UNIVERZITET SINGIDUNUM

Departman za
poslediplomske studije

SAVREMENE INFORMACIONE TEHNOLOGIJE
MASTER STUDIJSKI PROGRAM

MASTER RAD

FIREFOX OS

- budućnost web, sistemski specifičnih i hibridnih aplikacija -

Mentor:
Prof. dr Ranko Popović

Kandidat:
Nikola Vuković
br. indeksa:
2013/410246

Beograd, 2014.

SADRŽAJ

1 Uvod ..... - 4 -

2 Mobilne web aplikacije i sistemski specifične aplikacije ..... - 7 -

2.1 Sistemski specifične aplikacije ..... - 7 -

2.2 Mobilne web aplikacije ..... - 7 -

2.3 Hibridne aplikacije ..... - 8 -

2.4 Prednosti i mane mobilnih web i sistemski specifičnih aplikacija ..... - 8 -

2.4.1 Bogatstvo mogućnostima ..... - 8 -

2.4.2 Performanse ..... - 10 -

2.4.3 Iskustva razvojnih timova ..... - 12 -

2.4.4 Izgled i način korišćenja aplikacije (*look and feel*) ..... - 13 -

2.4.5 Dostupnost ..... - 13 -

2.4.6 Monetizacija ..... - 14 -

2.5 Statistike razvoja u odnosu na izabrane tehnologije i platforme ..... - 15 -

3 Web tehnologije u industriji video igara i edukaciji ..... - 19 -

3.1 OpenGL, Open GL ES i WebGL ..... - 19 -

3.2 3D modelovanje uz pomoć web tehnologija ..... - 20 -

3.3 Performanse ..... - 20 -

3.4 Potencijalna upotreba WebGL-a ..... - 21 -

3.5 WebGL u oblasti video igara ..... - 23 -

3.5.1 Primeri WebGL video igara ..... - 23 -

3.6 WebGL u oblasti edukacije ..... - 25 -

3.6.1 Primeri WebGL edukativnih aplikacija ..... - 25 -

4 Potreba za otvorenom Web platformom i uloga Mozilla-e ..... - 27 -

4.1 Istorija Mozille ..... - 28 -

4.2 Softver ..... - 29 -

4.2.1 Softverske komponente ..... - 29 -

4.3 Druge Mozilla aktivnosti ..... - 30 -

4.4 Zajednica ..... - 31 -

5 Firefox OS ..... - 32 -

5.1 Istorija Firefox OS-a ..... - 33 -

5.2 Hardver uređaja za Firefox OS ..... - 33 -

6 Arhitektura platforme ..... - 35 -

6.1 Gonk ..... - 36 -

6.2 Gecko ..... - 36 -

6.2.1 Kritike i prihvaćenost Gecko layout engine-a ..... - 39 -

6.3 Gaia ..... - 39 -

7 WebAPI ..... - 42 -

7.1 WebTelephony ..... - 44 -

7.1.1 Implementacije WebTelephony standarda ..... - 44 -

7.2 WebSMS ..... - 45 -

7.3 Battery Status API ..... - 46 -

7.3.1 BatteryManager.level ..... - 46 -

8 Mozilla Web aplikacije za Firefox OS ..... - 47 -

8.1 Sigurna implementacija sistema i ažuriranje sistema i aplikacija ..... - 49 -

8.2 Bezbednost aplikacija ..... - 50 -

9 Razvoj aplikacija za Firefox OS ..... - 51 -

9.1 Web aplikacija ..... - 51 -

9.2 Manifest aplikacije ..... - 53 -

9.3 Firefox OS simulator ..... - 54 -

9.3.1 Pokretanje simulatora ..... - 55 -

9.3.2 Dodavanje alatki za razvoj ..... - 55 -

9.3.3 Prebacivanje aplikacije na uređaj ..... - 56 -

9.4 Ograničenja Simulatora ..... - 56 -

9.5 Revizija aplikacije ..... - 57 -

10 Boilerplate aplikacija ..... - 57 -

10.1 WebActivities ..... - 58 -

10.1.1 Pick image ..... - 58 -

10.1.2 Pick anything ..... - 58 -

10.1.3 Take picture ..... - 59 -
10.1.4 Dial number ..... - 59 -
10.1.5 Send SMS ..... - 60 -
10.1.6 Add contact ..... - 60 -
10.1.7 Share URL ..... - 61 -
10.1.8 Share image ..... - 61 -
10.1.9 View URL ..... - 62 -
10.1.10 Compose mail ..... - 62 -
10.1.11 Save bookmark ..... - 63 -
10.1.12 Open video ..... - 63 -
10.2 WebAPIs ..... - 64 -
10.2.1 Add notification ..... - 64 -
10.2.2 Orientation lock ..... - 65 -
10.2.3 Vibrate 2 sec ..... - 66 -
10.2.4 Check connection ..... - 66 -
10.2.5 Check battery ..... - 67 -
10.2.6 Geolocation ..... - 67 -
10.2.7 Ambient light ..... - 68 -
10.2.8 Proximity ..... - 68 -
10.2.9 Device orientation ..... - 69 -
10.2.10 Keep screen on ..... - 69 -
11 Budućnost Firefox OS-a ..... - 70 -
12 Zaključak ..... - 71 -
Literatura ..... - 72 -

1 Uvod

Predmet istraživanja

Ovim radom su obuhvaćeni pojašnjenja, razlike, prednosti i mane različitih tipova aplikacija za mobilne uređaje. Reč je sistemski specifičnim, web i hibridnim aplikacijama. U drugom delu rada su obuhvaćeni razvoj, struktura i upotreba novog operativnog sistema za mobilne platforme. Reč je Firefox OS operativnom sistemu, projektu organizacije Mozilla. Ovaj sistem je potpuno baziran na najnovijim Web tehnologijama, kao što su HTML5, CSS3 i JavaScript.

Cilj istraživanja

Cilj istraživanja u ovom radu je da se pokažu sve prednosti i mane određenih pristupa pri razvoju aplikacija za mobilne platforme i odgovarajuće uređaje.

U prvom delu rada, data su pojašnjenja svake od ove tri vrste aplikacija, zatim se detaljno obrađuju prednosti i mane svakog pristupa kroz aspekte:

- broja mogućnosti koje takav pristup donosi;
- performansi;
- iskustva razvojnih timova;
- izgleda i načina korišćenja;
- dostupnosti;
- monetizacije.

U drugom delu rada, dat je pregled mogućnosti web aplikacija na polju edukacije, kao i u industriji video igara. Detaljno se objašnjavaju tehnologije OpenGL, OpenGL ES i WebGL, zatim se daju prikazi mogućnosti 3D modelovanja uz pomoć web tehnologija, performanse takvih aplikacija. U narednom odeljku ovog dela rada dat je prikaz potencijalne upotrebe aplikacija razvijanih uz pomoć WebGL tehnologije.

Kao poseban odeljak izdvojen je deo sa primerima video igara razvijenih potpuno uz pomoć web tehnologija i web aplikacija u oblasti edukacije.

U trećem delu rada, obrazloženi su razlozi za upotrebu otvorenih Web standarda i naprednih Web tehnologija u razvoju softvera za mobilne platforme i ulozi Mozilla organizacije. Takođe, napravljen je i osvrt na strukturu Mozilla organizacije, zatim na istoriju i promene koje je organizacija doživela. Zatim se ukratko predstavljaju softveri i softverske komponente koji su produkti Mozilla-e, kao i druge aktivnosti organizacije.

U četvrtom delu rada, dat je kratak uvod i pojašnjenje Firefox OS-a sa posebnim osvrtom na pogodnosti koje sa sobom donosi. Ovaj deo rada je podeljen na dve celine:

- Istorija Firefox operativnog sistema, koja je započela sredinom 2011. godine;
- Specifikacije i karakteristike hardverske podloge potrebne za pokretanje ovog sistem.

U petom delu rada, detaljno je opisana struktura samog sistema koja se sastoji iz tri dela:

- Gonk (najniži sloj koji se sastoji od Linux jezgra i komponenata koje sistemu omogućavaju pristup hardveru);
- Gecko (srednji sloj, komponenta koju Mozilla koristi u mnogim svojim aplikacijama, a služi sa prikazivanje sadržaja koji se generiše iz HTMLa, XMLa, CSS, JSa, slika itd.);
- Gaia (najviši sloj, korisnički interfejs kojim korisnik upravlja uređajem, potpuno otvorenog koda i pogodan za modifikaciju).

U šestom delu rada, dat je prikaz WebAPI-ja, komponenti koje omogućavaju ono što pre Firefox OS-a nije bilo moguće, a to je pristup hardverskim resursima uz pomoć neke od Web tehnologijama. Dat je pregled nekih od najbitnijih WebAPI-ja koje je Mozilla kreirala, kao što su:

- WebTelephony;
- WebSMS;
- Battery Status API.

U sedmom delu rada, prikazane su sistemske aplikacije koje dolaze predefinisane uz sistem i koje se mogu očekivati od pametnog telefona. To su, primera radi, aplikacije za pozivanje, slanje poruka, pretragu Interneta, slanje elektronske pošte, fotografisanje, navigaciju itd. Zatim se daje objašnjenje različitih nivoa privilegija koje aplikacijama mogu biti dozvoljene, a to su:

- Sertifikovana (sistemska aplikacija sa visokim nivoom poverenja);
- Privilegovana (aplikacija sa određenim restrikcijama, ali i dozvolama, smatra se pouzdanom);
- Web aplikacija (aplikacija sa gotovo nikakvim privilegijama, smatra se nepouzdanom).

U osmom delu rada, detaljno je opisan proces razvoja aplikacija za Firefox OS i ukazano je na pogodnosti koje ovaj proces donosi. Razvoj aplikacija za ovaj sistem se ne razlikuje od razvoja Web stranice i Web aplikacije, što znači da na svetu postoji više developera koji su sposobni da naprave aplikaciju za ovaj sistem, nego za sve druge sisteme zajedno. To je ujedno i najveća prednost Firefox OS-a u odnosu na konkurenciju.

U devetom delu rada, dat je prikaz Boilerplate aplikacije koja prikazuje mogućnosti Firefox OS platforme.

U desetom delu rada, ukratko je iskazano razmišljanje o budućnosti različitih pristupa razvoju aplikacija prikazanih u ovom radu.

U jedanaestom delu rada, dat je zaključak na osnovu svih materijala i iskaza iz prethodnih delova rada.

Hipoteza istraživanja

Web kao platforma uz pomoć otvorenih standarda nudi lakši razvoj i distribuciju aplikacija, a sa druge strane omogućava korisnicima isti doživljaj bez obzira na uređaj ili sistem putem kojih pristupaju aplikaciji.

Firefox OS

Sažetak: U ovom radu obrađena je tema budućnosti i daljeg razvoja aplikacija za mobilne uređaje. Obrađene su tri vrste aplikacija: sistemski specifične (*native*), web i hibridne aplikacije, koje su kombinacija prethodne dve. U korist svake od njih postoji mnoštvo dobrih argumenata, ali svaka ima i svoje mane. Sistemski specifične imaju prednost bržeg izvršavanja i direktan pristup hardverskim resursima uređaja, ali sa druge strane donose dosta kompleksniji proces razvoja, kao i veće troškove. Web, kao pozitivne strane imaju prost proces nastanka, jer su razvojni timovi upoznati sa web tehnologijama, zatim manju hardversku zahtevnost i cenu. Najvažnija karakteristika web aplikacija je mogućnost pokretanja na više platformi bez prepravljanja izvornog koda. Hibridne aplikacije igraju ulogu međusloja, jer one predstavljaju kombinaciju prethodne dve i uzimaju najbolje od svake. U drugom delu rada obrađena je tema Mozilla Firefox operativni sistem (Firefox OS). Iako je to sistem u nastajanju, iza njega stoje ideja i motivacija koje bi mogle da preokrenu tok kojim se Web trenutno kreće i otvore ga ka mogućnostima koje mnogi još uvek i ne slute. Misija koju je Mozilla zacrtala za Firefox OS je ujedinjavanje i oslobađanje Web-a, tako da svako može da uživa svoje slobode bez obzira koju platformu se odlučio da koristi. Web otvoren za sve znači brži i lakši razvoj standardizovanih aplikacija, a sve to sa ciljem da se Web konstantno kreće napred i osvaja nove granice.

Ključne reči: Firefox OS, sistemski specifične, web i hibridne aplikacije, otvoreni Web standardi, nezavisnost od platforme

Firefox OS

Abstract: This paper explores the future and further development of mobile applications. Three types of applications are taken into consideration: native, web and hybrid applications, which are the combination of the first two. There is a multitude of very good arguments for each of them, but also there are some shortcomings. Native applications have the advantage of better performance and direct access to device's hardware, but on the other hand they are more complex to develop and more costly. Web applications, on their positive side, are easier to develop, because the developers are already familiar with web technologies, they have lesser hardware requirements and are cheaper to develop. Their main advantage is the ability to work across different platforms without modifying the source code. Hybrid applications can be viewed as a middle layer, being the combination of the former two, they take all of their advantages. The second part of this thesis explores the Mozilla Firefox operating system (Firefox OS). Although the system is still emerging, the idea and the motivation behind it could turn the current course of the Web and open it up to possibilities most can't even foresee. The mission Mozilla has set for Firefox OS is uniting and liberating of the Web, so that anyone can enjoy their freedoms, no matter what platform they choose to use. The Web that is open to all means a faster and easier way to develop standardized applications, with a view to constantly push the Web forward and conquer new limits.

Keywords: Firefox OS, native, web and hybrid applications, Open Web standards, Cross platform

2 Mobilne web aplikacije i sistemski specifične aplikacije

Aplikacije za mobilne uređaje i HTML5 su trenutno dve najpopularnije tehnologije i između njih postoji mnogo preklapanja. Web aplikacije se pokreću u mobilnim pregledačima, ali takođe mogu biti i prepakovane kao sistemski specifične aplikacije za razne mobilne platforme. Kako je potrebno pokriti širok spektar platformi, i kada se u obzir uzmu mogućnosti današnjih mobilnih pregledača, razvojni timovi se opredeljuju za HTML5, jer ga smatraju rešenjem po principu "napiši jednom, pokreni na mnogo uređaja".

Da li je ova ideja zaista održiva? Još uvek je mnogo dobrih i razumnih razloga u korist sistemski specifičnih aplikacija, i zapravo, mnogi razvojni timovi su i dalje na tom putu.



slika 1.1: HTML5 i najrašireniji mobilni operativni sistemi

2.1 Sistemski specifične aplikacije

Kako im i samo ime kaže, sistemski specifične aplikacije su predviđene za rad pod određenim operativnim sistemom, odnosno na određenoj platformi. Njima se pristupa putem ikonica na glavnom ekranu (*home screen*) uređaja. Ova vrsta aplikacija se instalira putem specifične aplikacije, npr. App Store za Apple uređaje ili Google Play za Android uređaje. Razvijaju se zasebno za svaku platformu i mogu u potpunosti koristiti sve mogućnosti uređaja, kao što su kamera, GPS, akcelerometar, kompas, listu kontakata itd. Takođe, mogu inkorporirati gestove (bilo one standardne za dati sistem, bilo specifične za tu aplikaciju). Pored svega ovoga, sistemski specifične aplikacije imaju mogućnost korišćenja sistema za notifikacije, kao i rada u offline režimu. [8]

2.2 Mobilne web aplikacije

Web aplikacije nisu aplikacije u pravom smislu reči; to su pre web sajtovi koji, po mnogo čemu, izgledaju ili se ponašaju kao sistemski specifične aplikacije. S druge strane način implementacije ovih aplikacije se razlikuje. One se pokreću u pregledaču i najčešće su pisane u HTML5. Korisnici im pristupaju kao što bi pristupili bilo kojem web sajtu, tako što otvore određeni URL i tada im se pruža opcija za instalaciju na home screen uređaja, tako što se kreira prečica, odnosno oznaka (*bookmark*) koja vodi do početne stranice te aplikacije.

Ovaj tip aplikacije je postao veoma popularan sa pojavom HTML5 i realizacijom da se kroz pregledač mogu dobiti funkcionalnosti približne onima u sistemski specifičnim

aplikacijama. Danas, kada sve više i više sajtova koristi HTML5, razlika između web aplikacije i uobičajene web prezentacije je sve manje jasna.

Dobar primer za ovu tvrdnju je mobilna web aplikacija međunarodnog poslovnog lista Financial Times. Njihova web aplikaciju je, u mnogo čemu, veoma teško razlikovati od sistemski specifične aplikacije. Primera radi, ne postoje vidljivi dugmići karakteristični za pregledač, iako se ona pokreće u Safari-ju (kada se otvara sa iPhone-a). Korisnici imaju mogućnost da horizontalnim prevlačenjem pristupe novim odeljcima aplikacije. Takođe, zahvaljujući keširanju moguće je čitati novine u offline režimu.

Sve ove mogućnosti su odstupne kroz HTML5. Takođe, su dostupni GPS, tap-to-call (*mogućnost pozivanja telefonskog broja jednostavnim pritiskom*), zatim korišćenje kamere kroz Camera API itd. U početku neke od sistemski specifičnih mogućnosti nisu bile odmah dostupne. Među njima su bile: notifikacije, pokretanje u pozadini, informacije koje pruža akcelerometar (osim detekcije horizontalne ili vertikalne orijentacije uređaja), kao i kompleksni gestovi.

Naravno, mnoge aplikacije i ne koriste sve ili neke od ovih naprednih funkcionalnosti. Sa druge strane, dobro je znati da im se može pristupiti i putem web aplikacije, jer bi u suprotnom, za slučaj da je potrebno iskoristiti neku od naprednih mogućnosti uređaja, razvojni tim morao razvijati sistemski specifičnu aplikaciju ili, u najboljem slučaju, hibridnu aplikaciju. [8]

2.3 Hibridne aplikacije

Ovaj tip aplikacija predstavljaju aplikacije koje su delom sistemski specifične, delom web aplikacije (zbog čega ih mnogi pogrešno nazivaju web aplikacijama). Kao i sistemski specifične i hibridne aplikacije se preuzimaju sa Marketplace-a i mogu koristiti mnoge od mogućnosti koje su dostupne na uređaju. Kao i web aplikacije i hibridne se oslanjaju na HTML za prikazivanje sadržaja u pregledaču, uz preduslov da pregledač bude ugrađen u samu aplikaciju.

Često kompanije kreiraju hibridne aplikacije kao omotače za već postojeće web stranice. Na taj način se nadaju da će biti prisutni na Marketplace-u bez potrebe za upotrebom značajnih sredstava za razvoj druge aplikacije. Hibridne aplikacije su popularne jer omogućavaju razvoj aplikacije koja ne zavisi od platforme na kojoj se pokreće, čime se značajno smanjuju troškovi razvoja. To upravo znači da se istovetne komponente HTML koda mogu, uslovno rečeno, reciklirati za različite mobilne operativne sisteme. Alati poput PhoneGap-a i Sencha Touch-a dozvoljavaju dizajn i kodiranje nezavisno od platforme koristeći moć HTML-a. [8]

2.4 Prednosti i mane mobilnih web i sistemski specifičnih aplikacija

2.4.1 Bogatstvo mogućnostima

Funkcionalnost mobilnih uređaja se može podeliti u dve dimenzije: korišćenje i rad sa aplikacijom kao zasebnom celinom i način na koji koristi mogućnosti iz okvira ekosistema uređaja, npr. Android operativnog sistema, kao što su widget-i i notifikacije. Sistemski specifične aplikacije su izvanredne u obe dimenzije.

Što se tiče doživljaja korišćenja aplikacije, sistemski specifične korisniku nude više. Sa lakoćom se mogu koristiti događaji prevlačenja preko ekrana, istovremenog dodira na više tačaka (*multitouch*), na uređajima koji to podržavaju. Takođe, mogu reagovati na aktiviranje hardverskih dugmadi, kao što je Search button na Android telefonima ili dugmad za povećanje i smanjenje nivoa zvuka. Mogu pristupiti i hardveru, kao što su GPS i kamera, a uz dozvolu korisnika neke platforme obezbeđuju neometan pristup operativnom sistemu. [9]

Mada, nije sve u korišćenju samo date aplikacije. Operativni sistemi, poput Android-a obezbeđuju različite načine kako bi aplikacije stvarale interakciju sa korisnicima, ali i međusobno. Recimo, postoje widget-i na početnom ekranu uređaja, postoje notifikacije koje se prikazuju na statusnoj liniji. Takođe, aplikacije mogu imati i intent-e, koji dozvoljavaju aplikaciji da se ostalim aplikacijama najavi kao servis koji druge aplikacije mogu koristiti po potrebi.

Sa druge strane, istina je da mnoge funkcionalnosti jednostavno nisu dostupne HTML5 aplikaciji. Bez obzira koliko je vešt web programer, ukoliko, primera radi, ne postoji camera API, njegova aplikacija neće biti u mogućnosti da pravi fotografije. Srećom, ukoliko je zaista potrebno da aplikacije ima mogućnost fotografisanja moguće je napraviti sistemski specifičnu aplikaciju, koja u sebi sadrži web view koji omogućava ceo okvir za korisnički interfejs. Na ovaj način PhoneGap, okvir otvorenog koda za razvoj aplikacija, funkcioniše: popunjava prazninu tako što specifične mogućnosti sistema otkriva kao web servise, koje web view poziva uz pomoć standardnih mrežnih API-ja. Kada se na ovaj način napravi hibridna aplikacija moguće je koristiti mogućnosti sistema kao što su widget-i, notifikacije i intent-i.

Pravljenje hibridne aplikacije je daleko od idealnog rešenja. Ovaj način kreiranja aplikacije dodaje kompleksnost i primenljiv je samo na web aplikacije upakovane kao sistemski specifične, ali ne i na tradicionalne web sajtove kojima se pristupa putem pregledača. To neće još dugo biti tako. Web standardi veoma brzo napreduju i moderni pregledači drže korak sa tim razvojem. Mogućnost offline skladištenja, geo lokacija, audio i video reprodukcija su samo neki od primera mogućnosti koje imaju široku podršku na modernim pametnim telefonima. Čak i kamera postaje dostupna. Od Android-a v3.1 moguće je praviti fotografije i video zapise uz pomoć web standarda. Najnovije verzije pregledača na iOS-u podržavaju WebSocket za dvosmerni streaming, kao i detekciju orijentacije uređaja.

Sveukupno, mobilne platforme evoluiraju. Sa druge strane i web takođe evoluira, i to rapidno. Samo na polju pregledača na desktop platformama pet velikih takmaca proširuju i poboljšavaju standarde neverovatnom brzinom. Iako nije lako preneti sve ove mogućnosti na mobilne uređaje, mnoge od njih su našle svoje mesto i na mobilnim pregledačima.

Sistemski specifične aplikacije brzo napreduju, ali web sve više zatvara taj raskorak.

Instaliranje sistemski specifične aplikacije može biti naporno za korisnike. Često ih je potrebno motivisati kako bi napravili korak i instalirali aplikaciju na svoj uređaj. “Instaliranje” web aplikacije podrazumeva kreiranje bookmark-a na početnom ekranu uređaja. Iako je ovaj način preuzimanja aplikacije jednostavniji od preuzimanja aplikacije, on je i manje poznat korisnicima.

Održavanje sistemski specifične aplikacije može biti komplikovano ne samo za korisnike već i za razvojne timove, naročito ako moraju da se nose sa više verzija iste informacije na različitim platformama. Promene koda moraju biti zapakovane i postavljene na portal za distribuciju aplikacija. Sa druge strane, održavanje web ili hibridne aplikacije je podjednako jednostavno kao i održavanje web strane i moguće ga je izvršiti onoliko često koliko je potrebno.

Iako različiti pregledači podržavaju različite verzije HTML5, nezavisnost od platforme je veoma bitna. Ova nezavisnost se definitivno može lakše postići uz pomoć web ili hibridne aplikacije. Kada su ova dva tipa aplikacija u pitanju moguće je, ako ne ceo onda bar delove koda iskoristiti u slučaju prelaska na drugu platformu.

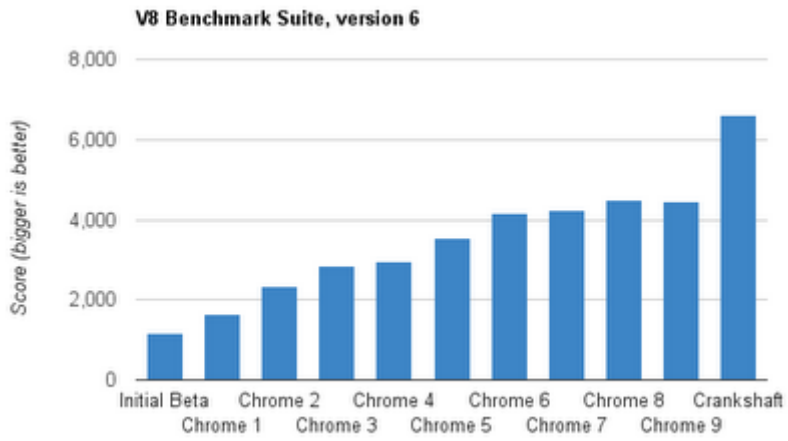
Kada se razvija softver koji zavisi od biblioteka i alata koji određuju sadržaj i dizajn aplikacije to se odražava i u pogledu finansija i utrošenog vremena. Sistemski specifične i hibridne aplikacije moraju da prođu proces odobravanja kao i restrikcije sadržaja koje određuju pravila portala za distribuciju, dok su web aplikacije svega toga oslobođene. Zato i ne čudi odluka mnogih velikih kompanija da se opredele za web aplikacije kako bi se izbegle oštre cenzure sadržaja koje postavljaju kompanije poput Apple-a. Postoji još jedan razlog zašto web aplikacije dobijaju sve veći primat koji je razvojnim timovima sigurno i bitniji. Često je slučaj da portali za distribuciju uzimaju procenat od svake prodaje ili pretplate na aplikaciju, koji ponekad može da iznosi čak i 30% što je nekim timovima previše.

Sa velikom sigurnošću se može reći da je razvoj web i hibridnih aplikacija jeftiniji. Za ove aplikacije je potrebno imati veštine koje su se stekle prethodnim radom na Web-u. Razvoj potpuno specifičnih aplikacija je daleko skuplji i zahteva veću specijalizaciju. Ipak, sa druge strane HTML5 je još uvek nova tehnologija koja, da bi se potpuno shvatila i da bi se razumeo postupak razvoja mobilne web ili hibridne aplikacije, donekle zahteva napredne veštine.

2.4.2 Performanse

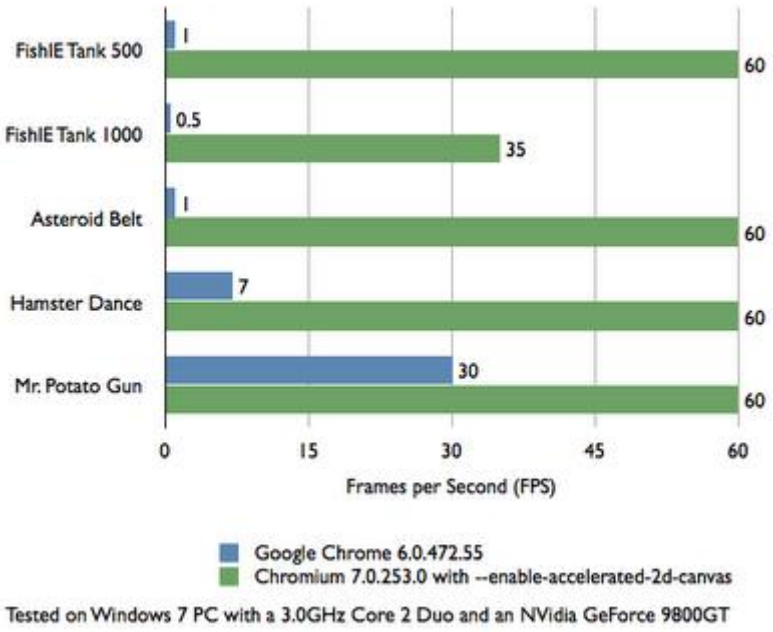
Sistemski specifične aplikacije ne moraju da se nose sa barijerom pokretanja programskog sloja za pokretanje web aplikacija. One su bliže hardverskom sloju uređaja i mogu da iskoriste pogodnosti performance booster-a kao što su GPU akceleracija i rad sa više niti.

