1. Introducere

Jocurile au devenit una din principalele surse de divertisment in societatea moderna. Ele ne pot imbunatati capacitatea de a lua o decizie intr-un timp scurt, dezvolta inteligenta, reduce stresul sau chiar depresia. Cu toate acestea, exista jocuri de tip simulatoare, cu ajutorul carora oamenii se pot pregati pentru diverse diverse actiuni ce sunt greu de reprodus (cum ar fi in cadrul armatei unde soldatii folosesc simulatoare pentru masinarii sau arme).

Jocurile video sunt prezente peste tot, spaţiul pe care-l oferă este din ce în ce mai frumos, istoriile pe care le povestesc sunt din ce in ce mai captivante, iar adepţii lor, singuri sau în reţea, trăiesc emoţii excepţionale.

In zilele nostre, tehnologia avanseaza tot mai repede, iar dispozitivele devin indispensabile si tot mai performante. Majoritatea oamenilor detin deja un dispozitiv mobil **cu un sistem de operare avansat, ce permite instalarea unor aplicaţii complexe şi personalizarea extensivă a interfeţei grafice.** Astfel, industria jocurilor se axeaza foarte mult pe aceste dispozitive mobile, iar jocurile au inceput sa fie compatibile cu majoritatea dispozitivelor de acest gen, impreuna cu website-urile.

Motivul pentru care am ales aceasta tema a fost pentru a dezvolta o aplicatie web-based care sa fie compatibila pe mai multe platforme. Jocul este de tip farming si are scopul de a construi piese mecanice necesare imbunatatirii unui robot.

Jocurile de tip farming constau in a construi anumite obiecte pentru a obtine profit, fiind un ciclu infinit, iar jocul neavand un final. Fiecare obiect ce poate fi construit intr-un astfel de joc are un cost, un timp necesar construirii si un profit. Aceste jocuri au devenit foarte populare in ultima vreme, iar website-uri mari precum „Facebook” ofera acces la o intreaga categorie de acest gen.

Intentia lucrarii este de a prezenta un schelet pentru un astfel de joc in care obiectele pot fi configurate cu usurinta chiar si de o persoana care nu are multe cunostinte in limbajul Javascript.

Aplicatia are la baza un framework de Javascript numit „LimeJS”, pentru care am avut nevoie de un serviciu de Apache, limbajul Python si putin HTML5.

~~~~~~~~~~~~~~ insert chapters here ~~~~~~~~~~~~~~~~~~~~~~~~~~~

Cap 2:

2.1.1 Despre HTML  
HyperText Markup Language (HTML) este un limbaj de marcare utilizat pentru paginile web afisate in navigatoare (browsere). Este o forma de marcare orientata catre prezentarea unor documente text pe o singura pagina, respectandu-se un anumit format dupa anumite specificatii. Aceste specificatii sunt dictate de World Wide Web Consortium, un consorțiu internațional, format din membri permanenti si parteneri care dezvolta standardele pentru WWW (World Wide Web).

 HTML furnizează mijloacele prin care conținutul unui document poate fi adnotat cu diverse tipuri de metadate și indicații de redare. Aceste indicații de redare pot varia de la decorațiuni minore ale textului, cum ar fi specificarea faptului că un anumit cuvânt trebuie subliniat sau că o imagine trebuie introdusă, până la scripturi sofisticate, hărți de imagini și formulare, fiecare element putand fi manevrat cu usurinta.

Desi HTML-ul reprezinta un format de text proiectat pentru a putea fi citit si editat de oameni cu ajutorul unui editor de text simplu, scrierea si citirea lor solicita cunostinte solide in HTML.

Cu toate acestea, exista anumite editoare grafice cum ar fi „Macromedia Dreamweaver”, „Adobe GoLive” sau „Microsoft FrontPage” care permit ca paginile web sa fie tratate asemănător cu documetele scrise in Word, dar cu observația că aceste programe generează un cod HTML care este de multe ori de proastă calitate sau chiar plin de bug-uri.

HTML-ul s-a nascut în urma cu aproximativ 30 de ani, într-o tentativa de a rezolva unele probleme ivite la transportul documentelor între diferite computer. Limbajul hypertext a evoluat încet, fiecare versiune avand o contributie importanta.

-Nivelul 1 este obligatoriu pentru toate navigatoarele si înseamna posibilitatea de a interpreta (hyper)text si imagini.

- Nivelul 2 (HTML 2.0) a introdus formularele (forms).

- HTML 3.0 (cunoscut anterior ca HTML+) aduce în plus tabelele, formatarea paragrafelor (alinieri stanga, centru si dreapta), formatarea textului pe langa alte elemente, formule matematice, taburi, note.

- HTML 4.0 a adus cele mai importante modificari si îmbunatatiri, versiune care îmbunatateste totodata si conceptele de accesibilitate si structuralitate a limbajului de marcare. A aparut in 1999 si a avut un impact foarte mare asupra internetului.

- HTML 5 este ultima versiune aparuta si a fost contruita pentru a inlocui atat HTML4 cat si XHTML, alaturi de HTML DOM Level 2. Se axeaza in principal pe animatii, elemente grafice, muzica, filme si poate fi folosit pentru a construi aplicatii web complexe.

Cu toate acestea, grupul care lucreaza la dezvoltarea limbajului HTML a decis ca nu se va mai continua numararea versiunilor de HTML, adica dupa HTML 5 nu va veni HTML6. Decizia vine la scurt timp dupa ce ([World Wide Web Consortium](http://www.w3.org/)) [W3C a introdus un nou logo pentru HTML5](http://megahost.ro/blog/logo-ul-oficial-html5-de-la-w3c.html). Asadar nu vom avea versiunea 6 sau subversiuni, si HTML va fi efectiv un considerat un standard in continua dezvoltare care va fi updatat permanent.

Scoaterea numerotarii versiunilor de HTML a fost decisa in urma unui proces de dezvoltare extrem de dinamic al acestui standard. Aceasta initiativa este asemanatoare cu cea adoptata de dezvoltatorul kernelului de Linux. Fiind updatata foarte des (aproape in fiecare luna apare cate o noua versiune), s-a renuntat la numerotare, iar sistemul de operare isi face update automat.

