ("**red**")
- və ya rənglərin rgb kodlarından istifadə edərək (**rgb (255,0,0)**)

təyin edilə bilər.

‘background-color’ xüsusiyyəti

Background-color xüsusiyyəti elementin fonun rəngini təsvir edir. **<body>** Elementdə HTML-sənədin bütün tərkibi yerləşir. Beləliklə, bütün səhifənin fonun rənginin dəyişikliyi üçün **background-color** xüsusiyyəti **<body>** elementinə tətbiq etmək lazımdır . Siz həmçinin başqa elementlərdə bu xüsusiyyəti tətbiq edə bilərsiniz, məsələn – başlıqlara və mətnə. Növbəti nümunədə müxtəlif fonun rəngləri **<body>** və **<h1>** elementlərinə tətbiq edilir .

```
body {  
background-color: #FFCC66;  
}
```

```
h1 {  
color: #990000;  
background-color: #FC9804;  
}
```

Burada diqqət olunmalı məsələ biz **h1** elementinə 2 stil xüsusiyyəti verdik. Və bunların arasına ; işarəsi qoyulur.

Fon təsvirləri — background-image xüsusiyyəti

background-image CSS-xüsusiyyəti fon təsvirinin yerləşdirməsi üçün istifadə olunur.

Web-səhifənin fon təsviri kimi hər hansı şəkili yerləşdirməsi üçün sadəcə <body> teqində background-image xüsusiyyətini tətbiq edin və şəkilin yerini göstərin.Məsələn

```
body {  
background-color: #FFCC66;  
background-image: url("butterfly.gif");  
}
```

```
h1 {  
color: #990000;  
background-color: #FC9804;  
}
```

Nümunədə əsas ona diqqət edin ki *url (" butterfly.gif")* ilə biz fayl olan yeri göstəririk. Bu yazı bildirir ki, şəkil stillərin cədvəli ilə eyni qovluqdadır . Burada əlbəttə ki digər qovluqdakı şəkil fayllarınada, istinad edə bilərik məsələn, *url (" . /images/butterfly.gif")*, və ya Internet-də olan fayllara, faylın tam ünvanını göstərərək: *url (" <http://www.html.net/butterfly.gif>")*.

FON TƏSVİRİNİN TƏKRARLANMASI — BACKGROUND-REPEAT XÜSUSİYYƏTİ

Əvvəlki nümunəni sınaqdan keçirsəz görəcəksiz ki fondakı şəkil bütün səhifəni tutmaq üçün horizontal və vertikal olaraq təkrarlanır.Bunu edən **background-repeat** xüsusiyyətidir.Onun mümkün dəyərləri **repeat-x,repeat-y,repeat** və **no-repeat**

- **repeat-x** şəkil horizontal olaraq təkrarlanır
- **repeat-y** şəkil vertikal olaraq təkrarlanır
- **repeat** şəkil həm vertikal həmdə horizontal olaraq təkrarlanır(susmaya görə bu dəyər aktiv olur)
- **no-repeat** şəkil təkrarlanmır

Fon təsvirinin bloklanması– background-attachment xüsusiyyəti

background-attachment xüsusiyyəti səhifənin fırladılması (scrolling) zamanı fon şəkilin blok edilməsini müəyyən edir.Bu xüsusiyyətin iki dəyəri var *scroll* və *fixed*

Fon şəkilinin yerləşməsi — background-position xüsusiyyəti

Susmaya görə fon şəkili ekranın sol yuxarı küncündə yerləşdirilir. **background-position** xüsusiyyəti bu susmaya görə mənanı dəyişdirməyə icazə verir, və fon şəkili ekranın istənilən yerində yerləşdirməyə kömək edir. **background-position** xüsusiyyətinin dəyərini çox formada vermək olar. Məsələn, '100px

200px' dəyəri fon şəkili sol yuxarı küncdə soldan 100px və yuxarıdan 200px məsafədə yerləşdirilir.

Koordinatları faiz ilə, piksellər, santimetrilər, və ya top, bottom, center, left və right sözlərindən istifadə edərək təqdim etmək olar. Məsələn

```
body {  
background-color: #FFCC66;  
background-image: url("butterfly.gif");  
background-repeat: no-repeat;  
background-attachment: fixed;  
background-position: right bottom;  
}
```

```
h1 {  
color: #990000;  
background-color: #FC9804;  
}
```

Qısaldılmış yazı background xüsusiyyəti

Background-un köməyi ilə siz bir neçə xüsusiyyəti sıxa və ixtisar edilmiş halda yazı bilərsiniz. Bu sizin stil cədvəllərin oxunmasını yüngülləşdirir. Məsələn, bu sətirlərə baxın:

```
background-color: #FFCC66;  
background-image: url("butterfly.gif");  
background-repeat: no-repeat;  
background-attachment: fixed;  
background-position: right bottom;  
background xüsusiyyətini istifadə edərək bunları cəmi bir sətir vasitəsi ilə yazmaq olar
```

background: #FFCC66 url("butterfly.gif") no-repeat fixed right bottom;
bir sətir vasitəsi ilə background xüsusiyyətini verərkən aşağıdakı ardıcılıq fikrini vermək lazımdır.

[background-color] | [background-image] | [background-repeat] | [background-attachment] | [background-position]

Ardıcıl göstərməli xüsusiyyətlərdən hansısa göstərmərsə o hal ona susmaya görə olan dəyər mənimsənilir. Məsələn aşağıdakı nümunədə **background-attachment** və **background-position** təyin edilməyib.

```
background: #FFCC66 url("butterfly.gif") no-repeat;
```

CSS DƏ ŞRİFTLƏR

Şrift ailəsi- font-family xüsusiyyəti

Bu xüsusiyyət səhifədəki mətn məlumatının təsviri vaxtı istifadə ediləcək prioritet şriftlərin verilməsi üçün istifadə edilir. Mətn istifadəçinin brauzerində nümayiş edilərkən ilk sadalanan şriftdə təsvir edilə bilmirsə (yəni istifadəçinin kompyuterində verilən şrift yoxdursa) onda növbəti şrift götürülür. Şriftlər verilərkən ad olaraq şrift ailəsinin (family name) və ya şrift nəslinin adı (generic family) verilir.

Family-name-ə nümunə olaraq Arial, Times New Roman, Tahoma və s.-ni göstərmək olar

Generic family-name-ə isə məsələn sans-serif-i nümunə göstərmək olar.

Times New Roman Garamond Georgia	These three font-families belong to the generic family serif . They are characterized by all having "feet".
Trebuchet Arial Verdana	These three font-families belong to the generic family sans-serif . They are all characterized by not having "feet".
Courier Courier New Andale Mono	These three font-families belong to the generic family monospace . They are all characterized by all characters having a fixed width

Şriftlər verilərkən saytın hansı formada təsvir ediləcəyini düşünüb əvvəlcə daha çox tərcih edilən şrift sonra isə növbə ilə daha aşağı prioritetli şriftlər daxil edilir. Ən sonda isə generic family name-in verilməsi məsləhət görülür. Belə ki əgər daxil edilən şriftlər istifadəçinin kompyuterində olmasa həmin ailəyə uyğun digər şriftlə təsvir ediləcək. Məsələn

```
h1 {  
font-family: arial, verdana, sans-serif;  
}  
h2 {  
font-family: "Times New Roman", serif;  
}
```

Nümunədən göründüyü kimi h1 elementi üçün şrift olaraq arial olmazsa verdana oda olmazsa sans-serif ailəsindən şrift təyin edilmişdir. Diqqət yetirilməli məsələ Times New Roman şrifti cüt dirnaq arasında təqdim edilmişdir. Buna səbəb şriftin adının arasındakı boşluqlardır.

*Şriftin stili-**font-style** xüsusiyyəti*

Bu xüsusiyyət şriftin stilini təyin edir. Dəyərləri normal, italic, oblique,

This is a paragraph, normal.

This is a paragraph, italic.

This is a paragraph, oblique.

Şriftin variantı-**font-variant** xüsusiyyəti

Font-variant xüsusiyyəti vasitəsi ilə normal və ya small-caps variantları arasında seçim edilir. small-caps variantı ilə təyin edilən məndə kiçik hərflər yuxarı reqistrdə amma daha kiçik ölçüdə verilir.

My name is Hege Refsnes.

MY NAME IS HEGE REFSNES.

Şriftin çəkisi – **font-weight** xüsusiyyəti

Bu xüsusiyyət ilə bold və ya normal arasında şriftin çəkisi üçün seçim edilir. Şriftin çəkisi rəqəm ilə də ifadə edilə bilər. Məsələn

```
p.normal {  
  font-weight: normal;  
}
```

```
p.light {  
  font-weight: lighter;  
}
```

```
p.thick {  
  font-weight: bold;  
}
```

```
p.thicker {  
  font-weight: 900;  
}
```

This is a paragraph.

This is a paragraph.

This is a paragraph.

This is a paragraph.

Şriftin ölçüsü-**font-size** xüsusiyyəti

Şriftlərin ölçüsü faizlə piksellə ,pt,em vəya ingiliscə small,medium,large,smaller,larger və digər sözlərlə təqdim edilə bilər.

Şriftlə bağlı olan bu xüsusiyyətləri qisaldılmış olaraq font yazısı ilə də təqdim etmək olur.Məsələn aşağıdakı

```
p {  
font-style: italic;  
font-weight: bold;  
font-size: 30px;  
font-family: arial, sans-serif;  
}  
yazını qisaldılmış olaraq
```

```
p {  
font: italic bold 30px arial, sans-serif;  
}  
göstərmək olar.
```

Font xüsusiyyətlərinin ardıcılığını

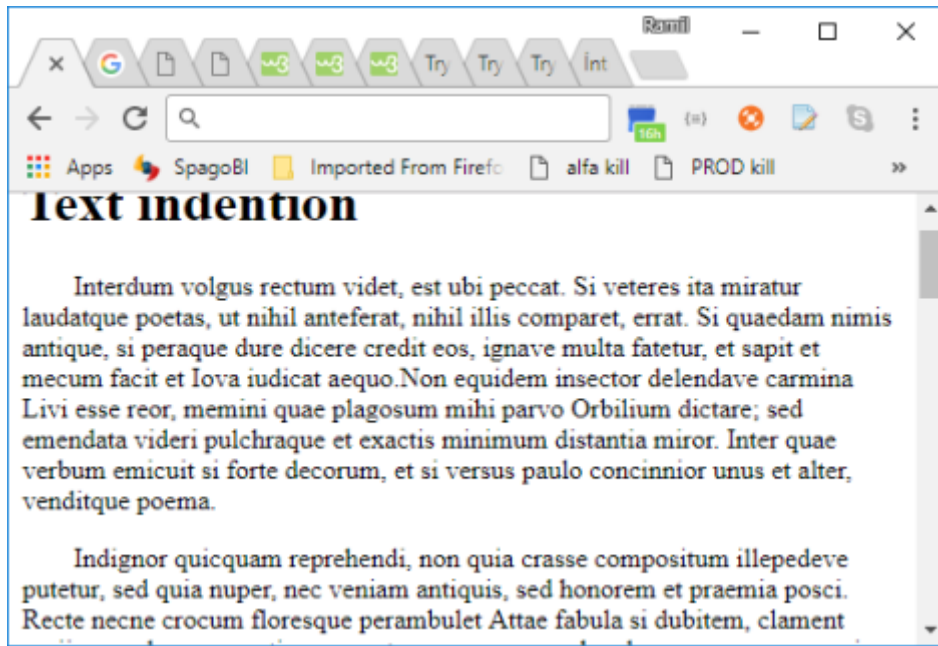
font-style | font-variant | font-weight | font-size | font-family
olaraq vermək lazımdır.

MƏTNLƏRİN FORMATLANMASI

Boşluqlar-text-indent xüsusiyyəti

Bu xüsusiyyət vasitəsi ilə məsələn paraqraf p teqi üçün onun birinci sətirinin daha yuxarıdakı mətnlə arasında boşluq qoyulması imkanı verir.Məsələn

```
p {  
text-indent: 30px;  
}
```



Mətnin düzləndirilməsi –text-align xüsusiyyəti

Bu xüsusiyyətdən mətnin horizontal olaraq düzləndirilməsi zamanı istifadə edilir.Mümkün dəyərləri **center**, **left** ,**right** və **justify**-dir.

```
h1 {
  text-align: center;
}
```

This is heading 1

```
h2 {
  text-align: left;
}
```

This is heading 2

```
h3 {
  text-align: right;
}
```

This is heading 3

Mətnin dekorasiyası-text-decoration xüsusiyyəti

Dəyərləri:**overline**,**line-through**,**underline**.Aşağıda hər 3 hal nümunədə göstərilmişdir.

```
h1 {
  text-decoration: overline;
}
```

```
h2 {
  text-decoration: line-through;
}
```

```
h3 {
```

text-decoration: underline;
}

This is heading 1

~~**This is heading 2**~~

This is heading 3

*Hərflər arasında məsafə-**line-spacing** xüsusiyyəti*

Əgər məsələn paraqraf və ya başlıqlarda sözlərdə hərflər arasında məsafəni artırıb azaltmaq istədikdə bu xüsusiyyət istifadə edilir. Hərflər arasında məsafə piksel ilə verilir. Məsələn

```
h1 {  
letter-spacing: 6px;  
}
```

```
p {  
letter-spacing: 3px;  
}
```

*Mətnin transformasiyası-**text-transform** xüsusiyyəti*

Bu xüsusiyyət mətnin aşağı və ya yuxarı reqistra çevrilməsi məqsədi ilə istifadə edilir. Mümkün dəyərləri :**capitalize,uppercase,lowercase,none**

- **Capitalize** sözlərin ilk hərfləri yuxarı reqistrə sonrakı hərfləri isə aşağı reqistra transformasiya edilir
- **Uppercase** sözün bütün hərflərini yuxarı reqistra çevirir
- **Lowercase** hərfləri aşağı reqistra çevirir.
- **None** sözlər heç bir transformasiyaya məruz qalmır

Məsələn

```
uppercase {  
text-transform: uppercase;  
}
```

```
lowercase {  
text-transform: lowercase;  
}
```



```
capitalize {  
text-transform: capitalize;  
}  
...
```

```
<p class="uppercase">This is some text.</p>  
<p class="lowercase">This is some text.</p>  
<p class="capitalize">This is some text.</p>
```

THIS IS SOME TEXT.

this is some text.

This Is Some Text.

İSTİNADLARIN FORMATLANMASI

Yuxarıda sadaladığımız format xüsusiyyətlərini istinad teqi olan a elementi üzərində daha ətraflı nəzər yetirək.

Psevdoklass HTML-teqlərinin xüsusiyyətlərinin təyini vaxtı müxtəlif şərtləri və ya hadisələri nəzərə almağa icazə verir. HTML-də istinadları a teqi vasitəsi ilə təqdim edirlər. Bu teqi CSS-də selector olaraq da göstərmək olur. Məsələn

```
a {  
color: blue;  
}
```

İstinadlar müxtəlif vəziyyətdə nümayiş oluna bilər. Məsələn baş çəkilməmiş (click edilməmiş), ziyarət edilməmiş (click edilməmiş). Bu halda CSS-də əlavə xüsusiyyətlər verilərkən aşağıdakı nümunədəki kimi psevdoklasslar istifadə edilir.

```
a:link {  
color: blue;  
}  
a:visited {  
color: red;  
}
```

Burada link və visited yazıları psevdoklassları göstərir. Adı halda linklər əgər klik edilməyibsə (**a:link**) göy ziyarət edilibsə (**a:visited**) isə qırmızı rəngdə təsvir edilir. Aktiv linklər üçün **a:active** üzərinə mouse ilə gəlinən linki isə **a:hover** psevdoklassı vasitəsi ilə stil vermək olar.