Bilo bi premalo reći da su se web aplikacije ubrzale u poslednjih nekoliko godina. Primera radi, V8, the JavaScript engine koji dolazi uz Chrome, već kada se pojavio predstavljao je veliki pomak, a od tada je postajao sve brži i brži.



slika 2.1 prikaz povećanja brzine izvršavanja JavaScript skripti u novijim verzijama Google Chrome-a

Engine-i za render-ovanje grafičkog sadržaja su se takođe ubrzali, a sada postoji i mogućnost hardverske akceleracije.



slika 2.2 uporedba broja slika po sekundi sa i bez hadrverske akceleracije

Dodatno, novi Web Workers API omogućava rad sa više niti, a savremeni razvojni timovi mogu koristiti i veliki broj biblioteka koje su optimizovane za što bolje performanse, kao i ispitane tehnike za optimizaciju performansi. Iako su mnoge od njih u početku razvijane samo za desktop platforme, relevantne su i za mobilne, jer se sve veća pažnja poklanja ovim platformama.

Naravno, nisu svi napreci sa desktop automatski prešli i na mobilne uređaje, ali trend ukazuje da će uskoro biti tako. Važno je primetiti i većina aplikacija za mobilne uređaje nisu najnovije 3D vide igre, već se fundamentalno baziraju na informacijama: vesti, e-mail, rasporedi, društvene mreže itd. Dovoljno je posetiti nekoliko sajtova putem mobilno telefona, kao na primer Gmail, Amazon, Twitter i već je jasno da su performanse mobilnih uređaja više nego adekvatne. Što se igara tiče, 2D igre su sasvim ostvarive, dok se sa

razvojem WebGL za mobilne uređaje pojavljuje sve više prelepih 3D igara. Dok se WebGL sasvim ne razvije, na raspolaganju je sve veći broj framework-a koji kompajliraju WebGL apps u sistemski specifične koje mogu da iskoriste prednosti OpenGL-a.

2.4.3 Iskustva razvojnih timova

Za kreiranje sistemski specifičnih aplikacije koriste se robusni programski jezici, kao što su Java, Objective C, C++, C# itd., koji su predviđeni za razvoj kompleksnih aplikacija i na sceni su već dugi niz godina. API-ji se prave od nule i zasebno za svaku platformu. Vrlo je lako debug-ovati aplikaciju uz pomoć emulatora koji predstavlja dati uređaj.

Razvoj web aplikacija je dodatno otežan raznovrsnošću i velikim brojem postojećih pregledača. Čak iako neka aplikacija radi kako treba, ne znači da neka njena mogućnost biti dostupna na različitom pregledaču ili čak različitoj verziji istog pregledača. Pa iako i bude dostupna, veliko pitanje je kako će je pregledač implementirati. Web standardi i dalje ostaju otvoreni za interpretaciju.

Istina je da su web standardi prvobitno začeti u eri kada je web prvenstveno bio sadržan od dokumenata, a ne aplikacija. Primera radi, JavaScript je razvijen i pušten u korišćenje za samo 10 dana. Vreme je pokazalo da su ti standardi u stanju da urade mnogo više nego što se u početku smatralo. Web developer-i su pronašli način da ojačaju dobre, a ukrote loše strane. Osim toga, i standardi su vremenom evoluirali, a tehnologije kao što su HTML5 i CSS3 donose sve više poboljšanja. Preferiranje i biranje između jezika poput C++, Java i JavaScript se svelo na religioznu debatu i u mnogome zavisi od jezika koje je developer učio na početku karijere. Danas se JavaScript mora uključiti kao ozbiljan takmac.

Loša strana fragmentacije pregledača je upravo postojanje svih tih različitih ekosistema. Kada se aplikacija razvija za Android platformu u Java programskom jeziku, mora je celu port-ovati u Objective C za iOS uređaje. Kada se web aplikacija jednom razvije, ona će moći da se pokrene na Android i iOS uređajima, kao i na WebOS, BlackBerry, Windows Mobile i ostalima. Za sada, tako je u teoriji. U praksi, ipak, neke stvari moraju biti doterane kako bi se na svakom uređaju sadržaj prikazao identično. Sa druge strane, isti proces bi morao biti ponovljen i za sistemski specifične aplikacije, jer mnogi mobilni uređaji dolaze sa različitim verzijama operativnog sistema, kao i hardverskim komponentama.

Dobra strana se ogleda tu činjenici da je fragmentacija odavno prisutna na Web-u i da postoje dobro poznate tehnologije koje se time bave. Što je još važnije, princip stalnog napretka tera razvojne timove da se baziraju prvobitno na bazični uređaj, a zatim dodaju slojeve ekstra mogućnosti koje su specifične za datu platformu, ukoliko i tamo gde je to moguće. Ova, kako je neki nazivaju, mantra otkrivanja mogućnosti uređaja je veoma korisna. Uz pomoć nje, danas postoje pomoćne biblioteke poput Modernizr koja podržava responsive web dizajn. Razumnim korišćenjem ovih tehnika moguće je proširiti skup uređaja koje aplikacija može da pokrije, poput starijih telefona koji datiraju pre ere pametnih, zatim pametnih satova ili čak TV uređaja, bez obzira na proizvođača ili operativni sistem. [9]

2.4.4 Izgled i način korišćenja aplikacije (*look and feel*)

U svetu dizajna softvera look and feel je termin koji se koristi u pogledu grafičkog interfejsa i obuhvata aspekte dizajna aplikacije, što uključuje elemente kao što su boje, oblici, raspored i izbor izgleda oblika slova (deo koji se odnosi na izgled), kao i ponašanje dinamičkih elemenata kao što su dugmići, box-ovi i meniji (deo koji se odnosi na osećaj).

Jedna od odlika koje definišu neku platformu je upravo taj look and feel aspekt. Korisnici očekuju konstantnost kada je raspored i korišćenje kontrola u pitanju. Postoje određeni idiomi koji variraju od platforme do platforme; primera radi - šta će se desiti prilikom izvršavanja *long hold*-a (kada se prst zadrži na nekom elementu nekoliko sekundi). Platforme imaju standardne idiome za ovakve događaje i nije ih sve moguće pokriti jednom HTML5 aplikacijom.

Osim toga, look and feel neke platforme je određen bibliotekom za sistemski specifičan softver, čiji widget-i obuhvataju izgled i način korišćenja koji korisnici očekuju. Dosta od očekivanog izgleda i načina korišćenja se dobija već samim korišćenjem osnovnih alata za razvoj aplikacija za datu platformu.

Kao što je već pomenuto, način na koji se razvijaju web aplikacije je da se napravi jedna osnovna verzija koja bi odgovarala svim platformama, te je progresivno poboljšavati. Iako se poboljšanja tipično odnose na mogućnosti, takođe je moguće poboljšati aplikaciju tako što se ciljaju tačno one platforme do kojih je razvojnom timu najviše stalo. Na ovo favorizovanje pregledača se u web zajednici ne gleda uvek blagonaklono, baš zbog velikog broja pregledača koji postoje i koji se svakodnevno koriste. Mada, ukoliko je razvojni pridaje veliki značaj za dve ili tri platforme, i ima želju da se dodatno potrudi da pridoda broju alternativa u odnosu na sistemski specifične aplikacije, ovo je dobar način razmišljanja.

Što se osnovnog pravca razvoja tiče, i Web ima svoj izgled i način korišćenja aplikacije, kao što se može reći da svaka mobilna platforma ima sopstveni „web look-and-feel“ koji uspostavlja fabrički pregledač. Upravo taj „web look-and-feel“ je pogodan za korisnike i zapravo omogućava da se postigne veći stupanj konsistencije sa načinom korišćenja pregledača na desktop platformi ili nekom drugom sličnom uređaju koji korisnik koristi.

Osim toga, postoje mnoge uspešne aplikacije koje ne prate izgled i način korišćenja sistemski specifičnih aplikacija. Ovo je naročito tačno kada su video igre u pitanju, jer retko koja prati logiku rasporeda, oblika, boja ili ponašanja elemenata interfejsa platforme na kojoj se pokreće. Naravno, ova tvrdnja se ne odnosi samo na video igre. Primera radi, dovoljno je pogledati aplikaciju popularne društvene mreže, Twitter, koja je razvijena za sve poznate mobilne platforme, i primetiti veoma različite mehanizme korisničkog interfejsa. [9]

2.4.5 Dostupnost

Mehanizmi za distribuciju aplikacija, kao što su Google Play za Android ili App store za Apple uređaje, postaju nadmoćni i sve popularniji u odnosu na druge načine distribucije. Istovremeno, postaju glavna pokretačka snaga celokupne mobilne industrije. Svaki razvojni tim može postaviti svoju aplikaciju na neki od ovih portala, gde je korisnici mogu pronaći kombinacijom razgledanja, pretraživanja ili putem preporuka. Pored toga, ukoliko

je aplikacija dobro napravljena i dovoljno korisna, interesantna i ugodna oku, dobre ocene i komentari će pomoći da se korisnici odluče baš za nju u moru sličnih aplikacija.

Web je, od svih medijuma ikada kreiranih, medijum putem kojeg je ubedljivo najlakše pronaći neki resurs. U običnom URL-u, sadržan je jedinstveni identifikator za sve što je ikada objavljeno na Web-u, što uključuje bilo koju aplikaciju koja je objavljena na standardnom web sajtu. Pretraživači omogućavaju da se ove aplikacije lako pronađu, kao i web sajtovi koji su sa njima povezani, tako što uključuju kataloge web aplikacije slične mobilnim marketplace-ovima. Zaista, svako može da podeli svoju web aplikaciju prostim link-ovanjem u e-mail-u ili na nekoj društvenoj mreži. Link se takođe može poslati i putem SMS-a, gde je dovoljno da korisnik klikne na poslati link kako bi pokrenuo aplikaciju u pregledaču svog uređaja. Istina je da još uvek ne postoje portali putem kojih korisnici mogu da ocenjuju i komentarišu aplikacije, ali i ta činjenica se brzo menja. Odličan primer tih promena je Marketplace za Firefox OS. [9]

2.4.6 Monetizacija

Mnogi razvojni timovi žele da unovče svoj trud i prodaju svoju aplikaciju pod što povoljnijim uslovima. Mobilne platforme nude nekoliko načina da se direktno naplate svoje aplikacije. Najjednostavniji način je jednokratna uplata, tj. da se aplikacija otključa zauvek. Takođe određene platforme nude načine plaćanja kroz samu aplikaciju (*in-app payment*) i mehanizme pretplate koji su integrisani u konzistentne i bezbedne mehanizme. Ove nove forme naplaćivanja omogućavaju razvojnim timovima da preobrate instant hit aplikacije u dugoročne izvore prihoda.

Pored naplaćivanja aplikacije postoji mogućnost zarađivanja i kroz tradicionalne metode na Web-u, kao što su reklamiranje i sponzorstvo.

Web ne bi bio pokretač moderne industrije da ne nudi obilje mogućnosti za zaradu. Iako direktni mehanizmi naplaćivanja po korišćenju nisu još uvek procvetali, postoje mnogi primeri gde su se rešenja tipa “softver kao servis”, koja se baziraju na pretplatama, pokazala održivim. Takvih primera je zaista mnogo, velika imena kao što su Google Apps, skup proizvoda 37Signals, kao i svi veliki e-mail servisi, kao što su Gmail, Yahoo itd. Pored toga, direktno plaćanje nije jedini način da se profitira na od web aplikacija. Postoji online reklamiranje, spajanje sa sličnim delatnostima, sponzorstvo, međusobno promovisanje zajedno sa drugim proizvodima i servisima.

Kada se stvari sagledaju iz ove perspektive, savršeno je razumno da razvojni tim koji se bavi razvojem web aplikacija oseti zavist. Kako nije moguće postaviti URL do web aplikacije na neki Marketplace, razvojni timovi pribegavaju triku. Kreiraju sistemski specifičnu aplikaciju koja služi kao omotač , tj. za svaku platformu koja se cilja, kreira se prazna ljuštura sistemski specifične aplikacije koja sadrži web view. Web view je tu da bi okružio pravu aplikaciju. Zatim se ovakva aplikacija postavi na razne portale za distribuciju aplikacija, i uz malo sreće dolazi i zarada. Na glavnim Markteplace-ima postoji na stotine, ako ne i hiljade ovih hibridnih aplikacija. Neke od njih su toliko dobro asimilovane da je nemoguće posumnjati da su u pitanju web aplikacije.

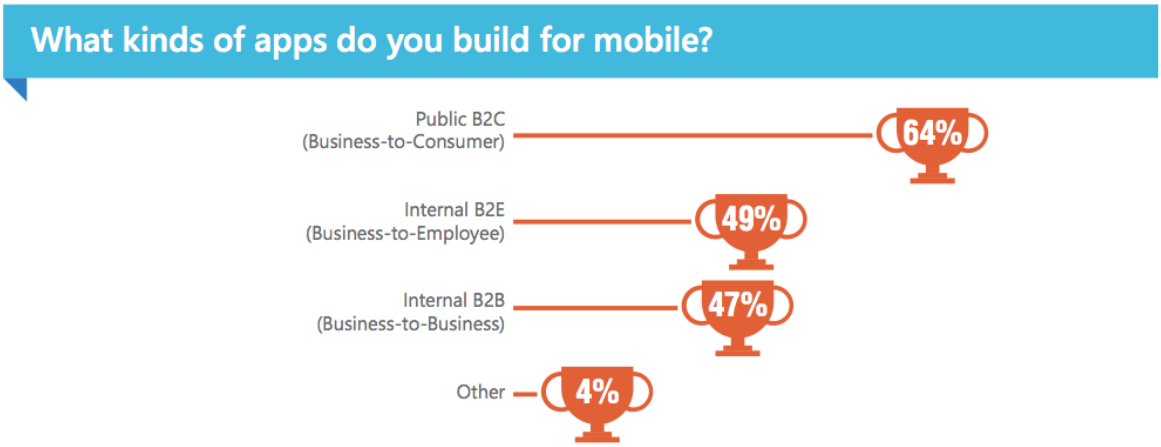
Loša strana je teret kompajliranja za svaku platformu zasebno. U ovom slučaju framework kao što je PhoneGap priskače u pomoć. Pored toga, razvijaju se web servisi poput PhoneGap Build i Apparatio. Dovoljno je usmeriti ove web sajtove ka kodu i dobija

se Android aplikacija, iOS aplikacija i sl. koju je moguće postaviti na portal za distribuciju aplikacija koji važi za tu platformu. Nema potrebe za instaliranje SDK alata, sve što je potrebno je program za pisanje koda i web pregledač.

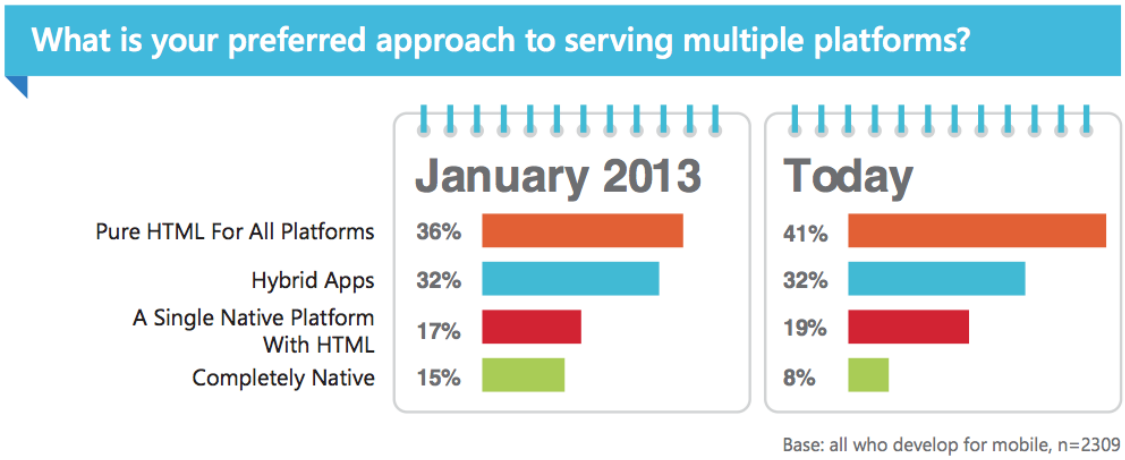
Da li će portali za distribuciju aplikacija ikada podržati direktno preuzimanje web aplikacija bez glavolomki oko umotavanja web aplikacije u okvire sistemski specifičnih? Do pre dve godine odgovor bi bio negativan, ili u najbolju ruku neodređen, ali od pojave Firefox OS-a i Firefox Marketplace-a odgovor je pozitivan. Google je takođe predstavio svoj Chrome Web Store. Iako je, za sada, ovaj portal pokrio samo desktop platforme, pokrenuta je lavina interesovanja drugih proizvođača pregledača, i može se smatrati delom sveukupnog trenda ka katalozima web aplikacija, uključujući tu i neke pokušaje koji se odnose na mobilne platforme. Koncept web marketplace-a je još uvek nov, ali znakovi su obećavajući. [9]

2.5 Statistike razvoja u odnosu na izabrane tehnologije i platforme

Primetan je lagani porast broja razvojnih timova koji su se odlučili za čist HTML5 u odnosu na sistemski specifične aplikacije, dok 41% timova radi na aplikacijama za više platformi istovremeno. Ovaj broj je porastao sa 36% iz januara 2013. godine. Takođe, primetan je i pad broja razvojnih timova koji su se opredelili čisto za sistemski specifične aplikacije, sa 15% početkom 2013. na 8% početkom 2014. godine. Ipak, većina se opredeljuje za hibridne metode, gde neke aplikacije sadrže HTML5 kôd, neke inkorporiraju specifične komponente sa deljenim komponentama za više platformi, a neke su pisane kodom koji je predviđen za rad na više platformi.



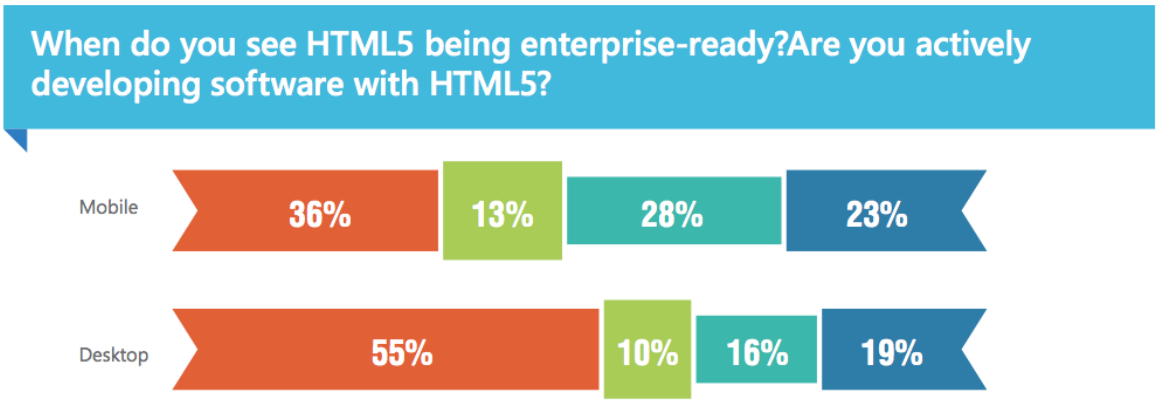
slika 2.3 odnos vrsta aplikacija koje se razvijaju za mobilne platforme



slika 2.4 poređenje izbora platforme od strane razvojnih timova u januaru 2013. i 2014. godine

Sveprisutno pitanje je da li je HTML5, koji je sposoban da ponudi interaktivnost i mogućnosti pravih aplikacije u format koji je čisto web baziran, dovoljno zreo za velike biznis aplikacije.

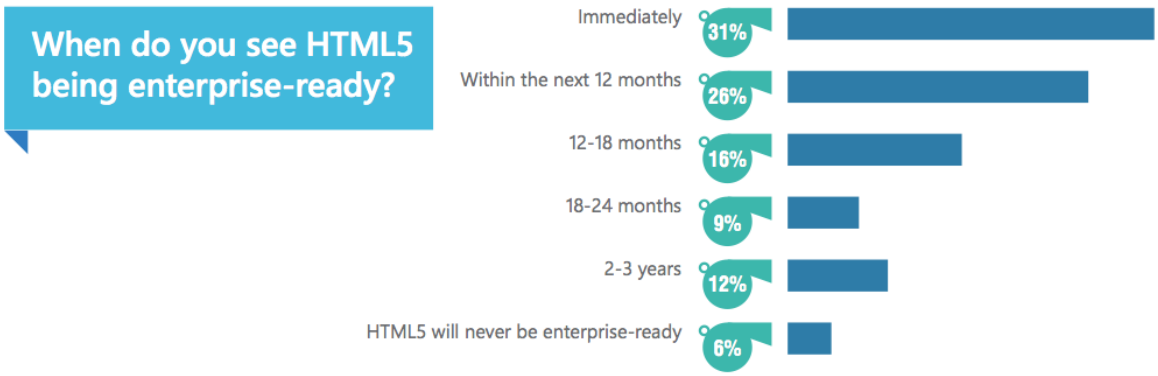
Prema mnogim istraživanjima, trećina razvojnih timova smatra da jeste, već u ovom trenutku, dok četvrtina tvrdi da će za takav nivo biti spreman kroz nekoliko meseci. Veliki broj, ipak - 43% - smatra da će HTML5 biti spreman za godinu dana. Ima i onih koji smatraju da nikada neće biti spreman za ovaj zadatak, ali takvi su u manjini i čine oko 6% ispitanika.



slika 2.5 prikaz mišljenja o zrelosti HTML5 za velike biznis aplikacije

Ove brojke su pomalo čudne kada se uzme u obzir da 91% ispitanika tvrdi da već razvija aplikacije uz pomoć HTML5 i da se većina projekata odnosi na desktop platforme, a manji broj na mobilne. 53% ispitanika smatra da je HTML5 siguran put za razvoj aplikacija koje podržavaju više platformi i kao dodatnu pogodnost navode što i dalje ostaju u poznatom svetu web tehnologija. [10]

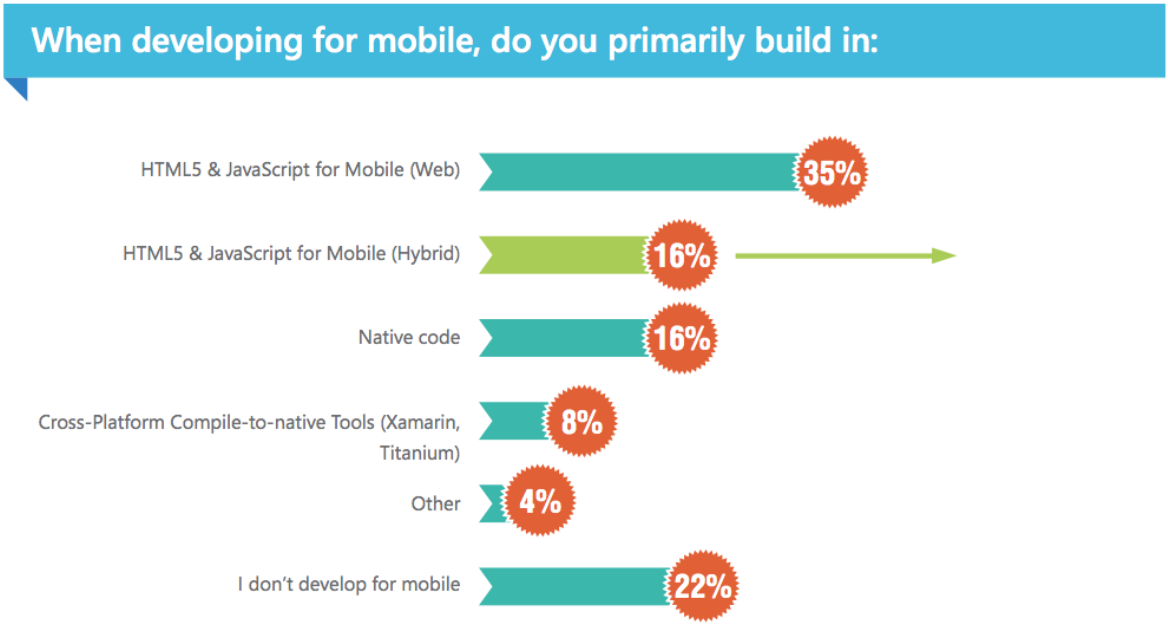
Tek 4% ispitanih se bavi razvojem za iOS dok 13% simultano razvija sistemski specifične aplikacije za iOS i Android, ali i za web. [10]



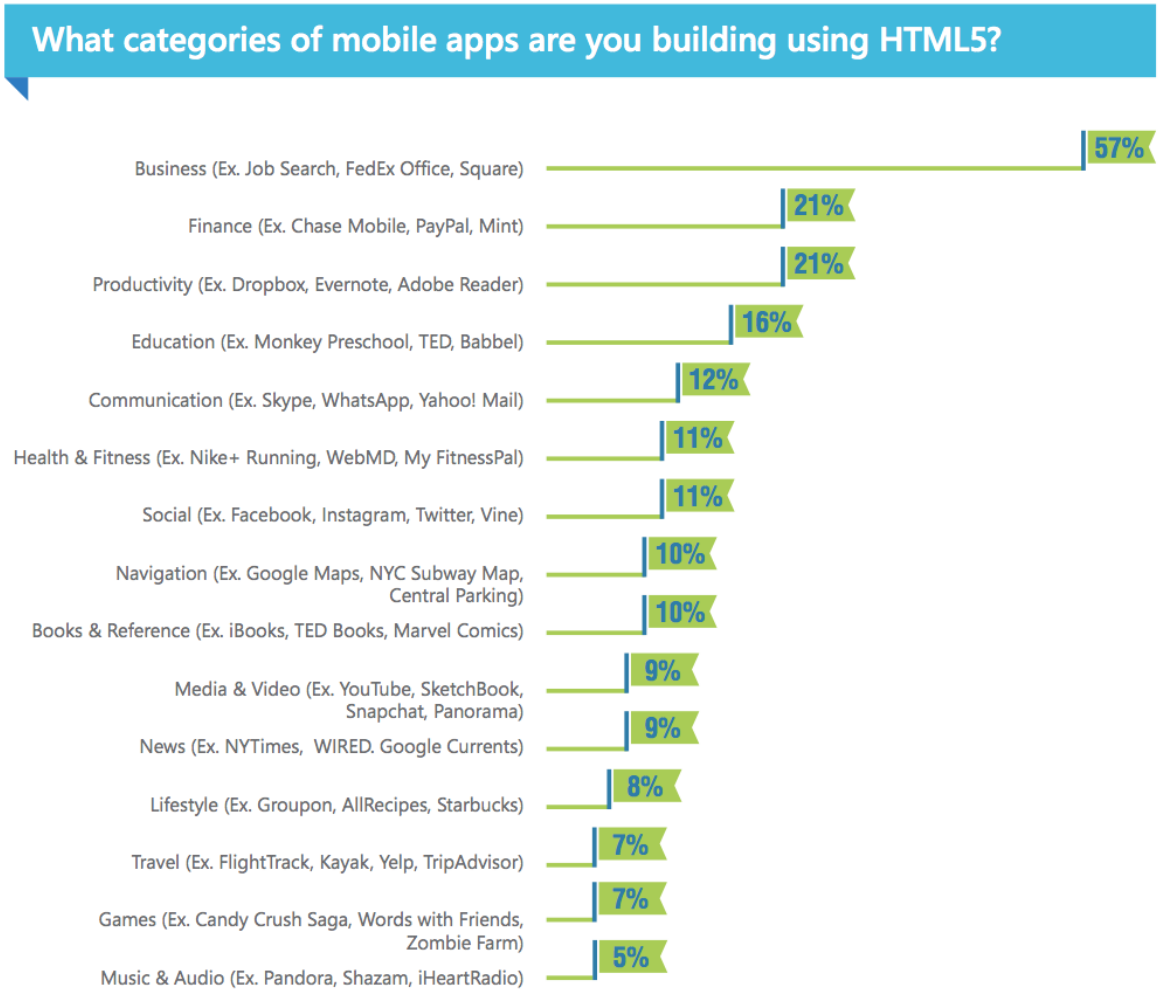
slika 2.6 prikaz mišljenja o trenutku kada će HTML5 biti spreman za velike biznis aplikacije

Interesantno je da Stiv Džobs u početku nije želeo specifične aplikacije za iPhone i da je preferirao web aplikacije. Ipak, posle pojavljivanja iOS app store-a sve se izmenilo. Veliki uspeh Google Play-a pokazatelj je velikog interesovanja za specifične aplikacije uopšte, a ne samo kada je iPhone u pitanju. Ipak, danas razvojni timovi nisu toliko fanatični kada je u pitanju odabir između sistemski specifičnih, višeplatformskih ili hibridnih aplikacija u pitanju, već se više fokusiraju na tehnologije koje zadovoljavaju potrebe pojedinog projekta.