2.1.2 Despre HTML5

HTML5 este a cincea revizuire a standardului HTML, fiind in curs de dezvoltare din octombrie 2011, obiectivele sale principale fiind de a imbunatati limbajul pentru a suporta elemente multimedia si a ajuta la dezolvatea aplicatiilor web complexe. Astfel, HTML5 isi propune sa insumeze HTML4, XHTML si DOM2HTML (mai ales Javascript).

Acest limbaj a devenit practic o incercare de a amesteca mai multe caracteristici si specificatii diferite intr-un singur limbaj de marcare care poate fi scris in oricare dintre sintaxele HTML sau XHTML.

Pentru a putea incuraja mai multe implementari interoperabilem acesta include modele detaliate de prelucrare, extinde, imbunatateste si rationalizeaza disponibilitatile pentru documentele web si introduce marcarea si diverse aplicarii API (application programming interfaces).

Din acest motiv, HTML5 este un candidat preferat pentru aplicatiile de platforme mobile, acesta avand multe caracteristici capabile sa ruleze pe dispozitive precum smartphone sau tablete.

Dintre cele mai importante caracteristici sintactice noi aduse de acest limbaj, avem <video>, <audio>, <header>, <canvas>, dar HTML5 a integrat continutul SVG pentru inlocuirea tag-ului generic <object>. Toate aceste noutati avand ca scop facilitatea manipularii continuturilor multimedia si a celor grafice. De asemenea, au fost introduse si elemente precum <section>, <article>, <header>, si <nav> pentru a imbunatati continutul semantic al documentelor. Alte elemente, fie au fost inlaturate, fie redefinite, precum <a>, <cite>, <menu>.

Chiar daca HTML5 a fost bine recunoscut printre dezvoltatorii de aplicatii web, in Aprilie 2010, el devine sugiect general de discutie in media dupa ce Steve Jobsm CEO la Apple Inc a dat publicitatii o scrisoare cu titlul „Ganduri despre Flash”, in care acesta concluziona ca Flash nu mai este necesar pentru a urmari filme sau pentru a consuma resursele web, deoarece standardelor noi create si deschise in era mobilului, cum ar fi HTML5. „Adobe Flash”, sau simplu „Flash” este o aplicatie utilizata pentru dezvoltarea obiectelor-applet de tip Flash disponibile in cadrul unor pagini web.

Cu toate ca HTML5 ofera mobilitate si usurinta in a formata un document web, dezvoltatorii trebue sa fie foarte atenti la compatibilitatea browserelor, precum si diferentele functionale dintre HTML5 si Flash. Mai multe despre compatibilitatea limbajului in capitolul 2.1.4.

Un browser care suporta HTML5 (text/html) va fi flexibil in tratarea sintaxelor incorecte. Acest limbaj de marcare este proiectat in asa maniera incat browserele vechi sa poata ignora in siguranta noile constructii HTML5. In cazul unei sintaxe gresite, limbajul ofera reguli detaliate pentru lexinf si parsare pentru ca browserele sa le poata rezolva automat. Cu toate acestea, faptul ca browserele pot interpreta corect o sintaxa gresita, poate fi si un dezavantaj deoarece un dezvoltator poate sa treaca cu vederea peste o eroare ce se poate reproduce doar pe anumite versiuni de interpretoare.

2.1.3 Canvas

HTML5 defineste elementul <canvas> ca "o pânză bitmap dependenta de rezoluție, care poate fi folosita pentru redare grafică, jocuri, sau alte imagini vizuale din mers." Un canvas este ca o panza, unde se poate utiliza JavaScript pentru a picta.

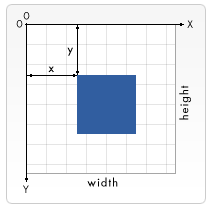
Tag-ul <canvas> a fost introdus pentru prima dată de Apple pentru Mac OS X Dashboard și puse în aplicare mai târziu în Safari și Google Chrome. Browserele bazate pe Gecko 1.8, cum ar fi Firefox 1.5, susțin, de asemenea, acest element. Elementul <canvas> este parte a specificatiilor WHATWG Web applications 1.0, de asemenea, cunoscut sub numele de HTML5.

Nu este dificil a folosi elementul <canvas>, dar este nevoie de o înțelegere de bază a limbajului de marcare HTML și a limbajului de programare JavaScript. Elementul <canvas> nu este acceptat (compatibil) în unele browsere mai vechi, dar este susținut în versiunile recente de toate dimensiunile majore de browsere.

Dimeansiunea standard este de 300px \* 150px (lățime \* înălțime), dar se pot defini dimensiuni personalizate în funcție de înălțimea CSS și proprietatea de lățime. Elementul <canvas> poate fi stilizat la fel ca orice imagine normală (margin, border, background, etc). Cu toate acestea, regulile nu afectează desenul real pe „canvas”.

Elementul <canvas> creează o suprafață de desen de dimensiuni fixe, care expune una sau mai multe contexte de redare, care sunt utilizate pentru a crea și manipula conținutul prezentat. Initial, acest element este gol. Pentru a afișa ceva, este nevoie de un script pentru a accesa contextul de randare și a desena pe ea. Elementul <canvas> are o metoda numita getContext (), utilizata pentru a obține contextul de randare și funcțiile sale de desen. getContext () are un singur parametru, si anume tipul de context. De exemplu, pentru grafica 2D se specifica "2D".

Înainte de a desena ceva pe canvas, trebuie să avem in vedere grila de pânză sau de coordonate spațiu. În mod normal, o unitate din rețea corespunde cu 1 pixel pe panza. Originea acestei grile este poziționat în colțul din stânga sus (de coordonate (0,0)). Toate elementele sunt plasate în raport cu această origine. Astfel încât poziția de colțul din stânga sus al pătrat albastru devine x pixeli de pixeli stânga și y din partea de sus (de coordonate (x, y)).