Aşağıdakı nümunəyə baxaq. Bildiyimiz kimi linklər susmaya görə altdan xətt çəkilmiş vəziyyətdə nümayiş olunur səhifədə.

```
<html>
<head>
<style>
ex1:hover, a.ex1:active { color: red}
ex2:hover, a.ex2:active { font-size: 150%;}
ex3:hover, a.ex3:active { background: red;}
ex4:hover, a.ex4:active { font-family: monospace;}
ex5:visited, a.ex5:link { text-decoration: none;}
ex5:hover, a.ex5:active { text-decoration: underline;}
</style>
</head>
<body>
<p>Mouse over the links to see them change layout.</p>
<p><a class="ex1" href="default.asp">This link changes color</a></p>
<p><a class="ex2" href="default.asp">This link changes font-size</a></p>
<p><a class="ex3" href="default.asp">This link changes background-
color</a></p>
<p><a class="ex4" href="default.asp">This link changes font-family</a></p>
<p><a class="ex5" href="default.asp">This link changes text-
decoration</a></p>
</body>
</html>
```

Burada class-I ex1 olan linkin üzərinə gəldikdə yazının rəngi.

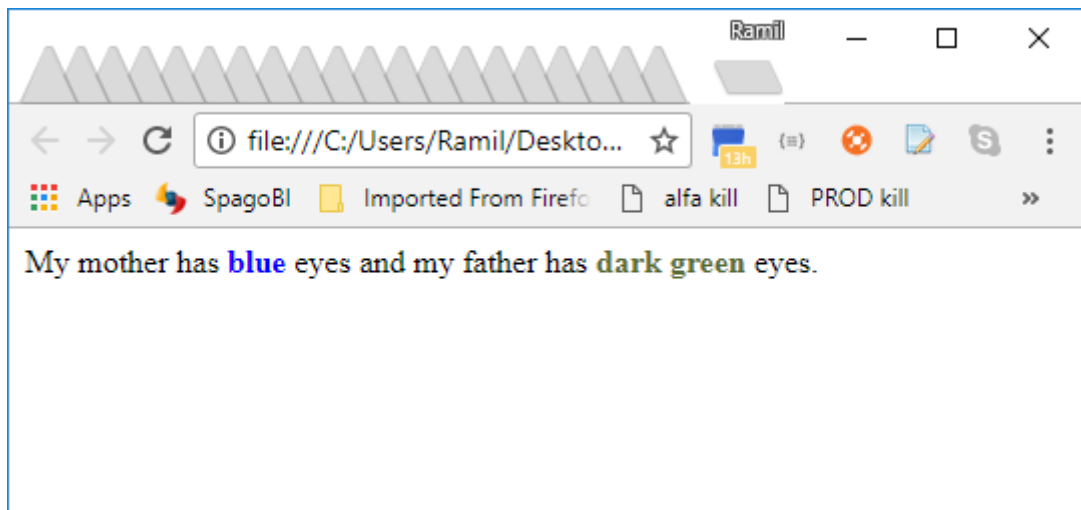
Ex2 classlı elementin yazısının ölçüsü dəyişir.

Ex3 classlı elementin yazısının arxa fonunun rəngi dəyişir.

Ex4 classli elementin yazısının şriftinin ailəsi dəyişdirilir.(monospace edilir)

Ex5 classli element əgər ziyarət edilibsə altdan çəkilmiş xətsiz nümayiş edilir.

ELEMENTLƏRİN QRUPLAŞDIRILMASI(SPAN VƏ DİV)



span və **div** elementlərindən səhifənin strukturlaşması zamanı istifadə edilir. Əsasən class və id atributları da təyin edilir.

**** elementi HTML-in neytral elementlərindən sayılır. Yəni vizual olaraq səhifəyə heç bir şey əlavə etmir. Amma CSS vasitəsi ilə dizayn verildikdə isə sətir tipli (inline) elementlərin qruplaşdırılaraq vizual olaraq effektlər verilməsinə yardım edir. Məsələn səhifədə hər hansı bir sitat yerləşdirib onun hansısa hissələrini vizual olaraq fərqləndirmək istədikdə istifadə edilir.

<p>My mother has ****blue**** eyes
and my father has ****dark green**** eyes.**</p>**
css faylında isə

```
blueclass{  
color:blue;  
font-weight:bold  
}
```

```
greenclass{  
color:darkolivegreen;  
font-weight:bold  
}
```

Yazılırsa o halda browserdə nəticə olaraq aşağıdakı səhifə təsvir olunar.

elementi -dan fərqli olaraq blok tipli elementlərin qruplaşdırmağa imkan verir. Bu fərqdən başqa işləmə məntiqi analojiidir.

Məsələn

```
&lt;div id="democrats">  
&lt;ul>  
&lt;li>Франклин Д. Рузвелт</li>  
&lt;li>Гарри Трумэн</li>  
&lt;li>Джон Ф. Кеннеди</li>
```

```

<li>Линдон Б. Джонсон</li>
<li>Джимми Картер</li>
<li>Билл Клинтон</li>
</ul>
</div>
<div id="republicans">
<ul>
<li>Дуайт Д. Эйзенхауэр</li>
<li>Ричард Никсон</li>
<li>Джэралд Форд</li>
<li>Роналд Рейган</li>
<li>Джордж Буш</li>
<li>Джордж У. Буш</li>
</ul>
</div>

```

Css-də

```

#democrats {
background:blue;
}

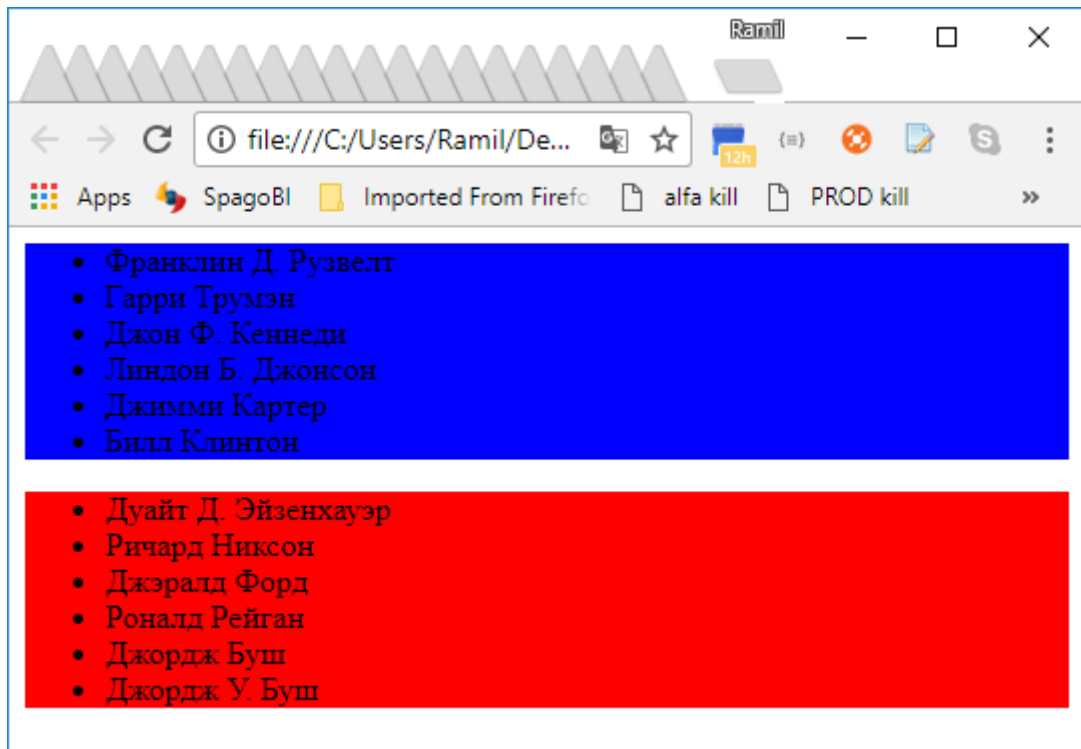
```

```

#republicans {
background:red;
}

```

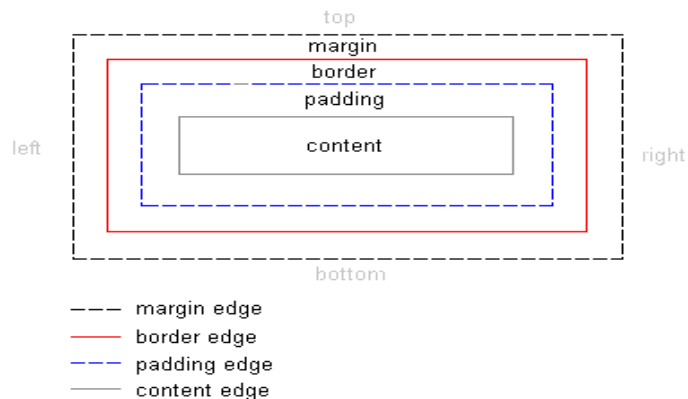
Nəticə browserdə



Bu nümunələrdə biz **span** və **div** elementlərindən sadə effektlərin verilməsi üçün istifadə etdik.

QUTU MODELİ

CSS-də qutu modeli HTML-elementlər üçün yaradılan qutuları təsvir edir. Qutu modeli həmçinin sahələrin, çərçivələrin, və hər elementin tərkibinin təyini üçün ətraflı seçimlərə malikdir. Növbəti diaqramda qutu modelinin necə qurulduğu göstərilmişdir:



Bu modeli nümunə üzərində öyrənək. Məsələn tutaq ki bizim səhifədə Ümumi dünya insan hüquqları bəyannaməsi ilə bağlı başlıq və mətn var.

<h1>Article 1:</h1>

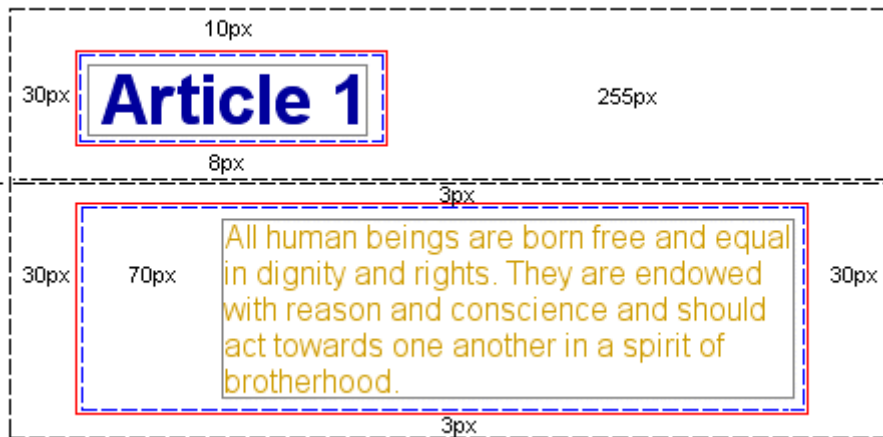
<p>All human beings are born free
n equal in dignity n rights.
They are endowed with reason n conscience
n should act towards one another in a
spirit of brotherhood**</p>**

Şrift və rənglərlə bağlı CSS xüsusiyyətləri artırıqdan sonra səhifəmiz təqribən aşağıdakı kimi olur.

Article 1

All human beings are born free and equal
in dignity and rights. They are endowed
with reason and conscience and should
act towards one another in a spirit of
brotherhood.

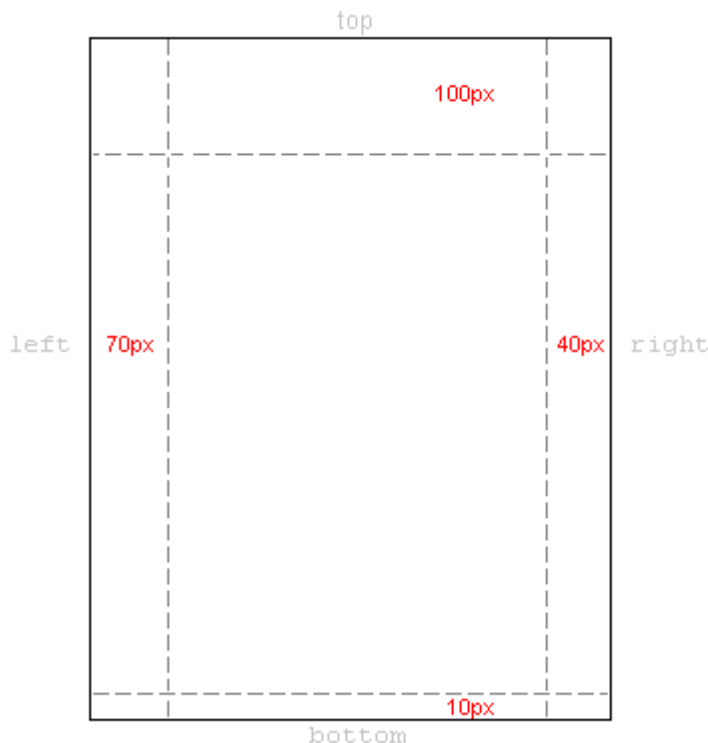
Bu nümunədə iki **h1** və **p** elementləri istifadə edilmişdir. Bu elementlərin qutu modeli aşağıdakı kimi olacaq



Göründüyü kimi hər bir elementin CSS ilə tənzimləmə bilən qutu modeli var.

MARGIN VƏ PADDING XÜSUSİYYƏTLƏRİ

Hər bir elementin 4 tərəfi var **.right,top,left,bottom**.Margin sahəsi sənədin kənarından və ya qonşu elementlər arasındakı məsafəni təyin edir.İlk olaraq sənədin özünün yəni body elementini margin sahələrinin necə təyin edildiyinə nəzər yetirək.



Bu nümunənin CSS kodu təqribən aşağıdakı kimi olacaq.

```
body {
margin-top: 100px;
```

```
margin-right: 40px;  
margin-bottom: 10px;  
margin-left: 70px;  
}
```

və ya

```
body {  
margin: 100px 40px 10px 70px;  
}
```

Biz bu vasitə ilə demək olar ki istənilən elementə margin sahəsi təyin edə bilərik. Məsələn paraqraf teqləri üçün margin sahəsi vermək üçün CSS-də

```
p {  
margin: 5px 50px 5px 50px;  
}
```

yazmaq lazımdır.

Elementlər üçün padding göstəricisi digər elementlərlə arasındakı məsafə yox elementin özünün kənar çərçivəsi ilə daxilindəki tərkibi arasındakı məsafəni təyin edir. Məsələn **h1** elementi üçün bu məsafələri vermək üçün

```
h1 {  
background: yellow;  
padding: 20px 20px 20px 80px;  
}
```

yazmaq lazımdır.