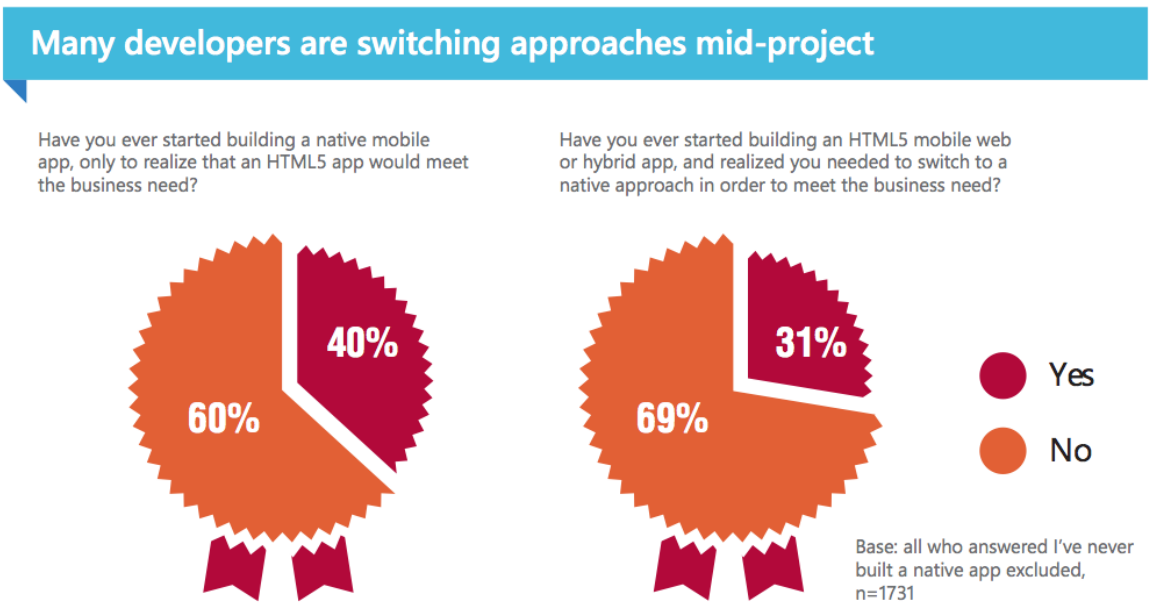
Kada se gleda u kontekstu razvoja, u mnogim slučajevima hibridne aplikacije su pravi izbor za datu aplikaciju, ali ima i mnogo slučajeva gde je sistemski specifična aplikacija najrazumniji izbor. Ono što je razvojnim timovima potrebno jesu alati koji im omogućavaju da budu efektivni bez obzira na pristup za koji se opredele. [10]



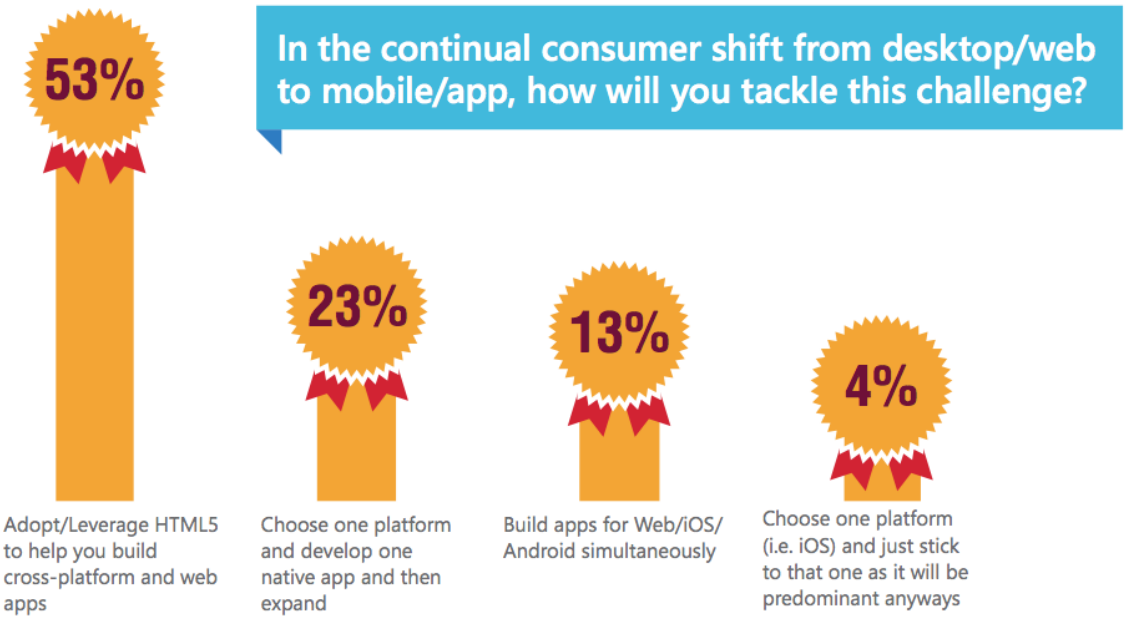
slika 2.7 prikaz odnosa pristupa razvoju mobilnih aplikacija



slika 2.8 kategorije mobilnih aplikacija koje se razvijaju HTML5 jezikom



slika 2.9 odnos razvojnih timova koji menjaju pristup tokom već započetih projekata



slika 2.10 odnos mišljenja o izboru platforme u odnosu na neprekidan prelaz korisnika sa desktop na mobilne platforme

3 Web tehnologije u industriji video igara i edukaciji

<canvas> je HTML5 element koji pruža mogućnost iscrtavanja 2D i 3D objekata uz pomoć JavaScript API-ja. Jedan od takvih API-ja je WebGL koji se koristi za iscrtavanje u 3D kontekstu <canvas> elementa.

3.1 OpenGL, Open GL ES i WebGL

WebGL specifikacija je objavljena od strane Khronos Group-a, organizacije koju su 2000. godine formirale velike medijske kompanije kao što su ATI, Intel i Sun Microsystems. Khronos grupa sebe opisuje kao kompaniju koja je posvećena kreiranju standardnih API-ja kako bi se omogućila autorizacija i reprodukcija medija na velikom broju platformi i uređaja. [7]

Od 2000. godine objavili su veliki broj široko korišćenih specifikacija poput OpenGL, OpenGL ES i OpenVG. OpenGL je specifikacija za kreiranje bogatog 3D sadržaja na više platformi i veoma je popularna, pogotovo u industriji video igara. Ova specifikacija je objavljena 1992. godine.

2003. godine Khronos Group je objavio OpenGL ES (Embedded Systems). Ova specifikacija je predviđena za manje i slabije uređaje. Verzije 1.0 i 1.1 su koristile pipeline sa fiksnim funkcijama i implementirale tradicionalne modele za osvetljavanje i osenčenje. Koristile su se na uređajima kao što su Nintendo 3DS, Android od verzije 1.6 i starije verzije iOS-a. Sony Playstation 3 takođe koristi OpenGL ES, ali u sopstvenoj varijanti jer je dodat programabilni pipeline i druga poboljšanja, koja su kasnije iskorišćena za verziju 2.0 koja je nastala godinu dana posle Playstation 3. [7]

OpenGL 2.0 specifikacija je nastala 2007. godine i sadržala je mnogo poboljšanja u odnosu na OpenGL 1.1. API je baziran na jeziku C i dodata je podrška za datoteke sa

plutajućim zarezom i programabilni pipeline umesto fiksnog iz verzije 1.1. Ovo je razlog zašto OpenGL 2.0 nije povratno kompatibilan sa verzijom 1.0. Verzija 2.0 podržava iPad, iPhone 3GS i novije, Android 2.2 i novije, Maemo uređaje itd. OpenGL ES 2.0 funkcioniše samo na uređajima koji podržavaju hardversku akceleraciju.

WebGL 1.0 specifikacija je objavljena 2011. godine od strane Khronos-a i od tada je postala standard za iscrtavanje 3D elemenata u canvas elementu HTML5. WebGL ima JavaScript API koji je gotovo identičan sa onim koji se koristi u OpenGL ES 2.0. Razlika leži u drugačijoj semantici zvog razlike u jezicima C i JavaScript. Ova predstavlja veliku prednost, jer WebGL može da računa na veliki broj potencijalnih developera koji već poznaju OpenGL i OpenGL ES, uzevši u obzir da je proces tranzicije lagan.

Kôd je veoma sličan - razlike uglavnom leže u semantici i načinu prikazivanja grešaka. Iako je kôd toliko sličan, ne postoji konverter koji bi izvršio prebacivanje iz OpenGL ES u WebGL. I pored nepostojanja konvertera, zapravo je veoma lako prepisati aplikaciju u WebGL ukoliko je pravilno implementirana u OpenGL ES.

3.2 3D modelovanje uz pomoć web tehnologija

Pre nego što je predstavljen WebGL postojala su dva načina da za prikaz 3D sadržaja u pregledačima. Uglavnom su se koristili Flash ili Java apleti. Kako su vremenom Java apleti potisnuti, Flash je ostao kao jedina ozbiljna konkurencija WebGL-u.

Najveća mana Java apleta i Flash-a je potreba da se u pregledač instalira plugin za njihovo pokretanje. Ova činjenica je oduvek predstavljala problem, jer na nekim računarima ne postoji mogućnost instaliranja plugin-a (na primer na školskim računarima na kojima učenici nemaju administratorske privilegije, te ne nemaju mogućnost instaliranja novih ili ažuriranja postojećih aplikacija). Mobilni uređaji takođe predstavljaju problem, jer mnogi od njih nemaju podršku za Java aplete ili Flash.

Postoji mogućnost prikazivanja 3D objekata za obe platforme (WebGL i Flash) korišćenjem aplikacija kao što su Blender, CopperTube itd. Svaki od ovih programa ima svoje 3D engine - Away3D za Flash i Three.js za WebGL. Međutim, WebGL zahteva da klijentska strana ima GPU podršku kako bi se shader-i mogli render-ovati. Trenutno WebGL nema dobru podršku u pregledačima (Internet Explorer ga ne podržava, kao ni stabilne verzije Safarija i Opere za OS X).

3.3 Performanse

Feliks Tarner je u oba jezika napisao jednostavnu aplikaciju za render-ovanje kocke kako bi testirao performanse pregledača kada su Flash i WebGL u pitanju. [1] U poređenje je uključen novi Flash 3D API, Stage3D koji je posvećen 3D render-ovanju u realnom vremenu. [2] Tabela prikazuje performanse različitih pregledača na računaru MacBook Pro 15” sa Intel Core i7@2GHz quad-core, 4GB RAM, AMD Radeon HD 6490M.

Broj kocki	Chrome 16.0.9		Firefox 9.0.1	
	WebGL	Stage3D	WebGL	Stage3D
100	60fps	40fps	60fps	60fps
500	60fps	20fps	56fps	~30fps
1000	30fps	14fps	30fps	~10fps
2000	22fps	7fps	18fps	-

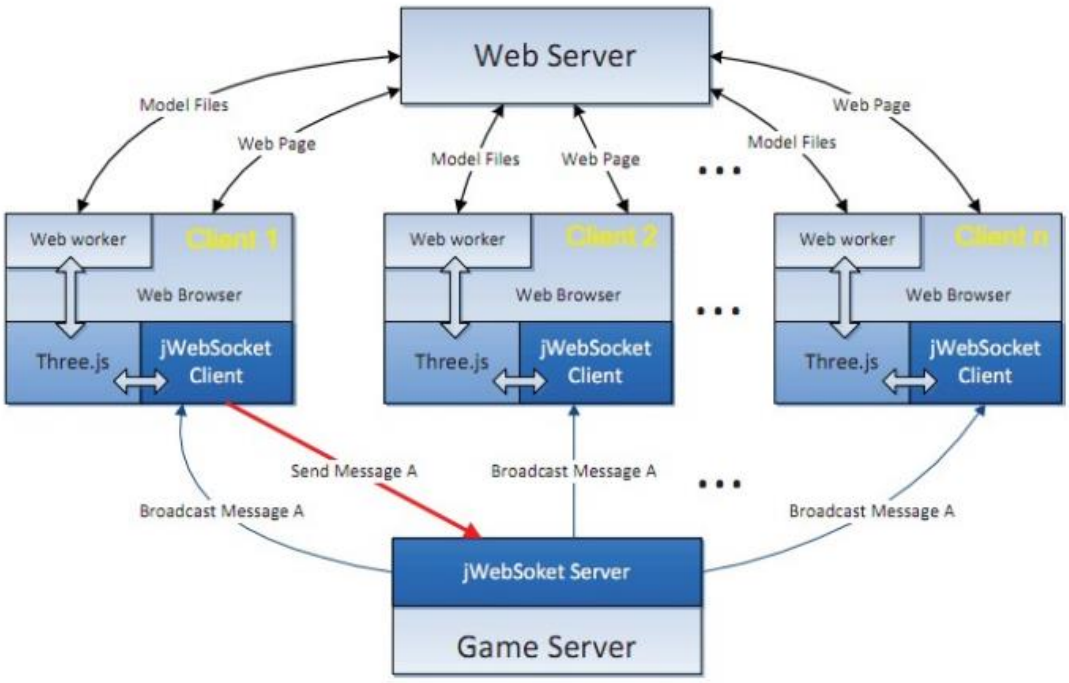
tabela 3.1 tabelarni prikaz performansi WebGL i Flash Stage3D

WebGL je sa lakoćom nadmašio Stage3D na ovom testu. Dodatno, WebGL je prikaz je bio daleko glatkiji, dok je broj slika po sekundi (*fps*) kod Flash-a bio veoma nestabilan. Čak i prikazu 100 kocki kod Flash-a u nekim trenucima animacija nije bila glatka. Suprotno tome, WebGL je i pri prikazivanju 2000 kocki je izgledao glatkije, čak i sa manjim fps-om. Firefox je postao nestabilan pri prikazu 1000 kocki, te iz tog razloga nedostaju rezultati za 2000. Naravno, ovaj test se ne može uzeti kao jedina referenca kada je testiranje performansi u pitanju, ali na dobar način prikazuje snagu WebGL-a.

3.4 Potencijalna upotreba WebGL-a

Postoje tri oblasti u kojima bi WebGL mogao pronaći svoje mesto, a to su: različite vrste vizualizacije informacija, video igre i 3D modelovanje. Kao što se može videti iz tabele performansi (tabela 3.1) WebGL stoji bolje u odnosu na Flash. Međutim, podrška pregledača još uvek ne ide u korist WebGL-a.

Ranija tvrdnja da je bolje razvijati aplikacije u Flash-u, a WebGL ostaviti za kasniji period kada još sazri sve manje stoji i WebGL lagano preuzima primat u odnosu na Flash. Slično važi i za video igre. Raniji razvoj igara, primera radi za Facebook platformu, je preferirao razvoj u Flash-u u odnosu na WebGL. Međutim, kao odličan primer da budućnost video igara na Web-u i pregledačima na mobilnim uređajima upravo leži u WebGL tehnologiji služe istraživanja Bijin Chena. [3] Bijin je lansirao video igre bazirane na Web platformi sopstvenom implementacijom framework-a za online multiplayer igre koje se pokreće u mobilnom pregledaču. Za ovaj projekat je iskoristio komunikaciju između WebGL i WebSockets. Kako Bijin tvrdi, ovo bi mogla da bude budućnost multiplayer igara jer igre ne zahtevaju zasebne instalacije i mogu se pokrenuti sa bilo kog računara. Arhitektura framework-a je prikazana sa slici.

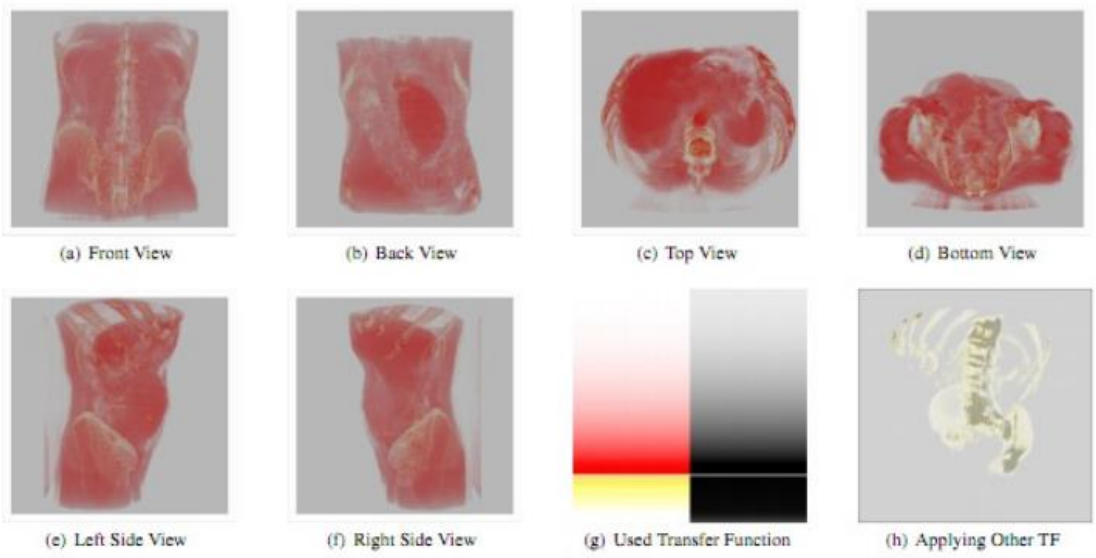


slika 3.1 arhitektura framework-a

Web server upravlja interakcijom i obezbeđuje datoteke za model, dok jWebSocket upravlja komunikacijom između igrača.

Druga potencijalna upotreba je razvoj aplikacija za 3D modelovanje. Iako na ovu temu ne postoji mnogo istraživanja, radova ili članaka, postoji zanimljiv projekat pod nazivom Tinkercad. Uz pomoć Tinkercad-a moguće je iscrtati 3D model u okviru pregledača i poslati ga kompaniji koja bi mogla da izradi fizički model iz crteža. Iako su funkcionalnosti ograničene projekat obećava, jer je u pitanju tek beta faza. Jedna od funkcionalnosti o kojoj se razmišlja je import objekata iz drugih programa.

Treći način na koji bi WebGL mogao da se iskoristi je 3D vizualizacija kompleksnih podataka. Jedan od takvih projekata je projekat Dzon Kongotea koji je primenio WebGL u aplikaciji za prikaz medicinskih podataka. [4]



slika 3.2 Aplikacija za 3D prikaz medicinskih podataka

Prema objavljenom članku, render-ovanje medicinskih podataka se najčešće vrši na centralizovanom serveru. Korišćenjem WebGL-a render-ovanje se može vršiti na klijentskoj strani, korišćenjem CPU i GPU snage klijentskog računara. Na ovaj način se smanjuju hardverski zahtevi, a samim tim i cena servera. Server se dalje može koristiti prosto za čuvanje i pribavljanje informacija. Mogućnost prikaza u više različitih pregledača je veliki plus za ovaj pristup, čak iako se na trenutnim Android uređajima prikaz svodi na 2-3 slike po sekundi. U svakom slučaju, članak dokazuje da je moguće prikazati volumetrijske podatke na mobilnim uređajima u realnom vremenu.

Još jedan odličan primer je projekat WebGLot čija je svrha brzo plovovanje matematičkih premisa u pregledaču. [5]

Weigang Zhang i Hao Yua u svom projektnom radu predlažu upotrebu WebGL-a za prikaz električne mreže i praćenje njenih performansi na različitim geološkim lokacijama. [6]

3.5 WebGL u oblasti video igara

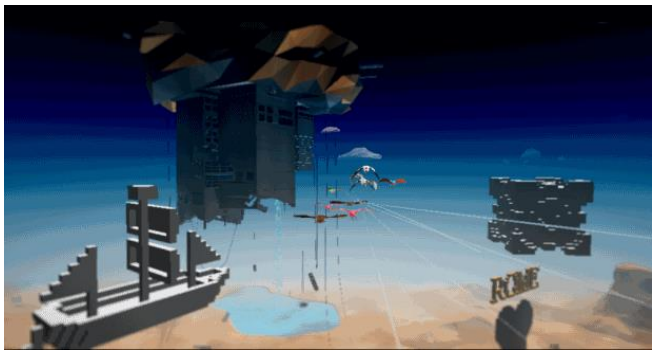
Gotovo svaki moderni računar i većina pametnih telefona imaju jake grafičko procesorske jedinice. Ovi čipovi su često moćniji od CPU čipova. Međutim, do skoro web stranice i mobilne verzije sajtova nisu imale mogućnost korišćenja GPU čipova. To je značilo spore grafike niskog kvaliteta i gotovo po pravilu u 2D.

Sve se to izmenilo kada se WebGL-a distribuiran sa novijim verzijama Firefox i Chome pregledača. Kao što je već napomenuto, WebGL, koji je baziran na poznatom OpenGL 3D standardu, omogućava pristup grafičkom hardveru putem JavaScript-a i canvas elementa koji je deo HTML5, bez upotrebe plugin-a, što dalje omogućava prikaz 3D grafičkih elemenata na web stranama u realnom vremenu.

3.5.1 Primeri WebGL video igara

1. 3 Dreams of Black

3 Dreams of Black je polu interaktivni film Kris Milka koji koristi tehnologije koje su razvili Google i drugi. Film vodi igrača kroz tri različita sna uz kombinovanje 2D i 3D elemenata sa videom. Može se pogledati na adresi http://www.ro.me/



slika 3.3 3 Dreams of Black

2. Angry Birds

Postoje dve verzije igre Angry Birds - verzija sa niskom rezolucijom (SD) koja je bazirana na 2D Canvas-u koji ne zahteva hardversku akceleraciju i WebGL HD verzija. Može se igrati na adresi: http://chrome.angrybirds.com/



slika 3.4 Angry Birds

3. Ironbane MMO

Ironbane je 3D platformska igra za više igrača u online režimu u kojoj se perspektiva menja u odnosu na položaj igrača. Glatkost animacije i odziv su neverovatni. Ironbane je upravo ono što HTML5 igra treba da bude. Može se igrati na adresi: http://www.ironbane.com/game.php



slika 3.5 Ironbane MMO

4. Pearl Boy

Poenta igre je pokretanje dečaka koji vesla predivnim morem sa zalaskom sunca u pozadini. Može se igrati na adresi: http://www.gooengine.com/demofiles/pearl-boy/index.html



slika 3.6 Pearl Boy

5. Google Maps Cube Game

Google Maps tim je u saradnji sa B-Reel stvorio Google Maps Cube Game, igru u kojoj je svrha sprovesti plavu kuglicu do cilja kroz ulice Njujorka, Tokija, Pariza ili San Franciska. Kada korisnik pomera kursor miša kocka se rotira i kuglica se pomera niz ulice, odbija od zgrada i kreće ka cilju. Ova igra koristi three.js za svoje rendering-e. Kao što je već pomenuto, three.js je odlična biblioteka za svakoga ko planira da se komercijalno bavi razvojem WebGL aplikacija.
Može se igrati na adresi: http://www.playmapscube.com/



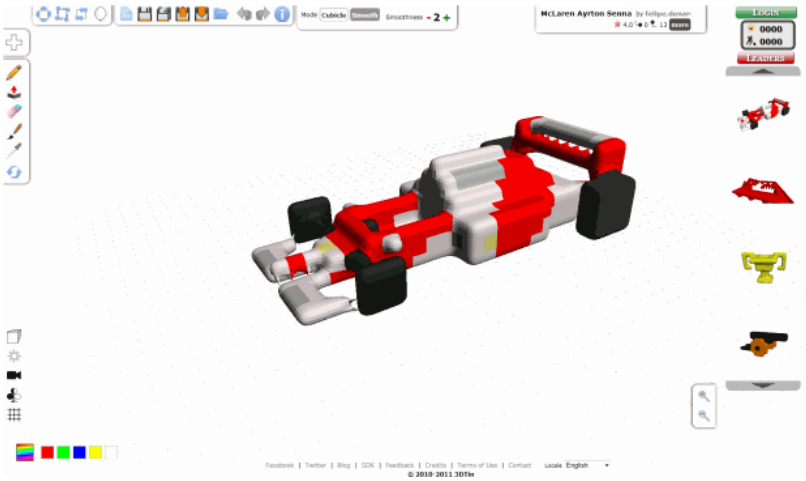
slika 3.7 Google Maps Cube Game

3.6 WebGL u oblasti edukacije

3.6.1 Primeri WebGL edukativnih aplikacija

1. 3Dtin

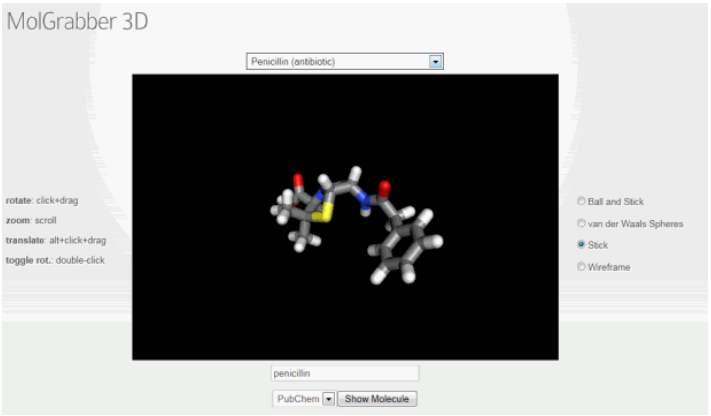
3Dtin je intuitivan i lak za učenje. Uz pomoć njega moguće je kreirati 3D sadržaj voxel-oriented pristupom (*najpribližniji prevod bi bio - 3D pikseli*). Spremni objekti se lako mogu pretvoriti u fizičke putem 3D štampe. Može se pokrenuti na adresi: http://www.3dtin.com/



slika 3.8 3Dtin

2. Chemdoodle

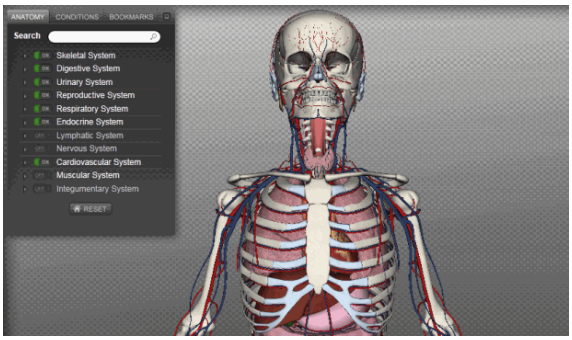
iChemLabs, kompanija koja se bavi vizuelizacijom naučnih istraživanja, predstavila je alat za prikaz molekula u okviru web stranice; na demonstrativnoj stranici moguće je odabrati neki od brojnih lekova i videti 3D prikaz njegove hemijske strukture u pregledaču. Može se pokrenuti na stranici: http://web.chemdoodle.com/demos/molgrabber-3d



slika 3.9 Chemdoodle

3. The Biodigital Human

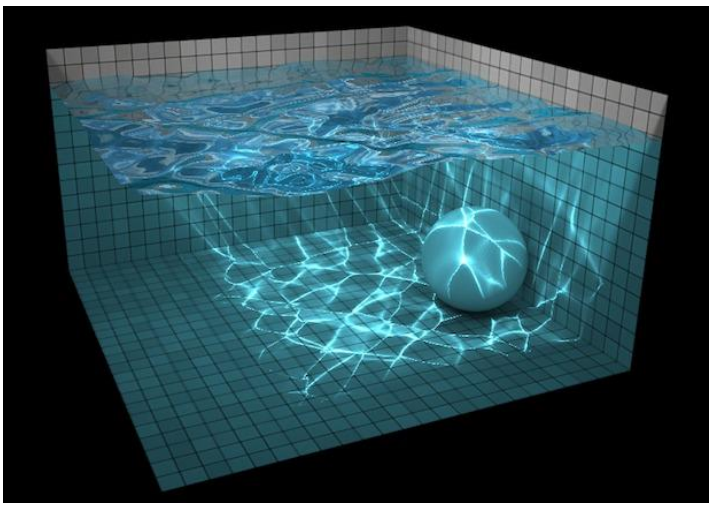
Postoji nekoliko web aplikacija posvećenih anatomiji ljudskog tela. Neke od njih su: The Biodigital Human, koji se može pokrenuti na adresi: http://www.biodigitalhuman.com/; i Google-ov Body Browser, koji se može pokrenuti na adresi: http://bodybrowser.googlelabs.com/.



slika 3.10 The Biodigital Human

4. WebGL Water

WebGL Water je eksperiment Evana Valasa koji predstavlja interaktivnu scenu u kojoj se lopta potapa u bazen. Scena sadrži refleksije, prelamanja svetlosti, kaustičnosti i okulziju ambijenta. Cilj eksperimenta je testiranje krajnjih granica različitih efekata u okviru WebGL shader-a. Shader-i su sastavljeni programi, pisani u GLSL-u i pokreću se u grafičkom procesoru svaki put kada se iscrta scena putem WebGL-a i veoma su optimizovani baš za kalkulacije koje podupiru ove vrste efekata. Može se pokrenuti i testirati na adresi: http://madebyevan.com/webgl-water/



slika 3.11 WebGL Water

5. WebGL Terrain Editor

WebGL Terrain Editor je kreiran od strane Rod Čedvika i predstavlja impresivnu tehničku aplikaciju. Ova demonstracija omogućava stvoranje sopstvenih pejzaža u realnom vremenu i odlikuje se efektima kao što su volumetrijske senke, meke senke praćene zracima i okulzije ambijenta.



slika 3.12 WebGL Terrain Editor

4 Potreba za otvorenom Web platformom i uloga Mozilla-e

Web, kakvim ga danas poznajemo, je veoma fragmentovan. Manjim delom zbog različitih brzina kojima mu pristupamo, a većim zbog raznolikosti uređaja i platformi koje svakodnevno koristimo kako bismo došli od određenog sadržaja ili informacije.