Spre deosebire de SVG, <canvas> acceptă doar o formă primitivă: dreptunghiuri. Toate celelalte forme trebuie să fie create prin combinarea unuia sau mai multor căi. Din fericire, avem un set de funcții pentru a face posibila compunerea unor forme foarte complexe.

De exemplu, pentru a desena un dreptunghi, avem la dispozitie 3 functii: **fillRect(x, y, width, height) – deseneaza un dreptunghi**

**strokeRect(x, y, width, height) – deseneaza conturul dreptunghiului**

**clearRect(x, y, width, height) – curata o zona dreptunghiulara, facand-o transparenta**

Fiecare dintre aceste trei funcții are aceeași parametri. x și y specifica pozitia pe panza (în raport cu originea) din colțul din stânga sus al dreptunghiului, iar **width** și **height** ofera dimensiunea dreptunghiului.

Pentru a crea forme folosind path-uri trebuie luate unele măsuri suplimentare. În primul rând, trebuie creat un path, apoi de utilizat comenzile pentru desenat pentru a desena in path, dupa care acesta se închide. Dupa toate acestea, se poate folosi sau umple (colora) path-ul pentru randare. Funcțiile utilizate pentru a face acest lucru:

**beginPath() – creaza un path nou**

**closePath() – inchide path-ul**

**stroke() – deseneaza forma, conturand forma**

**fill() – deseneaza o forma solida**

Primul pas pentru a crea o cale este de a apela beginPath (). Pe plan intern, path-urile sunt stocate ca o listă de sub-căi (linii, arce, etc), care formează împreună o forma. De fiecare dată când această metodă este apelata, lista este resetata și se poate desena noi forme.

În cazul în care path-ul curent este gol, cum ar fi imediat după apelul metodei beginPath (), sau pe un canvas nou creat, prima comanda de construcție cale este întotdeauna tratată ca o metoda moveTo (), indiferent de ceea ce este de fapt. Din acest motiv, este bine să se stabilească în mod specific poziția de plecare după resetarea unui path.

Al doilea pas consta in a apela metodele care specifica care path-uri sa fie desenate.

Al treilea pas (care este de altfel optional), este de a apela closePath (). Această metodă încearcă să închidă forma prin desenarea unei linii drepte de la punctul curent la început. În cazul în care forma a fost deja închisă sau nu e doar un punct din lista, această funcție nu face nimic.

Ultimul pas se refera la momentul când se apeleaza metoda fill (), iar toate forme deschise sunt închise în mod automat, astfel încât să nu trebuiasca a fi apelata si metoda closePath ().

O funcție foarte utilă, care nu deseneaza de fapt nimic, dar devine parte a listei cale descrisă mai sus, este functia moveTo. Un exemplu bun ar fi sa ne imaginam acest lucru ca ridicarea unui pix sau creion de la un loc pe o bucată de hârtie și plasând-o pe următoarea.

**moveTo(x, y) – muta obiectul la coordonatele specificate de x si y**

**De asemenea, exista o multitudine de functii care ajuta la desenarea canvasului.**

**lineTo(x, y) – deseneaza o linie dreapta pe pozitiile specificate de x si y**

**arc(x, y, raza, unghiDeInceput, unghiDeSfarsit, inversCeas) - deseneaza un arc**

**quadraticCurveTo(cp1x, cp1y, x, y) – deseneaza o curba** Bézier

Una dintre cele mai interesante caracteristici ale elementului <canvas> este abilitatea de a folosi imagini. Acestea pot fi utilizate pentru a face fotografii dinamice sau ca fundaluri grafice, de sprite în jocuri, și așa mai departe. Imaginile externe pot fi utilizate în orice format acceptat de browser, cum ar fi PNG, GIF, sau JPEG. Se poate utiliza chiar și imaginea produsă de alte elemente de panza pe aceeași pagină ca și sursa!

Importarea imaginilor într-un canvas este de fapt un proces în două etape:

* Se ia o referință la un obiect HTMLImageElement sau la un alt element de panza ca sursă. Nu se poate utiliza o imagine prin utilizarea unui URL catre aceasta.
* Se deseneaza imaginea pe canvas folosind functia drawImage().

Din moment ce se foloseste JavaScript pentru a controla elementele <canvas>, este de asemenea foarte ușor a face animații (interactive), desi a face animații mai complexe poate lua un pic mai mult de lucru.

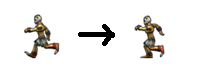
Probabil cea mai mare limitare este că odată ce o formă este desenata, ea rămâne așa. În cazul în care avem nevoie de a muta obiectul, trebuie să-l redesenam si tot ce a fost desenat inaintea lui. Este nevoie de o mulțime de timp pentru a redesena cadre complexe și performanța depinde în mare măsură de viteza de pe computerul pe care se execută.

2.1.4 Sprites

Un sprite este o colecție de imagini puse într-o singură imagine. O pagina web, cu multe imagini poate dura o lungă perioadă de timp pentru a se încărca și generează multiple cereri de servere. Din acest motiv, sprite-urile au devenit o modalitate eficienta de a incarca imaginile. Se va reduce numărul de cereri de server și se va salva mai multa lățime de bandă.

Sprite-urile sunt folosite mai ales in jocurile create cu ajutorul limbajului HTML5. Ele pot fi manipulate cu usurinta pentru a forma o serie de imagini dinamice, o animatie.

Un bun exemplu este aceasta imagine ~~~insert reference here~~~~~~~~care poate fi folosita pentru a simula o persoana care se deplaseaza. Imaginea este compusa din 10 spriteuri si va fi pozitionata in asa fel incat sa se obtina efectul dorit.

Folosind metoda drawImage() pe un canvas, putem schimba poziția imaginei pentru a lasa la vedere doar o porțiune decupată numita " sprite sheet ".