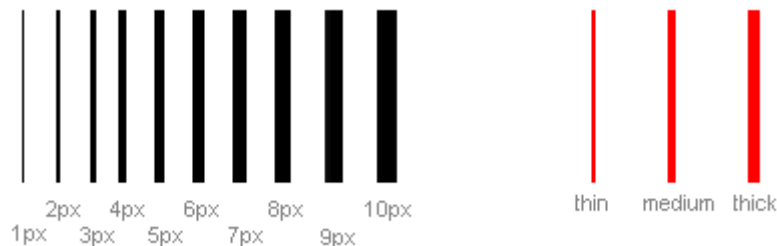
Çərçivələr(Border)

Çərçivələr ilə bağlı xüsusiyyətlər aşağıdakılardır

Çərçivənin qalınlığı(border-width)

Mümkün dəyərləri thin, medium, thick və ya piksellərlə olan rəqəmli ifadədə

Aşağıdakı şəkildə bu dəyərlərin müqaisəsi verilmişdir.

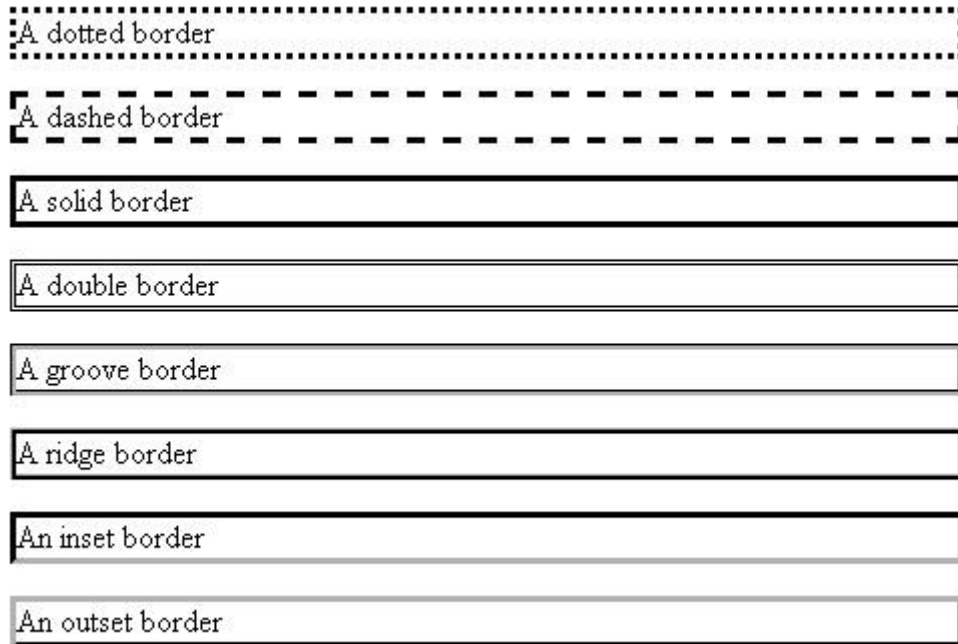


Çərçivənin rəngi (border-color)

Dəyərli 16 liq ədədlərlə(*#FF0000*),ingiliscə rəng adları ilə(*yellow,pink,black*) və ya RGB kodları (*rgb(200,0,100)*) ola bilər.

Çərçivənin növü(border-style)

Dəyərləri aşağıda sadalananlardır



Əlavə olaraq əgər çərçivənin göstərilməməsi lazımdırsa o halda **border-style** olaraq none və ya hidden daxil edilməlidir.

Border xüsusiyyətlərini qısaldılmış olaraq da təqdim etmək olur.Aşağıda həm uzun həm qısaldılmış formada border xüsusiyyətləri verilmişdir.

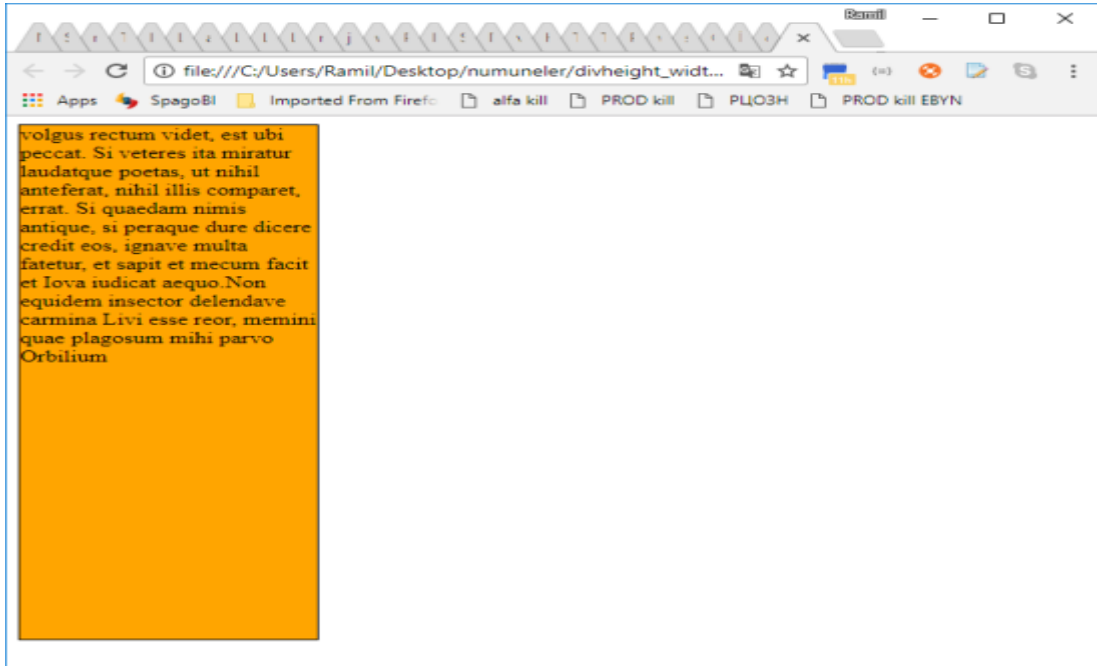
```
p {  
border-width: 1px;  
border-style: solid;  
border-color: blue;  
}  
və ya  
p {  
border: 1px solid blue;  
}
```

ELEMENTLƏRİN HÜNDÜRLÜYÜ(HEIGHT) VƏ ENİ (WIDTH)

Elementlərin hündürlük və eni ölçüləri verilməmişdirsə onda bu ölçülər elementin tərkibinin ölçüsü ilə təyin ediləcək.Olçu piksel və ya faizlə verilə

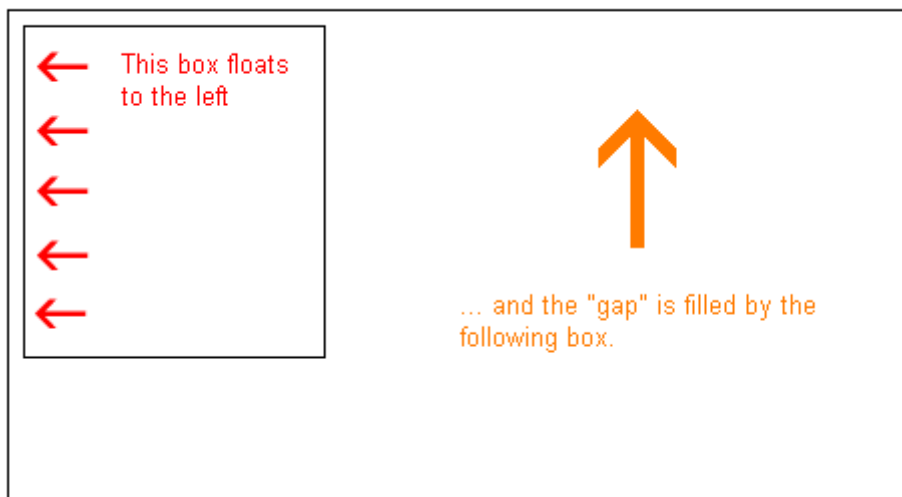
bilər.Aşağıdakı nümunədə div elementinin eni və hündürlüyü verilərək təqdim olunmuşdur.

```
div.box {  
height: 500px;  
width: 200px;  
border: 1px solid black;  
background: orange;  
}
```



ÜZƏ ÇIXAN ELEMENTLƏR(FLOAT XÜSUSİYYƏTİ)

Element sağa və ya sola **float** xüsusiyyətinin köməyi ilə yerləşdirilə bilər.Üzə çıxmanın əsas prinsip şəkildə göstərilmişdir.



Məsələn biz şəkli sol tərəfə qarşısına sağ tərəfə mətni yerləşdirərkən float xüsusiyyətindən istifadə edilir.Nəticə aşağıdakı kimi olur.



Bunun HTML kodu təqribən aşağıdakı kimi olur

```
<div id="picture">

</div>
<p>causas naturales et antecedentes,
```

idcirco etiam nostrarum voluntatum...</p>

Şəkilin sol tərəfdə və mətnidə onun ətrafında nümayiş etdirmək üçün qutunun ölçüsü verilərək həmçinin float xüsusiyyətinin dəyəri left verilməlidir.Yuxarıdakı nümunə üçün CSS aşağıdakı kimi olacaq

```
#picture {
float:left;
width: 100px;
}
```

float xüsusiyyətindən istifadə edilərək mətni səhifədə sütunlar(kolonlar) şəkilində nümayiş etdirməyə imkan verir .Məsələn

```
<div id="column1">
<p>Haec disserens qua de re agatur
et in quo causa consistat non videt...</p>
</div>
```

```
<div id="column2">
<p>causas naturales et antecedentes,
idcirco etiam nostrarum voluntatum...</p>
</div>
```

```
<div id="column3">
```

<p>nam nihil esset in nostra
potestate si res ita se haberet...</p>

</div>

HTML kodu üçün CSS-də hər sütunun səhifənin 33%-ni tutması üçün

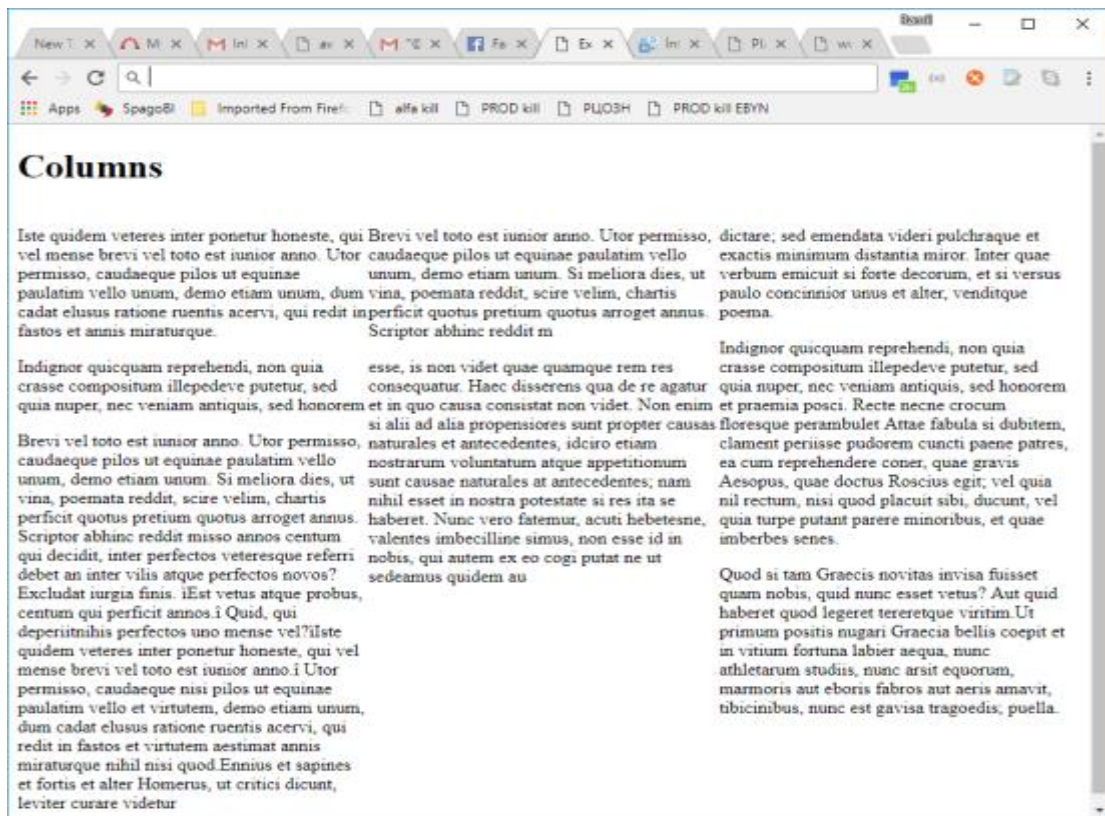
```
#column1 {  
float:left;  
width: 33%;  
}
```

```
#column2 {  
float:left;  
width: 33%;  
}
```

```
#column3 {  
float:left;  
width: 33%;  
}
```

yazmaq lazımdır.

Nəticədə səhifəmiz təqribən aşağıdakı şəkildəki kimi olacaq.



float xüsusiyyətini dəyəri **left, right, both** və **none** ola bilər. Susmaya görə element sağ və ya sol tərəfə sürüşdürülərək yerləşdiriləndə onda növbəti element yuxarı boşalan yeri doldurmaq üçün sürüşür.

Clear xüsusiyyəti vasitəsi ilə bu prosesi idarə etmək olur. **clear** –in dəyərləri **left, right, both** və **none** olur. Dəyərinin both olması yerləşəcəyi qutunun yuxarı çərçivəsinin bir əvvəlki elementin çərçivəsindən aşağıda olacağı anlamına gəlir.

ELEMENTLƏRİN YERLƏŞDİRİLMƏSİ (CSS POSITIONİNG)

CSS vasitəsi ilə elementlər səhifədə tam istənilən nöqtədə yerləşdirilə bilər. Əsas prinsip ondan ibarətdir ki siz elementin qutusunu səhifəni bir koordinat sistemi kimi təsəvvür eləsək əgər bu koordinat sistemində istənilən yerdə yerləşdirə bilərsiniz. Məsələn bir səhifədəki başlığı yuxarıdan *100px* soldan *200px* məsafədə yerləşdirmək istəyiriksə o halda

```
h1 {
  position: absolute;
  top: 100px;
  left: 200px;
}
```

yazmamız lazımdır.

Elementlərin yerləşdirilməsi 2 formada olur

1. Absolut yerləşdirilmə
2. Nisbi(relative) yerləşdirmə
3. Static yerləşdirmə
4. Sabit(fixed) yerləşdirmə
5. Yapışqan(sticky) yerləşdirmə

Absolut yerləşdirmədə element səhifədə yerləşdirilərkən özünə bağlı boş sahə saxlamır.absolut yerləşmə üçün CSS-də **position** xüsusiyyətinin dəyərini **absolute** olaraq təyin etmək lazımdır.Bundan başqa left,right,top və bottom xüsusiyyətləri vasitəsi ilə element səhifədə lazım olan nöqtədə yerləşdirilir.

