TALLINNA ÜLIKOOL

Informaatika Instituut

PhoneGap analüüs -

Eelseadistused bakalaureusetöö tarbeks

Seminaritöö

Autor: Toomas Naaber

Juhendaja: Jaagup Kippar

Autor: …...…….”…….” ….…….2015

Juhendaja:……..…..“…….” …….2015

Tallinn 2015

Autorideklaratsioon

Deklareerin, et käesolev seminaritöö on minu töö tulemus ja seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on viidatud.

.... (kuupäev) (autor)

Sisukord

Sissejuhatus

Bakalaureusetööna võtab autor ette rakenduse loomise. Enne kui saab rakenduse loomise juurde asuda, tuleb teha eeltöö ning uurida võimalikke lahenduskohti, mis aitaks rakendust luua. Seminaritöös teeb autor eelseadistused vajalike programmide ja lisandite jaoks.

Käesolev töö on jagatud neljaks peatükiks. Esimeses peatükis teeb autor lühitutvustused bakalaureusetööle, PhoneGapi olemusele ja rakendamisvõimalustele ning annab ülevaate Parse andmebaasist.

Teises peatükis võtab autor süvitsi ette PhoneGap seadistuse, installides või seadistades viit erinevat lisa või programmi. Samuti toob autor välja väiksemad probleemid, mis võivad esineda programmide installatsiooni käigus.

Kolmandas peatükis loob autor esmase stamprakenduse ning kirjeldab tegevusi lisaks tekstile ka illustreerivalt. Autor toob välja rakenduse loomise ning käivitamise juhendi nii emulaatoril kui ka seadmel endal.

Neljandas peatükis toob autor välja rakenduslikud lisaväärtused PhoneGap pluginate toel, mida saab hõlpsasti kasutada rakenduse loomisel. Näidete baasil on toodud illustreerivad tulemused. Teisalt kirjeldab autor peatükis PhoneGap Events'e, millega on võimalik lisada rakendusele kindlustavat tuge.

Käesoleva seminaritöö eesmärgiks on ära teha eeltööna eelseadistused PhoneGap rakenduse loomiseks. Lisaks leida esialgsed võimalused, mida saab bakalaureusetöös edukalt kasutada.

1 Tutvustused

Antud peatükis tutvustab autor bakalaureusetööd, annab lühiülevaate Phonegapile ja Parse andmebaasile.

1.1 Bakalaureusetöö tutvustus

Bakalaureusetööks on autor seadnud eesmärgiks luua piletimüügi rakendus Rally Estonia läbiviimiseks. Rakendus leiab kasutust juba sel suvel, 17-19.07.2015, mil sõidetakse taas rallit Lõuna-Eestis. Piletite müük toimub vabas õhus. Rakendusega võimaldatakse parem ja kiirem teadete edastamine piletite müügist ning probleemidest müügikohtades.

Rakendusele tuleb kaks erinevat poolt. Esimene neist rakendub müügipunktides, teine aga piletikoordinaatorite seadmetes. Piletipunktides on mõeldud, et piletite müüjad saavad sisestada müügi tulemusi, end positsioneerida ning saavad jätta teateid. Lähemalt antud funktsionaalsustest kirjeldab autor bakalaureusetöö raames.

Koordinaatorite funktsionaalsus pole veel täielikult kinnitatud, kuid eeldatavalt peaks olema müügiprotsessi kuvamine numbrites ning graafikuna, näha positsioneeritud müügipunktide asukohti, saata teateid müügipunktidesse ning näha punktidest tulenevaid teateid.

1.2 Phonegap tutvustus

PhoneGap on avatud lähtekoodiga raamistik ehitamaks kiirelt erinevate mobiilsete platvormidele funktsionaalseid rakendusi (native applications), kasutades selleks HTML5, Javascripti või jQuery't ja CSS3'e.

PhoneGap tuli turule 2011. aastal ning on tänaseks alla laetud üle miljoni korra ning seda kasutab rohkem kui 400 000 arendajat.