U eri smo kada gotovo da ne postoji osoba koja može da zamisli svoj radni dan bez stalnog prisustva barem jednog uređaja koji ima pristup Internetu. Klasične desktop konfiguracije lagano odlaze u istoriju, jer im nedostaje ključna karakteristika, a to je mobilnost. Na scenu stupaju uređaji novijih generacija poput laptopa, notebook-a, tablet računara i, naravno, nenadmašivi sadrug moderne komunikacije – mobilni telefon. Sami telefoni su, za relativno kratak vremenski period, u tehnološkom smislu, prešli veliki put. Dok je do pre nešto više od decenije mobilni telefon bio glomazna kvadratasta struktura, često teška i po više stotina grama, i služio uglavnom za pozive, kada je čak i slanje poruka bila napredna opcija, danas je bezmalo hardverski moćniji nego računar star nekoliko godina, a i sposobniji je za obavljanje mnogih naprednih funkcija. Zahvaljujući, najpre, Symbian-u koji je promovisala Nokia, ali ipak najviše Apple-ovom iOS-u i Google-ovom Android-u, pametni telefoni su postali uobičajena pojava. Trend pametnog telefona je doživeo bum pre nekoliko godina, baš zahvaljujući razvoju ovaj dva operativna sistema za mobilne platforme, koji su trenutno i najviše zastupljeni na tržištu.

Užurbani stil života je kod skoro svake osobe polako razvio naviku da informacija gotovo u svakom trenutku bude nadohvat ruke, da u pokretu može da se obavi transakcija, proveri stanje na računu, rezerviše nešto, kontaktira neko i sl. Upravo u tome leži snaga i razlog sve bržeg razvoja mobilnih platformi.

Za razliku od drugih platformi poput iOS, Android, Windows Phone i sl. Mozilla, budući kompanija koja se primarno zalaže za razvoj i standardizaciju Web-a, nastupa sa idejom za stvaranje otvorenog Web ekosistema, kako za developere (ljude koji se bave

razvojem softvera) tako i za korisnike, kroz predstavljanje novih rešenja koja bi bila nezavisna od platforme. Broj platformi raste gotovo svakodnevno i, dok sa jedne strane, njihov veći broj obezbeđuje raznolikost i veću slobodu za developere, sa druge strane ih i ograničava za razvoj aplikacije samo za datu platformu. Naravno, aplikacija se može paralelno razvijati i za više platformi, ali to zahteva ili angažovanje većeg tima developera, ili developera koji vlada sa više programskih jezika i sistema.

Kako je ideologija Mozilla fondacije uvek akcentovala promociju otvorenih Web standarda, ne čudi što iste ideje ugrađuju u svoj novi operativni sistem za prenosne uređaje. Firefox OS je u osnovi Web pretraživač koji funkcioniše kao platforma, tako što koristi HTML5, CSS3 i JavaScript standarde za kreiranje aplikacija koje su u osnovi Web aplikacije. Ovo prosto znači da svako ko zna da napravi Web sajt, zna i da napravi aplikaciju za Firefox OS.

Mozilla je zajednica za razvoj besplatnog softvera. Njihov najpoznatiji proizvod je Firefox Web Browser za desktop. Fondacija se bavi razvojem, širenjem, podržavanjem i korišćenjem Mozilla proizvoda i radi na unapređivanju ciljeva Open Web-a na način koji je opisan u Mozilla Manifestu. Zajednica je institucionalno podržana od strane Mozilla fondacije i oporezive podružnice, Mozilla Company.

Pored Firefox-a, Mozilla u svoje proizvode svrstava i Thunderbird e-mail klijent, Firefox Mobile Web Browser za mobilne platforme, zatim Firefox OS operativni sistem, kao i sistem za praćenje grešaka pod nazivom Bugzilla i mnoge druge.

4.1 Istorija Mozille

23. februara 1998. godine, Netscape Communications Company je pokrenula projekat pod imenom Mozilla (što je razvojno ime Netscape Navigator pretraživača), kako bi se koordinirao razvoj integrisanog Internet programskog paketa za veći broj platformi, pod nazivom MAS (Mozilla Application Suite). [11]

MAS je predviđen da bude *open source* (kompletno dostupnog koda) verzija Netscape-ovog pretraživača Netscape Navigatora. Paket je bio baziran na izvornom kôdu Netscape Communicator-a i sadržao je:

- Navigator browser, Communicator (Mozilla Mail i Newsgroup klijent);
- Mozilla Composer (alat za razvoj Web stranica);
- ChatZilla (IRC klijent) i elektronski imenik.

Razvoj paketa je vođen od strane Mozilla organizacije od 1998.-2003., a od 2003.-2006. od strane Mozilla fondacije. Zamenjen je projektom pod imenom SeaMonkey Internet Suite, paketom koji razvija zajednica i koji je baziran na istom izvornom kodu i unapređenoj bazi kodova.[13]

Prvobitno, cilj Mozille je bio razvoj tehnologija za kompanije poput Netscape-a, koje bi zatim komercijalizovale njihov otvoreni izvorni kod. Kada je u julu 2003. AOL (partnerska kompanija Netscape-a) drastično smanjio svoj udeo u Mozilla-i, osnovana je Mozilla fondacija kao pravni zastupnik projekta. Ubrzo, Mozilla napušta projekat u korist stvaranja nezavisnih aplikacija za svaku od funkcija, najpre Firefox pretraživača i Thunderbird e-mail klijenta.[20]

4.2 Softver

- **Firefox Web browser** je Mozilla-in najpoznatiji i najvažniji softverski proizvod. Dostupan je kako u desktop, tako i u mobilnoj varijanti. Potpuno je besplatan i otvorenog koda. Razvija se uporedo za Windows, OS X, Linux i Android operativne sisteme. Kao i mnoge druge aplikacije koje se razvijaju od strane Mozilla-e, koristi Gecko layout engine za prikaz Web stranica. Gecko implementira aktuelne, ali i Web standarde za koje se tek očekuje da će biti standardizovani od strane W3C (World Wide Web Consortium). Prema istraživanjima, Firefox približno zauzima 20-24% tržišnog udela Web pretraživača, što ga stavlja na treću poziciji po broju korisnika.[20]

- **Firefox mobile** (razvojno ime Fennec) je verzija Firefox Web browser-a za mobilne uređaje poput smart phone-a i tablet računara. I u ovoj verziji se koristi isti Gecko layout engine. Kao i kod desktop verzije, glavne karakteristike su HTML5 podrška, Firefox Sync, podrška za dodatke (*add-ons*) i pregled stranica po *tabovima* (jezičcima). Mobilna verzija je dostupna na uređajima sa Android OS v2.2 i naviše, koji imaju ARMv6 ili ARMv7 procesorske jedinice, dok x86 arhitektura nije u potpunosti podržana.

- **Thunderbird** je klijent za e-mail, vesti i chat. Besplatan je i otvorenog koda i razvija se za više platformi. Postoje verzije za Windows, Linux, OS X, OpenSolaris, FreeBSD i dr. Strategija za ovaj projekat je preuzeta od Firefox-a. Kao i kod ovog pretraživača, i osnovne mogućnosti Thunderbird-a se mogu proširiti dodacima (*extensions*). Preuzimanje novih dodataka, kao i ažuriranje postojećih se na lak način vrši pomoću XPInstall-a (XPI) putem Web sajta. Među najpoznatijim dodacima se nalazi Mozilla Lightning koji dodaje mogućnost korišćenja kalendara, kao i PIM (Personal Information Manager). [20]

U protokole koje Thunderbird podržava spadaju POP, IMAP, zatim LDAP (Lightweight Directory Access Protocol). U sebi ima ugrađen RSS/Atom čitač za sakupljanje novosti. U oktobru 2012. Mozilla je najavila smanjenje prioriteta za razvoj Thunderbird-a, jer uloženi napor za razvoj novih mogućnosti nije urodio plodom. Ponuđen je Extended Support Releases model, po kojem će Mozilla obezbeđivati sigurnosne i servisne zakrpe, dok će zajednica na sebe preuzeti razvoj novih mogućnosti.[11]

- **SeaMonkey**, ranije poznat pod imenom Mozilla Application Suite, je besplatna platforma za više sistema, otvorenog je koda i od komponenata sadrži Web pretraživač, e-mail i USENET klijent, HTML editor pod nazivom Mozilla Composer, kao i ChatZilla IRC klijent. U martu 2005. Mozilla je objavila da neće nastaviti razvoj MAS iznad verzije 1.7 i od tada razvoj preuzimaju SeaMonkey Council i zajednica, dok Mozilla obezbeđuje hosting developerima.[15]

- **Bugzilla** je sistem za praćenje bug-ova (softverskih grešaka) opšte namene i baziran je na Web-u. Objavljen je 1998. od strane Netscape-a kao softver otvorenog koda. Usvojen je od strane mnogih organizacija za potrebe praćenja grešaka, kako pri razvoju softvera otvorenog koda, tako i softvera koji je u vlasništvu neke kompanije (*proprietary software*). Bugzilla-u, između ostalih, koriste Mozilla fondacija, Wikimedia fondacija, WebKit, NASA, Yahoo!, GNOME, KDE, Apache, Red Hat i dr.

4.2.1 Softverske komponente

- **NSS** (Network Security Services) obuhvata komplet biblioteka dizajniranih tako da mogu da podrže razvoj sigurnosnih klijent/server aplikacija sa opcionom podrškom za

hardversko SSL ubrzanje na serverskoj i pametnim karticama na klijentskoj strani, i to na različitim platformama.

Podržana je kompletna open source implementacija kriptografskih biblioteka sa podrškom za SSL, TLS i S/MIME. Pored Mozille i druge kompanije poput AOL, Red Hat, Sun Microsystems/Oracle Co., i Google, zajednički razvijaju i koriste NSS u svojim aplikacijama. Mozilla obezbeđuje *repository* (skladište) za izvorni kod, sistem za praćenje grešaka i infrastrukturu za mailing liste i grupe za diskusiju. Pored biblioteka i API-ja, NSS obezbeđuje alate potrebne za debugging, dijagnostiku, kao i za upravljanje sertifikatima, ključevima i kriptografskim modulima. Programeri mogu da koriste NSS dinamičke biblioteke koje su povratno kompatibilne, tako da korisnici mogu da ažuriraju kôd na nove biblioteke bez potrebe za rekompajliranjem. Mozilla je sertifikovala NSS 3.x na 18 različitih platformi.

- **SpiderMonkey** je prvi JavaScript runtime engine koji je ikada razvijen. Razvijen je prvobitno pod okriljem Netscape Communications-a, a kasnije objavljen kao open source i od tada ga razvija Mozilla fondacija. Trenutno, SpiderMonkey pruža podršku za JavaScript u Firefox-u i raznim dekstop okruženjima (*Desktop Environment*) poput GNOME 3.

- **Rhino** je open source JavaScript engine koji razvija Mozilla fondacija. Potpuno je zavijen u Javi. Rhino pretvara JavaScript u Java klase i radi kako sa kompajliranim, tako i sa interpretiranim kodom.

- **Gecko** je layout engine koji podržava Web stranice napisane pomoću HTMLa, SVGa ili MathMLa. Napisan je u C++-u i koristi NSPR (Netscape portable runtime) kako bi se obezbedila nezavisnost od platforme. Firefox koristi Gecko za render-ovanje Web stranica, ali i za renderovanje korisničkog interfejsa. Pored Firefox-a i drugi proizvodi, poput Thunderbird-a, SeaMonkey-a, ali i neke aplikacije koje ne razvija sama Mozilla, koriste Gecko.

4.3 Druge Mozilla aktivnosti

- **Mozilla Persona** je mehanizam za autentifikaciju koji omogućava korisniku da se pomoću istog korisničkog imena i lozinke (ili nekog drugog metoda autentifikacije) na siguran način prijavi na više različitih sajtova. Ovaj sistem je sličan već postojećim rešenjima poput OpenID ili Facebook Connect. Persona se može implementirati pomoću CMS-ova kao što su Drupal, Serdendipity, SPIP i dr. Takođe, postoji i podrška u Phonegap-u. [21]

- **Firefox Marketplace** prvenstveno služi kao portal za preuzimanje HTML5 aplikacija predviđenih za Firefox OS.

- **Webmaker** je deo Mozilline obrazovne inicijative. Osnovan je sa ciljem da se "milionima ljudi pomogne da pređu sa korišćenja Web-a na razvijanje Web-a". Kao deo Mozilla-ine neprofitne misije, Webmaker pomaže da se ljudi "opismene" kada je Web u pitanju, tako što će bolje upoznati Web i preuzeti veću kontrolu nad svojim aktivnostima na mreži.

- **Mozilla Developer Network** (MDN), je zvanični Web portal Mozilla fondacije čiji je primarni cilj sastavljanje dokumentacije vezane za razvoj Web standarda i Mozillinih projekata. Organizovan sistem vođenja dokumentacije je bitan, jer Mozilla održava obimne dokumente vezane za tehnologije poput HTML, CSS, SVG, JavaScript, kao i dokumente za tehnologije koje Mozilla razvija. Pored ovog portala Mozilla objavljuje i veliki broj

video materijala, na Air Mozilla sajtu, vezanih za Web tehnologije i razvoj Mozilla-inih projekata.

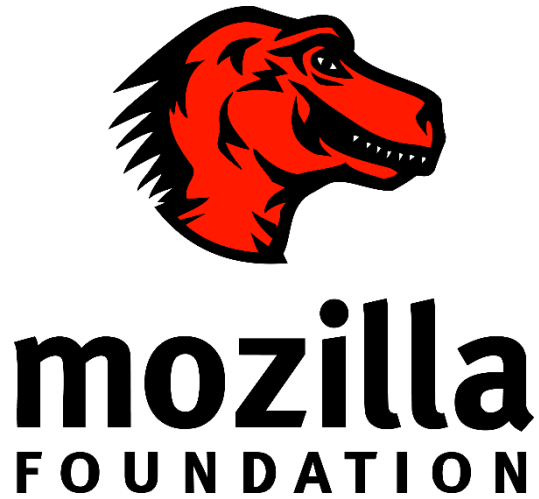
4.4 Zajednica

Mozilla zajednica se sastoji od preko 40.000 aktivnih saradnika širom sveta. Ovaj broj obuhvata i zaposlene i volontere koji rade ka postizanju ciljeva postavljenih u Mozilla Manifestu. Mnogi volonteri su se podelili u pod-zajednice i rade na lokalizaciji, marketingu i promociji Mozilla proizvoda ili kao korisnička podrška.

Mozilla Reps je program osnovan sa ciljem da osposobi i pruži podršku volonterima koji žele da postanu zvanični predstavnici Mozilla-e u svom regionu. Program obezbeđuje okvir i set alata koji pomažu volonterima da organizuju događaje, regrutuju i obuče nove saradnike, dokumentuju aktivnost i pruže bolju podršku lokalnim zajednicama.

Mozilla festival je godišnji događaj tokom kojeg hiljade ljudi istražuju, uče i zajedno rade na poboljšanju Web-a. Festivalu prisustvuju novinari, developeri, dizajneri, predavači, gejmeri i uopšte entuzijasti iz preko 40 zemalja i svi zajedno rade na ukrštanju slobode, Web-a i teme zadate za taj godišnji skup. Ceo događaj je koncentrisan oko dizajnerskih izazova čiji je cilj rešavanje ključnih pitanja na osnovu zadate teme skupa.

MozCamps je bitan deo inicijative pod nazivom *Grow Mozilla initiative* koja za cilj ima rast i razvoj Mozilla zajednice. Cilj ovih kampova je da se pronađu novi saradnici. Sami kampovi su intenzivni višednevni skupovi tokom kojih zvaničnici iz Mozilla-e prezentuju svoja uvodna izlaganja, organizuju se radionice, tematski skupovi (koje vodi plaćeno i volontersko osoblje), kao i mnogi izleti. Sve aktivnosti su predviđene tako da nagrade saradnike za njihov težak rad, angažuju ih oko novih projekata i inicijativa i usmere sve učesnike na Mozilla-inu misiju.



slika 4.1: Logotip Mozilla fondacije



slika 5.2: Mozilla Firefox OS štand na Mobile World Congress 2014.

5 Firefox OS

Firefox OS (poznat i pod razvojnim imenom Boot to Gecko ili B2G) je Mozilla-in operativni sistem otvorenog koda koji se zasniva na Linux-u i Mozilla-inoj Gecko tehnologiji. Ovaj operativni sistem za mobilne uređaje je oslobođen od tehnologija koje pripadaju bilo kojoj kompaniji (*proprietary tehnologije*), dok ujedno pruža moćnu platformu koja omogućava stvaranje ozbiljnih i kvalitetnih aplikacija. Osim toga, sistem je fleksibilan i sposoban da pruži napredno multimedijalno iskustvo krajnjem korisniku.

Najvažniji deo se tiče Web developera, odnosno činjenice da je kompletan korisnički interfejs u stvari Web aplikacija, sposobna da prikaže ili otvori druge Web aplikacije. Sve modifikacije interfejsa koje se naprave i sve aplikacije koje developer napravi tako da rade na Firefox OS-u su Web stranice, ali sa poboljšanim pristupom servisima i hardveru uređaja.[16]

Svakoj kompaniji koja teži da se iz korena promeni treba odati dužno poštovanja. Takav korak nije lak i često je potrebno pretrpeti mnogo kritika i preležati dosta dečijih bolesti, pa ako sve na kraju bude u redu, kompanija više nikada neće biti ista. Nešto slično se dešava na tržištu mobilnih platformi prethodnih šest godina, a počinje da se dešava i na tržištu računara uopšte.

Tako da, kada organizacija poput Mozilla-e, tvoraca veoma popularnog Firefox pregledača, odluči da izađe iz svoje zone komfora i zaželi da napravi samostalni operativni sistem za mobilne platforme, to neće ostati neprimećeno.

Firefox OS je rezultat dugogodišnjeg planiranja, i iako je još uvek u ranim fazama razvoja, zadobio je priličan broj ljubitelja i entuzijasta. U početku se sistem mogao videti, i to samo delimično, samo na svetskim IT sajmovima ili kao port na postojeće Android uređaje (budući da koriste isto Linux jezgro), ali iz dana u dan sve više proizvođača, kao što su ZTE, LG i Alcatel, proširuje ponudu telefona sa ovim operativnim sistemom.

5.1 Istorija Firefox OS-a

25. jula 2011. dr Andreas Gal, *Director of Research* u Mozilla-i, je putem mailing liste na *mozilla.dev.platform* najavio projekat Boot to Gecko. Predlog projekta je bio da se napravi kompletan, nezavisan operativni sistem za otvoreni Web, kako bi se pronašle i prevazišle prepreke koje sprečavaju Web developere da pišu aplikacije koje su po svemu jednake aplikacijama koje se izvorno prave za iOS, Android ili WP. U samom predlogu se navode oblasti koje treba istražiti i na kojima treba raditi. Među prvima se pominju novi Web API-iji koji bi radili sa hardverom uređaja i mogućnostima operativnog sistema, poput poziva ili rada sa kamerom. Zatim se pominju privilegovani modeli putem kojih bi API-ji na bezbedan način predstavljali most između Web stranice, tj. aplikacije i hardvera, kao i kod na niskom nivou koji bi omogućio pokretanje na Android uređajima.[16]

U julu i septembru 2012. Boot to Gecko menja naziv u Firefox OS i analitičari predviđaju da će u 2013., prvoj godini svog zvaničnog pojavljivanja, zauzeti oko 1% ukupnog tržišta pametnih mobilnih uređaja.

U februaru 2013. Mozilla je najavila planove za iskorak na globalno tržište. Na press konferenciji pred početak *Mobile World Congress*-a u Barseloni najavljeno je da će prvi talas Firefox OS uređaja biti dostupan u Brazilu, Kolumbiji, Mađarskoj, Meksiku, Crnoj Gori, Poljskoj, Srbiji, Španiji i Venecueli. [14]



slika 5.1: Jedno od grafičkih rešenja koje se koristi za promociju Firefox OS-a

5.2 Hardver uređaja za Firefox OS

Za potrebe demonstracije i razvoja Firefox OS-a koristi jedan od dva telefona španske kompanije Geeksphone koja su specijalno razvijena za developere aplikacija za Firefox OS, modeli kod nazivima Keon i Peak, zatim Alcatel One Touch Fire i ZTE Open, ali i mnogi Android uređaji i Raspberry Pi.

Većina uređaja sa ARM procesorima ima dovoljno hardverskih resursa za pokretanje port-ovane verzija Firefox OS-a. U softverskom inženjerstvu, porting je proces adaptiranja softvera tako da se izvršni program može napisati i za računarsko okruženje koje se razlikuje od onog za koje je softver prvobitno bio napisan (npr. drugačiji CPU, operativni sistem i sl.). Ovaj termin se takođe koristi i kada se softver ili hardver izmene tako da mogu biti iskorišćeni i u drugim okruženjima. Softver se može portovati onda kada je trošak portovanja na novu platformu manji od troška pisanja softvera ispočetka.

Pred kraj 2013. godine, za kada je i bio zakazan zvaničan izlazak Firefox OS platforme, nekoliko poznatih proizvođača mobilnih uređaja, kao što su LG Electronic, ZTE, Huawei i TCL, prikazalo je svoje modele sa ovim sistemom. Ubrzo zatim prikazan je i prvi tablet sa Firefox OS-om, iz kompanije Foxconn. [14]

Još jedna stvar koja umnogome ide u prilog Mozilla-inom operativnom sistemu je cena. Kako mnogi autori objašnjavaju, Firefox OS nije nastao da bi uklonio iOS, Android ili Windows Phone, već da bi obezbedio jeftiniju alternativu koja će pružiti iste, ako ne i bolje performanse. Mozilla cilja na sasvim drugačije tržište u odnosu na Apple. U najboljem slučaju pokolebaće prodaju iPhone-a u zemljama koje su u razvoju, poput Filipina, Brazila, Srbije itd., kao što to Android čini.

Sa druge strane, Google je mnogo izloženiji. I Google i Mozilla se principijelno zalažu za otvorene platforme i niže cene, sa bitnom razlikom što Firefox OS nudi potpunu otvorenost i dostupnost koda.

Component	Minimum	Recommended
CPU	ARMv6	Cortex A5 class or better ARMv7a with NEON
GPU	—	Adreno 200 class or better
Connectivity	—	WiFi 3G
Sensors	—	Accelerometer Proximity Ambient light A-GPS

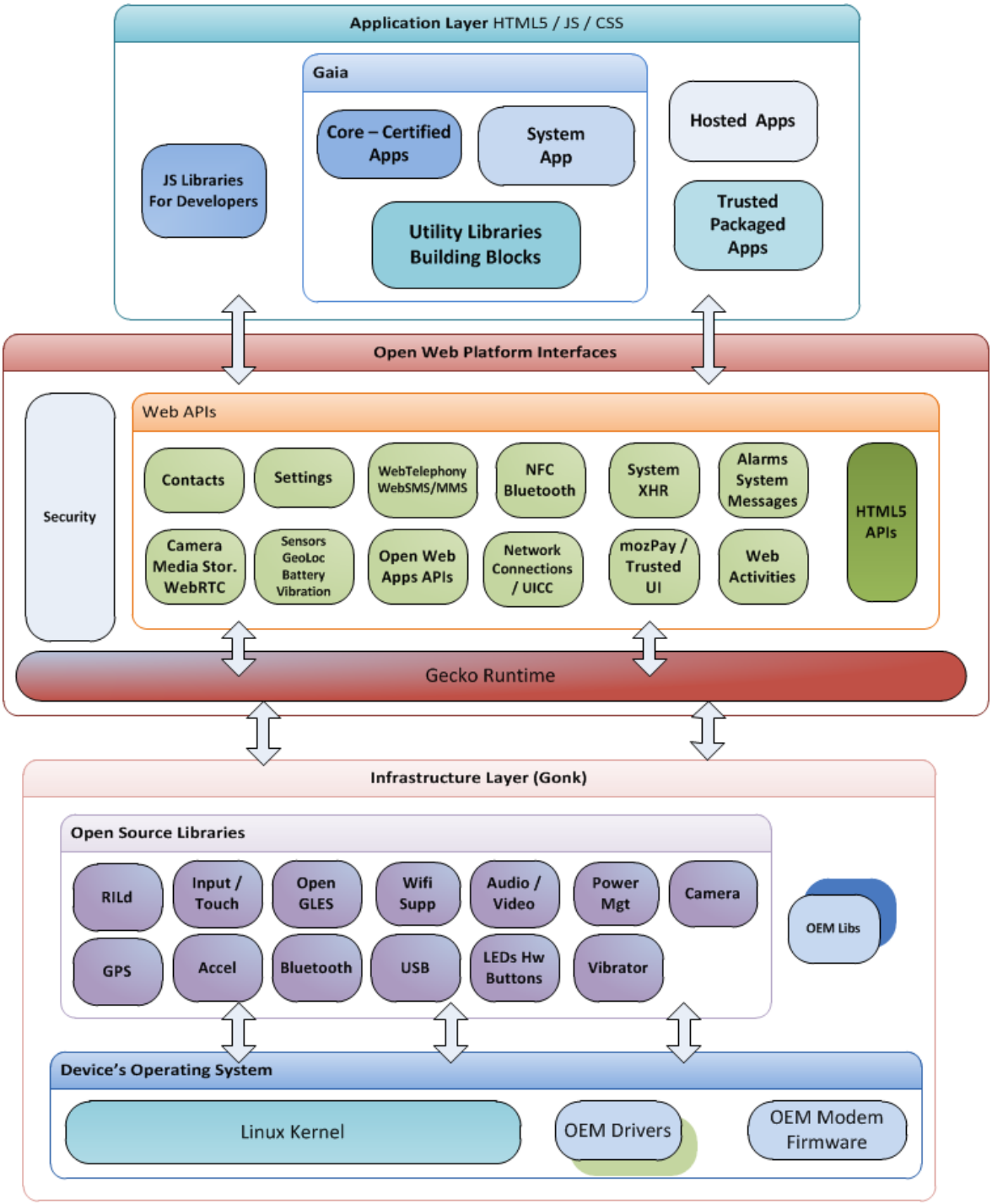
slika 5.2: Tabelarni prikaz minimalnih hardverskih zahteva za Firefox OS platformu



Slika 5.3: Geeksphone modeli Keon i Peak

6 Arhitektura platforme

Arhitektura Firefox OS-a se sastoji od 3 sloja, pod nazivima **Gonk** (najniži sloj u koji su smešteni Linux kernel i HAL), **Gecko** (sloj za obradu otvorenih standarda) i **Gaia** (korisnički interfejs).[18]



slika 6.1: Šematski prikaz arhitekture Firefox OS platforme

6.1 Gonk

Gonk je najniži sloj Firefox OS platforme i sastoji se od Linux jezgra i HAL-a. HAL (Hardware abstraction layer) je set softverskih rutina koje služe za emulaciju određenih detalja koji su usko vezani za neku platformu i daju aplikacijama direktan pristup hardverskim resursima. Na ovaj način programerima je omogućeno da, koristeći standardizovane pozive operativnog sistema, pišu aplikacije visokih performansi koje su nezavisne od uređaja. Najčešće se ovaj proces vrši nad procesorima, jer svaki tip CPU-a ima specifični set instrukcija.