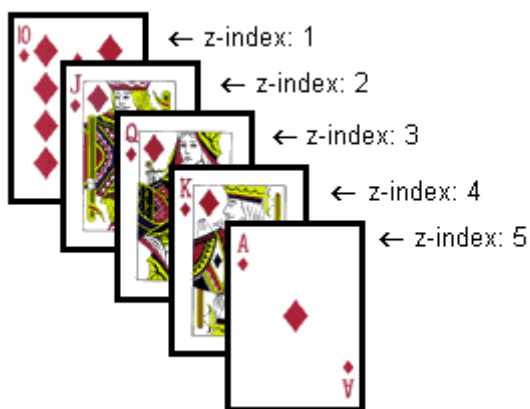
Nisbi yerləşdirilmədə isə yerləşmə koordinatları ana elementə nisbətən verilir.

Static yerləşdirmədə elementin yerləşəcəyi nöqtə yuxarı aşağı sol və sağ –a nisbətdə deyil ümumi səhifədə elementlərin düzülmə ardıcılığına görə yerləşdiriləcək.

Sabit yerləşdirmədə element sabit olaraq səhifənin müəyyən hissəsində yerləşəcək və hətta səhifə scroll edilsə belə yeri dəyişməyəcək.

Yapışqan yerləşdirmə scrollun vəziyyətindən asılı olaraq nisbi və sabit yerləşdirmə arasında olan yerləşdirmədir.

CSS səhifənin hər 3 ölçüsü ilə əməliyyat edə bilər.Hündürlük,en və dərinlik.İndiyə qədər danışdığımız məlumatlar səhifənin ancaq 2 ölçüsünü nəzərə alırdı.Səhifədə elementləri qat qat bir birinin üzərinə yerləşdirmək üçün **z-index** xüsusiyyətindən istifadə edilir.Bunun üçün elementlərin z-index dəyərləri təyine dilir.Daha böyük z-index dəyərli element daha kiçik z-index-li bütün elementlərin üstündə olacaq.



Oyun kartları ilə bağlı yuxarıdakı səhifənin CSS –i təqribən aşağıdakı kimi olacaq.

```
#ten_of_diamonds {  
position: absolute;  
left: 100px;  
top: 100px;  
z-index: 1;  
}
```

```
#jack_of_diamonds {  
position: absolute;  
left: 115px;  
top: 115px;  
z-index: 2;  
}
```

```
#queen_of_diamonds {  
position: absolute;  
left: 130px;  
top: 130px;  
z-index: 3;  
}
```

```
#king_of_diamonds {  
position: absolute;  
left: 145px;  
top: 145px;  
z-index: 4;  
}
```

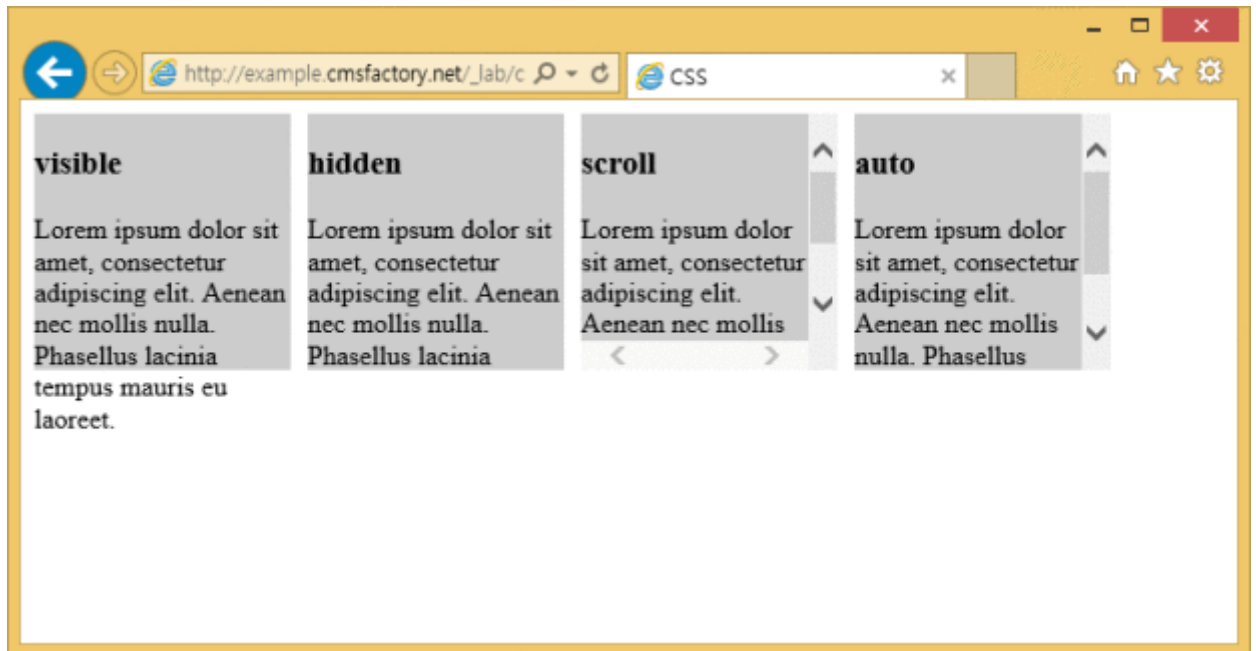
```
#ace_of_diamonds {  
position: absolute;  
left: 160px;  
top: 160px;  
z-index: 5;  
}
```

z-index xüsusiyyəti vasitəsi ilə şəkli yazının üzərinə və ya əksinə yazını şəkilin üzərinə yerləşdirmək olar.

CSS-də əgər elementin tərkibi elementin qutusunda daha böyük olduğu vaxtı elementin necə nümayiş ediləcəyini **overflow** xüsusiyyəti təyin edir. Mümkün dəyərləri **scroll, hidden, auto, visible, inherit**

- **Visible**- bu halda elementin tərkibi əgər elementin qutusunda böyükdürsə qutudan kənara çıxaraq nümayiş olunur
- **Scroll** – bu halda tərkib elementin ölçüsünü aşarsa avtomatik olaraq scroll artırılacaq Bu halda həm horizontal və vertikal scroll-lar artırılacaq

- Auto-scroll kimidir fərqi ancaq ehtiyac olan tərəfə scroll artırılır
- Hidden –tərkibin kənarda qalan hissəsi görünməz olur
- Inherit – bu overflow dəyəri ana elementə uyğun təyin edilir.



JavaScript nədir?

HTML dili hər nə qədər mətnlər üstündə istədiyimiz hər işi görməyimizə icazə versə də, zəif olduğu bəzi tərəfləri var; məsələn HTML bizə veb səhifələrin bir “iş” gördürmək, onları fərqli hadisələrə və vəziyyətə həssas formaya salmağımıza imkan vermiş. Bu çatışmamazlığı Netscape firmasının nümayəndələri də fərqlənə varmış, 1995 –ci il Dekabr ayında C dilinin brauzerlərə uyğunlaşdırılmış növü deyəbiləcəyimiz JavaScripti bazara çıxartdılar.

Həmin vaxtlarda Sun Microsystems Pascal və Delphi dillərindən ilhamlanaraq yaratdığı “Java” adlı bir proqramlaşdırma dilini bazara çıxarmaq üzrə idi. Netscape-in bazara çıxartdığı script dilinin adını JavaScript qoyması, o vaxtlarda çox tələffüz edilən “Java” sözünün məşhurluğundan istifadə etmək üçün tətbiq etdiyi bir marketinq strategiyasıdır, yəni bir çox adamın düşündüyünün əksinə Java ilə JavaScript arasında ad oxşarlığı xaric heç bir oxşarlıq yoxdur.

Bir qədər sonra Microsoft da bu rəqabətə qoşuldu və JavaScriptlə hardasa eyni olan Jscripti bazara çıxartdı. Ancaq, sonra bu dillər fərqli istiqamətlərə doğru inkişaf etdirildi. Hal-hazırda Netscape brauzeri Jscript-i tanımır, amma Explorer hər iki scripti də tanıyır. Yenə də iki brauzerin JavaScript-i şərh etmələri bəzən fərqlilik kimi göstərilir.

JavaScript populyar bir proqramlaşdırma dilidir. HTML dili ilə istifadə olunur. İnternetdə, kompüterlərdə, tabletlərdə və telefonlarda işləyir. Bu gün JavaScript-i dəstəkləməyən heç bir brauzer yoxdur. JavaScript internet səhifələrin yaradılmasında geniş istifadə olunan proqramlaşdırma dilidir. 1995-ci ilə Brendan Eys tərəfindən yazılmışdır. JavaScript Obyekt Yönlü Proqramlaşdırma (OYP) dilidir.

JavaScript – obyekt yönümlü proqramlaşdırma imkanları olan interpretativ proqramlaşdırma dilidir. Sintaksis nöqtəyi-nəzərindən JavaScript-in baza dili C, C++ və Java PD-lərinin proqram konstruksiyalarına bənzəyir (məs.: if, while və && operator). Ancaq bu oxşarlıq yalnız sintaktis oxşarlıq ilə məhdudlaşır.

JavaScript – tipləşdirilməmiş dildir, yəni bu dildə dəyişənlərin tiplərini müəyyən etmək tələb olunur. JavaScript-də obyektlər, ad xüsusiyyətlərində sərbəst qiymətləri əks etdirir. Bu xüsusiyyətinə görə Perl PD-sinin assosiativ massivlərini, C strukturlarını və ya C++ və ya Java obyektlərini xatırladırlar. JavaScript dilinin nüvəsi sadə məlumat tipləri, ədədlər, sətirlər və Null qiymətlərini dəstəkləyir.

Bundan başqa bu dil massivlər, tarixlərin və müntəzəm ifadə obyektlərinin inteqrasiya edilmiş dəstinə malikdir. Adətən JavaScript veb-brauzerlərdə tətbiq edilir. Obyektlərin tətbiqi nəticəsində bu dilin imkanları daha genişdir. JavaScript, istifadəçi ilə qarşılıqlı təsiri təşkil etməyə, veb-brauzeri idarə etməyə və veb-brauzerin pəncərəsinin daxilində əks etdirilən sənədin

tərkibini dəyişdirməyə imkan verir. JavaScript veb səhifələrin HTML-koduna tətbiq edilmiş ssenarilər vasitəsilə inteqrasiya edir. Bir qayda olaraq, bu versiya JavaScript-in kliyent dili adlanır. Qeyd etmək lazımdır ki, ssenari kliyenti veb-serverdə deyil, kompüterinizdə (oflayn rejimdə) icra edilir. JavaScript və bu dilin dəstəklədiyi məlumat tipləri dili beynəlxalq standartlara əsaslanır.

Bunun sayəsində reallaşdırmalar arasında çox gözəl uyğunluq yaranır. Bunun sayəsində reallaşdırmalar arasında çox gözəl uyğunluq yaranır. JavaScript-kliyentinin bəzi hissələri rəsmi standart, bəzi hissələri hissələr de-fakto standartdır, amma kliyentin elə hissələri də var ki, brauzerin konkret versiyasına uyğun spesifik genişlənmədir. Müxtəlif brauzerlərdə JavaScript reallaşdırmalarının uyğunluğu JavaScript-in kliyent dilindən istifadə edən proqramçıları tez-tez düşündürür və narahat edir. Bu fəsildə dilin imkanlarının faktiki öyrənməsinə keçməzdən əvvəl JavaScript-in qısa icmal və giriş informasiyası verilir. Bundan başqa, praktik veb-proqramlaşdırma JavaScript-in kliyent dilində bir neçə kod fragmenti bu fəsildə nümayiş etdirilir.

JavaScript istifadə edərək, edə bilərsiniz;

HTML obyektlərinin dəyişdirilməsi

HTML obyektlərinin silinməsi

Yeni HTML obyektləri əlavə olunur

HTML obyektlərinin kopyalanması

Və daha çox...

HTML sənədində JavaScript kodları istifadə olunur. Kod yazmağa başlamazdan əvvəl `<script>` teqi əlavə olunur və kod yazma prosesi bitdikdən sonra `</script>` yazılaraq bağlanır.

JavaScript kodu üçün yaratacağınız `<script> </script>` obyektini `<head>` bölməsində və ya `<body>` bölümündə istifadə edə bilərsiniz.

DİQQƏT: BODY elementi daxilində istifadə etmək səhifə yükləmə sürətini artıracqdır.

`<!DOCTYPE html>`

`<html>`

`<head>`

`<script>`

```
function funk() {  
document.getElementById("demo").innerHTML = "Paragraf dəyişdi!"; } </script>  
  
</head>  
  
<body>  
  
<h1>Web Səhifə</h1>  
  
<p id="demo">Bir paragraf</p>  
  
<button type="button" onclick="funk()">Düyməyə bas!</button>  
  
</body>  
  
</html>
```

İstəsəniz javascript faylı yaradıb HTML sənədinizdən çağıraraq istifadə edə bilərsiniz. Bu istifadə üsulu mənbə kodunuzun sadə olmasını təmin edəcəkdir. Bunu etmək üçün <script> kodunuza src = "" xassəsini əlavə edin və əsas qovluğa görə fayl adını göstərin. Məsələn, skript sənədimizin adı "script.js" dirsə;

```
<script src="script.js"></script>
```

Qeyd: <script> və </script> elementləri skript sənədində istifadə edilmir.

JavaScript-də çap və ya çıxarma xüsusiyyəti yoxdur. JavaScript yalnız HTML Sənədindəki obyektləri idarə etmək üçün istifadə olunur.

HTML sənədinə mətni yalnız test məqsədilə çap etmək üçün document.write () istifadə edə bilərsiniz.

Brauzerinizdə ayıklama xüsusiyyəti varsa, console.log () kodunu istifadə edərək brauzerinizin konsol hissəsindəki JavaScript dəyərlərinə baxa bilərsiniz. Konsol bölməsinə daxil olmaq və səhvləri düzəltməyə başlamaq üçün brauzerinizdə F12 düyməsini basmalısınız.

Qeyd: document.write yalnız test məqsədləri üçündür. Tamamilə yüklənmiş bir HTML sənədində istifadə etsəniz, bütün elementlər sıfırlanacaq.

JavaScript-in başqa sahələrdə istifadəsi

JavaScript – ümumi təyinatlı proqramlaşdırma dilidir və onun istifadəsi tək veb-brauzerlər ilə məhdudlaşmır. Əvvəllər JavaScript, istənilən proqrama ssenarilər ilə yerləşdirilir və icra edilir. İlk günlərdən Netscape şirkətinin veb-serverləri JavaScript ssenarilərini icra etməyi bacaran

JavaScript interpretatoru dəstəkləyirdi. Oxşar üsulla Microsoft korporasiyası Internet Explorer-ə əlavə olaraq öz veb serverlərində IIS və Windows Scripting Host məhsulu üçün JScript interpretatoru istifadə edir. Adobe şirkəti öz Flash- fayllarının oxuyucusunun idarə etməsi üçün JavaScript-dən törəyən dili cəlb etdi. Həmçinin Sun şirkəti Java 6.0 distributivinə JavaScript interpretatoru quraşdırdı və bunun sayəsində istənilən Java-proqramına ssenarilərin yerləşdirilməsi imkanı əhəmiyyətli dərəcədə yüngülləşdi.

Netscape və Microsoft öz JavaScript interpretatorlarının reallaşdırmalarını, öz proqramlarını əlavə etmək istəyən şirkətlər və proqramçılar üçün əlçatan etdi. Netscape şirkəti tərəfindən yaradılmış interpretator açıq mənbəli və azad yayılan PT olub, hal-hazırda Mozilla (<http://www.mozilla.org/js/>) təşkilatı vasitəsilə yayılır. Mozilla faktiki olaraq JavaScript 1.5 interpretatorunun iki müxtəlif versiyasını yayır: biri C dilində yazılmışdır və SpiderMonkey adlanır, o biri isə Java dilində yazılmışdır və kitabın müəllifinin fikrincə çox təkmil şəkildə hazırlanmış və Rhino (kərgədan) adı verilmiş interpretatordur.