Lisaks tavalisele HTML5 kirjutamisel saab rakendustele anda funktsionaalsusi juurde Java rakendustega. PhoneGap omab suurel hulgal erinevaid vidinaid (plugin). Sisse ehitatud vidinaid on 12, kuid kõik ei tööta igal mobiilsel platvormil (Joonis 1). Lisaks neile on PhoneGapile erinevate arendajate poolt avatud lähtekoodiga loodud 779 erinevat pluginat (Apache Cordova Plugins Registry). Igal arendajal on võimalus luua endale meelepärane plugin.

	iPhone / iPhone 3G	iPhone 3GS and newer	Android	Blackberry OS 6.0+	Blackberry 10	Windows Phone 8	Ubuntu	Firefox OS
Accelerometer	✓	✓	✓	✓	✓	✓	✓	✓
Camera	✓	✓	✓	✓	✓	✓	✓	✓
Compass	X	✓	✓	X	✓	✓	✓	✓
Contacts	✓	✓	✓	✓	✓	✓	✓	✓
File	✓	✓	✓	✓	✓	✓	✓	X
Geolocation	✓	✓	✓	✓	✓	✓	✓	✓
Media	✓	✓	✓	X	✓	✓	✓	X
Network	✓	✓	✓	✓	✓	✓	✓	✓
Notification (Alert)	✓	✓	✓	✓	✓	✓	✓	✓
Notification (Sound)	✓	✓	✓	✓	✓	✓	✓	✓
Notification (Vibration)	✓	✓	✓	✓	✓	✓	✓	✓
Storage	✓	✓	✓	✓	✓	✓	✓	✓

Joonis 1- sisse ehitatud pluginate töökindlus erinevatel platvormidel

PhoneGap töötab üldiselt lihtsal põhimõttel. Arendaja kirjutab valmis HTML koodi antud rakendusele – võimalik luua ka mitmeosaline rakendus, näiteks mitme sisulehega. Peale HTML'i kirjutamist saab antud tulemust muuta visuaalselt meeldivamaks CSS rakendamisel. Lisaks on võimalik kasutada JavaScripti funktsioone andmaks rakendusele veel suurema funktsionaalsuse.

Kui eelmainitud tegevused on tehtud, tuleb PhoneGap rakendus töödelda PhoneGap Build programmi poolt, mille tulemuseks on väljund rakenduse platvormide valikutest – Android, IOS webOS, Windows jne. Rakenduse väljundit saab otse kuvada nii seadmes endas kui ka kasutades SDK tööriistu kuvamaks tulemust emulaatoris.

PhoneGap võimaldab kõigele eelnevale kasutada ka Java tuge rakendustes. Java kasutamine on tehtud võimalikuks vaid läbi pluginate. Lisa vidinad annavad rakendusele vastava võimaluse muuta antud rakendus veelgi funktsioonide rohkeks.

1.3 Parse.js andmebaas

Autor valis uudsema lähenemise, võttes kasutusele Parse andmebaasi lahenduse tavalise SQL andmebaasi asemel. Parsel on ka Java abil kasutatav Androidi dokumentatsioon, mis annab arendajale võimaluse kasutada Parse võimalusi Android seadmes. Lisaks on olemas teised võimalused, kuidas antud andmebaasi kasutada erinevatel operatsioonisüsteemidel.

Parse on pilveserveril töötav rakenduslik liides, mis muudab andmete vahendamise kiireks ning turvaliseks. Parse võimaldab luua ka lokaalset andmebaasi ning kasutada antud andmeid ka siis kui puudub interneti ühendus. Kogu info salvestatakse kohalikku baasi ning interneti ühendamisel lisatakse kõik uued andmed pilve. Selline võimalus on hetkel ainult Android ja IOS platvormidele. Parsel on ka sisse ehitatud analüüsi võimalus, mis annab igapäevase ülevaate rakenduse kasutusest.

2 PhoneGap seadistamine

PhoneGap seadistamine nõuab mitmete lisandite allalaadimist ning nende installimist või seadistamist, nii et need toetaks PhoneGap tööd.