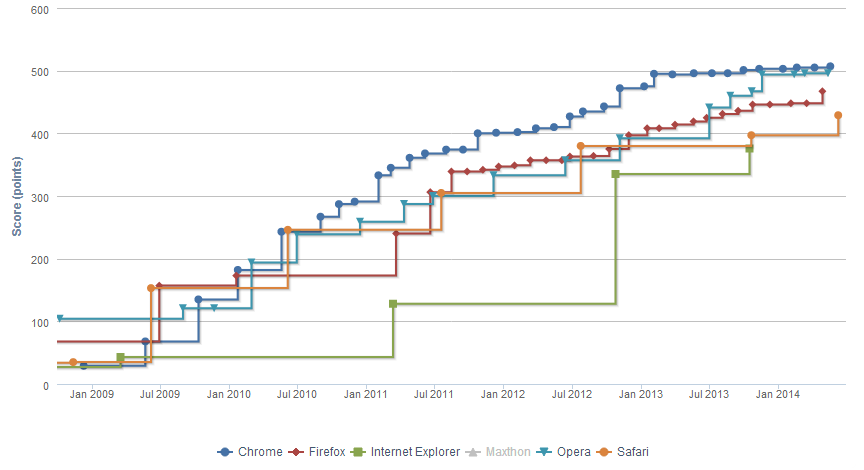
2.1.6. Compatibilitatea

Un browser (navigator) este o aplicatie software ce permite utilizatorilor sa afiseze text, grafica, video, muzica sau alte informatii pe o pagina specifica WWW, dar si sa comunice cu furnizorul de informatii sau invers.

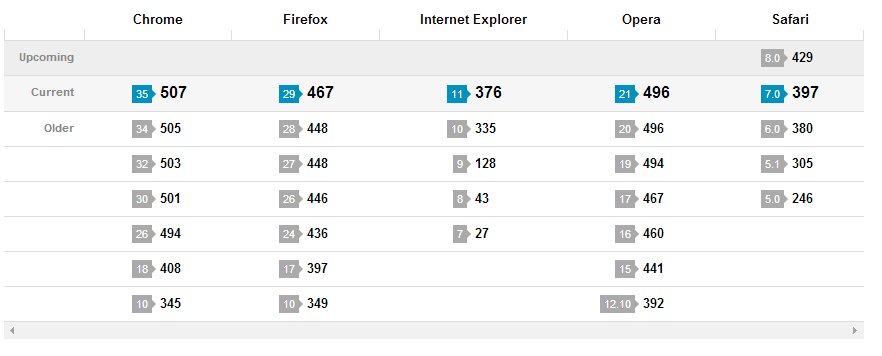
Unele dintre cele mai cunoscute aplicații *browser* sunt „Microsoft Internet Explorer”, „MozillaFirefox”, „Apple Safari”, „Google Chrome”, „Mozilla Camino”, „Opera Software – Opera”, „NintendoDS browser” și „Flock”.

Desi HTML5 ofera multiple solutii la probleme vechi, ofera posibilitatea de a construi aplicatii web complexe, ajuta dezvoltatorul sa corecteze mici probleme sintactice, acestia trebuie sa aiba grija in ce masura folosesc acest limbaj deoarece pot exista probleme de compatibilitate. Chiar si cele mai noi browsere nu reusesc sa interpreteze corect toate functionalitatile oferite de acest limbaj. Multe versiuni vechi de browsere sunt inca folosite, iar de acceea dezvoltatorii trebuie sa isi ia o masura de precautie in momentul in care scriu cod folosind HTML5.

Conform unei statistici, aceasta este evolutia browserelor cu privire la compatibilitatea cu HTML5 din ultimii 5 ani. (http://html5test.com/results/desktop.html).



Scorul fiecarei versiuni de browser a fost calculat in functie de semantica (reguli de parsare, elemente, formulare, microdata), multimedia (video, audio, peer-to-peer), elemente grafice/efecte (grafica 2D, grafica 3D si animatii), accesul pe dispozitive (localizare si orientare, date de intrare si de iesire), conectivitate (comunicarea), stocarea datelor (aplicatii web, sistem de cacheing, fisiere), integrare si performante (interactiunea userilor, performante, securitate, istoric si navigare) si altele (scripturi si detalii legate de partea grafica)

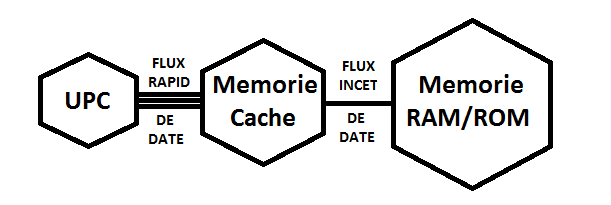


Avem mereu nevoie să fie atenți atunci când facem o alegere din punct de vedere tehnologic. Avem la dispozitie toate aceste instrumente de mare folos pe care le putem utiliza astăzi, dar asta nu înseamnă neaparat că ar trebui.

3.1 Cache

3.1.1. Despre cache

Memoria cache (sau simplu un cache), este o colectie de date ce reprezinta o copie a valorilor originale stocate altundeva sau calculate la un anumit moment, unde operatia de aducere din memorie a datelor este costisitoare (din cauza timpilor mari de acces la memorie) sau costul remanipularii lor este mare in comparatie cu costul citirii acestora din memoria cache.



De asemenea, memoria cache mai este numita si RAM cache, deoarece aceasta este o memorie de tip static RAM. Fiind de tip SRAM, viteza si costul sunt mai ridicate decat la RAM-ul dinamic (DRAM). S-a observat ca prin pastrarea a cat mai multe informatii pe SRAM, sistemul functioneaza mai rapid avand in vedere faptul ca programele acceseaza memorile de date sau instructiuni in repetate randuri.

3.1.2. Web cache

Web cache-ul este un mecanism pentru stocarea temporară (caching) de documente web, cum ar fi pagini HTML și imagini, pentru a reduce utilizarea lățimii de bandă, sarcina serverului ~~~~~~~~~~~~~~~, și lag percepute. Un cache-ul stochează copii ale documentelor care trec prin ea; cererile ulterioare pot fi satisfăcute din cache-ul în cazul în care sunt îndeplinite anumite condiții. [1] Google link cache în rezultatele de căutare oferă o modalitate de accesare a informațiilor de la site-uri care au plecat recent în jos și o modalitate de recuperare de date mai rapid decât făcând clic directe link-ul.