JavaScript öyrənilməsi

İstənilən yeni proqramlaşdırma dilinin metodikası zamanı praktikadan istifadə etmək lazımdır. Bu kitabı oxuduğunuzda, sizə JavaScript imkanlarını yoxlamağı məsləhət görürəm. Məsələn, sadə funksiyalardan ibarət kodlardan başlaya bilərsiniz. Proqram kodunu oxumağa və anlamağa çalışın. JavaScript öyrənilməsinə ən asan yanaşma – sadə ssenarilərin yazılmasıdır. JavaScript kliyentinin üstünlüklərindən biri ondan ibarətdir ki, işləmə mühiti hər hansı, veb-brauzer və ən sadə mətn redaktoru ilə qurulmuşdur. JavaScript-də proqramlar yazmaq üçün, xüsusi proqram təminatının yüklənməsinə ehtiyac yoxdur. Məsələn, faktorialların yerinə Fibonaççi ədədlərinin ardıcılığını göstərmək üçün, aşağıdakı nümunəyə baxmaq olar.

```
<script>
document.write("<h2>Fibonaççi ədədləri</h2>");
for (i=0, j=1, k=0, fib =0; i<50; i++, fib=j+k, j=k, k=fib)
{
    document.write("Fibonacci (" + i + ") =" + fib);
    document.write("<br>");
}
</script>
```

Bu kod sizə qəliz görünə bilər (əgər kodu anlama bilmədisə, narahat olmayın), amma belə qısa proqramlarla təcrübə etmək üçün, kodu olduğu kimi kopyalamaq və lokal URL-ünvanlı fayl kimi veb- brauzerdə onu icra etmək kifayətdir. Nəzərə alın ki, hesablamaların nəticəsini göstərmək üçün document.write() metodundan istifadə olunur. Bu metoddan istifadə, JavaScript ilə tanışlıq zamanı faydalıdır. Alternativ olaraq dialog pəncərəsində sadə mətn nəticəsini göstərmək üçün alert() metodunu tətbiq etmək olar:

```
alert ("Fibonacci (" + i + ") = " + fib);
```

Qeyd edək ki, JavaScript ilə belə sadə sınaq kodlarını HTML faylın daxilində , və teqlərinin içərisində yerləşdirmək olar.

JavaScript ilə sınaqların daha da sadələşdirməsi üçün URL-ünvana mənimsədilə bilən javascript: psevdoprotokol spesifikasiatoru yaradılmışdır. Bu üsul ayəsində JavaScript-də ifadənin və qiymətin nəticənin hesablamasıdır. Belə URL-ünvan psevdoprotokol spesifikasiatorundan (javascript:) ibarətdir, hansı ki, burada sərbəst olaraq JavaScript-kodu (təlimatlar biri-birindən nöqtəli vergül ilə ayrılır) göstərilir. URL-ünvanı psevdoprotokol vasitəsilə yükləndikdə, brauzer sadəcə JavaScriptkodunu icra edir. Belə URL- ünvanı ifadənin son qiyməti sətir tipinə dəyişdiriləcək və bu sətir yeni sənəd kimi veb-brauzer vasitəsilə göstəriləcək. Məsələn, bəzi operatorların və JavaScript dilinin təlimatlarını yoxlamaq üçün, veb-brauzerin ünvan sahəsində aşağıdakı URL-ünvanları yığmaq olar:

```
javascript:5%2
javascript:x = 3; (x<5)? "x qiyməti kiçikdir": "x qiyməti daha böyükdür"
javascript:d = new Date(); typeof d;
javascript:for (i=0, j=1, k=0, fib=1; i<5; i++, fib=j +k, k=j, j=fib) alert (fib);
javascript:s=""; for (i in navigator) s+=i+" "+navigator [i] +" \n"; alert (s);
```

Firefox təksətirli veb- brauzerində ssenarilər JavaScript-konsollarını ehtiva edir. Bu konsolu Alətlər menyusundan işə salmaq olar. Burada sadəcə yoxlamaq istədiyiniz ifadəni və ya təlimatı daxil etmək lazımdır. JavaScript- konsoldan istifadə zamanı psevdoprotokol spesifikasiatorunu (javascript:) işə salmaq olar.

JavaScript- kodun öyrənilməsinin baza metodikası, başqa dillərin metodikası ilə uyğun gəlir. Əgər siz JavaScript- ssenarilərində tez-tez xətalara rastlaşırsınızsa, ehtimal ki, JavaScript-in cari sazlayıcısı ilə maraqlanacaqsınız. Internet Explorer-də Microsoft Script Debugger, Firefox-da Venkman adlı məlum genişlənmənin modul sazlayıcısından istifadə etmək olar. Bu alətlərin təsviri kitabın əhatə mövzundan kənar olmasına baxmayaraq, siz asanlıqla İnternetdə, hər hansı bir axtarış sistemindən istifadə edib bu barədə məlumat toplaya bilərsiniz. Daha bir alət jslint-dir. Ciddi desək bu sazlayıcı deyil; Bu alət JavaScript-proqram kodunda olan xətalara axtarmağa yönəlmişdir

JavaScript Framework-ləri

JavaScript-in populyarlığının əsas səbəblərindən biri də JavaScript Framework'ləridir. Trend olan JavaScript frameworklərinə misal olaraq aşağıdakıları göstərə bilərik.

- AngularJS, sürətli tətbiqetmə inkişafı üçün bir sıra müasir inkişaf və dizayn xüsusiyyətləri təqdim edən Google-un veb inkişaf Frameworkdür.
- ReactJS, əsasən Facebook tərəfindən dəstəklənən və Facebook və Instagram istifadəçi interfeysinin arxasındakı başqa bir yaxşı JavaScript çərçivəsidir.
- jQuery veb saytınızı genişləndirmək və daha interaktiv etmək istədiyiniz zaman istifadə edilə bilər. Google, WordPress və IBM kimi şirkətlərin hamısı jQuery-yə etibar edirlər.

JavaScript kliyenti

JavaScript interpretatoru veb-brauzerə JavaScript kliyenti vasitəsilə inteqrasiya edir. Elə buna görə də, JavaScript dedikdə, insanların ağına ilk öncə JavaScript kliyenti gəlir. Bu kitabda JavaScript-in kliyent dili bu dilin alt çoxluğunu təşkil edən JavaScript-bazası ilə birlikdə təsvir

edilir. JavaScript kliyentinin daxilində JavaScript interpretatorunu və obyektiv-brauzerlə müəyyən edilən sənədin obyekt modeli (Document Object Model, DOM) yerləşir. Sənədlər JavaScript-senariləri özündə saxlaya bilər. Bu senarilər, öz növbəsində DOM modelindən istifadə edərək, sənədin və ya idarə etmənin üsulunun modifikasiyası üçün istifadə edilir. Başqa sözlə demək olar ki, JavaScript kliyenti veb səhifələrin statik tərkibin davranışını müəyyən etməyə imkan verir. JavaScript kliyenti veb-proqramların hazırlamasının texnologiyalarının əsasıdır, DHTML AJAX arxitekturaların). ECMA-262 spesifikasiyası JavaScript-in baza dilinin standart versiyasını müəyyən edir və World Wide Web Consortium (W3C) təşkilatı tərəfindən standartlaşdıran DOM spesifikasiyasını hazırlamışdır hansı ki, brauzer bu standartı öz obyekt modelində dəstəkləməlidir. W3C DOM standartı ən məşhur brauzerlərlə tam dəstəklənir. Yalnız – Microsoft Internet Explorer tərəfindən dəstəklənmir; bu brauzerdə hadisələrin emalı mexanizminin dəstəyi yoxdur.

JavaScript kliyentindən istifadə nümunələri

JavaScript interpretatoru ilə təchiz edilmiş veb-brauzer İnternet vasitəsilə JavaScript-senarilər şəklində icra edilən tərkibi göstərə bilər. Aşağıdakı nümunədə JavaScript dilində veb-səhidəyə integrasiya edilmiş sadə proqram göstərilmişdir.

```
<html>

<head>

<title> Faktorialin hesablanması</title>

</head>

<body>

<h2>Faktorial cədvəli</h2>

<script>

Var fact=1

For(i=1;i<= 15; i++){

fact= fact*i

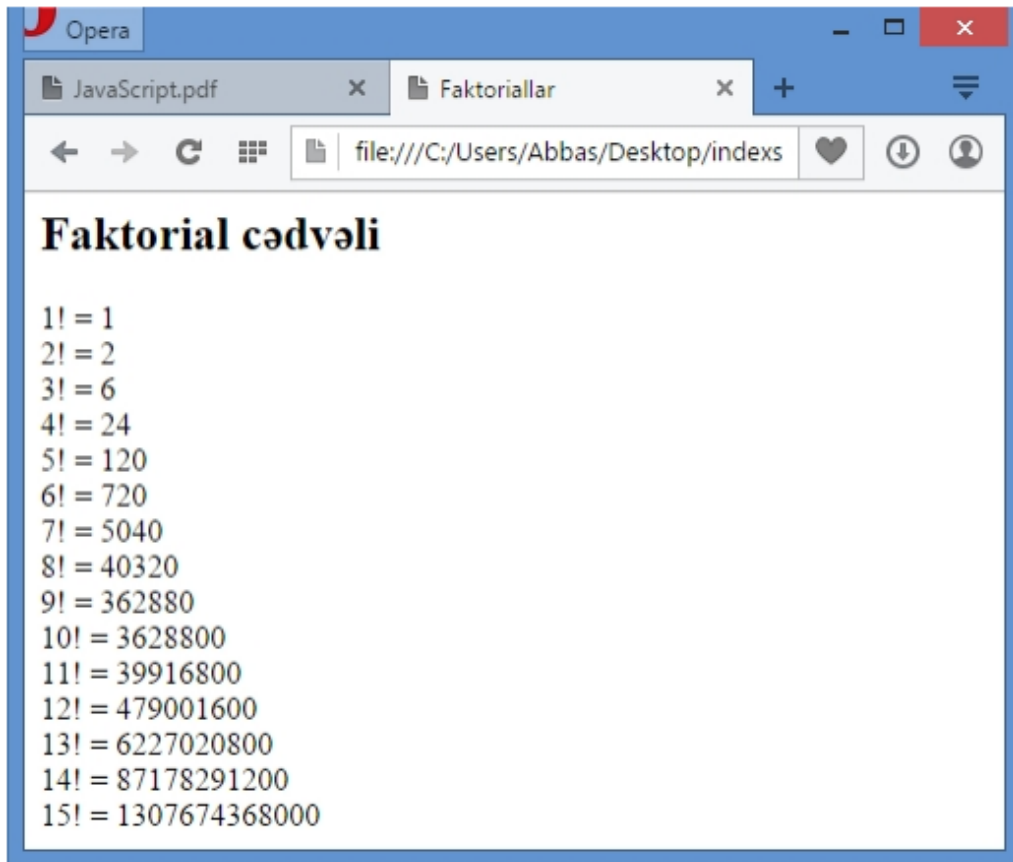
document.write(i+"! = " + fact + "<br>");

}

</script>
```

</body>

</html>

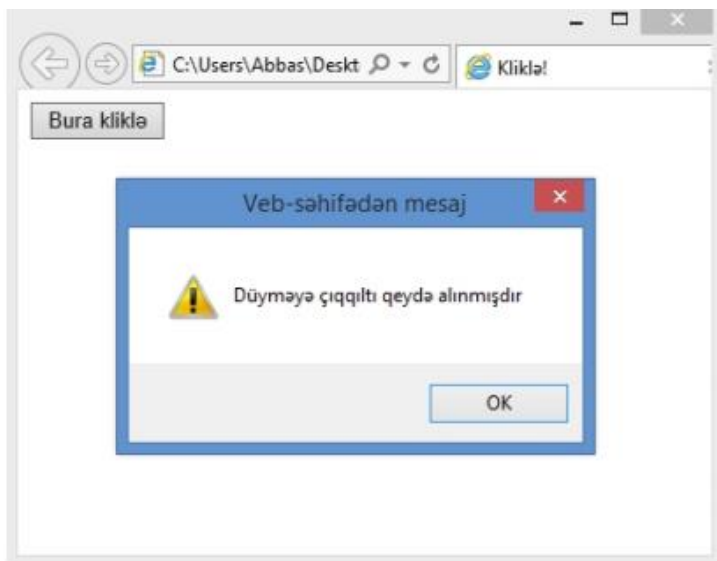


Sözügedən proqram JavaScript-i dəstəkləyən brauzer vasitəsilə icra edildikdə, şəkildəki nəticəni verəcək. Bu nümunədən göründüyü kimi, JavaScript-kodunun HTML-faylına yerləşdirilməsi üçün teqlərindən istifadə edilmişdir. Burada əsas – document.write() metodundan istifadə edilir. Bu metod, sənədin veb-brauzerlə yükləməsi anında HTML-sənədin daxilində dinamik HTML-mətn istehsal etməyə imkan verir. JavaScript yalnız HTML-sənədin tərkibini deyil, həm də onların davranışının idarəetməsini təmin edir. Başqa sözlə, JavaScript-proqram istifadəçisinin hərəkətlərinə reaksiya verə bilər (məs.: mətn sahəsinə qiymətin daxil edilməsi və ya sənəddə təsvir sahəsində siçandan icra olunan çıxqıltıya cavab). Bu sənəd üçün hadisələrin emalçılarının təyini yolu ilə əldə edilir.

JavaScript yalnız HTML-sənədin tərkibini deyil, həm də onların davranışının idarəetməsini təmin edir. Başqa sözlə, JavaScript-proqram istifadəçisinin hərəkətlərinə reaksiya verə bilər (məs.: mətn sahəsinə qiymətin daxil edilməsi və ya sənəddə təsvir sahəsində siçandan icra olunan çıxqıltıya cavab). Bu sənəd üçün hadisələrin emalçılarının təyini yolu ilə əldə edilir. Məsələn, müəyyən hadisənin yaranması anında icra edilən JavaScript-kodlarını nümunə göstərmək olar.

Nümunədə HTML-kodun sadə fraqmenti göstərilmişdir. Burada emalçı çıxqılıya cavab olaraq müəyyən hadisəni (burada xəbərdarlıq) icra edir.

```
<button onClick=""alert('Düymə çıxqılı qeydə alınmışdır');> Bura kliklə </button>
```



Leksik struktur

Proqramlaşdırma dilinin leksik strukturu – proqramların yazılma qaydalarını müəyyən elementarların dəstidir. Dilin aşağı səviyyəli sintaksisi; dəyişənlərin adları, şərhlər üçün istifadə edilən simvollar, bir təlimatı digərindən ayırmaq və s-dir. Bu qısa fəsil JavaScript-in leksik strukturunu sənədlərlə əsaslandırır.