Samo jezgro i više biblioteka korisničkog interfejsa su delovi uobičajenih open source projekata poput Linux-a, libusb, bluez itd. Libusb je biblioteka napisana u programskom jeziku C sa ciljem da se aplikacijama obezbedi pristup USB uređajima na različitim operativnim sistemima. Bluez je Bluetooth stack za Linux čija je uloga implementacija Bluetooth wireless standarda na Linux-u.

Nekoliko delova HAL-a su zajednički sa Android-om. To su moduli za GPS, kameru i dr. Može se reći da je Gonk vrlo jednostavna Linux distribucija. Postoji port Gecko-a za Gonk, kao što postoji i port Gecko-a za Mac OS X, Windows i Android. Kako Firefox OS ima punu kontrolu na Gonk-om, Gecko može da se poveže sa interfejsima sa kojima ne bi mogao na drugim operativnim sistemima. Primera radi, Gecko ima direktan pristup *telephony stack*-u (skupu biblioteka i servisa koje uređaj koristi kako bi upravljao pozivima) i *display frame buffer*-u (najčešće služi za prikaz okvira iz bafera memorije) kao delovima Gonk-a, što na drugim operativnim sistemima nema.[18]

6.2 Gecko

Gecko (razvojno ime Raptor i NGLayout (Next Generation Layout)) je besplatan i open source layout engine koji se koristi u mnogim aplikacijama koje razviju Mozilla fondacija i Mozilla Corporation, kao i mnogim drugim open source projektima. Layout engine je softverska komponenta koja od označenog sadržaja poput HTMLa, XMLa, slika, apleta i sl., i komponenata za formatiranje kao što su CSS, XSL, prezentacioni HMTL itd., prikazuje sadržaj stranice ili aplikacije na ekranu. Layout engine je najčešće ugrađen u Web pretraživače, e-mail klijente, e-book reader-e, sisteme za online pomoć i druge aplikacije koje zahtevaju prikaz i uređivanje Web sadržaja.

Među značajne mogućnosti Gecko-a spadaju: parsiranje različitih tipova dokumenata poput HTML, XML, SVG itd.; mogućnost naprednog rendering-a koje uključuju transformacije i kombinovanje vizuelnih elemenata iz različitih izvora u jednu sliku (*compositing*); kao i podrška za JavaScript i dodatke (*plugins*).

Gecko je toliko brz i moćan da se pored prikaza sadržaja koristi i za prikaz korisničkog interfejsa za pojedine aplikacije. Drugim rečima, Gecko će pored sadržaja dokumenta takođe prikazati i scrollbar, toolbar i menije. Ovo je omogućeno putem XUL-a (XML User Interface Language), Mozilla-inog jezika za izradu korisničkog interfejsa koji je baziran na XML-u. Gecko pruža API bogat opcijama koje ga čine pogodnim za širok spektar aplikacija koje su povezane na Internet.[12]

Iako na prvi pogled podseća na Web pretraživač, jer obezbeđuje komponente potrebne za prikaz sadržaja na ekranu, uključujući i layout engine i dopunski set komponenata pretraživača, Gecko to nije. U paket nisu uključene komponente modula za prikaz interfejsa, za razliku od aplikacija poput Firefox-a. Određene komponente Web

pretraživača, kao što su obeleživač, istorija pregleda i adresar nisu deo Gecko-a, ali se njihov izvorni kod može preuzeti besplatno sa mozilla.org.

Napisan je u programskom jeziku C++, radi na više platformi i operativnih sistema kao što su razne distribucije BSD-a, Linux, OS X, Solaris, OS/2, Windows i dr. Zauzima treće mesto na listi korišćenja layout engine-a, posle Trident-a (koristi se u Internet Explorer-u od verzije 4) i WebKit-a (koristi se u Safari-ju i Google Chrome-u).

Razvoj layout engine-a koje je danas poznat kao Gecko je započet 1997. godine od strane Netscape-a. Postojeći render engine, koji je prvobitno bio napisan za Netscape Navigator 1.0, je smatran inferiornim u odnosu na engine koji je korišćen u Internet Explorer-u. Bio je spor, nije se u potpunosti pridržavao W3C standarda, imao je ograničenu podršku za dinamički HTML i patio od nedostatka inkrementalnog reflow-a (mogućnost layout engine-a da preraspodeli elemente na ekranu kako se nov sadržaj preuzima i dodaje na stranicu). Novi engine je razvijan paralelno sa starim sa idejom da se implementira u Netscape Communicator kada postane dovoljno stabilan.[18]

Kada je početkom 1998. pokrenut projekat Mozilla, novi kôd novog engine-a je objavljen pod open source licencom. Prvobitno ime je bilo Raptor, ali je zbog povrede autorskih prava promenjeno u NGLayout. Razvoj prvog Netscape browser-a koji će koristiti Gecko je potrajao do novembra 2000. godine, jer je veliki deo koda za Netscape Communicator 5.0 morao biti ponovo pisan, dok su neki delovi bili potpuno napušteni. Kako se razvoj nastavio, tako su i druge kompanije, poput AOL, preuzele Gecko kao engine za svoje aplikacije. 15 jula 2003. AOL je otpustio sve developere koji su radili na razvoju Gecko-a. Na taj isti dan je osnovana Mozilla fondacija koja je postala glavni nadzornik za razvoj Gecko-a. Danas na razvoju ovog engine-a rade zaposleni u Mozilla korporaciji, zaposleni u kompanijama koje doprinose Mozilla projektu, kao i volonteri širom sveta.

Od samog početka, Gecko je razvijan tako da podržava otvorene Internet standarde. Neki od standarda koje Gecko podržava su:

- HTML4 (delimična podrška za HTML5);
- CSS 2.1 (delimična podrška za CSS3);
- JavaSript 1.8 ((ECMAScript 3, i delimična podrška za ECMAScript 5)
- DOM nivo 1 i 2 (delimična podrška za DOM 3);
- XML 1.0;
- XHTML 1.0;
- XSLT i XPath;
- MathML;
- XForms;
- RDF;
- SSL;
- Unicode;
- OJI (Open Java Interface);
- Slike u digitalnom formatu PNG, GIF, JPEG i PJPEG;
- SVG (delimična podrška).



Činjenica da Gecko neke od standarda podržava u potpunosti ne znači i da je on 100% bez greške. Kada se uzme u obzir da svaka Web stranica može biti kompleksna i imati proizvoljno mnogo ugnježdenog sadržaja, razumljivo je da se desi da se stranica ne prikaže u svakom pretraživaču baš onako kako je preporučeno u specifikacijama. Sve dok se budu vršila testiranja kvaliteta koda, postojaće i pronalaziće se i greške u Gecko open source bazi koda. Iz toga sledi da će svaki komercijalni softver koji se isporučuje sa Gecko-om imati poznate greške već i u samom trenutku objavljivanja. Ovaj princip, naravno, ne važi samo za Gecko već i za svaki drugi layout engine i softver koji se na njemu bazira. Svaka greška koja se ikada pojavila u Gecko-u je detaljno dokumentovana uz pomoć Bugzilla-e.

Kako bi prikazao stranice koje su dizajnirane za starije verzije Netscape-a i Internet Explorer-a, Gecko se koristi tehnikom DOCTYPE switching. Dokumenti sa savremenim DOCTYPE-om se prikazuju u modu koji se saglasan sa standardima i striktno prati W3C standarde. Dokumenti koji nemaju DOCTYPE ili koriste stariju verziju se render-uju u *quirks* modu. Taj mod se koristi kod nekih Web pretraživača kako bi se obezbedila povratna kompatibilnost stranicama koje su pravljene ili za starije pretraživače ili su pisane po standardima koji su zastareli.

Primarna upotreba Gecko-a je u Web pretraživačima. Prvi koji su ga koristili su Netscape 6 i Mozilla Application Suite (SeaMonkey). Takođe se koristi i u drugim pretraživačima koji su derivati originalnog Mozilla-inog pretraživača, kao što su Firefox, Camino, Flock, K-Meleon, kao i u implementaciji Internet Explorer-a koji je deo Wine-a. Linux verzija Google-ovog softvera za organizaciju slika Picasa, DevHelp, pretraživač dokumentacije za API-je za GTK+/GNOME, Thunderbird, Portable Firefox i HP Secure Web Browser su samo neki od proizvoda koji koriste Gecko za render-ovanje dokumenata.

Mnogi Web developer-i su opterećeni višestrukim razvojem i testiranjem svake Web stranice ili aplikacije kako bi ona bila u skladu sa različitim i često nekompatibilnim DOM-ovima različitih pregledača. Od proizvođača pregledača se zahteva potpuna podrška za otvorene standarde kako bi developer-i bili u mogućnosti da koriste moćan sistem za formatiranje i jedinstven DOM, tako da se kôd koji jednom napišu svuda prikaže isto (“write once, view anywhere”). Gecko-ova robusna podrška za ove standarde ga čini dobrim izborom kao platforme za razvoj Web aplikacija i sadržaja za Web stanice.

Gecko se može koristiti na skoro svakoj poznatoj platformi zahvaljujući XPCOM-u (Cross Platform Component Object Model), podskupom COM-a. COM je od firme Digital preuzeo Microsoft i on je *de facto* standard za modularne interfejse na Windows platformama. Pored toga, na Windows platformama, Gecko je uokviren ActiveX control-om, tako da ga Visual Basic developer-i mogu koristiti. ActiveX okviri nisu dostupni za druge platforme, jer je to tehnologija koja funkcioniše samo pod Windows-om. JavaBean okvir se trenutno ne razvija, ali arhitektura Gecko-a je takva da ne postoji ništa što bi u budućnosti kočilo takav razvoj.[18]

Gecko-ov XPCOM interfejs se razlikuje od onog koji koristi Microsoft. Najveća razlika leži u tretiranju DOM-a putem interfejsa. Interfejsi Microsoft-ovog Trident-a pristupaju DOM-u putem sopstvenog API-ja, dok Gecko koristi standarde preporučene od strane W3C-a.

Kao što je već napomenuto, Gecko radi na mnogim poznatim platformama. Među njima su Win32 (Windows XP Service Pack 2 i 3, Windows Vista i Windows 7), Mac OS X 10.5 i noviji i Linux. Saradnici, kao i mnogi proizvođači i prodavci hardvera koji

pripadaju Mozilla-inoj mreži putem mozilla.org rade na port-ovanju Gecko-a za druge platforme. U te platfome se ubrajaju Solaris, HP/UX, Irix, AIX, OS/2, OpenVMS, BeOS i Amiga. Za starije verzije Windows-a o Mac OS X-a se koriste starije verzije Gecko-a.

6.2.1 Kritike i prihvaćenost Gecko layout engine-a

Gecko je u početku teško prihvatan i imao je spor uspon na tržištu usled komplikovanosti koda. Kao što je rečeno, Gecko je od početka bio zamišljen kao nešto više od pukog HTML renderer-a za Web pretraživače. Pored te osnovne funkcije, Gecko pruža i mnogostrani okvir za generisanje korisničkog interfejsa, XUL. [21]

Još jedan od razloga zbog kojih je Geckov kôd obiman leži i u upotrebi XPCOM-a. XPCOM je nadplatformski model komponenata uz pomoć kojeg se gotovo svaka funkcionalnost Gecko-a dostupna kao serija komponenata ili višestruko iskoristiva kao biblioteka na drugim platformama. Svim komponentama se može pristupiti putem Web pretraživača ili skriptom iz bilo koje Mozilla aplikacije.

Na Windows i sličnim platformama, Gecko zavisi od kompajlera koji nisu besplatni. Stoga, FFOS (Free and open-source software) distribucije Linux-a često ne uključuju Gecko pakete koji se koriste u Wine-u, sloju za kompatibilnost sa Windows-om.

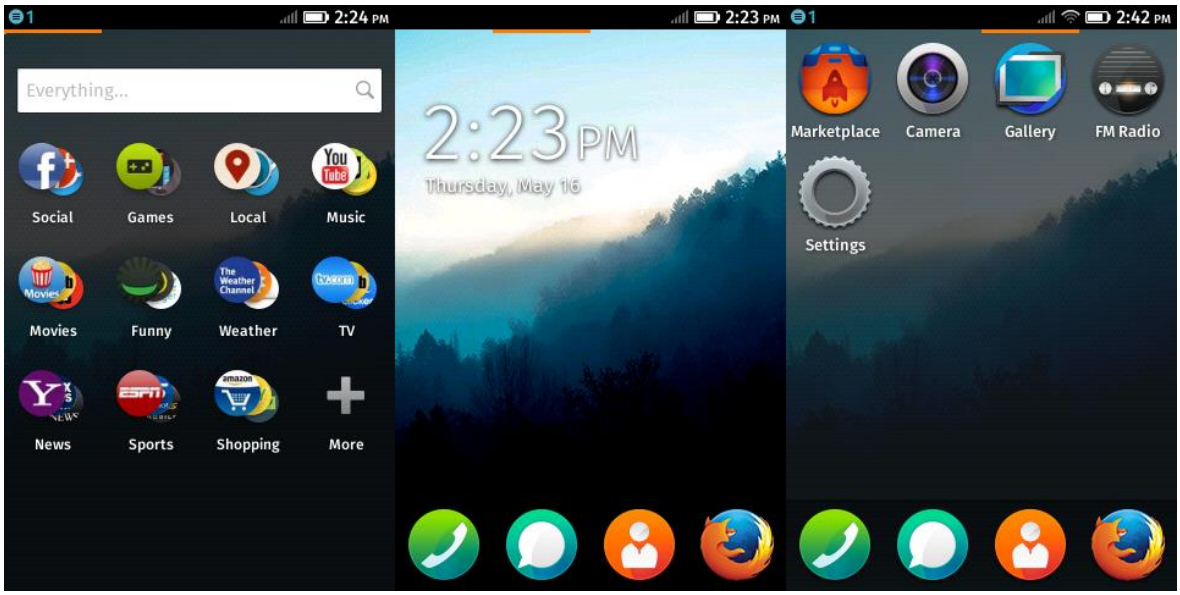
6.3 Gaia

Gaia je treći sloj Firefox OS-a, odnosno korisnički interfejs. U pitanju je set Web aplikacija koje se pokreću lokalno na mobilnom uređaju, emulator-u ili desktop-u. Interfejs je napisan isključivo korišćenjem Web tehnologija kao što su HTML, CSS i JavaScript. Sve što se vidi na ekranu pošto se Firefox OS startuje, uključujući Lock screen, Home screen, Dialer i mnoge druge aplikacije, je prikazano uz pomoć Gaia-e. Sve što je potrebno da se zna kako bi se unapredio ili izmenio ovaj sloj operativnog sistema su gorepomenute Web tehnologije. Kako bi komunicirala sa drugim slojevima operativnog sistema i hardverom uređaja Gaia koristi Web API-je koji su implementirani u Gecko.[21]

Zahvaljujući svom dizajnu Gaia nije vezana samo za Firefox OS uređaje, već se može pokrenuti na bilo kom operativnom sistemu i u drugim Web pregledačima. U drugim pregledačima potencijalno preti opasnost da se prikaže sa manjkom funkcionalnosti u zavisnosti od mogućnosti samog pretraživača.



slika 6.2: Koncept korisničkog interfejsa



slika 6.3: Početni ekran i meniji

Interfejs Firefox OS-a se nekoliko puta menjao tokom razvoja. Glavni ekran izgleda uobičajeno za jedan pametni telefon. Statusna linija na vrhu ekrana prikazuje ime mreže na kojoj se telefon nalazi, jačinu signala mreže ili WiFi, nivo baterije i trenutno vreme.

Neki elementi interfejsa su rešeni poput onih koji se koriste kod dva najveća takmaca na tržištu pametnih telefona, Android-a i iOS-a. Tako se, na primer, primećuje nedostatak dugmeta sa funkcijom za vraćanje unazad i ulaska u glavni meni, što su odlike iOS-a. Sa druge strane notifikacijama o propuštenim pozivima, porukama, update-ovima koji su se pojavili i sl. se pristupa spuštanjem notifikacione trake, dok se prelazak iz jedne aktivne aplikacije u drugu vrši kroz meni koji postaje dostupan dugim držanjem Home dugmeta, što je rešenje koje se može videti i na Android-u.[21]

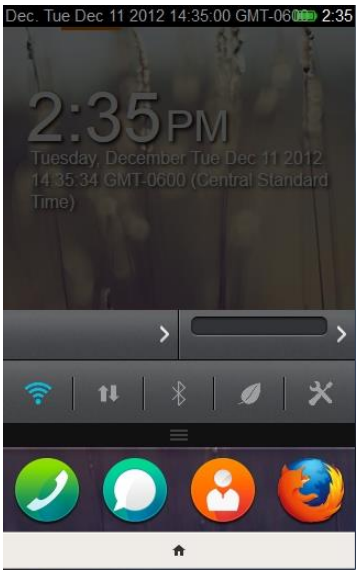
Kada se korisnik prebaci na levo od početne strane, otvara se Search page, odnosno stranica za pretragu, koja predstavlja najveću razliku između Firefox OS-a i konkurencije. Pretraga je zamišljena i ostvarena tako da bude jedan od najvažnijih aspekata korišćenja i generalnog iskustva koje ovaj operativni sistem pruža.

Na vrhu se nalazi *Searh bar* (prazno polje za pretragu) u koji korisnici upisuju upite. Na dnu su smešteni redovi u kojima su stavke raspoređene po kategorijama kao što su Muzika, Kupovina, Finansijske vesti, Društvene mreže itd. Ove kategorije se mogu proširiti pritiskom na ikonicu More.

Po odabiru, primera radi kategorije Društvene mreže, prikazaće se Web aplikacije iz date kategorije. Trenutno, najatraktivnije aplikacije ovde su Facebook, Twitter, Google Plus i Tumblr. Korisnik sada može da otvori željenu aplikaciju ili da je doda na neku od stranica korisničkog interfejsa kako bi je koristio i kasnije.

Ukoliko se pojam pretražuje bez odabira kategorija prikazaće se mnoštvo izbora od kojih korisnik može odabrati onaj koji mu je potreban. Na primer, pretraga pojma “The Beatles” će prikazati Grooveshark, YouTube, SoundClound, ali će takođe ponuditi i

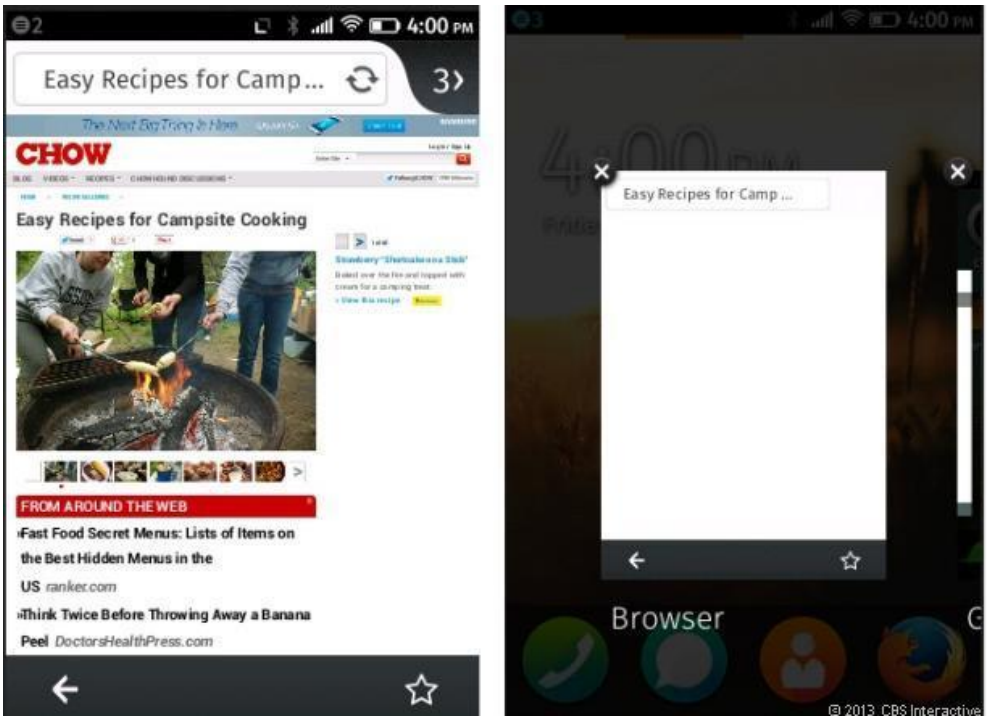
Wikipedia-ju, Instagram, Facebook i Twitter. Ovakav način pretrage aplikacija je veoma moćan i koristan i predstavlja jednu od najboljih odlika Firefox OS-a. [20]



slika 6.4: Meni sa notifikacijama

Središnji deo ekrana prikazuje ikonice za aplikacije. Za razliku od Androida, kao što je već napomenuto, ne postoji glavni meni, već se aplikacijama pristupa odabirom jednog od ekrana koji se nalaze levo ili desno od glavnog.

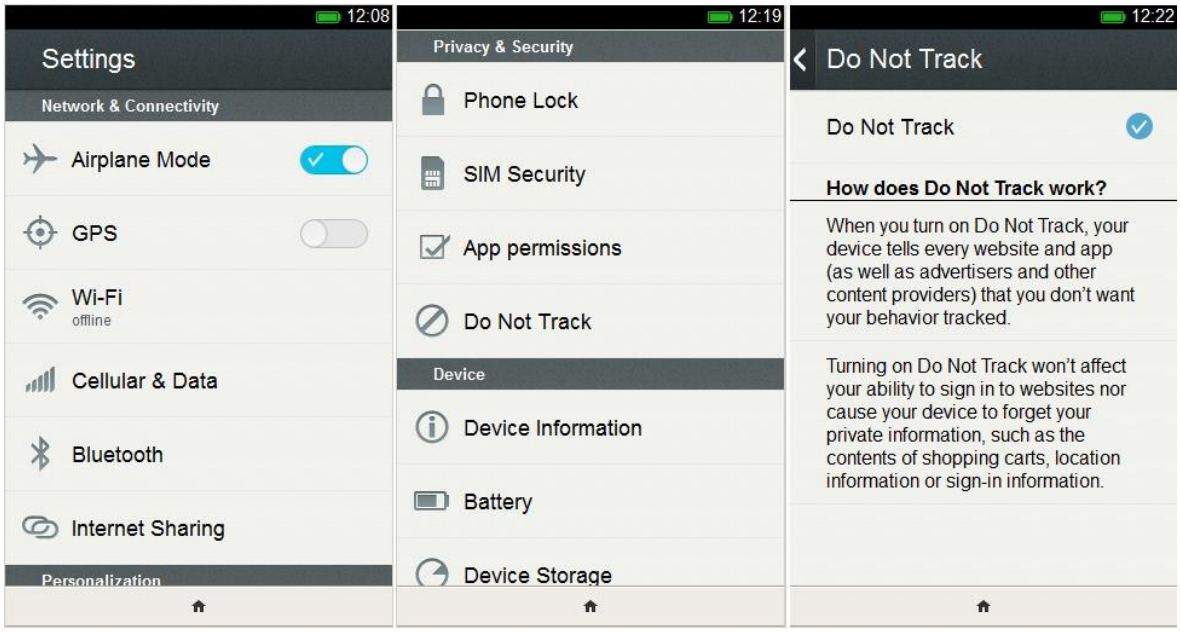
Na dnu ekrana se nalazi *dock* u koji se može smestiti do sedam najčešće korišćenih aplikacija. [21]



Slika 6.5: Prelazak iz jedne aktivne aplikacije u drugu

Podešavanja mobilnog uređaja se ne razlikuju mnogo od onih na koja su korisnici pametnih telefona već navikli. Pristupa im se odabirom ikonice *Settings* i podeljena su na kategorije kao što su na primer podešavanja mreže i povezivanja (Network & Connectivity), podešavanje izgleda (Personalization), privatnost i bezbednost (Privacy & Security) i mnoge druge. Pored ovih podešavanja u odeljku *Device* se mogu proveriti informacije o samom uređaju, kao što su verzija softvera i jezgra sistema, postojanje eventualnih ažurnijih verzija delova sistema, količine zauzetog i slobodnog prostora na memorijskim medijima, kao i nivo baterije.

Ono što Firefox OS odvaja od drugih operativnih sistema za mobilne uređaje je opcija *Do Not Track*, ista koja postoji i za Firefox browser. Ova opcija radi na principu zabrane praćenja aktivnosti korisnika na nekom sajtu ili kroz Web aplikaciju. Uključivanje opcije samo povećava sigurnost i obezbeđuje veća privatnost na Web-u, dok ne utiče na mogućnost prijavljivanja na sajtove niti briše lične podatke poput sadržaja potrošačke korpe, informacija o lokaciji ili sačuvanih korisničkih imena i lozinki.



Slika 6.6: Podešavanja uređaja

7 WebAPI

Boot to Gecko projekat je u potpunosti baziran na otvorenim standardima. To praktično znači da je kôd otvoren i dostupan svakome. Ukoliko neki standard ne postoji, Mozilla predlaže da se takav standard stvori, a zatim samostalno ili zajedničkim snagama sa drugim kompanijama i zvaničnim telima za standarde, poput W3C, radi na njihovom razvoju.

Mozilla nema tu snagu, niti nameru da nameće svoje standarde. Takav odnos bi se direktno kosio sa misijom Mozilla-e. Cilj te misije je da svi koji koriste Web, od proizvođača, preko developera do krajnjih korisnika imaju slobodu da učestvuju i unapređuju otvoreni Web. Kako zvanični stav Mozilla-e glasi: "Nije bitno ko će stvoriti standard, već je bitno ko će ga bolje implementirati.". Ukoliko neki standard već postoji i zadovoljava potrebe on će biti prihvaćen i korišćen od strane Mozilla-e.

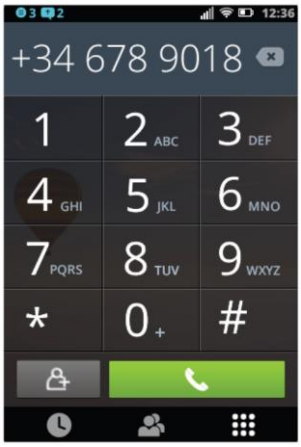
Web kao okruženje se veoma razlikuje od desktop okruženja. Dok se kod desktopa retko dešava da se koristi softver koji nije podrobno ispitan, na Web-u se koriste tehnologije koje nisu u potpunosti implementirane ili ih različite kompanije drugačije implementiraju. Dobar primer za ovu tvrdnju je CSS koji postoji već duži niz godina, ali ni u jednom pretraživaču nije potpuno implementiran. Zbog toga, ukoliko predlažu neki novi ili dodaju izmene na postojeći standard, kompanije ispred naziva standarda stavljaju određeni prefiks (moz u slučaju Mozille). Ukoliko se standard prihvati, a smatra se prihvaćenim ukoliko se većinskim delom implementira dok svih proizvođača, prefiks se ukida.