2.1 Java SDK

Android SDK vajab rakenduste käivitamiseks JDK'd (Java Development Kit). Mängude ja Java programmide käivitamiseks on arvutisse installitud Java Runtime Environment (JRE), kuid antud kontekstis ei ole see piisav. Android SDK töötamiseks läheb vaja JDK'd. Selleks tuleb Java Oracle lehelt alla laadida Java SE JDK (SE = Standard Edition) (Joonis 2).

Joonis 2 - Oracle SDK allalaadimine

Allalaadimiste loetelus tuleb valida lähtuvalt arvuti operatsioonisüsteemist. Loetelus ei pea võtma kaasa JDK demos and samples valikut, vaid piisab ainult JDK'st (Joonis 3).

Java SE Development Kit 8u40

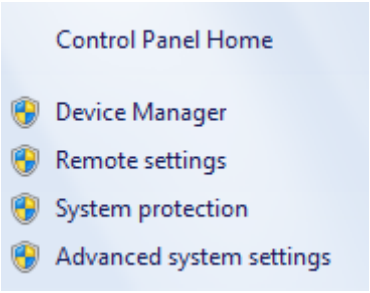
You must accept the Oracle Binary Code License Agreement for Java SE to download this software.

Accept License Agreement Decline License Agreement

Product / File Description	File Size	Download
Linux x86	146.85 MB	jdk-8u40-linux-i586.rpm
Linux x86	166.82 MB	jdk-8u40-linux-i586.tar.gz
Linux x64	145.14 MB	jdk-8u40-linux-x64.rpm
Linux x64	165.19 MB	jdk-8u40-linux-x64.tar.gz
Mac OS X x64	221.89 MB	jdk-8u40-macosx-x64.dmg
Solaris SPARC 64-bit (SVR4 package)	131.59 MB	jdk-8u40-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	92.83 MB	jdk-8u40-solaris-sparcv9.tar.gz
Solaris x64 (SVR4 package)	139.45 MB	jdk-8u40-solaris-x64.tar.Z
Solaris x64	95.82 MB	jdk-8u40-solaris-x64.tar.gz
Windows x86	175.71 MB	jdk-8u40-windows-i586.exe
Windows x64	180.19 MB	jdk-8u40-windows-x64.exe

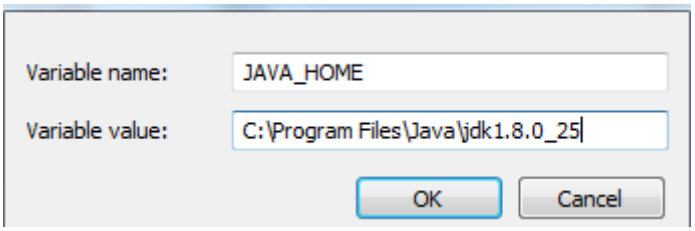
Joonis 3 - Oracle SDK valikud lähtuvalt operatsioonisüsteemist

Peale allalaadimist tuleb installida JDK ning seejärel peab JDK asukoha ära määrama süsteemi seadetes (System settings). Süsteemi seadmed leiab üles Control Panel alt, valides sealsest loendist System. Vasakul listist tuleb valida Advanced system settings (Joonis 4).



Joonis 4 - Control Panel valikud

Avanenud aknast tuleb valida all paremas nurgas olev Environment Variables. Järgnevas aknas tuleb lisada uus system variable. Esimesse lahtrisse kirjutatakse JAVA_HOME ning teise lahtrisse JDK asukoht (Joonis 5).