Simvol yığımı

JavaScript-də proqramların yazılışı zamanı Unicode simvol yığımından istifadə olunur. Yalnız ingilis dili üçün ASCII-yə və ISO Latin-1-ə yaxın kodlaşdırmadan və əsas qərbi avropa dillərinin 8-dərəcəli kodlaşdırmasının 7- dərəcəli kodlaşdırmasından fərqli olaraq, Unicode-un 16-dərəcəli kodlaşdırması praktik olaraq istənilən yazı dilinin tətbiqini təmin edir. Bu imkan beynəlmilləşdirmə üçün və xüsusilə "ingilis olmayan" proqramçılar üçün əhəmiyyətlidir. Amerikanlar və digər ingilis dilində danışan proqramçılar proqramları, adətən yalnız ASCII və ya Latin-1 kodlaşdırmasını dəstəkləyən mətn redaktorunun köməyi ilə yazırlar və buna görə onlarda Unicode tam simvolların dəsti əlçatan deyil. Ancaq əks tərəfin bu barədə heç bir çətinliyi, bir halda ki, çünki ASCII və Latin-1 kodlaşdırmaları Unicode kodlaşdırmasının alt çoxluqlarını təşkil edir və hər hansı bir simvol dəstinin köməyi ilə yazılmış JavaScript-proqramı tamamilə düzgündür. ECMAScript v3 standartı JavaScript-proqramlarında istənilən yerdə Unicode-simvolların mövcudluğuna imkan edir.

Registrə həssaslıq

JavaScript – registrə həssas dildir. Bu isə o deməkdir ki, əsas sözlər, dəyişənlər, funksiya adları və dilin ixtiyari bir identifikatoru həmişə böyük və kiçik hərflərin eyni cür yığımını ehtiva etməlidir. Məsələn, while açar sözü "While" və ya "WHILE" kimi deyil "while" kimi yazılmalıdır. Online, Online, OnLine və ONLINE – bu dörd müxtəlifin dəyişən adları da

analojidir. Qeyd edək, HTML dili, JavaScript-dən fərqli olaraq, registrə həssas deyil. HTML-in və JavaScript-kliyentinin yaxın əlaqəsini nəzərə alaraq bu fərq qarışıqlığa gətirib çıxara bilər. Nəzərə alın ki, JavaScript- obyektləri və onların xüsusiyyətləri ilə HTML dilinin teq və atributları eynidir. Əgər HTML-dilində bu teqlər və atributlar istənilən registrdə yazıla bildiyi halda, JavaScript-də adətən kiçik hərflərlə yazılmalıdır. Məsələn, HTML-də onclick hadisəsinin emalçısının atributu əksər hallarda onClick kimi, ancaq JavaScript- kodunda (və ya XHTML-sənədində) onclick kimi göstərilməlidir.

JavaScript – proqramın daxilində leksemləri arasında boşluqlara, tabulyasiyalara və sətir keçidlərinə məhəl qoymur. Buna görə də boşluq, tabulyasiya və yeni sətir simvolları proqram mətnində xüsusi simvolların köməyiylə məhdudiyyətsiz istifadə edilə bilər. Ancaq bu barədə, növbəti bölmədə danışacağıq.

Vacib olmayan nöqtəli vergüllər

Sadə JavaScript-də təlimatları C, C++, və Java PD-ləri kimi adətən nöqtəli vergül (;) simvolları ilə qurtarır. Nöqtəli vergül təlimatlar bir-birindən ayırmasına xidmət edir. Ancaq JavaScript-də əgər hər təlimat ayrı sətirdə yerləşirsə, nöqtəli vergülü qoymamaq olar. Məsələn, aşağıdakı fraqment nöqtəli vergüllərsiz ola bilər:

```
a = 3;
```

```
b = 4;
```

Ancaq əgər hər iki təlimat bir sətirdə yerləşdirilmişsə, onda məcburi olaraq nöqtəli vergül qoyulmalıdır:

```
a = 3; b = 4;
```

Proqramlaşdırmada nöqtəli vergüllərin qoyulma kriteriyalarını düzgün anlamaq çətindir və buna görə istənilən məqamda nöqtəli vergüllərdən istifadə etməyi özünüzdə vərmiş edin. Nəzəri alın ki, JavaScript, istənilən iki leksem arasında sətir boşluğu güman edir, amma JavaScript-in sintaktis analizatorunun avtomatik nöqtəli vergülləri qoyma vərmişinin bəzi istisnaları mövcuddur. Əgər proqram kodunun sətirində bitirilmiş təlimatdan sonar yeni sətirə keçilibsə, JavaScript-in sintaktis analizatoru nöqtəli vergülü avtomatik qoyur.

Məsələn, aşağıdakı fraqmentə baxaq:

```
return
```

```
true;
```

JavaScript-in sintaktis analizatoru güman edir ki, proqramçının yazdığı kod aşağıdakı kimidir:

```
return;
```


true;

Hərçənd ki əslində proqramçı, return true; yazmaq istəyirdi. Gördüyünüz kimi, diqqətli olmaq lazımdır – bu kod sintaktis cəhətdən səhv deyil, amma kodda aşkar olmayan nasazlığı mövcuddur. Oxşar xoşagəlməz hadisə: break outerloop; yazanda da yarana bilər. JavaScript break açar sözündən sonra nöqtəli vergülü avtomatik qoyur və növbəti sətiri icra edildikdə xəta ilə üzləşir. Analoji səbəblərə görə postfix operatorlarını (++ və --) (5-ci fəsilə baxmaq), aid olduğu ifadənin sətirində yerləşdirmək məcburidir.

Şərhlər

JavaScript, həmçinin Java və C++, C stilində olan şərhləri dəstəkləyir. Sətir sonunda // simvolları arasında daxil edilən istənilən mətnə, şərh kimi baxılır. Şərhlərin icraedici funksiyası yoxdur. Həmçinin /* və */ simvolları arasında daxil edilən istənilən mətnə şərh kimi baxılır. C stilində sözügedən şərhlər bir neçə sətirdən ibarət ola bilər və qoyulmuş ola bilmir. Aşağıdakı kod sətirlərində düzgün JavaScript-şərhləri göstərilmişdir:

```
// Bu təkşətirli şərhdir.  
/* Bu da həmçinin şərhdir */ // və bu başqa cür şərh formasıdır.  
  
/*  
 * Bu da daha bir şərh formasıdır.  
 * Bu formada olan şərhlər bir neçə sətirdə yerləşdirilə bilər.  
 */
```

Literallar

Literal – proqramın mətnində bilavasitə göstərilmiş qiymətdir. Aşağıda bəzi literal nümunələri göstərilmişdir:

```
12           // On iki ədədi  
1.2          // Bir tam onda iki onluq ədədi  
"hello world" // Mətn sətiri  
'Hi'        // Başqa cür sətir  
true         // Məntiqi qiymət  
false        // Digər bir məntiqi qiymət  
/javascript/gi // Müntəzəm ifadə (şablon üzrə axtarış üçün)  
null         // Obyektin yoxluğu
```

ECMAScript v3-də ifadələrdə həmçinin massiv literalları və obyekt literalları dəstəklənir. Məsələn:

```
{ x:1, y:2}   // Obyektin inisializatoru  
[1,2,3,4,5]   // Massivin inisializatoru
```

Literallar – istənilən proqramlaşdırma dilinin mühüm hissəsidir, çünki, literalsız proqram yazmaq mümkün deyil. JavaScript-də olan müxtəlif literallar 3-cü fəsildə təsvir edilmişdir.

İdentifikatorlar

İdentifikator – sadəcə verilən addır. JavaScript-də identifikatorlar, dəyişənlərin və funksiyaların adları kimi, həmçinin bəzi dövrlərin nişanları kimi çıxış edir. Mümkün identifikatorların formalaşması qaydaları Java və bir çox başqa proqramlaşdırma dillərinin qaydaları ilə eynidir. Birinci simvol hərflər, sətir xətti () simvolu və ya dollar (\$) işarəsi

olmalıdır. Birinci simvoldan sonra istənilən hərf, rəqəm, sətir xətti simvolu (\\) və ya dollar işarəsi ola bilər. (Birinci simvol heç bir halda rəqəm ola bilməz, çünki bu halda şərhçi ədədləri identifikatorlardan ayırmağa çətinlik çəkir.) Mümkün identifikator nümunələri:

```
i
my_variable_name
v13
_dummy
$str
```

ECMAScript v3-də identifikatorlar Unicode simvol dəstində olan bütün hərfləri və rəqəmləri ehtiva edə bilər. Standartın bu versiyasına qədər JavaScript- identifikatorları ASCII dəsti ilə məhdudlaşmışdı. Bu işarə yalnız kodun generasiyası vasitələri üçün nəzərdə tutulmuşdur, buna görə də identifikatorlarda bu işarənin istifadəsindən çəkinmək lazımdır. 2.8. Ehtiyata saxlanılan sözlər Unicode escape- ardıcılıqları – identifikatorlarda 16 dərəcəli 4 onaltılıq rəqəmdən ibarət simvol kodu olan \\u simvol birləşməsi vasitəsilə icra edilir. Məsələn, π identifikatoru üçün \\u03c0 kimi yazmaq olar. Bu sintaksis bu cür narahat görünə bilər, amma mətnlər ilə iş zamanı tam Unicode simvol dəstinə dəstəkləməyən redaktorlarda və ya digər vasitələrdə bu Unicode simvollarının JavaScript- proqramların transliterasiyasını təmin edir.

Nəhayət, identifikatorlar ilə JavaScript-də digər məqsədlər üçün istifadə edilən açar sözləri bir-birilə uyğun gəlmir. Aşağıdakı bölmədə JavaScript-in xüsusi ehtiyacları üçün ehtiyata saxlanmış açar sözlər göstərilmişdir.

Ehtiyata saxlanılan sözlər

JavaScript-də bir neçə ehtiyata saxlanılan söz mövcuddur. Bu sözlər JavaScript-proqramlarında identifikator kimi (dəyişən, funksiya və dövrlərin nişan adları kimi) çıxış edə bilməz. Cədvəldə ECMAScript v3-də standartlaşdırılmış açar sözləri göstərilmişdir. JavaScript interpretatoruna görə bu sözlər xüsusi qiymətə malikdir və dilin sintaksis hissəsini təşkil edir.

break	do	if	switch	typeof
case	else	in	this	var
catch	false	instanceof	throw	void
continue	finally	new	true	while
default	for	null	try	with
delete	function	return		

Digər cədvəldə başqa açar sözləri göstərilmişdir. Hal-hazırda bu sözlər JavaScript-də istifadə olunmur, amma dilin gələcək inkişaf etdirmələri üçün ECMAScript v3-də ehtiyatda saxlanmışdır.

Abstract	double	goto	native	static
Boolean	enum	implements	package	super
byte	export	import	private	synchronized
char	extends	int	protected	throws
class	final	interface	public	transient
const	float	long	short	volatile

ECMAScript v4 standartının cari layihələri as, is, namespace və use açar sözlərinin tətbiqinə imkan verir. Hərçənd ki, JavaScript cari interpretatorları bu dörd açar sözündən identifikator kimi istifadə edilməsini qadağan etmir, ancaq identifikatorlarda bu sözlərin istifadəsindən çəkinmək lazımdır.

Bundan başqa, JavaScript dilində qabaqcadan müəyyən edilmiş qlobal dəyişən və funksiya identifikatorlarının istifadədən çəkinmək lazımdır. Aşağıdakı cədvəldə ECMAScript v 3 standartı ilə müəyyən edilən qlobal dəyişənlər və funksiyalar göstərilmişdir. Konkret reallaşdırmalar öz qlobal görünmə sahəsi vasitəsilə qabaqcadan müəyyən edilmiş elementləri ehtiva edə bilər. Bundan başqa, JavaScript-in konkret platforması üzrə (müşəri, server və başqalar) bu siyahını genişləndirə bilərik.

<code>argument</code>	<code>encodeURIComponent</code>	<code>Infinity</code>	<code>Object</code>	<code>String</code>
<code>Array</code>	<code>Error</code>	<code>isFinite</code>	<code>parseFloat</code>	<code>SyntaxError</code>
<code>Boolean</code>	<code>escape</code>	<code>isNaN</code>	<code>parseInt</code>	<code>TypeError</code>
<code>Date</code>	<code>eval</code>	<code>Math</code>	<code>RangeError</code>	<code>undefined</code>
<code>decodeURI</code>	<code>EvalError</code>	<code>NaN</code>	<code>ReferenceError</code>	<code>unescape</code>
<code>URIComponent</code>	<code>Function</code>	<code>Number</code>	<code>RegExp</code>	<code>URIError</code>

Məlumat tipləri və qiymətlər

Kompüter proqramları qiymətlərin manipulyasiya nəticəsində işləyir. Proqramlaşdırma dilində təqdim edilmiş və emal edilə bilən, qiymətlər məlumatlar tipindən (data types) və proqramlaşdırma dilinin ən fundamental xarakteristikalarından biri olan bu tipi dəstəkləyən məlumat tiplərinin dəstindən ibarətdir. JavaScript üç elementar məlumat tipi ilə işləməyə imkan verir: ədədlər, mətn sətirləri (və ya sadəcə sətirlər) və məntiqi doğruluq qiymətləri (və ya sadəcə məntiqi qiymətlər). JavaScript-də həmçinin iki bayağı məlumat tipi təyin edilir: null və undefined. Bu qiymətlərin hər biri yalnız bir qiymətə malikdir.

JavaScript bu elementar məlumat tiplərindən savayı obyekt (object) kimi bilinən tərkib məlumat tipini də dəstəkləyir. Obyekt (yəni, obyekt məlumat tipinin üzvü) qiymətlərin (ədədlər sətirlər və ya elementlər kimi və ya daha mürəkkəb, məsələn başqa obyektlər) kolleksiyasını təşkil edir. Obyektlər JavaScript-də ikili təbiətə malikdir: obyekt adlandırılmış qiymətlərin qaydaya salınmamış kolleksiyası kimi və ya nömrələnmiş qiymətlərin qaydaya salınmış kolleksiyası kimi. Obyektin sonuncu vəziyyətində obyekt massiv (array) adlanır. Hərçənd ki, JavaScript-də obyektlər və massivlər ayrı-ayrı məlumat tipidir və bu kitabda ayrı tiplər kimi baxılır.

JavaScript-də obyektin daha bir xüsusi tipi müəyyən edən funksiya (function). Funksiya – icra edilən kodun bağlandığı obyektidir. Funksiya (invoked), müəyyən əməliyyatın icra edilməsi üçün çağırıla bilər. Massivlər kimi, funksiyalar da, digər obyekt növlərindən fərqlər və JavaScript-də funksiyalar ilə işləmək üçün xüsusi sintaksis müəyyən edilmişdir. Buna görə biz obyektlərdən və massivlərdən asılı olmayaraq funksiyalarla tanış olacağıq.

JavaScript-in baza dilində funksiyalardan və massivlərdən başqa obyektlərin bir qədər xüsusi növləri də vardır. Bu obyektlər yeni olmayan məlumat tiplərini və yalnız obyektlərin yeni siniflərini (classes) dəstəkləyir

Bu fəsilin qalan hissəsində elementar tiplərin hər biri təfərrüatı ilə təsvir edilmişdir.

Fəsil başlanğıc səviyyə üçün bir qədər dar ixtisaslaşdırılmış təfərrüatları özündə ehtiva edir.