Neki od standarda koje je Mozilla predložila:

- WebTelephony;
- Vibration API;
- WebSMS;
- Idle API;
- Screen Orientation;
- Settings API;
- Resource lock API;
- Power Management API;
- Mobile Connection API;
- Socket API;
- Sensor API
- WiFi Information API;
- Device Storage API;
- Contacts API;
- Camera API;
- Peer to Peer API;
- Open WebApps;
- WebNFC;
- WebBluetooth;
- WebUSB;
- Network Information API;
- Battery Status API;
- Alarm API;
- Browser API:
- Time/Clock API;
- Intents/Activities/Actions;
- Keyboard/IME API.

7.1 WebTelephony

Cilj WebTelephony standarda je da se uspostavi DOM API koji bi omogućio sadržaju sa Web stranice da pozove određeni broj, odgovori ili prekine poziv, stavi pozivaoca na čekanje ili nastavi poziv. Ovaj standard se kod W3C vodi kao projekat u razvoju i još uvek nije prihvaćen.



slika 7.1: Ekran za pozivanje

7.1.1 Implementacije WebTelephony standarda

```
// Telephony object
var tel = navigator.mozTelephony;

// Check if the phone is muted (read/write property)
console.log(tel.muted);

// Check if the speaker is enabled (read/write property)
console.log(tel.speakerEnabled);
```

slika 7.2: Inicijalizovanje Telephony objekta

```
// Place a call
var cal = tel.dial("123456789");
```

slika 7.3: Hardkodovan poziv

```
// Receiving a call
tel.onincoming = function (event) {
  var incomingCall = event.call;

  // Get the number of the incoming call
  console.log(incomingCall.number);

  // Answer the call
  incomingCall.answer();
};

// Disconnect a call
call.hangUp();


// Iterating over calls, and taking action depending on their changed status
tel.oncallschanged = function (event) {
  tel.calls.forEach(function (call) {
    // Log the state of each call
    console.log(call.state);
  });
};
```

slika 7.4: Upravljanje pozivima

7.2 WebSMS

Jedna od osnovnih funkcionalnosti mobilnog telefona je slanje i primanje SMS poruka, a to je u Firefox OS-u postignuto putem WebSMS API-ja. Ovaj API omogućava Web sadržaju da kreira, šalje i prima SMS poruke. API je dostupan korišćenjem window.navigator.mozSms, a vraća objekat SmsManager. WebSMS još uvek nije standardizovan, ali je u procesu provere od strane W3C.

Kako je predviđen za slanje SMS poruka, ovaj API je primarno podržan od strane mobilnih pretraživača.

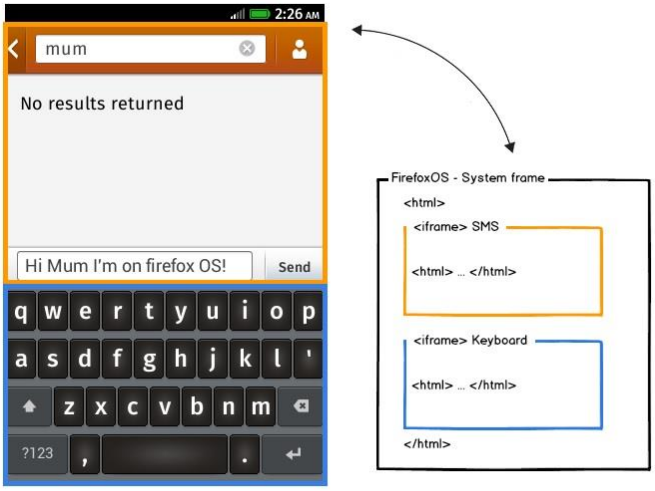
```
// SMS object
var sms = navigator.mozSMS;

// Send a message
sms.send("123456789", "Hello world!");
```

slika 7.5: Inicijalizovanje WebSMS objekta

```
// Recieve a message
sms.onreceived = function (event) {
   // Read message
   console.log(event.message);
};
```

slika 7.6: Primanje i čitanje poruke



slika 7.7: Arhitektura Web aplikacije za slanje SMS-ova

7.3 Battery Status API

Battery Status API, ili kako se češće naziva – Battery API, obezbeđuje informacije o nivou baterije uređaja i prikazuje obaveštenja o događajima koja se šalju kada se izmeni nivo baterije. Ovim putem se može podesiti nivo iskorišćavanja resursa za datu aplikaciju kako bi se smanjila potrošnja kada je nivo baterije nizak ili kako bi se sačuvale izmene na dokumentima, kako bi se izbegla mogućnost gubljenja podataka, pre nego što se baterija isprazni.

Battery API osobinom (property-jem) window.navigator.battery, koja je objekat BatteryManager-a, proširuje window.navigator i dodaje nove događaje koji se mogu prikazati kako bi se pratio status baterije.

7.3.1 BatteryManager.level

Pokazuje nivo baterije kao vrednosti od 0.0 do 1.0.

```
var time = navigator.battery.dischargingTime
```

Povratna vrednost promenljive time je broj koji predstavlja nivo baterije uređaja skaliran na vrednosti između 0.0 do 1.0. Vrednost 0 znači da je baterija prazna i da će se sistem uskoro ugasiti. Vrednost 1.0 predstavlja punu bateriju, ali takođe predstavlja

povratnu vrednost u slučaju da implementacije nije u mogućnosti da utvrdi nivo baterije ili je u pitanju sistem koji se ne napaja putem baterije.

```
var battery = navigator.mozBattery
if (battery) {
      var batteryLevel = Math.round(battery.level * 100) + "%",
        charging = (battery.charging)? "" : "not ",
        chargingTime = parseInt(battery.chargingTime / 60, 10,
        dischargingTime = parseInt(battery.dischargingTime / 60, 10);

   // Set events
   battery.addEventListener("levelchange", setStatus, false);
   battery.addEventListener("chargingchange", setStatus, false);
   battery.addEventListener("chargingtimechange", setStatus, false);
   battery.addEventListener("dischargingtimechange", setStatus, false);
}
```

slika 7.8: Battery Status API

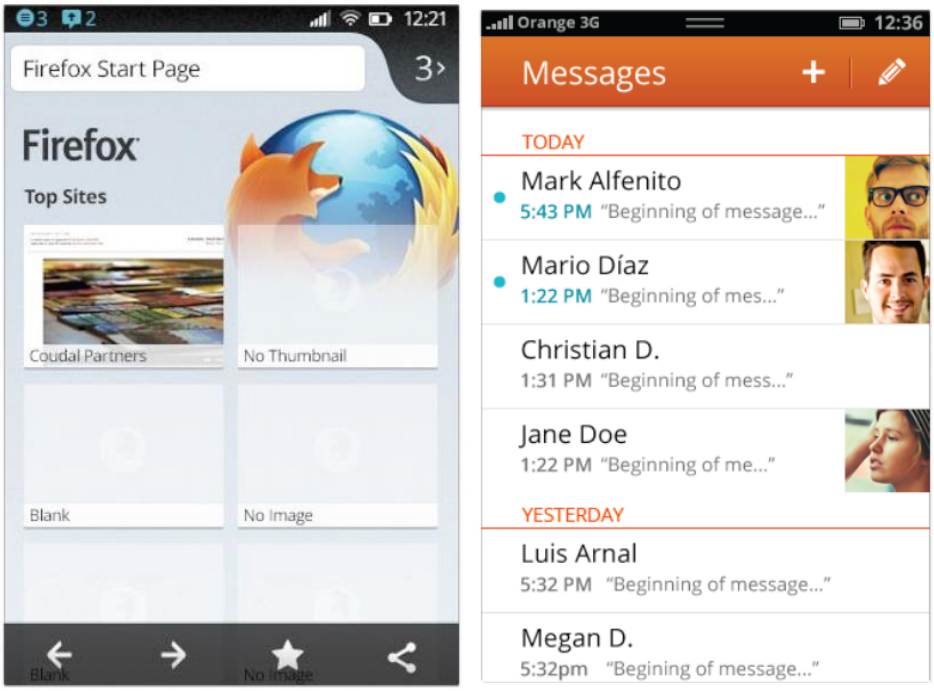
8 Mozilla Web aplikacije za Firefox OS

Firefox OS, već i u radnim verzijama, dolazi popunjen aplikacijama i opcijama koje se mogu očekivati od pametnog telefona. Ono što je interesantno i što Mozillin operativni sistem razlikuje od drugih je što se u podrazumevanim sistemskim aplikacijama dobijaju Video player, Radio, Usage aplikacija za proveru količine razmenjenih podataka, Nokia HERE Maps i Accuweather.

Kao što je već više puta napomenuto, u pitanju su isključivo Web aplikacije, od kojih se većina izvršava lokalno.

U te aplikacije, između ostalih, spadaju:

- Dialer;
- Contacts;
- Settings;
- SMS;
- Web browser (Firefox);
- Gallery;
- Music player;
- E-mail (POP);
- Calendar;
- Alarm clock;
- Camera;
- Notes;
- Notifications;
- Home Screen;
- Mozilla Marketplace;
- System Updater;
- Localization Support.



slika 7.9: Prozori Firefox Web browser-a i Messaging aplikacije



slika 7.10: Prozori Calendar i Dialer aplikacija

Web aplikacije su postale gotovo opsesija za neke IT kompanije u poslednjih nekoliko godina. Dešava se i da se neke od njih priključe trendu, ali i da se vrlo brzo povuku iz toka. Web aplikacije koje se razvijaju uz pomoć HTML5 i otvorenih Web standarda sa sobom nose magično obećanje da je moguće iskodirati nešto samo jedanput a zatim pokrenuti na bilo kojem uređaju koji poseduje internet konekciju, bez obzira na proizvođača, operativni sistem ili verziju itd.

Nažalost, zbog osnovne pretpostavke da bi ove aplikacije trebalo da rade na svakom uređaju i zbog njihove oslanjanja na HTML umesto na nativni kôd, Web aplikacije ispadaju daleko inferiornije u odnosu na nativne aplikacije, naročito sa stanovišta korisnika.

Kako je operativni sistem trenutno zamišljen, korisnici se uglavnom oslanjaju na Web alikacije kako bi završili neki zadatak. Bez obzira da li su instalirane putem Marketplace-a ili putem Search panela, aplikacije koje Firefox OS koristi su "lažne", odnosno mahom hibridne.

Većina ovih aplikacije su prosto omotači oko mobilne verzije sajta neke druge kompanije. Primera radi, Facebook aplikacija je samo korak ispred otvaranja m.facebook.com adrese. Isto važi i za Twitter aplikaciju. Napredak leži u činjenici da su mobilne verzije sajtova na ovaj način optimizovane za konkretan uređaj na kojem se otvaraju.

8.1 Sigurna implementacija sistema i ažuriranje sistema i aplikacija

Firefox OS dolazi instaliran na uređaju. Originalne instalacije sistema su kreirane od strane poznatog, pouzdanog izvora, najčešće proizvođača uređaja, koji je odgovoran za razvoj, testiranje i digitalno potpisivanje distribucionih paketa.

Sigurnosne mere se koriste kroz celokupan skup tehnologija. Privilegije koje se tiču file system-a se primenjuju pomoću *Linux access control lists* (Listi za kontrolu pristupa pod Linux-om). Sistemske aplikacije su instalirane na medij nad kojim je dozvoljen samo upis (read-only), osim tokom ažuriranja kada je privremeno dozvoljen i upis i ispis (read-write). Samo delovi skladišne memorije na kojima se nalaze korisnički sadržaji smeju da budu read-write.

Mnoge komponente u samom hardveru uređaja imaju ugrađene mehanizme zaštite. Na primer, proizvođači čipova koriste razne tehnike kako bi smanjili ranjivosti. Osnove Firefox platforme, Gecko i Gonk, su takođe ojačani posebnim tehnikama kako bi se odbranili od potencijalnih pretnji.

Operativni sistemi mobilnih uređaja u velikom broju slučajeva do krajnjeg korisnika dolaze sa neažurnim verzijama sistema i aplikacija i samim tim su podložniji napadima. Svi novi update-i i sigurnosne zakrpe za Firefox OS platformu se prosleđuje kroz bezbedni proces kojim Mozilla garantuje integritet instalacije sistema.

Ažuriranje sistema može da zahvati sve ili samo određene delove Firefox OS arhitekture. Ukoliko se ažurira neki deo Gonk-a, što će se desiti samo u slučaju da se pojave neke kritične sigurnosne zakrpe ili veliki pomak ka stabilnosti i sigurnosti jezgra, jer Mozilla ne radi na razvoju Linux jezgra, koristi se FOTA (Firmware Over the Air) sistem za ažuriranje. Uz pomoć FOTA sistema je moguće ažurirati i druge delove Firefox OS-a, uključujući upravljanje uređajem (firmware/driver-i), podešavanja Firefox OS-a, sigurnosne zakrpe, kao i Gaia i Gecko nivoe.

Ažuriranja koja ne uključuju Gonk mogu se vršiti putem Mozilla System Update Utility-ja. Mozilla za Firefox OS koristi potpuno isti okvir za ažuriranje, procese i Mozilla Archive (MAR), format koji se koristi za pakete za ažuriranje, koji koristi i za desktop varijantu Firefox browser-a.

Gecko engine je relativno mali i zauzima oko 8MB. Mozilla će ažurirati Gecko kada se otkriju sigurnosni propusti. Za slučaj da ažuriranje iz nekog razloga ne uspe, na uređaju se uvek čuva pomoćna verzija engine-a kojoj se pribegava.

Korisnički interfejs je host-ovan u cloud-u, na poverljivom SSL domenu, a lokalno se samo kešira. Ovo praktično znači da do ažuriranja dolazi u realnom vremenu. Na uređaju se keš ažurira transparentno.

Na svakom uređaju postoji ugrađeni servis koji periodično proverava da li je dostupna nova verzija sistema. Kada sistemski paket postane dostupan i servis za ažuriranje ga prihvati, od korisnika se traži da potvrdi instalaciju. Pre nego što se update instalira na mobilni uređaj prvo se proverava da li na uređaju postoji dovoljna količina slobodnog prostora, a zatim se kod distribucije verifikuje:

- poreklo paketa (proveravaju se izvorna lokacija, domen, port i manifest);
- integritet datoteke (SHA-256 hash check);
- digitalni potpis (provera sertifikata).

Sledeće sigurnosne mere se koriste tokom procesa ažuriranja:

- Mozilla preporučuje da se ažuriranje vrši putem SSL konekcije;
- Zahteva se jaka kriptografska verifikacija pre instaliranja firmware paketa;
- Kompletan paket update-a se mora preuzeti i smestiti na sigurnu lokaciju pre nego što ažuriranje počne;
- Sistem mora biti u stanju sigurnosti pre nego što ažuriranje počne. To znači da nijedna Web aplikacija ne sme biti upaljena;
- Kriptološki ključevi moraju biti smešteni na sigurnu lokaciju na samom uređaju.

8.2 Bezbednost aplikacija

Firefox OS ograničava i primenjuje okvir resursa koji će biti dostupni aplikaciji, dok sa druge strane podržava veliki broj aplikacija sa različitim nivoima dozvola.

Firefox OS koristi strategiju dubinske bezbednosti kako bi se uređaj zaštitio od napada ili malicioznih aplikacija. Ova strategija podrazumeva raznovrsne mehanizme koji uključuju robusne modele dozvola koji se baziraju na *Application Trust* modelu, zatim modele po kojima samo verifikovani API-ji imaju pristup niskim slojevima, kao što je hardver uređaja, te bezbedne procese instalacije i ažuriranja softvera, kao i pokretanje aplikacija po sistemu sandbox-a. Sandbox je poseban sigurnosni mehanizam za odvajanje pokrenutih programa. Najčešće se koristi za pokretanje nepoverljivog koda, programa koji potiču od neproverenih proizvođača ili Web sajtova.[19]

Kako su Firefox OS aplikacije Web aplikacije i ne postoje aplikacije pisane samo za tu platformu, koje bi korisnik instalirao, svi sistemski pristupi se posreduju striktno putem WebAPI-ja. Čak je i pristup file system-u kontrolisan kroz WebAPI i back-end SQLite bazu podataka. Ovo, dakle, znači da ne postoji direktan pristup iz aplikacije do datoteka koje se nalaze sačuvane na SD kartici ili drugom memorijskom mediju.

Tip aplikacije	Nivo poverenja	Opis
Sertifikovana (Certified)	Visok	Sistemske aplikacije koje je odobrio operater ili proizvođač. Ovaj nivo je rezervisa za mali broj sistemskih aplikacija, poput aplikacija za SMS, Bluetooth, kameru i telephony, i nije dostupan svim developer-ima.
Privilegovana (Privileged)	Pouzdana	Aplikacije koje su razvijene od treće strane a proverene su, odobrene i digitalno potpisane od strane ovlašćenog Marketplace-a.
Web (sve ostale)	Nepouzdana	Podrazumeva instalirane, kao i online aplikacije sa uobičajenim Web sadržajem. Kako lokalne tako i host-ovane (samo manifest aplikacije se nalazi na uređaju).

tabela 8.1: Prikaz tipova aplikacija po nivoima privilegija

Nivo poverenja određuje nivo pristupa funkcionalnostima uređaja. Kod Firefox OS-a se primenjuje princip najmanje dozvole. To znači da se, za početak, aplikaciji daje apsolutni minimum dozvola, a zatim selektivno dodeljuju dodatne privilegije samo kako je to neophodno i razumno. U početku aplikacija ima najniži nivo dozvola, što je izjednačava sa nepouzdanim Web sadržajem. Ukoliko aplikacija putem API-ja zahteva dodatne dozvole, one moraju biti taksativno nabrojane u manifestu aplikacije. Gecko će razmotriti davanje privilegije pristupa Web API-ju samo ukoliko su te privilegije eksplicitno nabrojane u manifestu. Gecko će takođe proveriti i da li aplikacija pripada tipu koji bi smeo da ima zatraženi pristup.[19]

9 Razvoj aplikacija za Firefox OS

Kao što je više puta napomenuto, aplikacije za ovu platformu su Web aplikacije i za njihov razvoj je potrebno poznavanje nekih od Web tehnologija kao što su HTML, CSS, JavaScript, Ajax i dr. Minimum koraka koji je potreban kako bi Web aplikacija postala Firefox OS aplikacija, je:

- razviti aplikaciju korišćenjem HTML, CSS i JavaScript tehnologija;
- kreirati manifest dokument;
- publikovati ili instalirati aplikaciju.[17]

9.1 Web aplikacija

Kroz gorepomenuta tri koraka stvara se Web aplikacija koja se može instalirati na uređaj. Strogo gledano, ovakva Web aplikacija je mnogo više od pukog dodavanja manifesta na Web sajt ili Web aplikaciju. Tehnologije koje se koriste u otvorenim Web

standardima se mogu posmatrati kao kompletna platforma koja koristi engine Web pregledača za prikaz korisničkog interfejsa i interpretaciju koda, a koristi Web protokole za komunikaciju sa serverom.[21]

Nekoliko pitanja treba uzeti u obzir pre početka razvoja. A to su, na primer:

- da li aplikacija treba da bude povezana sa Internetom;
- da li aplikacija treba da koristi podatke i kako i gde će oni biti čuvani;
- da li performanse aplikacije mogu da imaju koristi od naprednih mogućnosti platforme, poput WebSockets.

Kako bi se potpuno iskoristile mogućnosti aplikacija, sami developeri mogu dosta da učine sa svoje strane. Tako, na primer:

- preporučuje se upotreba *responsive* Web dizajna, kako bi aplikacije mogle da se koriste na velikom broju različitih uređaja;
- moguće je omogućiti *offline caching*, tako da se podacima može pristupati i bez konekcije na Internet;
- podatke je moguće čuvati lokalno korišćenjem DOM-ova IndexedDB ili localStorage;
- aplikacija se može razviti tako da se pokreće pomoću ikonice iz menija ili glavnog ekrana;
- mogu se koristiti API-ji za interakciju sa delovima hardvera, poput geo lokacije i orijentacije ekrana;
- dobro je dati korisnicima mogućnost da daju povratne informacije razvojnom timu.

Firefox OS ne primorava developera da kreira decentralizovane aplikacije, ali takođe ni na koji način ga ne sprečava da to uradi. Kao što je pomenuto, trenutno postoje tri glavne kategorije aplikacija za Firefox OS. Ovaj operativni sistem je prvi koji dozvoljava instalaciju aplikacija iz bilo kog izvora. Firefox Marketplace je samo referenca implementacije portala za distribuciju aplikacija, ali nije centralni reper za develepere. Oni mogu da odaberu brojne načine za distribuciju svojih aplikacija. Na primer, mogu sami da host-uju aplikaciju na Internetu ili je distribuiraju putem zasebnog Marketplace-a. Kako je Firefox Marketplace potpuno otvorenog koda svako ko je zainteresovan da vodi svoj Marketplace može slobodno da ga prilagodi svojim potrebama, ukoliko, recimo, smatra da aplikacije treba implementirati prateći određene smernice i tako ih decentralizovati.

Jedan od glavnih principa Firefox OS misije je izbegavanje zavisnosti od biblioteka podataka. Ideja je da Web stane na svoje noge i da prirodno evoluira. Ukoliko bi se, kao kod drugih operativnih sistema, odlučili da svima podele SDK (Software development kit - skup alata i biblioteka za razvijanje softvera za datu platformu) i odgovarajuće biblioteke kako bi kreirali decentralizovane aplikacije, ironija bi bila u tome što bi se na taj način stvorila sklonost i jaka zavisnost u odnosu na date biblioteke. Firefox OS je mnogo više od pukog operativnog sistema - to je decentralizovana platforma koja developer-ima pruža potpunu kontrolu. Zadatak koji se stavlja pred developer-e je da stvore zdraviji, decentralizovani Web, jer bi bio oksimoron očekivati od operativnog sistema ili centralizovanog autoriteta da samostalno reše problem decentralizacije.

9.2 Manifest aplikacije

Manifest aplikacije sadrži informacije potrebne Web pretraživaču kako bi mogao da sarađuje sa aplikacijom. Upravo je manifest ono što razlikuje Web aplikaciju za Firefox OS od Web sajta. Manifest je JSON datoteka, sa ekstenzijom .webapp, koja sadrži ime i opis aplikacije, a može da sadrži i izvor aplikacije, ikonice i dozvole koje aplikacija zahteva. Pretraživač koji pokreće aplikaciju mora da sadrži Web runtime.

Ukoliko je potrebno, Mozilla nudi alat pod nazivom Manifest Validator koji služi za proveru manifesta.

Ukoliko developer želi, sam može da objavi aplikaciju. Nije u obavezi da je objavljuje na zvaničnom Marketplace-u. Tada se instalaciju pokreće putem dugmeta na Web stranici. Najčešće se to izvodi pozivanjem navigator.mozApps.install() funkcije. Ukoliko je aplikacije objavljena na Marketplace-u, instalacija se pokreće tako što se pretraživaču prosleđuje URL do manifesta aplikacije.[17]



```
{
 "version": "1.0",
 "name": "MozillaBall",
 "description": "Exciting Open Web development action!",
 "icons": {
  "16": "/img/icon-16.png",
  "48": "/img/icon-48.png",
  "128": "/img/icon-128.png"
 },
 "developer": {
  "name": "Mozilla Labs",
  "url": "http://mozillalabs.com"
 },
 "installs_allowed_from": ["*"],
 "appcache_path": "/cache.manifest",
 "locales": {
  "es": {
   "description": "¡Acción abierta emocionante del desarrollo del Web!",
   "developer": {
    "url": "http://es.mozillalabs.com/"
   }
  },
  "it": {
   "description": "Azione aperta emozionante di sviluppo di fotoricettore!",
   "developer": {
    "url": "http://it.mozillalabs.com/"
   }
  }
 },
 "default_locale": "en"
}
```

slika 9.1: Primer manifesta aplikacije

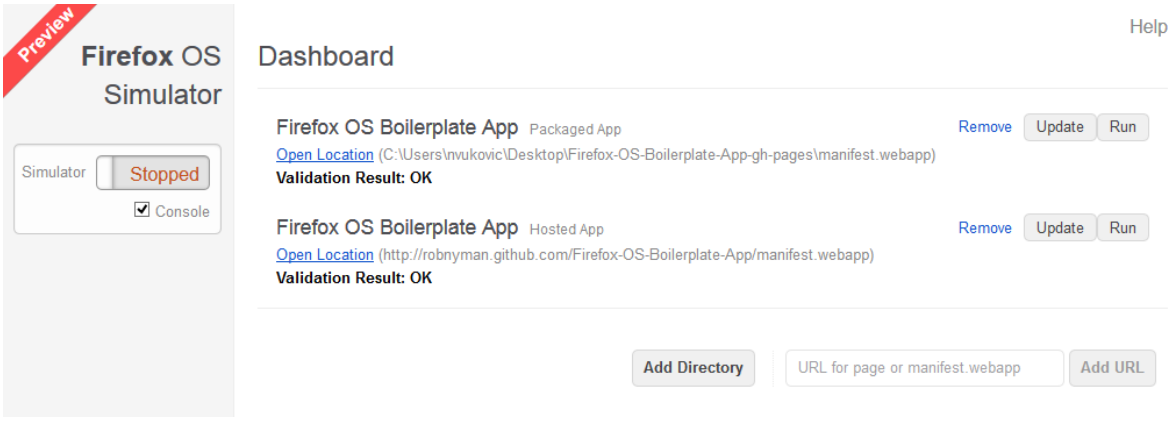
9.3 Firefox OS simulator

Firefox OS simulator je dodatak za Firefox desktop browser i služi da omogući testiranje i otklanjanje grešaka u Firefox OS aplikacijama na desktop računaru. Ciklus kodiranje-testiranje-otklanjanje grešaka je mnogo brži putem simulatora nego na pravom uređaju, i naravno nije potrebno imati sam uređaj.

Dodatak Simulator se sastoji od:

- **Simulator-a:** što uključuje Firefox OS desktop klijent, koji je u stvari verzija viših slojeva Firefox OSa koja se pokreće ne desktopu.
- **Dashboard-a (komandne table):** alat koji Firefox pretraživaču omogućava da pokrene i zaustavi Simulator, a zatim i da instalira i deinstalira aplikacije, kao i da otklanja greške koje se u njima pojavljuju. Putem Dashboard-a se aplikacije prebacuju iz simulatora na fizički uređaj.[17]

Simulator se distribuira i instalira kao dodatak za Firefox pretraživač. Nakon instalacije se može pokrenuti iz Web Developer dela menija.



slika 9.2: Dashboard Simulatora

Aplikacija se u Simulator dodaje ili lokalno putem dugmeta "Add Directory" i odabira manifesta aplikacije ili preko Web-a dodavanjem URL-a u textfield u kojem piše "URL for page or manifest.webapp". Ukoliko URL vodi do manifesta, taj manifest će biti preuzet. U suprotnom, Simulator će ga izgenerisati sam.