Joonis 5 - JAVA_HOME variable lisamine

Edasi tuleb avada genereeritud PATH variable ning lisada selle lõppu JDK bin kaust –;C:\Program Files\Java\jdk1.8.0_25\bin

Peale sisestusi tuleb kõikides akendes, mis avanesid peale Control Panelit, vajutada OK. Nüüd on vaja kontrollida, kas JDK aktsepteeritakse arvuti poolt. Avades CMD, kirjutades käsu javac –versioon ning vajutades enter, ilmub ekraanile java versioon (Joonis 6). Kui seda ei ilmu ning esineb veateade (Joonis 7), siis on eksitud sisestamisega. Eeldatavasti on PATH variable juurde lisatud kirjaveaga JDK, mistõttu tuleb üle kontrollida, kas JDK versioon vastab PATH all olevalt JDK versioonile.

```
C:\Users\Toomas>javac -version
javac 1.8.0_25
```

Joonis 6 - Java versiooni kuvamine

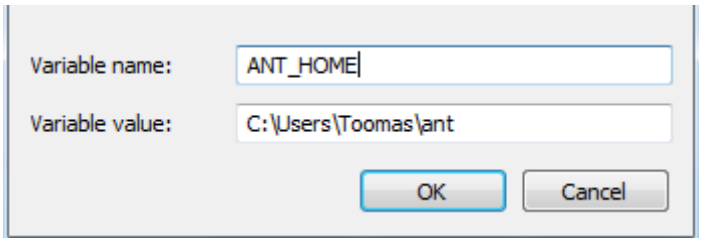
```
C:\Users\Toomas>javac -version
'javac' is not recognized as an internal or external command,
operable program or batch file.
```

Joonis 7 - Java versioon veateade

2.2 Apache Ant

Android SDK vajab rakenduste käivitamiseks lisaks JDK'le ka Apache Ant'i, mis võimaldab üles ehitada Java tarkvara. Apache Ant saab alla laadida lehelt http://ant.apache.org/bindownload.cgi. Soovituslikult tuleks valida viimati välja antud Ant zip faili kujul. Hetkel viimaseks versiooniks on 1.9.4. Peale alla laadimist tuleb zip fail lahti pakkida ning soovitavalt viia C kettale lähemale, et PATH link ei tuleks liiga pikk. Peale mainitud tegevusi tuleb nüüd lisada PATH'i juurde Ant bin kaust ning luua uus variable ANT_HOME.

Selleks avame Control Panelist System, sealt alt valik Advanced system settings ning lõpuks Environment Variables. Esimeseks avame PATH valiku ning lisame sellele lõppu Ant bin kausta asukoha – näiteks autoril asub see kasutas C:\Users\Toomas\ant\bin. Peale seda loome uue Variable ning paneme nimeks ANT_HOME ning sellele anname väärtuseks C:\Users\Toomas\ant. (Joonis 8)



Joonis 8 - ANT_HOME variable lisamine

Peale sisestusi tuleb kõikides akendes, mis avanesid peale Control Panelit vajutada OK. Nüüd tuleb kontrollida, kas Ant on arvuti poolt aktsepteeritud. Avades CMD ning kirjutades käsu ant –version ning vajutades enter ilmub ekraanile ant versioon (Joonis 9), kuid kui seda ei ilmu ja esineb veateade (Joonis 10), siis on sisestamisel läinud midagi valesti. Eeldatavasti sai PATH variable juurde lisatud Ant kaust kirjaveaga.

```
C:\Users\Toomas>ant -version
Apache Ant(TM) version 1.9.4 compiled on April 29 2014
```