Ədədlər

Ədədlər – xüsusi izah tələb etməyən əsas məlumat tipidir. JavaScript C və Java kimi PD-lərindən fərqli olaraq, o qədər ki, etmir tam və natural ədədlər arasında fərq qoymur. JavaScript-də bütün ədədlər 64-dərəcəli natural maddi qiymətlər ilə təqdim edilir. Bu format IEEE 754.1 standartı ilə təyin edilmişdir.² Bu format $\pm 1,7976931348623157 \times 10^{308}$ dən $\pm 5 \times 10^{-324}$ qədər olan intervalda olan ədədləri təyin edə bilər.

Bilavasitə JavaScript- proqramının kodunda olan ədəd, ədəd literalı adlanır. JavaScript bir neçə ədəd literalını dəstəkləyir. Bu haqda sonrakı bölmələrdə bəhs olunur. Diqqətli olun: istənilən mənfi ədəd literalı əvvəlində "mənfi" işarəsi qoyulur. Ancaq faktiki olaraq mənfi (minus) işarəsinin dəyişməsi ədəd literallarının sintaksis hissəsi olmayan unar operatorunu təşkil edir (5-ci fəsilə baxmaq) .

Tam literallar

JavaScript-də tam onluq ədədlər rəqəm ardıcılığı ilə yazılır. Məsələn

0

3

10000000

JavaScript-in ədəd formatı $9007199254740992 (-2^{53})$ dən $9007199254740992 (2^{53})$ qədər olan ədəd diapazonunda bütün tam ədədləri dəqiq təqdim etməyə imkan verir. Bu diapazondan kənar tam qiymətlərin kiçik dərəcədə dəqiqliyi itə bilər. Qeyd etmək lazımdır ki, JavaScript-də (xüsusən bit operatorları, 5-ci fəsildə təsvir edilmişdir) bəzi tam ədəd əməliyyatları 32-dərəcəli tam ədədlər ilə icra olunur və $2147483648 (-2^{31})$ dən $2147483647 (2^{31} - 1)$ -ə qədər olan qiymətləri alır.

Onaltılıq və səkkizlik literallar

JavaScript-in onluq tam literallarında başqa onaltılıq say sistemində (əsaslı 16 olan) olan qiymətləri də tanıyır. Onaltılıq literal "0x" və ya "0X" simvolların ardıcılığı ilə başlayır. Bu halda, sətir onaltılıq ədəddən təşkil olunur

Onaltılıq rəqəm – 0- dan 9-a qədər və ya a-dan (və ya A) f-ə qədər (və ya F) qədər hərflərdən ibarət rəqəmdir. Onaltılıq sistemin tam ardıcılığı (0-F) onluq say sisteminin 0-15 ədədlərinə uyğundur. Aşağıda onaltılıq tam ədəd literallarının nümunələri göstərilib:

0xff // $15 \times 16 + 15 = 255$ (10 əsaslı)

0xCAFE911

Hərçənd ECMAScript standartı səkkizlik say sistemində tam ədəd literallarını dəstəkləmir, yalnız JavaScript-in bəzi reallaşdırmaları bu tip məsələləri həll etməyə icazə verir. Səkkizlik literallar 0-7 aralığında olur və digər ədədlər bu rəqəmlərin vasitəsilə düzəlir. Məsələn:

$0377 // 3*64 + 7*8 + 7 = 255 (10)$

Bəzi reallaşdırmalar səkkizlik say sisteminin literallarını dəstəklədiyi halda, bəziləri dəstəkləmir.

Həqiqi ədəd literalları

Həqiqi ədəd literalları onluq say sisteminə malik olmalıdır; bu literallar həqiqi ədədlərin ənənəvi sintaksisindən ibarətdir. Tam qiymət tam ədədin, onluq kəsrini və ya qalıqlı ədədin tam hissəsi müəyyən edir. Həqiqi ədədlərin literalları həmçinin səciyyəvi nəsihətdə təqdim edilə bilər: həqiqi ədəd, e ədədini (və ya E), plus və ya minus və tam eksponenti dəstəkləyir. Bu nəsihət 10-da qiymətlə (mənayla) müəyyən edilən dərəcələrə vurulmuş (artırılmış) həqiqi ədədi ifadə edir eksponentlər. Belə sintaksisin daha yığcam təyini:

[rəqəmlər] [.rəqəmlər] [(E|e) [(+|-)] rəqəmlər]

Ədədlərlə iş

JavaScript-dilində proqramlarda ədədlərlə işləmək üçün hesab operatorlarından istifadə edilir. Əsas hesab operatorları, toplama, çıxma, vurma, bölmə operatorlarından ibarətdir

JavaScript-in göstərilmiş əsas hesab operatorlarından başqa daha mürəkkəb riyazi əməliyyatların icra edilməsinə kömək edən və dilin baza hissəsinə aid olan böyük miqdarda riyazi funksiya aiddir. Rahatlıq üçün bu funksiyalar Math obyektinin xüsusiyyətləri şəklində saxlanılır və funksiyalara giriş üçün həmişə Math hərfi adından istifadə olunur.

Məsələn, sinus x dəyişəninin ədədi qiymətini aşağıdakı qaydada hesablamaq olar:

```
sin_of_x = Math.sin (x);
```

Ədədin kvadrat kökü isə belə hesablanır:

```
hypot = Math.sqrt (x*x + y*y);
```

JavaScript-də dəstəklənən bütün riyazi funksiyalar, həmçinin Math obyektinin haqqında kitabın üçüncü hissəsində geniş məlumat verilir

Sətirlər

Sətir, JavaScript-də hərf, rəqəm, punktuasiya nişanları və digər Unicode-simvollar ardıcılıqlarını və həmçinin mətnin daxil edilməsinə imkan verən məlumat tipidir. Siz növbəti dərslərdə görə bilərsiniz ki, sətir literalları cüt və ya tək dırnaq (apostrof) simvolları arasında olur.

Diqqətli olun: C, C++ və Java-da simvolik məlumat tipi kimi bilinən char tipi JavaScript-də yoxdur. Tək simvol tək uzunluğa malik sətir ilə təqdim edilə bilər.

Sətir literalları

Sətir literalı – tək və ya ikiqat dırnaqlarla (' və ya ") əhatə olunmuş Unicode- simvolları ardıcılığıdır. Əgər mətnə ikiqat dırnaqlardan istifadə etmək lazımdırsa literalı tək dırnaqlarla (') əhatələmək (içərisində yazmaq) lazımdır. Eyni qaydanın əksi olaraq, əgər mətnə tək dırnaqlardan (apostrof) istifadə etmək lazımdırsa literalı cüt dırnaqlarla (") əhatələmək (içərisində yazmaq) lazımdır. Sətir literalı proqramın bir sətirində yazılmalıdır və ikinci sətirə keçirilmir. Sətir literalında yeni sətir daxil etmək üçün \n simvol ardıcılığından istifadə etmək lazımdır.

Nəzərə alın ki, tək dırnaq ilə məhdudlaşdırmış sətir ilə ehtiyatlı davranmaq lazımdır. Kiçik bir yiyəlik halı şəkilçisi (can't, O'Reilly kimi) olan apostrofun düzgün daxil edilməməsi nəticəsində xəta baş verə bilər. Çünki apostrof ilə tək dırnaq işarəsi eyni bir simvoldur və istisna halı metodu ilə əks sləş simvolu vasitəsilə istifadə edilməlidir. JavaScript kliyentində proqramlar adətən HTML-koddan ibarət ola bilər və öz növbəsində HTML-kodda, JavaScript-kodunun sətirlərini ehtiva edə bilər. Buna görə də JavaScript-i HTML teqlərə və hadisələrə tətbiq edərkən JavaScript kodu dırnaq içərisində yazılır. Aşağıdakı nümunədə JavaScript-ifadəsi kimi "Təşəkkür" sətiri tək dırnaqlarla əhatələnmişdir. Kodun özü isə HTML atributunun hadisə emalçısına mənimsədildiyinə görə cüt dırnaqlar ilə əhatələnmişdir:

```
<a href="" onclick=" alert ('Təşəkkürlər')">Məni klikləyin!</a>
```

Sətir literallarında idarəedici ardıcılıqlar

Əks sləş (\) simvolu JavaScript-sətirlərində xüsusi təyinatla malikdir. Onu yanında gələn simvollarla birlikdə, müvafiq simvol ifadə edir. Adətən sətirin daxilində yerləşdirilməsi mümkün olmayan simvolların sətirə mənimsədilməsi üçün istifadə edilir. Məsələn, \n – yeni sətirə keçmək üçün istifadə olunur.

Əvvəlki bölmədə qeyd olunmuş başqa nümunə – tək dırnaq simvolunu ifadə edən \' simvol birləşməsidir. Bu idarəedici sətir ardıcılığı tək dırnaq ilə əhatələnmiş sətir sahəsinə tək dırnağın simvolunun əlavə edilməsi üçündür. İndi bizə aydın olur ki, niyə bu ardıcılıqları idarə edici adlandırırıq. Burada əks sləş simvolu tək dırnaq simvolunun interpretasiyasını idarə etməyə

imkan verir. Aşağıdakı nümunədə biz tək dırnaq işarəsinin əks sləş simvolu ilə tətbiqi nəticəsində biz bu işarəni sətir sonluğu kimi deyil, apostrof kimi veririk:

```
'You\'re right, it can\'t be a quote'
```

Sətirlərlə iş

JavaScript-in integrasiya edilmiş imkanlarından biri də sətirləri birləşdirməkdir. Əgər + operatoru ədədlərə tətbiq edilsə, bu ədədlər toplanır, əgər sətirlərə tətbiq edilsə, bu sətirlər bitişdirilir, bu halda ikinci birinci sətirin sonuna əlavə edilir. Məsələn:

```
msg = "Hello, " + "world"; // Alınır: "Hello, world"
```

```
greeting = "Hörmətli, " + name + ". Mənim ana səhifəmə xoş gəlmisiniz";
```

Sətir uzunluğunun (sətirdə olan simvolların miqdarı) təyini üçün length xüsusiyyətindən istifadə olunur. Məsələn, əgər s dəyişəni sətiri tipindədirsə, bu sətirin uzunluğunu aşağıdakı qaydada göstərmək olar:

```
s.length
```

Sətirlərlə işləmək üçün bir neçə metod mövcuddur. s sətirinin son simvolu belə göstərmək olar:

```
last_char = s.charAt(s.length - 1)
```

İkinci simvolu çıxartmaq üçün, s sətirinin üçüncü və dördüncü simvolları, təlimata tətbiq edilir:

```
sub = s.substring (1,4);
```

S sətirində birinci simvol olan "a" hərfinin mövqeyini aşağıdakı qaydada müəyyən etmək olar:

```
i = s.indexOf ('a');
```

Sətirlərlə işləyən digər metodlar da vardır. Bu metodların təfərrüatı String obyektinin təsvirində və kitabın üçüncü hissəsinin listinqlərində sənədlərlə əsaslandırılmışdır. Əvvəlki nümunələrdən anlamaq olar ki, JavaScript-sətirləri (və gələcəkdə tanış olacağımız JavaScript massivləri) 0-dan başlayaraq indeksləşdirilir. Başqa sözlə, sətirin birinci simvolunun sıra nömrəsi sıfıra bərabərdir. C, C++ və Java-da işləmiş proqramçılara bu metod tanışdır. Bir məqam da var ki, bu dillərdə sətirlərin və massivlərin numerasiyası vahiddən başlanır. JavaScript-in bəzi (bir qədər) reallaşdırmalarında ayrı simvollar götürülə bilər sətirlər (amma sətirlərə yazılmamaq) sətirlərə müraciət (rəftar) vaxtı (yanında) necə massivlərə, daha əvvəl nəticədə göstərilən charAt() metodunun çağırışı aşağıdakı qaydada yazılmış ola bilər:


```
last_char = s [s.length - 1];
```

Ancaq bu sintaksis ECMAScript v3-də standartlaşdırılmamışdır, daşınan deyil və ondan çəkinmək lazımdır.

Biz obyekt məlumat tipi müzakirə edəndə, xüsusiyyətin əvvəlki nümunələri və sətirlər metodları obyektlər metodları kimi istifadə olunur. Bu o demək deyil ki, sətirlər – obyektlərin tipidir. Əslində sətirlər JavaScript-in ayrı bir məlumat tipidir. Onların xüsusiyyətlərinə və metodlarına giriş üçün obyekt sintaksisi istifadə olunur, amma sətir özü heç vaxt obyekt olmur! Bunun niyə belə olduğunu, biz bu fəsilin sonunda biləcəyik.

Ədəd-sətir dəyişikliyi

Sətir ədəd kontekstində istifadə olunduqda, avtomatik olaraq ədədə dəyişdiriləcək. Məsələn, aşağıdakı ifadə tamamilə mümkündür:

```
var product = "21" * "2"; // nəticədə 42 ədədi alınacaq.
```

Bu şəraitin əksi zəruri olduqda ədədi sətiri dəyişdirmək olar; bunun üçün kifayət qədər sadə metod, sətirdən 0 qiymətini çıxmaq lazımdır:

```
var number = string_value - 0;
```

(Diqqətli olun: bu vəziyyətdə toplama əməliyyatı sətirlərin bitişdirilməsi əməliyyatı kimi yerinə yetiriləcək.)

Ədəd sətir dəyişikliyinə daha az inkişaf etmiş və daha düzxətli üsulu Number() funksiya konstruktoruna müraciət ilə icra edilir:

```
var number = Number (string_value);
```

Ədəd-sətir dəyişikliyinə belə üsulunun çatışmazlığı ondan ibarətdir ki, bu qədər sadə əməliyyatı həddən artıq ciddi üsulla yerinə yetirir. Bu üsul yalnız onluq say sistemində oluna bilər və bu üsul yalnız rəqəm mövcudluğunu güman edərək, boşluq simvolları, sətirdə ədəddən sonra gələn başqa qeyri-rəqəmsal simvolların yaranmasına icazə verir. Dəyişikliyin daha elastik üsulu parseInt() və parseFloat() funksiyalarının köməyi ilə təmin olunur. Bu funksiyalar istənilən qeyri-rəqəmsal simvollarla məhəl qoymadan sətirin başlanğıcında duran sərbəst ədədləri dəyişdirəcək və ədədin ardınca yerləşdirilmiş simvollar qaytarılacaq. parseInt() funksiyası yalnız tam ədəd dəyişikliyinə yerinə yetirir. parseFloat() funksiyası isə həm tam, həm də həqiqi ədədlər ilə ədəd dəyişikliyinə yerinə yetirə bilər. Əgər sətir "0x" və ya "0x" simvollarından başlayırsa, parseInt() funksiyası sətiri onaltılıq ədəd kimi göstərir. Məsələn:


```
parseInt ("3 dovşan"); // 3 qiymətini alır  
parseFloat ("3.14 metr") ; // 3.14 qiymətini alır  
parseInt (" 12.34"); // 12 qiymətini alır  
parseInt ("0xFF"); // 255 qiymətini alır
```

İkinci argument kimi parseInt() funksiyası hesablama sistemlərini də qəbul edə bilər. 2-dən 36-a qədər olan ədəd diapozunda düzgün alınır⁴ , məsələn:

```
parseInt ("11", 2); // 3 (1*2 + 1) qiymətini alır  
parseInt ("ff", 16); // 255 qiymətini alır (15*16 + 15)  
parseInt ("zz", 36); // 1295 qiymətini alır (35*36 + 35)  
parseInt (" 077", 8); // 63 (7*8 + 7) Qaytaracaq  
parseInt (" 077", 10); // 77 (7*10 + 7) Qaytaracaq
```

Əgər parseInt() və parseFloat() metodlarında dəyişliyi yerinə yetirmək mümkün deyilsə, onlar NaN qiymətini alır:

```
parseInt ("eleven"); // NaN qiymətini  
parseFloat (" $72.47"); // NaN qiymətini alacaq.
```

Məntiqi qiymətlər

Ədəd və sətir məlumat tipləri böyük və ya sonsuz mümkün qiymətlər miqdarına malikdir. Məntiqi məlumat tipi isə, əksinə, yalnız iki true və false literalları ilə təqdim edilmiş mümkün məntiqi qiymətlərdən ibarətdir.