3.1 Javascript

JavaScript (prescurtat JS) este un limbaj de programare dinamic. Acesta este cel mai frecvent utilizat ca parte din browsere web, a căror implementare permite scripturilor client-side sa interacționeze cu utilizatorul, de a comunica asincron, și modifica conținutul documentului care este afișat. De asemenea, este utilizat in programarea server-side de rețea (cu Node.js), dezvoltarea de jocuri și crearea aplicațiilor mobile.

JavaScript este un prototip bazat pe un limbaj de scripting cu tastare dinamica și are funcții de primă clasă. Sintaxa sa a fost influențată de limbajul de programare C. JavaScript imprumuta multe nume și convenții de denumire din Java, dar cele două limbaje sunt diferite si nu au legătură. Este un limbaj multi-paradigma, suporta programarea orientat-obiect, imperativa și funcționala.

Aplicatia de JavaScript, în afara paginilor web, mai este utilizat si în documentele PDF, browsere specifice site-ului, și widget-uri desktop. Fiind construite masini virtualesi platforme pe JavaScript (în special Node.js), popularitatea acestui limbaj de programare a crescut considerabil pentru partea de aplicatii web server-side. Pe partea de client, JavaScript a fost pus în aplicare în mod tradițional ca un limbaj interpretat, dar compilarea just-in-time este acum realizată de browserele recente (după anul 2012).

JavaScript a fost formalizat în standardul ECMAScript și este utilizat în principal ca parte a unui browser web (client-side JavaScript). Acest lucru permite accesul programatic la obiecte într-un mediu gazdă.

JavaScript susține o mare parte din sintaxa de programare structurata de la C (de exemplu clauzele sau buclele cum ar fi „if”, „while”, „switch”, etc). O diferență sintactica de la C este de inserare automată a punctului și virgulei, care va termina în mod normal declarațiile să fie omise.

Ca in majoritatea limbajelor de scripting, tipurile de date sunt asociate cu valori, nu cu variabile. De exemplu, o variabila „x” poate fi interpretata ca si un numar, apoi ca si string. Javascript suporta numeroase modalitati de a testa tipul unui obiect, chiar si „duck typing”.

In programarea orientata-obiect, „duck typing” este un stil de dactilografiere în care metodele și proprietățile unui obiect determina o semantica valida, mai degrabă decât moștenirea de la o anumită clasă sau punerea în aplicare a unei interfețe. Numele conceptului se referă la testul rață, atribuit lui James Whitcomb Riley care poate fi formulat după cum urmează:

„Când văd o pasăre care merge ca o rață și înoată ca o rață și face ca rața, eu numesc asta o pasăre rață.” (<http://www.brainyquote.com/quotes/quotes/j/jameswhitc119818.html>)

În acest tip de dactilografiere, un programator este preocupat doar de asigurarea că obiectele se comporta cum ar trebui într-un context dat, mai degrabă decât asigurându-se că acestea sunt de un anumit tip.

JavaScript este aproape în întregime construit pe bază de obiect. Obiectele sunt tablouri asociative, aprofundate cu prototipuri. Numele de proprietatilor obiectelor sunt chei formate din stringuri. Acestea susțin două sintaxe echivalente: notația punct (obj.x = 10) și notația suport (obj ['x'] = 10). Proprietățile și valorile lor pot fi adăugate, modificate sau șterse în run-time. Cele mai multe proprietăți ale unui obiect (si cele de pe lanțul de moștenire prototip) pot fi enumerate folosind un o bucla „for”. JavaScript are un număr mic de obiecte built-in, cum ar fi „Function” si „Date”.

Acest limbaj include si o funcție de „eval” care poate executa declarațiile furnizate ca siruri de caractere in timpul executiei.

DHTML ( Dynamic Hyper Text Markup Language) permite limbajelor de scripting sa schimbe variabile în limbajul de definire al unei pagini web, care la rândul său, afectează aspectul și funcționalitatile paginii HTML "statice". Astfel, caracteristica dinamică a DHTML este modul în care aceasta funcționează în timp ce o pagină este vizualizata, nu în capacitatea sa de a genera o pagină unică cu fiecare încărcare a paginii.

Funcțiile sunt de primă clasă, ele sunt defapt obiecte. Ca atare, ele au proprietăți și metode, cum ar fi „call()” si „bind()”. O funcție imbricată este o funcție definită în altă funcție. Acesta este creata de fiecare dată când funcția exterior este invocată. În plus, fiecare funcție formează o închidere lexicala: domeniul de aplicare lexical al funcției din exterior, inclusiv constante, variabile locale și valorile de argument, devin o parte din starea internă a fiecărui obiect interior, chiar și după executarea funcției din exterior.

JavaScript foloseste prototipuri unde multe alte limbi orientate obiect folosesc clase de moștenire. Este posibil a simula mai multe caracteristici specifice unei clase cu ajutorul prototipurilor în JS.

Impreuna cu rolul lor tipic, functiile mai joaca si rolul de constructori. Prefixarea un apel de funcție cu „new” va crea o instanță a unui prototip, moștenind proprietăți și metode de la constructor (inclusiv proprietăți de prototip Object). ECMAScript 5 oferă metoda „Object.create”, care permite crearea în mod explicit a unei instanțe fără a moșteni automat de la prototipul „Object” (mediile de lucru mai vechi pot atribui prototipului valoarea null). Proprietatea constructorului „prototype” determină obiectul folosit pentru prototipul nou al obiectului. Noi metode pot fi adăugate prin modificarea prototipului obiectului folosit ca un constructor. Constructorii predefiniti din JavaScript, cum ar fi „Array” sau „Object”, de asemenea, au prototipuri care pot fi modificati. Deși este posibil să se modifice prototipul „Object”, nu se recomanda deoarece majoritatea obiectelor în JavaScript vor moșteni metode și proprietăți de prototip din „Object” și nu se pot aștepta ca prototipul sa fie modificat.