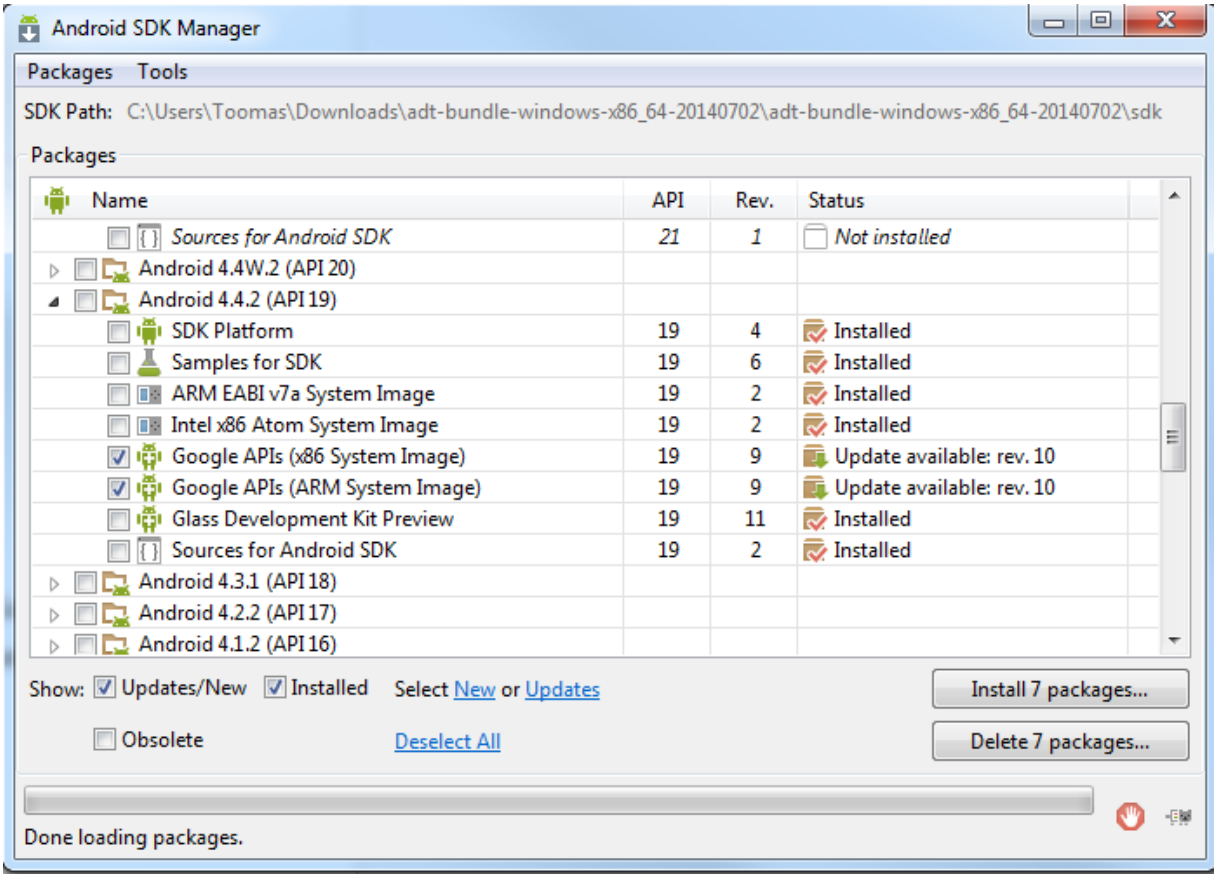
Joonis 9 - Ant versiooni kuvamine

```
C:\Users\Toomas>ant -version
'ant' is not recognized as an internal or external command,
operable program or batch file.
```

Joonis 10 - Ant versiooni veateade

2.3 Android SDK

Eclipse/Android SDK on nüüd nimetusega Android Studio ning on kättesaadav Android veebilehelt. Kui allalaadimine on lõppenud, tuleb antud .exe fail käivitada, et installida Studio arvutisse. Järgnevalt tuleb tööle panna Android Studio ning sealt avada Android SDK Manager (Joonis 11).



Joonis 11 - Android SDK Manager vaade

SDK Managerist tuleb üles leida Android 4.4.2 (API 19) package, mis on vaja lisada SDK'le, sest PhoneGap nõuab API 19 (Joonis 12).

```
Error: Please install Android target 19 (the Android newest SDK). Make sure you
have the latest Android tools installed as well. Run "android" from your command
-line to install/update any missing SDKs or tools.
```

Joonis 12 - PhoneGap veateade Android API 19 puudumise kohta

Peale API 19 installimist tuleb Windows PATH'i juurde lisada Android SDK platform-tools ning tools kaust. Selleks avame Control Panelist System, sealt edasi valime Advanced system settings ning siis Environment Variables. Seal avame valiku PATH ning lisame selle lõppu

eelpool mainitud kahe kausta asukohad. Näiteks, autoril asuvad need kasutad järgnevalt: C:\Android Studio\sdk\platform-tools ja C:\Android Studio\sdk\tools.