Za svaku aplikaciju sledeće informacije će biti dostupne:

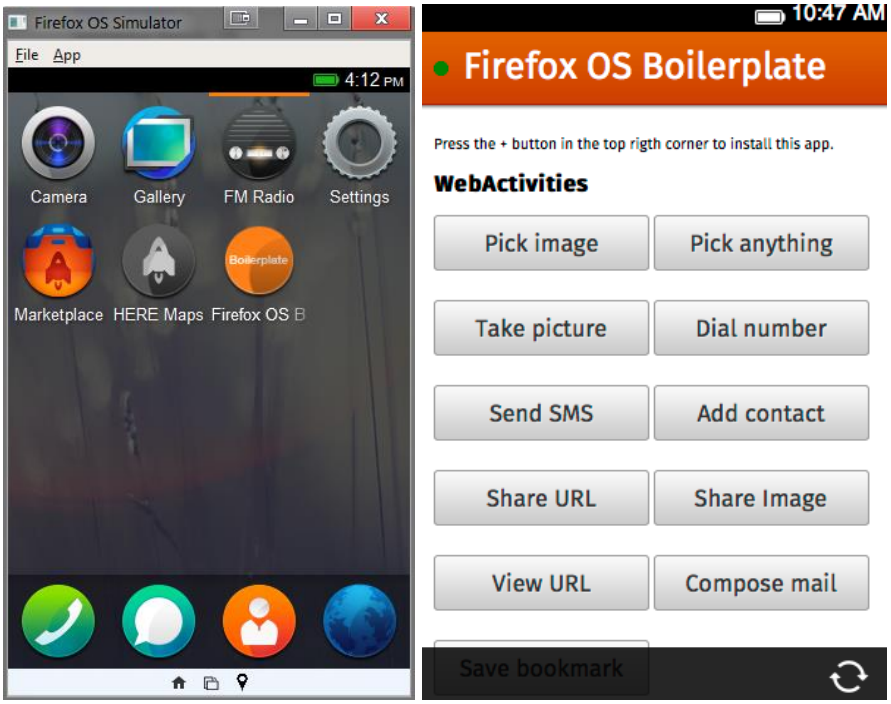
- ime aplikacije, preuzeto iz manifesta;
- njen tip;
- link do manifest datoteke;
- rezultat validacije manifesta;

Pored informacija dostupne su i četiri komande:

- **Remove** – služi za uklanjanje aplikacije iz Simulatora ili Dashboard-a. Može se poništiti sve dok je Dashboard otvoren;
- **Update** – služi za ažuriranje aplikacije, ukoliko su napravljene određene izmene;
- **Run** – pokretanje aplikacije u Simulatoru;
- **Push to device** – ukoliko je fizički uređaj zakačen na računar, ovim putem se aplikacije prebacuje na njega;

9.3.1 Pokretanje simulatora

Simulator se može pokrenuti ili odabirom komande Run pored aplikacije koju testiramo (na ovaj način se aplikacija automatski pokreće u simulatoru) ili klikom na crveno dugme sa natpisom "Stopped" (u ovom slučaju se putem menija mora doći do aplikacije). Kada se pokrene, Simulator se nalazi u zasebnom prozoru i simuliraće uređaj sa ekranom čija je rezolucija 320x480 piksela. Na dnu ekrana se nalazi linija sa alatima, a navigacija se obavlja mišem.



slika 9.3: Ekran simulatora i pokrenuta Boilerplate aplikacija

Ekran simulatora se ne razlikuje od ekrana koji se nalazi na fizičkom uređaju. Jedina razlika su dodatne opcije na dnu ekrana. Sleva na desno one su:

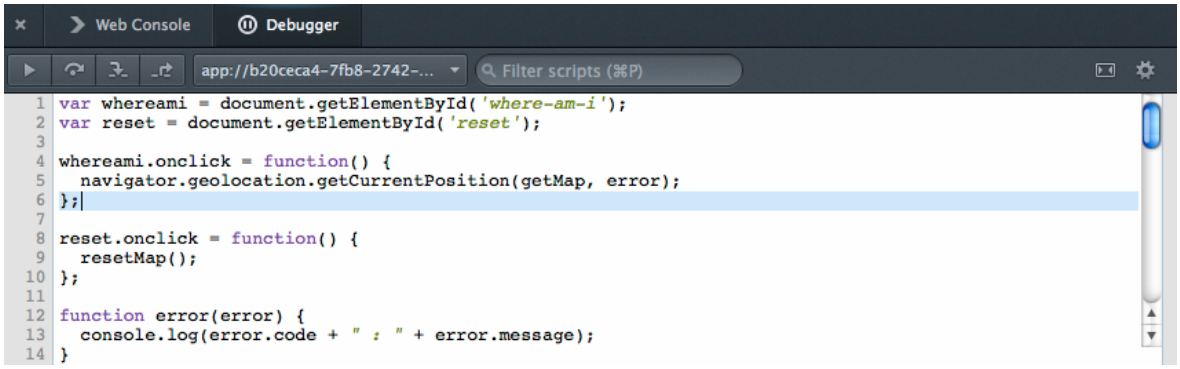
- **Home button**: dugme koje vraća korisnika na početni ekran;
- **Rotation Simulation button**: dugme koje simulira menjanje orijentacije uređaja. Klikom na ovo dugme se pokreće deviceorientation događaj;
- **Geolocation Simulation button**: pokreće se dijalog sa upitom za deljenje geografske lokacije. Radi uz pomoć Geolocation API-ja ili korišćenjem trenutnih koordinata ili upisivanjem proizvoljnih.

9.3.2 Dodavanje alatki za razvoj

Simulatoru je moguće dodati alatke za razvoj kako bi se otklanjale greške tokom razvoja aplikacije. Za sada je moguće dodati JavaScript Debugger i Web Console.

Ukoliko je Simulator pokrenut, pojaviće se još jedna opcija pod nazivom "Connect…" koja korisnika vodi do nove stranice na kojoj se povezuje na mašinu koja može biti i lokalna i bira port na koji se povezuje. Kada se Debugger pokrene, korisnik može da vidi sve skripte vezane za svoju aplikaciju, od kojih se glavna otvara kao

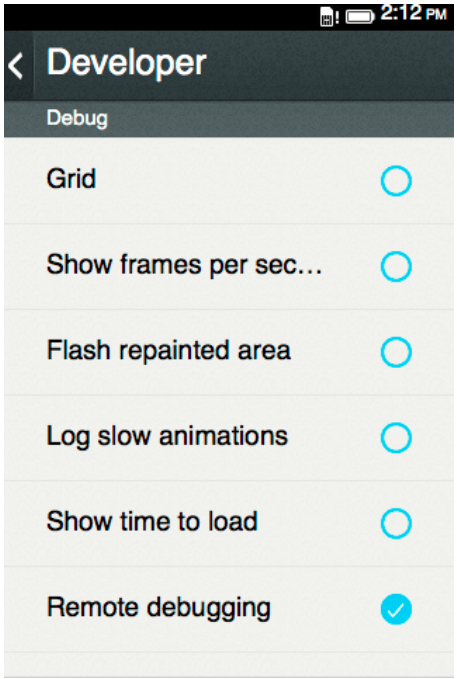
podrazumevana. Osim što može da vidi svoje skripte, korisniku je obezbeđen debugging celokupnog Firefox OS-a.[17]



slika 9.4: Konzola za proveru grešaka

9.3.3 Prebacivanje aplikacije na uređaj

Prebacivanje aplikacije na uređaj je jednostavan proces. Potrebno je željenu aplikaciju učitati u Simulator, a zatim je komandom "Push" prebaciti na uređaj. Pre toga je potrebno podesiti uređaj odabirom opcije Remote debugging koja se nalazi u Developer Settings delu podešavanja.



slika 9.5: Podešavanje uređaja za razvoj aplikacija

9.4 Ograničenja Simulatora

Firefox OS Simulator ne pruža savršenu simulaciju. Pored očigledne razlike u veličini ekrana, Simulator ne pokriva ni hardverska ograničenja uređaja kao što su količina dostupne memorije ili brzina procesora.

Osim toga, postoje kodeci poput MP3, AAC i H.264 koji zavise od hardverske akceleracije za dekodiranje, tako da ni oni nisu podržani. Takođe, ni određeni API-ji koji

rade na uređaju ne mogu da rade i u Simulatoru, prosto iz razloga što na desktopu ne postoji takav hardver. Neki od API-ja su dodati, poput onog za geo lokaciju, i očekuje se da će ih u budućnosti biti više, ali sledeći nisu podržani i njihovo korišćenje može dovesti do prikazivanja pogrešnih podataka:

- Telephony;
- WebSMS;
- WebBluetooth;
- Ambient Light;
- Proximity;
- Network Information;
- Vibration.

9.5 Revizija aplikacije

Da bi aplikacija postala privilegovana i odobrena za zvanični Marketplace, developer je mora poslati Mozilla-i na razmatranje. Aplikacija se zatim prosleđuje na rigoroznu proceduru provere koda. Proverava se autentičnost i integritet, a zatim da li se zatražene dozvole koriste upravo za ono za šta su i zatražene. Najviše pažnje se posvećuje proveri interfejsa između privilegovane aplikacije i sadržaja koji nema taj status, kako ne bi došlo do napada putem prekoračenja ovlašćenja. Dužnost Marketplace-a je da osigura korisnika od aplikacija koje bi mogle na maliciozan način da koriste dozvole.

Kada aplikacija uspešno prođe kroz proces revizije, ona je odobrena za korišćenje, njen manifest je digitalno potpisan od strane Marketplace-a i njeno preuzimanje je omogućeno svima. Aplikacije koje korisnik postavlja na, primera radi, svoj sajt ili bilo koje drugo mesto koje nije zvanični Marketplace su takođe dostupne svima i svi ih mogu preuzeti i instalirati. Ovakve aplikacije imaju najmanji nivo privilegija.[17]

10 Boilerplate aplikacija

Svrha Boilerplate aplikacije je da se na jednostavan način i na jednom mestu pokaže snaga web tehnologija, kao i mogućnosti koje nudi korišćenje ovih alata za izradu aplikacija za mobilne platforme. Kao što je više puta pomenuto, za izradu ove aplikacije, kao i mnogih aplikacija za Firefox OS, korišćene su tehnologije HTML, CSS i JavaScript.

Boilerplate se oslanja na WebAPI-je koje je Mozilla predložila, a neki od njih su i prihvaćeni od strane W3C, kako bi web tehnologijama pružio potpun pristup hardverskoj osnovi uređaja. Neke od akcija koje ova aplikacije nudi su upravo mogućnosti koje se i očekuju od pametnog telefona današnjice.

Akcije su podeljene u dve grupe: WebActivities i WebAPIs.

Među WebActivities spadaju:

- Fotografisanje;
- Pozivanje broja;
- Slanje SMS ili e-mail poruke;
- Deljenje sadržaja putem Bluetooth-a ili e-mail-a;
- Čuvanje bookmark-a;
- Reprodukovanje video materijala itd.

Među WebAPIs akcijama mogu se pronaći:

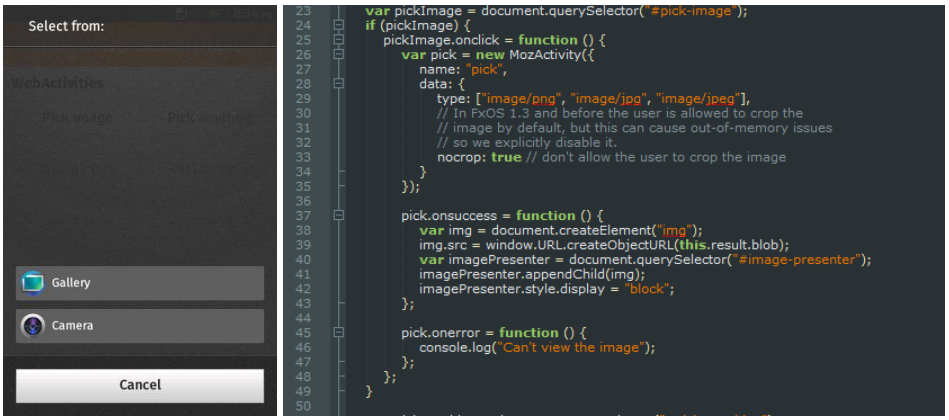
- Prikaz notifikacija;
- Zaključavanje orijentacije uređaja;
- Vibracija;
- Provera konekcije i statusa baterije;
- Udaljenost korisnika od uređaja;
- Ambijentalno osvetljenje itd.

10.1 WebActivities

WebActivities je deo Boilerplate aplikacije za Firefox OS operativni sistem. U ovom delu se nalaze web aplikacije koje putem određenih funkcija komuniriciraju sa hardverskom osnovom uređaja i vrše funkcije koje se i očekuju od pametnih telefona najnovije generacije.

10.1.1 Pick image

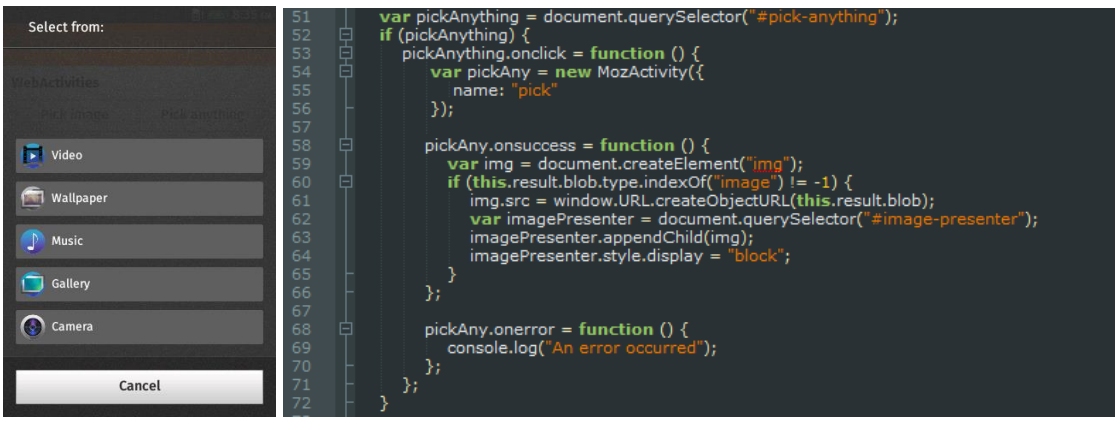
„Pick image“ je prva od stavki u odeljku WebActivities i služi za odabir bilo koje datoteke sa ekstenzijom koja predstavlja sliku, odnosno fotografiju, a nalazi se ili na memoriji uređaja ili memorijskoj kartici.



slika 10.1 - Akcija „Pick image“ i pripadajući JavaScript kôd

10.1.2 Pick anything

„Pick anything“ je druga od stavki u odeljku WebActivities koja služi za odabir bilo koje datoteke koja se nalazi ili na memoriji uređaja ili memorijskoj kartici.



slika 10.2: Akcija „Pick image“ i pripadajući JavaScript kôd

10.1.3 Take picture

„Pick anything“ je treća od stavki u odeljku WebActivities koja služi pokretanje Camera aplikacije za fotografisanje i snimanje video zapisa.



```
var record = document.querySelector("#record");
if (record) {
    record.onclick = function () {
        var rec = new MozActivity({
            name: "record" // Possibly capture in future versions
        });

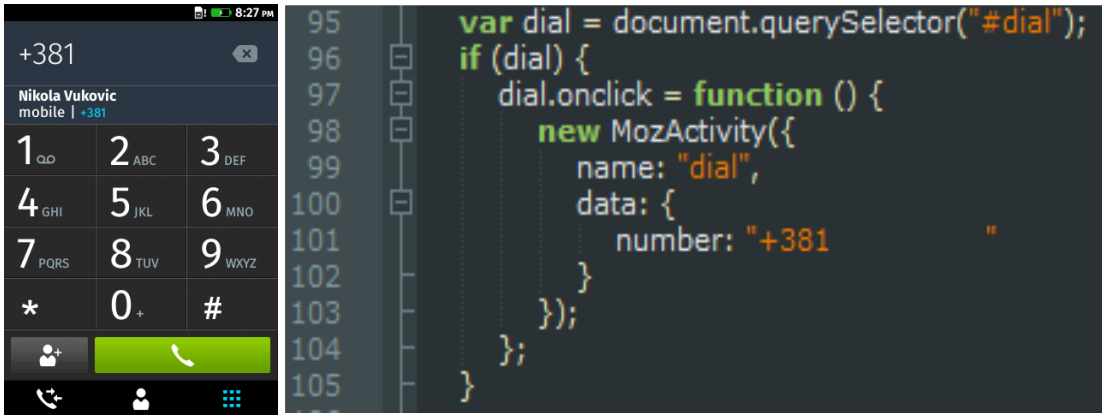
        rec.onsuccess = function () {
            var img = document.createElement("img");
            img.src = window.URL.createObjectURL(this.result.blob);
            var imagePresenter = document.querySelector("#image-presenter");
            imagePresenter.appendChild(img);
            imagePresenter.style.display = "block";
        };

        rec.onerror = function () {
            alert("No taken picture returned");
        };
    };
}
```

slika 10.3: Akcija „Take picture“ i pripadajući JavaScript kôd

10.1.4 Dial number

„Dial number“ je četvrta stavka u odeljku WebActivities koja služi za pokretanje Dialer aplikacije za pozivanje telefonskog broja.

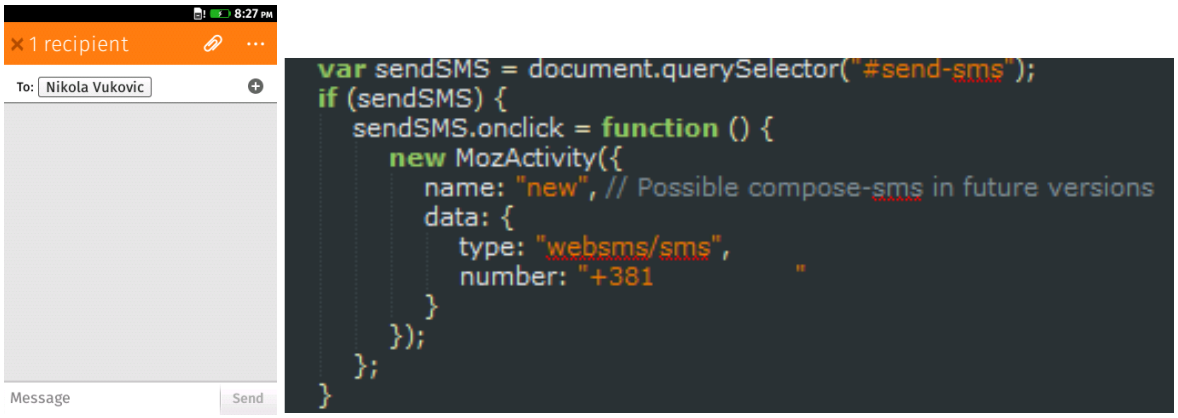



```
var dial = document.querySelector("#dial");
if (dial) {
    dial.onclick = function () {
        new MozActivity({
            name: "dial",
            data: {
                number: "+381          "
            }
        });
    };
}
```

slika 10.4: Akcija „Dial number“ i pripadajući JavaScript kôd

10.1.5 Send SMS

„Send SMS“ je peta stavka u odeljku WebActivities koja služi za pokretanje Messaging aplikacije za slanje SMS i MMS poruka.



slika 10.5: Akcija „Send SMS“ i pripadajući JavaScript kôd

10.1.6 Add contact

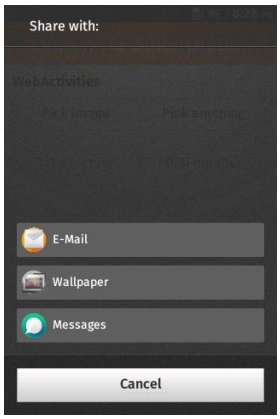
„Send SMS“ je šesta stavka u odeljku WebActivities koja služi za pokretanje Messages aplikacije za slanje SMS i MMS poruka.



slika 10.6: Akcija „Add contact“ i pripadajući JavaScript kôd

10.1.7 Share URL

„Share URL“ je sedma stavka u odeljku WebActivities koja služi za deljenje hiperlink-a sa drugima putem e-mail-a ili SMS, odnosno MMS poruke.

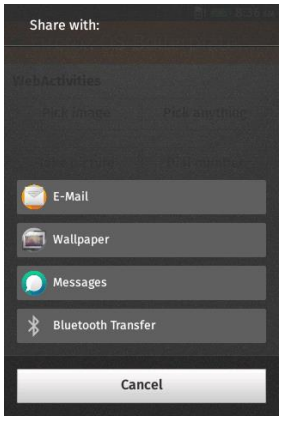



```
var share = document.querySelector("#share");
if (share) {
    share.onclick = function () {
        new MozActivity({
            name: "share",
            data: {
                //type: "url", // Possibly text/html in future versions,
                number: 1,
                url: "http://www.teamwerkz.rs"
            }
        });
    };
}
```

slika 10.7: Akcija „Share URL“ i pripadajući JavaScript kôd

10.1.8 Share image

„Share image“ je osma stavka u odeljku WebActivities koja služi za deljenje datoteke sa ekstenzijom koja predstavlja sliku, odnosno fotografiju, a nalazi se ili na memoriji uređaja ili memorijskoj kartici. Deljenje je moguće putem e-mail-a, SMS, odnosno MMS poruke, Bluetooth-a. Moguće je i sliku postaviti kao pozadinu ekrana uređaja.




```
var shareImage = document.querySelector("#share-image"),
    imgToShare = document.querySelector("#image-to-share");
if (shareImage && imgToShare) {
    shareImage.onclick = function () {
        if(imgToShare.naturalWidth > 0) {
            // Create dummy canvas
            var blobCanvas = document.createElement("canvas");
            blobCanvas.width = imgToShare.width;
            blobCanvas.height = imgToShare.height;

            // Get context and draw image
            var blobCanvasContext = blobCanvas.getContext("2d");
            blobCanvasContext.drawImage(imgToShare, 0, 0);
```

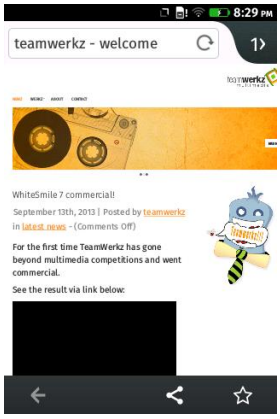
slika 10.8a: Akcija „Share image“ i pripadajući JavaScript kôd

```
            // Export to blob and share through a Web Activitiy
            blobCanvas.toBlob(function (blob) {
                new MozActivity({
                    name: "share",
                    data: {
                        type: "image/*",
                        number: 1,
                        blobs: [blob]
                    }
                });
            });
        }
        else {
            alert("Image failed to load, can't be shared");
        }
    };
}
```

slika 10.8b: Akcija „Share image“ i drugi deo pripadajućeg JavaScript kôda

10.1.9 View URL

„View URL“ je deveta stavka u odeljku WebActivities koja služi za pokretanje aplikacije Browser (u Firefox OS Simulator-u, dodatku za Firefox desktop varijantu pregledača) ili Firefox pregledača na Firefox OS uređaju koja omogućava pregled sadržaja na Internetu.

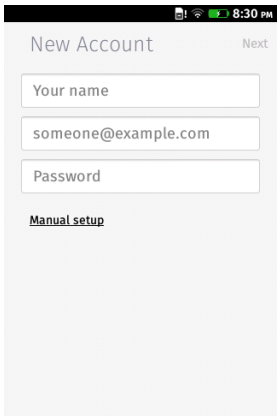



```
var viewURL = document.querySelector("#view-url");
if (viewURL) {
    viewURL.onclick = function () {
        new MozActivity({
            name: "view",
            data: {
                type: "url", // Possibly text/html in future versions
                url: "http://www.teamwerkz.rs"
            }
        });
    };
}
```

slika 10.9: Akcija „View URL“ i pripadajući JavaScript kôd

10.1.10 Compose mail

„Compose mail“ je deseta stavka u odeljku WebActivities koja služi za pokretanje aplikacije E-mail koja omogućava slanje e-mail poruka.

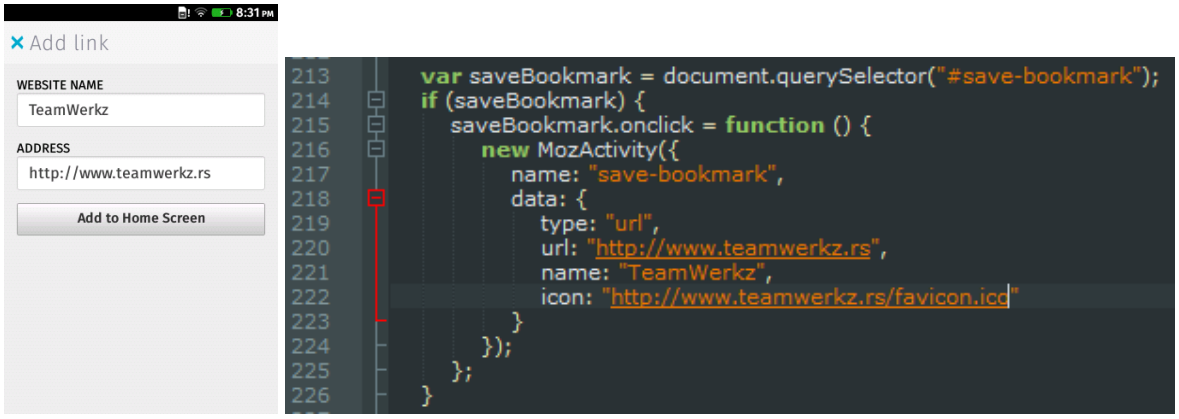



```
var composeEmail = document.querySelector("#compose-email");
if (composeEmail) {
    composeEmail.onclick = function () {
        new MozActivity({
            name: "new", // Possibly compose-mail in future versions
            data: {
                type : "mail",
                url: "mailto:example@example.org"
            }
        });
    };
}
```

slika 10.10: Akcija „Compose mail“ i pripadajući JavaScript kôd

10.1.11 Save bookmark

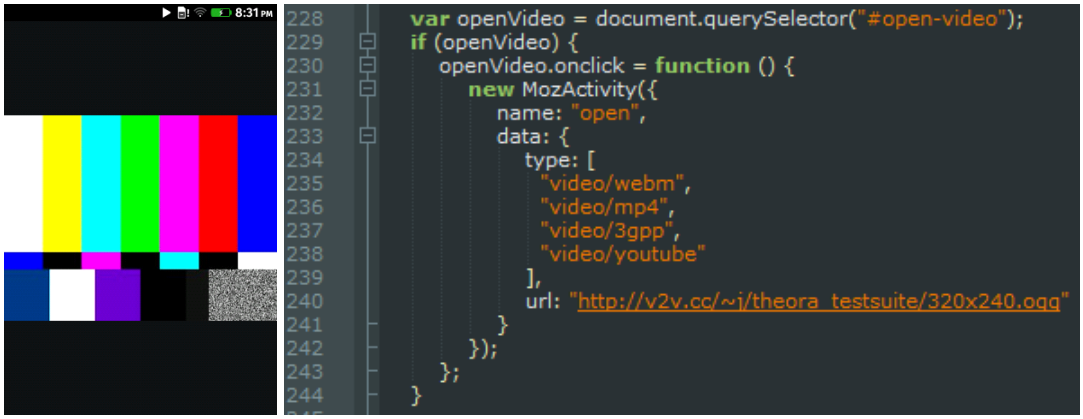
„Save bookmark“ je jedanaesta stavka u odeljku WebActivities koja služi čuvanje omiljenih web stranica putem obeleživača (*bookmark*) tako što se čuva veza do te stranice kao i ime radi lakšeg kategorizovanja i organizovanja.



slika 10.11: Akcija „Save bookmark“ i pripadajući JavaScript kôd

10.1.12 Open video

„Open video“ je dvanaesta stavka u odeljku WebActivities koja služi za pokretanje web aplikacije za reprodukovanje vidoe zapisa. U ovom slučaju otvara se video koji je hardkodovan, a nalazi se na adresi: http://v2v.cc/~j/theora_testsuite/320x240.ogg



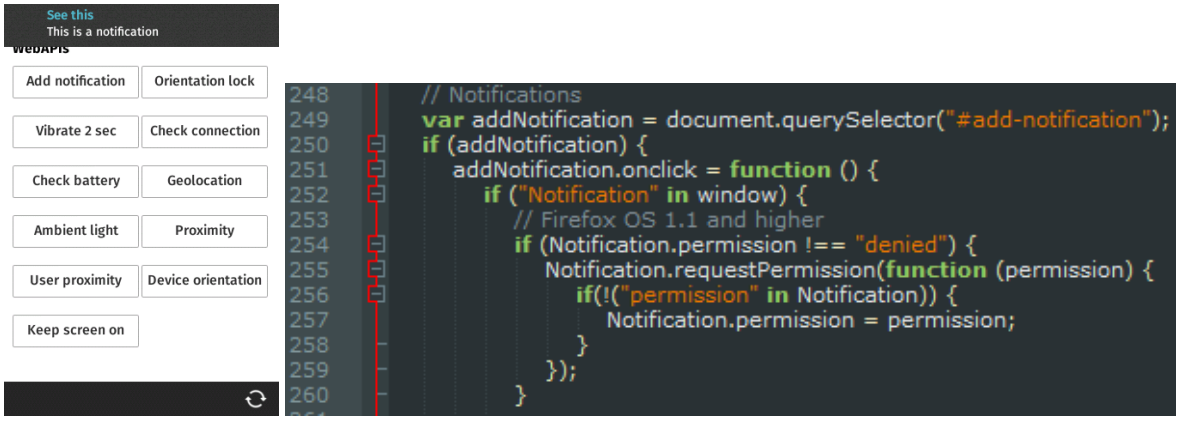
slika 10.12: Akcija „Open video“ i pripadajući JavaScript kôd

10.2 WebAPIs

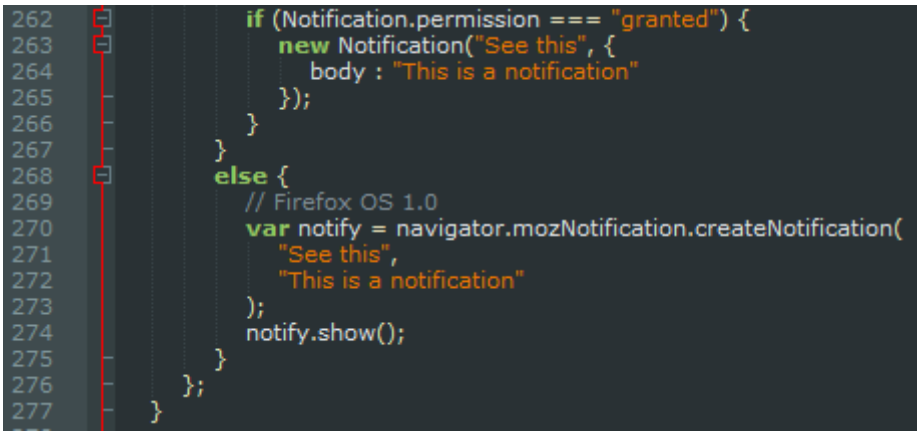
WebAPIs je drugi deo Boilerplate aplikacije za Firefox OS i svrha mu je da prikaže komunikaciju web aplikacija sa najnižim slojevima hardera mobilnog uređaja, a sve to putem WebAPI-ja koje je Mozilla predložila za standardizaciju. Neki od tih WebAPI-ja, kao što je već pomenuto su prihvaćeni i standardizovani, a neki su u fazi čekanja na odobrenje.