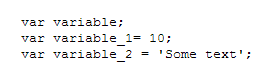
Spre deosebire de majoritatea limbajelor de programare orientate-obiect, nu se face o distinctie intre definitia unei functii si definitia unei metode. Putem spune mai degraba ca distinctia are loc in timpul apelului de functie; cand o functie este chemata ca si metoda a unui obiect.

JavaScript suportă nativ diverse implementari de funcții pe baza sabloanelor de tip „Role”, cum ar fi „Traits” și „Mixins”. O astfel de funcție definește comportamentul suplimentar pentru cel puțin o metodă legată de acest cuvânt cheie în corpul funcției sale . Un astfel de „Role” trebuie apoi să fie delegat în mod explicit prin „call” sau „apply” la obiectele care au nevoie de caracteristici suplimentare care nu sunt împărtășite prin intermediul prototipelor.

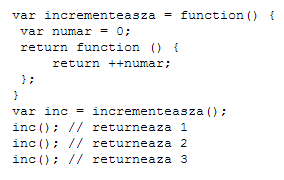
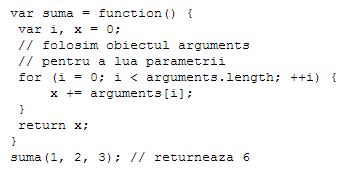
Limbajul se bazează de obicei pe un mediu run-time (de exemplu, un browser web) pentru a oferi obiecte și metode prin care script-urile pot interacționa cu mediul (de exemplu, o pagina de web). De asemenea, se bazează pe mediul de run-time pentru a oferi posibilitatea de a include / script-uri de import (cum ar fi elemente HTML <script>). Aceasta nu este o caracteristică a limbajului în sine, dar ea este comună în cele mai multe implementări JavaScript.

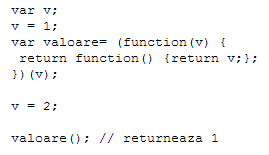
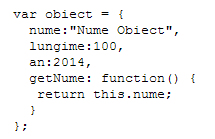
Un număr nedeterminat de parametri poate fi trecut la o funcție. Funcțiile pot fi accesate prin parametri formali și, de asemenea, prin obiectul „arguments”. Funcțiile „Variadic” pot fi create prin utilizarea metodei „apply”. La fel ca multe limbaje de scripting, tablourile și obiectele (tablouri asociative din alte limbaje), pot fi create cu o sintaxa de comenzi rapide. Defapt, acesti literali formează baza formatul de date JSON. ~~~~~~~~~~~~~~add json chapter~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

JavaScript suporta de asemenea si expresii regulare intr-o maniera similara lui Perl (un limbaj de programare dinamic, procedural care împrumută caracteristicile limbajelor C, shell, AWK,sed, Lisp și ale multor alte limbaje într-o măsură mai mică), ce ofera o sintaxa puternica si concisa pentru manipularea textului care este mai sofisticat decat functiile de stringuri predefinite.

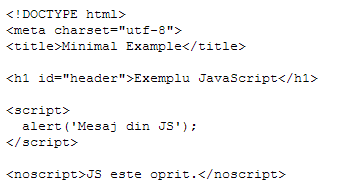
 Variabilele se pot defini folosind keyword-ul „var”. In javascript, variabilele nu necesita specificarea tipului, valoarea lor stabilind acest lucru (lucru specific in limbajele de scriptare).

O functie este defapt un bloc de cod cu scopul de a indeplini o anumita sarcina. Pentru ca aceasta functie sa fie executata, ea trebuie sa fie invocata (sa fie apelata). In JavaScript avem mai multe tipuri de functii: simple, anonime, „Variadic”, invocate-imediat.



În JavaScript, obiectele sunt date (variabile), cu proprietăți și metode. Aproape totul în acest limbaj poate fi un obiect: Siruri de caractere, Date, tablouri, funcții etc. De asemenea, se pot defini propriile obiecte cu proprietati si/sau metode.



Exemplu de cod JavaScript introdus intr-o pagina HTML. Se foloseste tagul de <script> pentru a introduce codul de Javascript, iar tagul de <noscript> este folosit in cazul in care JavaScript este oprit sau nu este compatibil pe browserul respectiv.

3.1.2 JSON

JSON sau JavaScript Object Notation, este un format standard deschis (public) care utilizează text lizibil pentru a transmite obiecte de date formate din perechi atribut-valoare. Acesta este utilizat în principal pentru a transmite date între un server și aplicații web, ca alternativă la XML.

Deși inițial derivat din limbajul de scripting JavaScript, JSON este un format de date independent de limbă, iar atat codul de analiză cat și generarea de date JSON sunt disponibile intr-o mare varietate de limbaje de programare.

Formatul JSON a fost inițial specificat de Douglas Crockford. Acesta este descris în prezent prin două standarde concurente, RFC 7159 și ECMA-404. Standardul ECMA este minim, descrie doar sintaxa gramatică permis, în timp ce RFC oferă, de asemenea, o serie de considerații semantice și de securitate. Tipul media oficial pentru JSON este application/JSON. Extensia numelui fișierului este „.json” .

Din moment ce JSON a fost derivat din JavaScript și sintaxa lui este un subset al limbajului, este adesea posibil să se utilizeze functia „eval” de JavaScript, pentru a analiza datele JSON cu toate ca acest lucru este considerat nesigur.

3.2.1 ce este un server

**Server**

La fel ca și minicomputerele în trecut (care au fost inlocuite), serverele ocupă un loc deosebit de important în tehnologia informaticii. Un server este o aplicație pe computer sau chiar un computer întreg, care ar trebui sa opereze continuu în rețeaua sa și așteaptă solicitări din partea altor calculatoare aflate in aceeasi rețea. Serverele pot servi si pentru alte scopuri, dar de cele mai multe ori, ele sunt rezervate exclusiv pentru funcția de server.

De exemplu, sa presupunem ca folosim un calculator (in cadrul unui birou) ca si stație de lucru, dar ca si server pentru celelalte calculatoare aflate in incapere. Astfel, un calculator poate indeplini mai multe functii.