Məntiqi qiymət, doğruluq ətrafında icra edilir. JavaScript-proqramlarda yerinə yetirilən müqayisələrin nəticələri adətən məntiqi qiymətlərdir. Məsələn:

```
a == 4
```

Bu ifadə a dəyişənin 4 ədədinə bərabər olmasını yoxluyur. Əgər a dəyişən həqiqətən də dördə bərabərdirsə, true məntiqi qiymətinin şərti ödənilir. Əgər a dəyişəni dördə bərabər deyilsə, müqayisənin nəticəsi false olacaq.

JavaScript-də məntiqi qiymətlər adətən idarəedici konstruksiyalarda istifadə olunur. Məsələn, JavaScript-də if/else təlimatı hər hansı bir argumenti yerinə yetirir, əgər məntiqi qiymət true-a bərabər olarsa, əməliyyat yerinə yetirilir, əks halda false qiyməti alınır. Adətən məntiqi qiyməti yaradan müqayisə bilavasitə istifadə olunan təlimatla birləşir. Bu sözü kodla ifadə edək:

```
if (a == 4)
```

```
b = b + 1;
```

else

a = a + 1;

Burada yoxlama yerinə yetirilir, əgər a dəyişəni 4 ədədinə bərabərdirsə b dəyişəninin qiymətinə bir vahid əlavə edilir; əks təqdirdə a dəyişəninin qiymətinə bir vahid əlavə edilir.

İki mümkün true və false məntiqi qiymətlərinə, bəzən "düzdür" (true) və ya "səhvdir" (false) və ya "bəli" (true) və "xeyr" (false) kimi baxılır.

Məntiqi qiymətlərin dəyişikliyi

Məntiqi qiymətlər başqa tiplərin qiymətlərinə asan dəyişdirilir, həm də bu tip dəyişikliklər əksər hallarda avtomatik yerinə yetirilir⁵. Əgər məntiq ədəd kontekstində istifadə olunur. true qiyməti 1- ədədinə, false qiyməti isə 0 ədədinə dəyişdiriləcək. Əgər məntiqi qiymət sətir kontekstində istifadə olunursa, onda true qiyməti "true" sətirinə, false qiyməti isə "false" sətirinə dəyişdiriləcək. Məntiqi qiymət birdən yuxarı ədəd kimi istifadə olunduqda, true qiymətinə dəyişdiriləcək. Əgər qiymətlər 0-a və ya NaN-a bərabərsə, false məntiqi qiymətinə dəyişdiriləcək. Məntiqi qiymətlərdə sətirlərdə istifadə edildikdə, əgər sətir boş sətir deyilsə, true qiymətinə dəyişdiriləcək, əks təqdirdə dəyişiklik nəticəsində false qiyməti alınacaq. Null və undefined xüsusi qiymətləri false qiymətinə dəyişdiriləcək və istənilən funksiya, obyekt və ya massivin qiymətləri null-dan böyükdür və true qiymətinə dəyişdiriləcəklər. Əgər siz dəyişikliyi açıq- aydın yerinə yetirmək istəyirsinizsə, Boolean() funksiyasından istifadə etmək olar:

```
var x_as_boolean = Boolean(x);
```

Açıq dəyişikliyin başqa üsulu ikiqat məntiqi inkarın operatorunun istifadəsi ilə mümkündür:

```
var x_as_boolean =!! x; 3.4.
```

Funksiyalar

Funksiya – icra edilən kodun fragmentidir, hansı ki, JavaScript-programında və ya JavaScript reallaşdırmasında qabaqcadan müəyyən edilmişdir. Funksiya JavaScript- programında yalnız bir dəfə təyin edilir, lakin istənilən icra edilə və ya səbəb çağırılı bilər Funksiyalar arqumentləri, qiyməti və ya qiymətləri müəyyən edən və hesablamaları yerinə yetirməli olan parametrləri ötürə bilər; həmçinin funksiya bu hesablamaların nəticəsini təşkil edən qiyməti qaytara bilər. JavaScript reallaşdırmaları bir çox qabaqcadan müəyyən edilmiş funksiya təklif edir. Bunlara misal olaraq, bucağın sinusunu hesablayan Math.sin() funksiyasını göstərmək olar. JavaScript, proqramların ehtiva etdiyi şəxsi funksiyaları da müəyyən edə bilər məsələn, belə bir kod:

```
function square (x) // Funksiya square adlanır. O bir x arqumentini qəbul edir.
```

```

{
    // Burada funksiyanın əsası başlanır.

return x*x;    // Funksiya öz argumentini kvadrata yüksəldir və alınmış qiymətə qaydır

}
    // Burada funksiya bitir.

```

Bir çox dillərdə, həmçinin Java-da funksiyalar – dilin məlumat tipi kimi deyil, yalnız sintaktis elementləridir: funksiyaları müəyyən etmək və çağırmaq olar. O halda ki, JavaScript-də funksiyalar əsl qiymətlər (təşkil edir, dilə böyüyü verir elastiklik. Bu bildirir ki, funksiyalar dəyişənlərdə saxlanıla bilər, massivlər və obyektlər, həmçinin argumentlər kimi başqa funksiyalara ötürülmək.

Funksional literallar

Əvvəlki bölmədə biz square() funksiyanının təyini gördük. Adətən JavaScript- proqramlarında funksiyaların əksəriyyəti bu sintaksis ilə təsvir edilir. Ancaq ECMAScript v3 standartı funksional literalların təyini üçün sintaksisə (JavaScript 1.2-də və daha gec versiyalarda reallaşdırılmış) malikdir. Funksional literal function açar sözünün köməyi ilə, funksiyanın adı ilə birlikdə verilir. Funksiyanın adının qarşısında duran mötərizələrin içərisində bu funksiyanın malik olduğu argumentlərin siyahısı göstərilir. Funksiya fiqurlu mötərizələr ilə başlayıb, fiqurlu mötərizələr ilə bitir. Funksional literal ilə təyin olun funksiyalara ad verilməyə bilər. Ən böyük fərq ondan ibarətdir ki, funksional literallar digər JavaScript- ifadələri kimi yazıla bilər. Yəni, square() bu şəkildə funksiyanı təyin etmək vacib deyil:

```
function square (x)
```

```

{

return x*x;

}

```

Bunu literalların köməyi ilə aşağıdakı kimi də etmək olar:

```
var square = function (x)
```

```

{

return x*x;

}

```

Belə, müəyyən edilmiş funksiyalar adətən lyambda-funksiya adlandırılır. Bu, üsul ilk dəfə LISP proqramlaşdırma dilində istifadə edilmişdir. Elementar səviyyədə siz bu literalların xeyirini

görməsənizdə, mürəkkəb ssenarilərdə siz görəcəksiniz ki, onlar kifayət qədər rahat və faydalıdır. Funksiyanın təyininin daha bir üsulu mövcuddur: argumentlərin siyahısını və funksiyanın əsasını Function()-konstruktura sətirlər şəklində vermək olar. Məsələn:

```
var square = new Function (" x", "return x*x;");
```

Funksiyaların bu cür təyindən nadir hallarda istifadə olunur. Adətən burada əsas hissəsini sətir şəklində vermək narahatdır və oxşar tərzdə müəyyən edilmiş funksiyalar yuxarı sadalanan iki üsul ilə daha az effektivdir.

PHP

PHP (ing. PHP: Hypertext Preprocessor) dinamik veb səhifələr yaratmaq üçün nəzərdə tutulmuş bir skriptləşdirmə dilidir. PHP Əsasən veb həllərin yaradılmasında istifadə edilən populyar server proqramlaşdırma dilidir. Məsələn, formadakı məlumatları qəbul edərək email edən skript yaratmaq, saytda axtarış sistemi reallaşdırmaq və ya qeydiyyat sistemi yaratmaq mümkündür. 1994-cü ildə Rasmus Lerdorf CGI alətlərini yazmağa başladı və daha sonra 1995-ci ildə onları mükəmməlləşdirərək PHP dilini ortaya çıxartdı. Əksər hallarda PHP, Linux əməliyyat sistemi, MySQL məlumatlar bazası və Apache veb serveri qısa olaraq LAMP kimi (Linux, Apache, MySQL, PHP) yazılır.

Tarixi

PHP dili 1995-ci ildə yaradılmışdır. İlk əvvəllər onun adı Personal Home Page sözlərinin baş hərflərinin birləşməsindən əmələ gəlmişdir. Daha sonralar daha ciddi ad fikirləşmək məqsədi ilə Hypertext Preprocessor adı verildi. Lakin HPP bir o qədər yaxşı səslənmədiyindən PHP adı qaldı. 2004-cü ildən PHP obyekt yönümlü dil kimi istifadə olundu. Məhz bu ildə Zend şirkəti PHP üçün yeni obyekt modeli tətbiq etməyə başladı. PHP5 versiyasında artıq obyektlər üzvləri (metodlar, xassələr) private, protected, public, static, final kimi ola bilər, interfeyslərdən istifadə oluna bilər. 2009-cu ildə Zend şirkəti PHP üçün Zend [Server](#) platformasını yaratdı. Zend Serverdə PHP kodlar kompilyasiya olunub bayt-kodlar şəklində qalır və bu da öz növbəsində kodun daha sürətli işləməsinə səbəb olur.

Üstünlükləri

1. PHP demək olar ki, hər platformada işləyə bilər. PHP eyni kod əsasını istifadə etdiyi üçün, UNIX, Windows (95/98/NT/2000) və Mac OS daxil olmaqla 25 platformada yığılıb qurula bilər. Kodlar eyni olduğundan scriptə platformadan müstəqil olaraq çalışacaq.
2. PHP, fayl uzantısı ala bilməkdədir. Tətbiqin içərisində iştirak edən nüvə mühərrik (Zend tərəfində yazıldı), bir sıra sadə kod modullarından və kod uzantılarından ibarətdir. Bu səbəblə programçılara PHP uzadılmaları yaradaraq bəzi xüsusi əməliyyatlarını edə bilmələri üçün iki variant təqdim edilir; ya uzantı modullarını yazaraq tətbiq oluna bilən bir yığma etmək, ya da PHP-nin dinamik yükləmə mexanizmi yüklənə tətbiq oluna uzatmalar yaratmaq.

3. PHP bir çox HTTP server interfeysi saxlayır. PHP apacheyə, AOL server'a, Roxen və THTTPD'ye birbaşa yüklənə bilər. Alternativ olaraq CGI modulu olaraq da istifadə edilə bilər.
4. PHP bir çox məlumat bazası interfeysi saxlayır. PHP, MySQL, MS SQL, Oracle, Informix, PostgreSQL və digərləri birbaşa işləyə bilər. Bunlar ikili ədəd düzənindəki interfeysləri ibarətdir və bu həllər üçün verilənlər bazasının dəstəklənmədiyi yerlərdə ODBC dəstəyi təmin edir.
5. Bir PHP istifadəçisi hər hansı bir kitabxana üçün interfeys təşkil çətinlik çəkməz. Bir çox istifadəçi bu yolu seçmiş, qrafik təkrarlananları, PDF faylları, Flash Movie'leri, Cybercash xətkəş / cədvəlləri, XML, IMAP, POP və digərləri əlaqədar modullar tapa bilmişdir.
6. Pear, PHP-nin davamı və Add-on anbarıdır. Pear, Perl üçün inkişaf etdirilən CPAN'e bənzəyir. Hələ də başlanğıc mərhələsində olmasına baxmayaraq pear, PHP-nin kurulumuyla birlikdə gələcək bir sıra PHP script'ini istifadəyə təqdim etməkdədir.

PHPdə kod quruluşu

PHP'nin quruluşu C, Perl və Java dilinə oxşayır. PHP, XXI əsrdə məşhurlaşmışdır. PHP daxilində HTMLdən də istifadə etmək mümkündür. Bunun üçün `<?php` açma tagindən və `?>` bağlama tagindən istifadə edilməlidir.

Açma və bağlama tagləri

PHPdə kod yazmağa başladığınız zaman açma və bağlama tagləri göstərilməlidir. Əks təqdirdə PHP bunu oxuya bilmir. Bunu

```
<?php
```

```
?>
```

yaza bilərsiniz. `<?php` ilə PHP kodları yazmağa başlayır və `?>` ilə bitir. Əgər yazacağınız hər hansı bir kod `?>` kənarında qalarsa, PHP onu qəbul etməz.

Şərhlərmə

PHPdə mövcud olan bu funksiya ilə siz yazdığınız kodlara şərh verə bilərsiniz. Məsələn;

```
<?php
// Burada tək sətirlik şərh yazı bilərsiniz.

/* Burada isə
Çox sətirli
Kod yazı
Bilərsiniz. */
?>
```

Ekrana yazdırma

PHPdə istədiyiniz bir şeyi ekrana yazdırmaq üçün iki yol mövcuddur. `print` və `echo`. Bunlar arasında əsasən `echo` dan istifadə olunur.

PHP, `echo` daxilində yazılan yazını ekranda göstərir. Yazılan yazı `"..."` aralığında yazılmağı və sonuna `;` qoyulmasını tələb edir. Əks təqdirdə kod işləməyəcək.

Lakin yazacağınız yazı rəqəmlərdən ibarətdirsə o zaman `"..."` işarəsinə görə qalmır. Məsələn;

Dəyişənlər

PHP'də dəyişən yaratmaq üçün `$` işarəsindən istifadə edilir.

HTML daxilində php istifadə etmək

HTML səhifələrin daxilə php proqramlar yazıla bilər. Bu zaman php bu faylı oxuyarkən onu ancaq yazı kimi qəbul edir. Lakin xüsusi teqlərə rast gəldikdə, artıq onu php kod kimi başa düşür və bağlayıcı teqə qədər olan sətirləri tərcümə etməyə başlayır. Başa düşdüyünüz kimi HTML içərisinə php kod daxil etdikdə onu xüsusi teqlər arasında yazmaq lazımdır ki, həmin hissələri php kimi tərcümə etsin. Əks halda onları da yazı kimi oxuyacaq. Bu teqlər müxtəlif formada ola bilər. Bunu etməyin bir çox yolu var. Məsələn:

```
<?php echo("HTML ilə işləyərkən belə edin.\n"); ?>

<? echo ("Daha sadə olması üçün belə də mümkündür.\n"); ?>

<?= ''operator'' ?> Bu ikisi bir-birinin eynidir "<? echo ''operator'' ?>"

<script language="php">
    echo ("Bu şəkildə də php kodu HTML içərisinə daxil etmək olar.");
</script>
```