10.2.1 Add notification

„Add notification“ je prva stavka u odeljku WebAPIs koja služi za prikazivanje notifikacija, odnosno da obavesti korisnika o propuštenim pozivima, nepročitanim SMS i MMS porukama, pristiglim e-mail-ovima ili eventualnom postojanju novijih verzija aplikacija ili samog operativnom sistema uređaja, te potrebi da se te aplikacije ili sistem ažuriraju.



slika 10.13a: Akcija „Add notification“ i pripadajući JavaScript kôd



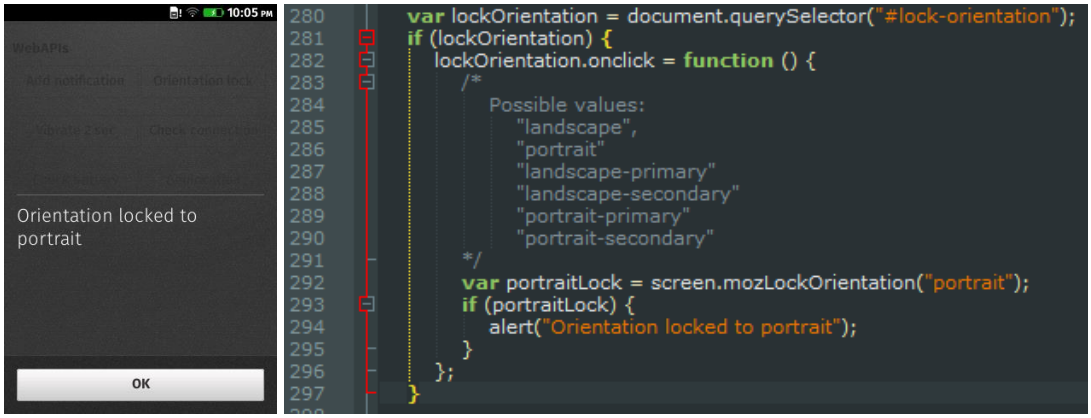
slika 10.13b: Akcija „Add notification“ i drugi deo pripadajućeg JavaScript kôda

10.2.2 Orientation lock

„Orientation lock“ je druga stavka u odeljku WebAPIs koja služi za zaključavanje orijentacije uređaja. Veliki broj mobilnih uređaja u sebi sadrži žirskom, a iščitavanjem podataka sa senzora žiroskopa aplikacije imaju uvid u trenutnu orijentaciju uređaja. U odnosu na te podatke aplikacija, ukoliko podržava tu mogućnost, će se učitati u vertikalnom (*portrait*) ili u horizontalnom (*landscape*) modu. Neke aplikacije nude različita rešenje interfejsa u odnosu na orijentaciju. Napredniji i uređaji višeg cenovnog ranga nude mogućnost prepoznavanja lica korisnika uz pomoć prednje kamere, te određuju orijentaciju u odnosu na položaj lica korisnika.

U aplikaciji je iskorišćena mogućnost zaključavanja orijentacije u Portrait modu. Sve dostupne opcije su:

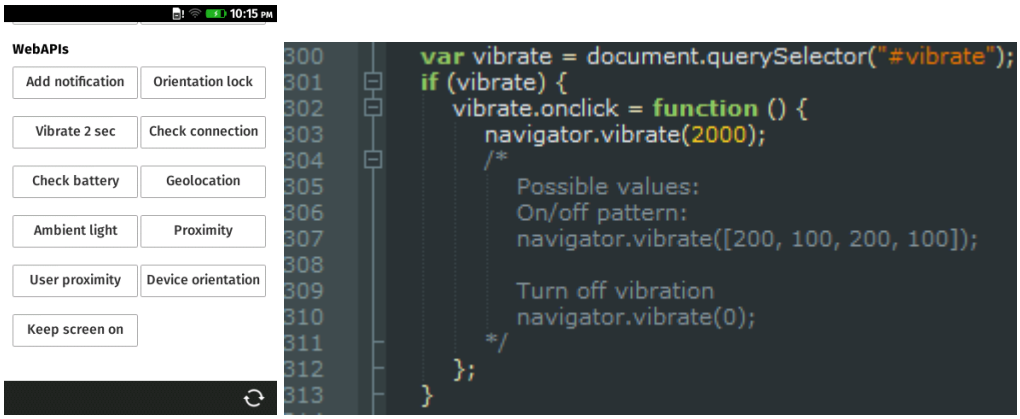
- Landscape;
- Portrait;
- Landscape-primary;
- Landscape-secondary;
- Portrait-primary;
- Portrait-secondary.



slika 10.14: Akcija „Orientation lock“ i pripadajući JavaScript kôd

10.2.3 Vibrate 2 sec

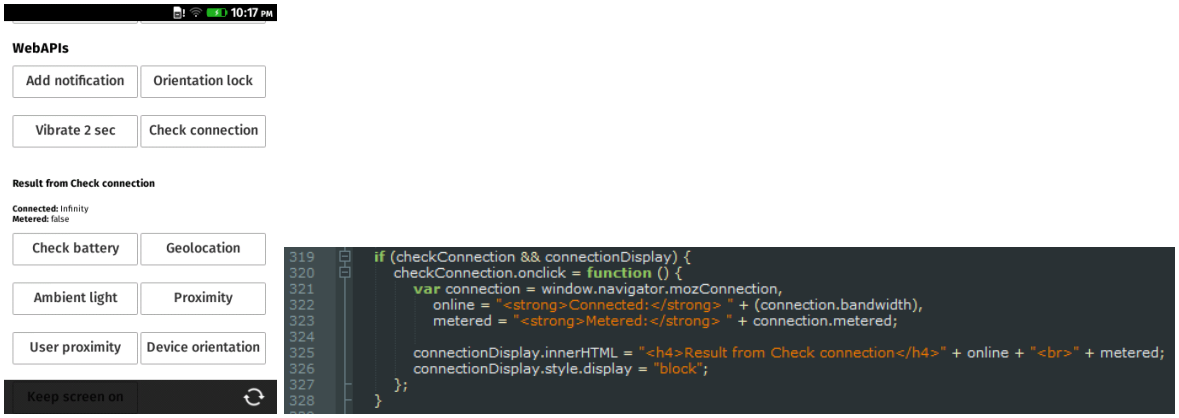
„Vibrate 2 sec“ je treća stavka u odeljku WebAPIs koja putem API-ja za vibraciju komunicira sa vibracionim motorom mobilnog uređaja. Vrednost dužine trajanja vibriranja je moguće izmeniti u kôdu i izražava se u milisekundama. Moguće je podesiti i ritam vibracije, tako što će, na primer, uređaj vibrirati redom 200, 100, 200, 100ms.



slika 10.15: Akcija „Orientation lock“ i pripadajući JavaScript kôd

10.2.4 Check connection

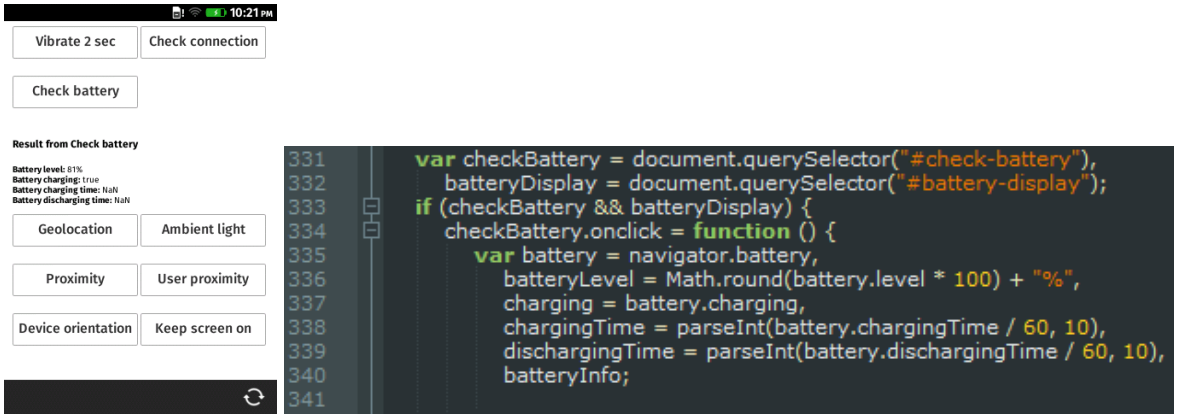
„Check connection“ je četvrta stavka u odeljku WebAPIs koja putem API-ja komunicira sa delovima mobilnog uređaja zaduženim za bežičnu komunikaciju.



slika 10.16: Akcija „Check connection“ i pripadajući JavaScript kôd

10.2.5 Check battery

„Check battery“ je peta stavka u odeljku WebAPIs koja putem API-ja za bateriju komunicira sa senzorom zaduženim za merenje nivoa baterije mobilnog uređaja. Ovo je jedan od WebAPI-ja koji je Mozillla predložila, a prihvaćen je od strane World Wide Web Consortium-a.



slika 10.17a: Akcija „Check connection“ i pripadajući JavaScript kôd

```
        batteryInfo = "<h4>Result from Check battery</h4><strong>Battery level:</strong> " + batteryLevel + "<br>";
        batteryInfo += "<strong>Battery charging:</strong> " + charging + "<br>";
        batteryInfo += "<strong>Battery charging time:</strong> " + chargingTime + "<br>";
        batteryInfo += "<strong>Battery discharging time:</strong> " + dischargingTime;

        batteryDisplay.innerHTML = batteryInfo;
        batteryDisplay.style.display = "block";
    };
}
```

slika 10.17b: Akcija „Check connection“ i drugi deo pripadajućeg JavaScript kôda

10.2.6 Geolocation

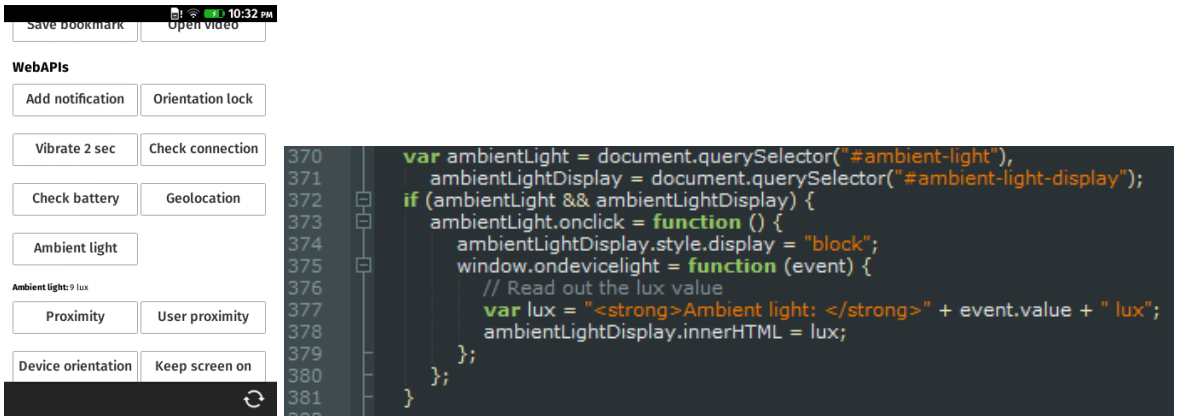
„Check battery“ je šesta stavka u odeljku WebAPIs koja putem API-ja za geolokaciju komunicira sa GPS senzorom mobilnog uređaja. Na ovaj način uređaj pronalazi svoju lokaciju uz pomoć satelita što je veoma korisno pri korišćenju aplikacije za navigaciju ili dodatavanja oznake fotografijama ili video zapisima načinjenim uz pomoć kamere mobilnog uređaja.



slika 10.18: Akcija „Geolocation“ i pripadajući JavaScript kôd

10.2.7 Ambient light

„Ambient light“ je sedma stavka u odeljku WebAPIs koja putem API-ja komunicira sa senzorom mobilnog uređaja za količinu okolne svetlosti. Na ovaj način uređaj, ukoliko je ta opija odabrana, može automatski da reguliše osvetljenost ekrana u osnosu na okolnu količinu svetla. Ova opcija je naročito korisna, jer u situacijama kada je sredina dobro osvetljena nije potrebno da i ekran sija punom snagom, a kako je upravo ekran najveći potrošač električne energije na mobilnim uređajima ovim se znatno produžava trajanje baterije. S druge strane, ukoliko je okolina loše osvetljena, osvetljene ekrana će se povećati.



slika 10.19: Akcija „Ambient light“ i pripadajući JavaScript kôd

10.2.8 Proximity

„Proximity“ je osma stavka u odeljku WebAPIs koja putem API-ja za blizinu komunicira sa senzorom mobilnog uređaja. Senzom meri udaljenost korisnika od prednje strane uređaja i na taj način pokreće određene akcije. Korisnik će prineti uhu uređaj tokom razgovora, senzor će prepoznati blizinu korisnika i privremeno isključiti ekran kako bi se izbeglo aktiviranje bilo kakvih drugih opcija dok razgovor traje. Vrednosti koje uređaj prepoznaje su u opsegu 0-10cm.



slika 10.20: Akcija „Proximity“ i pripadajući JavaScript kôd

10.2.9 Device orientation

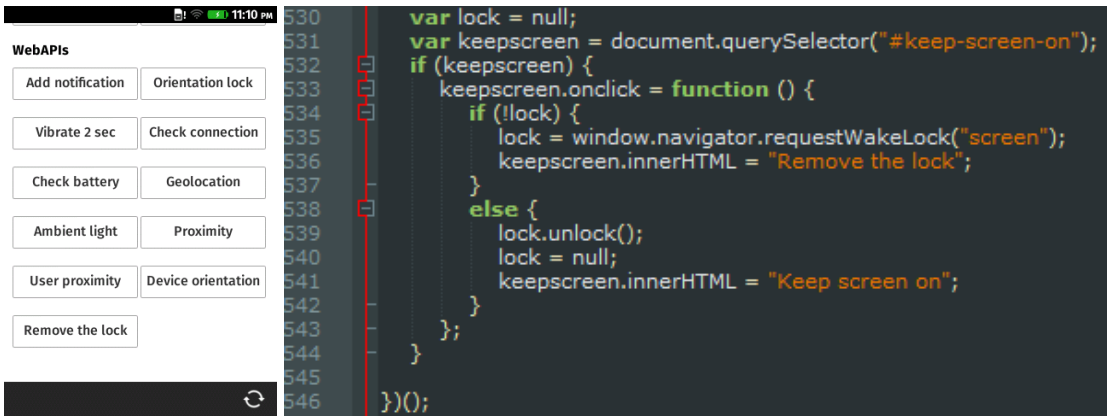
„Device orientation“ je deveta stavka u odeljku WebAPIs koja putem API-ja za orijentaciju komunicira sa senzorom mobilnog uređaja. Ovim putem je moguće odrediti položaj uređaja, kao i njegovu orijentaciju. Flat je u slučaju kada je uređaj položen horizontalno na nekoj površini, a Top kada stoji vertikalno, odnosno onako kako bi se inače držao u ruci.



slika 10.21: Akcija „Device orientation“ i pripadajući JavaScript kôd

10.2.10 Keep screen on

„Keep screen on“ je deseta stavka u odeljku WebAPIs koja putem API-ja komunicira sa senzorom mobilnog uređaja. Ovim putem je moguće isključiti mogućnost gašenja ekrana uređaja ili ga ponovo omogućiti, pokretanjem aplikacije za zaključavanje ekrana.



slika 10.22: Akcija „Keep screen on“ i pripadajući JavaScript kôd

11 Budućnost Firefox OS-a

Firefox OS nije platforma koja ima vlasnika. U pitanju nije čak ni novi ekosistem, jer je Firefox sinonim za Web, a Web je najveći postojeći ekosistem u današnjem digitalnom svetu. Trenutna situacija je mukotrpna za većinu developer-a, jer sistem u kojem se uopšte pomišlja na stvaranje aplikacije koja tek treba da se dokaže i to na više platformi, nije održiv. Mozilla Firefox OS vidi kao katalizator koji će otvoriti Web za sve.

Prema istraživanjima, postoji preko 8,000.000 developera za HTML5, nasuprot 100.000 za iOS i 400.000 za Android. Mozilla računa na ovu, uslovno rečeno, armiju da učini Web što slobodnijim mestom. Međutim, iako su neprofitne namere Mozille lepa zamisao, zatvoreni ekosistemi kao što je Apple-ov App store sa preko 20,000.000 preuzetih aplikacija samo tokom 2012. godine i Google Play sa približno istim brojem preuzimanja se i te kako čine primamljivim mladim developerima.

Problem kod postojećih platformi leži u činjenici da kada se korisnik jednom odluči za platformu i plati aplikacije koje želi da koristi, retko prelazi na drugu platformu, jer ne postoji način da iste aplikacije prenese na nju. Umesto da stvara novu platformu, Firefox OS koristi Web kako bi prevazišao baš ovakve prepreke. Kako u Mozilli tvrde, nema potrebe za stvaranjem novog ekosistema, jer Firefox OS koristi sve prednosti HTML5 i činjenice da developeri praveći Web aplikacije i Web sajtove, već prave Firefox OS aplikacije. Ova tvrdnja je potkrepljena demonstracijom na Mobile World kongresu na kojem je prikazana integracija sa Facebook-om i mapama koje se baziraju na Nokijinoj Here platformi.

Pored Firefox OSa postoje i mnogi drugi koji žele da razmrdaju tržište koje većinskim delom drže dva operativna sistema. To su, između ostalih, Tizen, Ubuntu i Windows Phone. Prema tvrdnjama analitičara, na polju alternativnih operativnih sistema koji tek nastaju, Mozilla ima najveću prednost. Napravili su najveći napredak i za korak su ispred drugih, jer su za samo dve godine od ideje došli do sistema koji se gotovo svakodnevno može koristiti.

Još jedna činjenica, ništa manje bitna od prethodnih, stavlja Firefox OS ispred drugih operativnih sistema. Zbog malih hardverskih zahteva, Firefox OS telefoni mogu da budu najjeftiniji na tržištu. Mozilla za početak ne planira ulazak na velika tržišta poput Severne Amerike i zapadne Evrope koja su većinski pokrivena Android i iOS operativnim sistemima, već cilja na tržišta u razvoju, kao što su Latinska Amerika i Istočna Evropa. Zato i ne čudi odluka o strateškom partnerstvu sa španskim operaterom Telefónica. Telefónica pokriva veliki deo Latinske Amerike, gde je udeo pametnih telefona na tržištu tek nešno iznad 16%, za razliku od, na primer, 50% u Ujedinjenom Kraljevstvu. Zahvaljujući Telefónici ostvareni su i dogovori sa drugim velikim operaterima poput América Móvil, China Unicom, Deutsche Telekom, Sprint, Telecom Italia and SingTel, kao i sa proizvođačem čipova Qualcomm. Sve ovo može samo da znači jeftinije telefone za korisnike, bilo da ih kupuju u trgovini ili preko operatera, koji će zbog velike ponude modela nuditi bolje subvencije.

Korišćenje Firefox OSa, u razvojnoj fazi, donekle podseća na stare verzije Androida. Pomalo je nezgodan, deluje nedovršeno i primećuje se nedostatak aplikacija i mogućnosti. Svakodnevno se radi na poboljšanju ovih i drugih aspekata sistema. Možda i najveća snaga platforme leži u činjenici da već postoji mnoštvo dobrih Web aplikacija, kao i ogroman broj developera koji žele da se oslobode stega Google-a i Apple-a. Za operativni sistem, dostupnost aplikacija i podrška od strane developera su među najbitnijim činiocima

uspeha. Mozilla kroz zajednicu, a najpre kroz mrežu Mozilla Developer Network, ima jedan od najbolje razvijenih sistema podrške, dok postavljanje aplikacija na Marketplace ili bilo gde na Internetu čini aplikacije dostupne svakome.

Postoje korisnici kojima možda i nije stalo do toga koji sistem koriste na svom uređaju, ali je zato mobilnim operaterima veoma stalo do pojavljivanja konkurencije na tržištu. Pojavljivanjem trećeg jakog operativnog sistema operateri dobijaju dobru podlogu za povoljnije pregovore sa velikim proizvođačima uređaja kao što su Apple i Samsung. Što je konkurencija bolja, njihov odnos će biti uravnoteženiji i na kraju će pobediti oni koji imaju najbolji proizvod, najniže cene i sposobnost da prepoznaju nove prilike.

Pravi test Firefox OSa i njegovih izgleda za budućnost leži u kvalitetu softvera, kao i u doživljaju za korisnike i one koji će razvijati aplikacije, a koje ova platforma podstiče.

12 Zaključak

Kada su sistemski specifične, hibridne i web aplikacije u pitanju, pravog pobednika je nemoguće odrediti. Pojedine aplikacije su bolje kao sistemski specifične, a neke kao web aplikacije. Web kao platforme trenutno ima veći zalet, ali kada se u obzir uzmu mogućnosti i kvalitet izvršavanja, sistemski specifične aplikacije takođe brzo napreduju. Dok ne dođe dan kada web aplikacije preuzmu primat na većini platformi i operativnih sistema, sistemski specifične aplikacije će uvek biti važne za razmatranje.

Jedna od tehnika koja je već pomenuta su hibridne aplikacije. One mogu biti dobra alternativa za neke razvojne timove: web view tamo gde je to moguće i komponente vezane za platformu gde nije moguće koristiti web.

Web kao platforma sa sobom donosi standarde i principe progresivnog poboljšanja i napretka. Web tehnologija je takva da tačno zna kako da cilja mnoštvo uređaja i operativnih sistema na tržištu. Bilo da se to smatra fragmentacijom ili raznovrsnošću, razvojni timovi mogu mnogo da se okoriste prethodnim radovima.



Literatura

[1] Blog Feliksa Tarnera „Stage 3D vs WebGL performance“ http://www.airtightinteractive.com/2011/10/stage3d-vs-webgl-performance/ (posećen apr. 2014.)

[2] Adobe Flash developer center Stage3D http://www.adobe.com/devnet/flashplayer/stage3d.html (posećen mart 2014.)

[3] Bijin, Chen, i Xu Zhiqi. “A framework for browser-based Multiplayer Online Games using WebGL and WebSocket.” *2011 International Conference on Multimedia Technology*. IEEE, 2011. 471-474.

http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=6001673 (posećen apr. 2014.)

[4] Džon Kongote „Interactive visualization of volumetric data with WebGL in real-time.” *Web3D 11 Proceedings of the 16th International Conference on 3D Web Technology*. 2011. 137-146. http://dl.acm.org/citation.cfm?id=2010425.2010449 (posećen apr. 2014.)

[5] Dan Lecocq „WebGLot : High-Performance Visualization in the Browser.” *Proceedings of International Conference on Computer Graphics and Interactive Techniques* (2010) : 4503-4503. http://dl.acm.org/citation.cfm?id=1837072 (posećen mart 2014.)

[6] Zhang i Weigang „*A WebGL-based method for visualization of intelligent grid*.” 2011 4th International Conference on Electric Utility Deregulation and Restructuring and Power Technologies (DRPT). IEEE, 2011. 1546-1548. http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=5994142 (posećen mart 2014.)

[7] WebGL and its applications *T-111.5502 Seminar on Media Techology B 5cr* Anton Zvonkov https://noppa.aalto.fi/noppa/kurssi/t-111.5502/lisatty7102/T-111_5502_webgl.pdf (posećen mart 2014.)

[8] Mobile: Native Apps, Web Apps, and Hybrid Apps, Raluca Budiu http://www.nngroup.com/articles/mobile-native-apps/ (posećen mart 2014.)

[9] HTML5 vs Native: The Mobile App Debate http://www.html5rocks.com/en/mobile/nativedebate/ (posećen april 2014.)

[10] HTML5 vs native vs hybrid mobile apps: 3500 developers say all three, please http://venturebeat.com/2013/11/20/html5-vs-native-vs-hybrid-mobile-apps-3500-developers-say-all-three-please/ (posećen mart 2014.)

[11] Gary B., Shelly, Thomas, J. Cashman, Steven G. Forsythe, Steven M. Freund, **Mozilla Firefox: Introductory Concepts and Techniques**, 1st Edition, Thomson Course Technology, 2005.

[12] Kenneth C. Feldt, **Programming Firefox: Building Rich Internet Applications with XUL**, 1st Edition, 2007.

[13] Nigel McFarlane, **Rapid Application Development with Mozilla**, 1st Edition, 2003.

[14] Fiza Irfan, (2013) *Firefox OS ready to hit the Stores,* http://sites.ieee.org/sb-neduet/megafuel-webmagazine/mega-fuel-5th-edition/tech-fest/firefox-os-ready-to-hit-the-stores/ (posećen feb. 2014.)

[15] Jevsikova, T. (2004) *Mozilla Internet application suite: developing for education,* *http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=1393654&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D1393654* (posećen feb. 2014.)

[16] Mozzila Developmer Network, (2013), *Introduction to Firefox OS,* https://developer.mozilla.org/en-US/docs/Mozilla/Firefox_OS/Introduction (posećen mart 2014.)

[17] Mozzila Developmer Network, (2013), *Creating Apps for Firefox OS,* https://developer.mozilla.org/en-US/docs/Mozilla/Firefox_OS/Apps (posećen jan. 2014.)

[18] Mozzila Developmer Network, (2013), *The Firefox OS platform,* https://developer.mozilla.org/en-US/docs/Mozilla/Firefox_OS/Platform (posećen mart 2014.)

[19] Mozzila Developmer Network (2013), *Firefox OS security documentation,* https://developer.mozilla.org/en-US/docs/Mozilla/Firefox_OS/Security (posećen dec. 2013.)

[20] Firefox OS guidelines http://www.mozilla.org/en-US/styleguide/products/firefox-os/ (posećen dec. 2013.)

[21] Firefox OS branding http://www.mozilla.org/en-US/styleguide/identity/firefoxos/branding/ (posećen mart 2014.)