Cuvântul *server* provine din cuvântul englez *to serve* – a servi: un server serveste in gerenal intreaga retea, pentru a asigura accesul la toată paleta de forme de conectare și servicii. De cele mai multe ori, un calculator joaca ambele roluri de server, și de client, în același timp. De asemenea, numele de *server* este un alt termen pentru *Host computer* – computer gazdă.

Pentru a avea o functionalitate de server, calculatoarele folosite cu acest scop au configuratiile hardware deseori optimizate pentru o functionare cat mai buna. Majoritatea componentelor hardware sunt identice cu cele dintr-un calculator personal, dar sistemele de operare difera de cele mai multe ori, iar accentul se pune pe partile de care are cea mai mare nevoie serverul respectiv.

3.2.2 servere web

**Serverul web** sau **web serverul** este serverul care stochează (găzduiește) pagini web și le pune la dispoziția solicitanților prin protocolul HTTP. Relația server-client se bazează pe o aplicație care este instalată pe server și care este programată să transfere paginile web găzduite. Putem observa aici că ideea de web server presupune și noțiunea de hosting (găzduire), deoarece serverul trebuie să dețină datele pe care urmează să le returneze la cerere.

Orice calculator poate fi transformat într-un server web prin instalarea software de tip server și conectare la Internet. Există mai multe aplicații software de tip web server, inclusiv software-ul de domeniu public de la NCSA și Apache, și pachete comerciale de la Microsoft, Netscape și altele.

De obicei, utilizatorii vizitează un site Web fie prin click pe un hyperlink care le aduce la site-ul sau tastând URL-ul site-ului direct în bara de adrese a unui browser. Dar cum apare în același site-ul de pe computerul oricui oriunde în lume și de multe ori pe multe calculatoare în același timp?

Relația este următoarea: utilizatorul (clientul) aflat în dreptul unui computer pe care are instalată o aplicație tip browser solicită (serverului) prin intermediul unui url o anumită pagină web; serverul rulează anumite linii de cod și returnează un rezultat. Acesta a fost cazul site-urilor (paginilor web) statice (adică cele bazate în exclusivitate doar pe limbajul de programare html și css). Există însă așa numitele site-uri dinamice care au în compunere alături de limbajul rudimentar de afișare a paginii web și un limbaj de comunicare între serverul web și o bază de date. Pentru acest caz relația este următoarea: utilizatorul (clientul) aflat în dreptul unui computer pe care are instalată o aplicație tip browser solicită (serverului) prin intermediul unui url o anumita pagină web; serverul web verifică solicitarea și prin intermediul unui limbaj de programare special interoghează o bază de date, dacă anumite condiții sunt intrunite, baza de date returnează serverului web datele solicitate care la rândul lui furnizează datele mai departe clientului inițial. În această ultimă relație între web server și utilizator sunt transmise doar informații destinate afișării în browser într-o formă prietenoasă a informațiilor solicitate. Până să ajungă la utilizator, așa cum am văzut mai sus, serverul web prin intermediul unui scripting special schimbă o serie de informații cu o bază de date stocată.

3.3.1 **HTTP**

**HTTP** (**H**yper**t**ext **T**ransfer **P**rotocol) este metoda cea mai des utilizată pentru accesarea informațiilor în Internet care sunt păstrate pe servere World Wide Web (WWW). Protocolul HTTP este un protocol de tip text, fiind protocolul "implicit" al WWW. Dacă un URL nu conține partea de protocol, aceasta se consideră ca fiind http.

HTTP presupune că pe calculatorul destinație rulează un program care înțelege protocolul. In general, fișierul trimis la destinație este un document HTML, sau un fișier grafic, de sunet, animație sau video, un program executabil pe server-ul respectiv sau și un editor de text.

Acest protocol oferă o tehnică de comunicare prin care paginile web se pot transmite de la un computer aflat la distanță spre propriul computer. Dacă se apelează un link sau o adresă de, atunci se cere calculatorului host să afișeze o pagină web. În prima fază numele (adresa) este convertit de protocolul DNS într-o adresă IP. Urmează transferul prin protocolul TCP pe portul standard 80 al serverului HTTP, ca răspuns la cererea HTTP-GET.

Informațiile suplimentare, cum ar fi indicații pentru browser, limba dorită sau altele, se pot adăuga în header-ul (antetul) pachetului HTTP. În urma cererii HTTP-GET urmează din partea serverului răspunsul cu datele cerute. Dacă dintr-un anumit motiv informațiile nu pot fi transmise, atunci serverul trimite înapoi un mesaj de eroare. Modul exact de desfășurare a acestei acțiuni (cerere și răspuns) este stabilit în specificațiile HTTP.

2.3.4 Apache

**Apache** este un server HTTP de tip open source. Apache joacă un rol deosebit de important în dezvoltarea webului, fiind folosit în prezent în circa 65.2 % din paginile web.

Ca si alternativă viabilă la Netscape Communications Corporation, Apache a evoluat rapid în funcționalitate și performanță ca un rival competitiv pentru alte servere web bazate pe Unix. Apache este dezvoltat de o comunitate deschisă de programatori sub emblema Apache Software Foundation.

Acest serviciu este disponibil pentru o mare varietate de sisteme de operare incluzând Unix, FreeBSD, Linux, Solaris, Novell NetWare, Mac OS X, Microsoft Windows si OS/2.

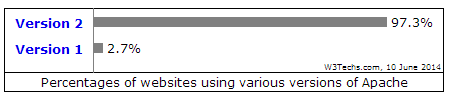
Incepand din luna aprilie 1996, serverul Apache a devenit cel mai popular server HTTP, fiind totodata un software gratuit si open source. Cu toate că în noiembrie 2005 a început să piardă din cota de piață, în aprilie 2008 Apache stătea încă la baza a peste 50 % din siturile web, iar în iunie 2013 a ajuns la 60.5%. ~~~~~~~~~~~~~~ <http://w3techs.com/> ~~~~~~~~~~~~~~~~

Apache suportă o varietate de caracteristici, multe implementate ca module compilate care îi extind funcționalitatea. Acestea pot varia de la limbaje de programare server-side la sisteme de autentificare. Câteva limbaje suportate sunt: mod\_perl, mod\_python, Tcl si PHP. Ca alte module putem enumera : SSL si TLS support (mod\_ssl), un modul *proxyun*, modul de rescriere URL (cunoscut ca un motor de rescriere mod\_rewrite), *custom log files*(mod\_log\_config) și suport de filtrare (mod\_include și mod\_ext\_filter). O altă calitate a serverului Apache este *virtual hosting* (găzduirea virtuală), care constă în posibilitatea de a găzdui mai multe situri simultan pe același server.

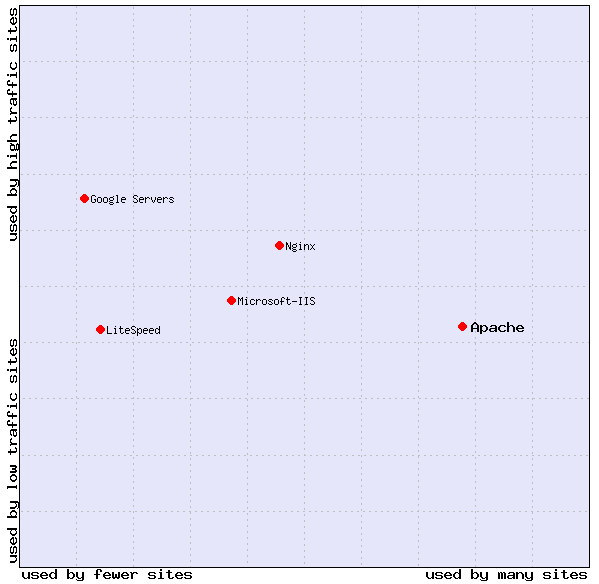
Apache este folosit de unele din cele mai mari situri din lume, chiar si motorul de căutare folosit de Google folosind o versiune modificată de Apache numită Google Web Server (GWS). De asemenea proiectele Wikimedia, inclusiv Wikipedia rulează tot pe un server Apache.

Serverul Apache HTTP a fost relicentiat la 2,0 Apache (de la licența anterioara 1.1) în ianuarie 2004 și Apache HTTP Server 1.3.31 și 2.0.49 au fost primele versiuni utilizând noua licență.

Conform unor statistici, Apache este folosit pentru 60.5% dintre toate websiturile, iar dintre acestea, 97.3% folosesc a doua versiune.



Ca si popularitate, Apache se claseaza pe primul loc, fiind cel mai popular webserver din prezent.



2.4 Python

2.4.1 Despre Python

Python este un limbaj de programare orientat-obiect clar și puternic, comparabil cu Perl, Ruby, Scheme, sau Java.

Ca si limbaj de programare, cele mai notabile functionalitati ale Python-ului ar fi:

- sintaxa eleganta, ce ofera posibilitatea de a scrie secvente de cod usor de citit.

- Este un limbaj ușor de utilizat, care face simpla scrierea unui program. Acest lucru face Python ideal pentru prototype developement și alte sarcini de programare ad-hoc, fără a compromite mentenabilitatea.

- Vine cu o librarie mare, standard, care acceptă mai multe sarcini de programare comune, cum ar fi conectarea la servere de web, căutarea de text cu expresii regulate, citirea și modificarea de fișiere.

- Modul interactiv Python îl face ușor pentru a testa fragmente scurte de cod. Există, de asemenea, un environement pentru developement numit IDLE.

- Se extinde ușor prin adăugarea unor noi module implementate într-un limbaj compilat, cum ar fi C sau C + +.

- Poate fi, de asemenea, încorporat într-o aplicație pentru a oferi o interfață programabilă.

- Ruleaza pe mai multe calculatoare diferite și sisteme de operare cum ar fi Windows, MacOS, mai multe versiuni de Unix, OS / 2, etc.

- Este un software public, în două sensuri. El nu costa nimic pentru a fi descărca sau pentru a fi folosit, sau pentru a fi inclus într-o aplicatie. Python poate fi, de asemenea, modificat în mod liber și re-distribuit, deoarece limbajul este protejat de copyright si este disponibil sub o licență open source.

- O varietate de tipuri de date de bază sunt disponibile: numere (în virgulă mobilă, complexe, numere de lungime nelimintata), siruri de caractere (atât ASCII și Unicode), liste și dicționare.

- Python suportă programarea orientat-obiect, cu clase și moștenire multiplă.

- Codul poate fi grupat în module și pachete.

- Limbajul susține creșterea și prinderea excepțiilor, rezultand un mod mai bun de a controla erorile.

- Tipurile de date sunt puternice și dinamice.Amestecarea tipurilor incompatibile (de exemplu, încercarea de a aduna un șir de caractere și un număr) determină o excepție, astfel încât erorile sunt prinse devreme.

- Python conține caracteristici avansate de programare, cum ar fi generatoare si liste imbricate.

- Managementul automat de memorie al Python-ului ajuta la renuntarea alocarii manuale și dealocarii acesteia în cod.

Un număr mare de platforme Web sunt construite cu Python. Abilitățile limbajului ca si limbaj pentru programarea CGI sunt în afara oricăror dubii. Un exemplu bun ar fi  YouTube, unul din site-urile cu cel mai mare trafic din lume, este construit pe baza limbajului Python.

Python permite extinderea funcționalității prin pachete adiționale programate de terți care sunt axate pe o anumită funcționalitate, cum ar fi pachetul *wxPython* care conține metodele și structurile necesare creării unei interfețe grafice.

Limbajul Python, ca si popularitate, este în creștere începînd cu anul 2000, datorită faptului că permite crearea mai rapidă a aplicațiilor care nu cer viteze înalte de procesare a datelor. Cu toate acestea, el este util si ca limbaj de scriptare, utilizat în cadrul aplicațiilor scrise în alte limbaje. Modulele Python pot fi de asemenea scrise în *C*, compilate și importate în Python pentru a mări viteza de procesare.