2.4 Node.js

PhoneGap rakenduse ehitamiseks ning tööle panemiseks on vaja arvutisse installida Node js. Node annab võimaluse installida arvutisse PhoneGap vajalikud lisandid. Node kasutab npm käsku programmis CMD, et välja kutsuda JavaScripti pakett üle interneti. Nmp on vaikimisi mõeldud kasutamaks Node.js'. Lisaks annab nmp võimaluse installida kasutajatel Node.js rakendusi, mis on kättesaadavad npm registris.

Node.js CMD kasutamiseks tuleb installimisel sisestada järgnev käsk: npm install. Selle järgi kirjutada mooduli nimi. Autori poolt loodava rakendusele sisestatakse järgnevad käsud: npm install –g cordova (Joonis 13) või npm install –g phonegap (Joonis 14).

```
C:\Users\Toomas>npm install -g cordova
```

Joonis 13 - Cordova installatsiooni näide

```
C:\Users\Toomas>npm install -g phonegap
```

Joonis 14 - PhoneGap installatsiooni näide

Edaspidi kasutab PhoneGap Cordova sisemisi funktsioone, alates rakenduste käivitamisest kuni rakendustele lisaväärtuste lisamiseni.

Rakenduse käivitamiseks tuleb anda käsk run (Joonis 15)

```
C:\Users\Toomas\PG rakendused\my-app>phonegap run --device
```

Joonis 15 - PhoneGap rakenduse käsk run

Lisades rakendusele platformi, plugina või muu lisandi, tuleb kasutada käsku add (Joonis 16, Joonis 17)

```
C:\Users\Toomas\PG rakendused\my-app>phonegap platform add android
```

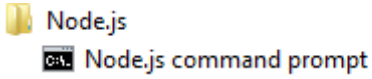
Joonis 16 - PhoneGap rakenduse käsk add

```
C:\Users\Toomas\PG rakendused\my-app>cordova plugin add org.apache.cordova.network-information
```

Joonis 17 - Cordova plugin käsk add

2.5 Cordova ja PhoneGap

Peale Node.js installimist tuleb avada Node.js command prompt, mille asukoht on järgmne: Start → All Programs → Node.js → Node.js command prompt (Joonis 18).



Joonis 18 – Node.js kaust Start all

Cordova installimiseks tuleb sisestada käsk npm install –g cordova ning vajutada enter. Algselt annab Node.js kuni 3 hoiatust, kuid need ei mõjuta sellel hetkel programmi tööd. Peale lõpetamist peaks Node väljastama tulemuse, mis on näidatud joonisel 19.

```
C:\Users\Toomas>npm install -g cordova
npm WARN engine cordova-js@3.8.0: wanted: {"node":"~0.10.x"} (current: {"node":"0.12.0","npm":"2.5.1
"})
npm WARN engine npm@1.3.4: wanted: {"node":">=0.6","npm":"1"} (current: {"node":"0.12.0","npm":"2.5.
1"})
npm WARN engine xmlbuilder@2.2.1: wanted: {"node":"0.8.x || 0.10.x"} (current: {"node":"0.12.0","npm
":"2.5.1"})
C:\Users\Toomas\AppData\Roaming\npm\cordova -> C:\Users\Toomas\AppData\Roaming\npm\node_modules\cord
ova\bin\cordova
cordova@4.3.0 C:\Users\Toomas\AppData\Roaming\npm\node_modules\cordova
├── underscore@1.7.0
├── q@1.0.1
├── nopt@3.0.1 (abbrev@1.0.5)
└── cordova-lib@4.3.0 (valid-identifier@0.0.1, osenv@0.1.0, properties-parser@0.2.3, bplist-parser@0
.0.6, mime@1.2.11, unorm@1.3.3, semver@2.0.11, dep-graph@1.1.0, shelljs@0.3.0, rc@0.5.2, npmconf@0.1
.16, through2@0.6.3, elementtree@0.1.5, glob@4.0.6, d8@0.4.4, xcode@0.6.7, request@2.47.0, init-pack
age-json@1.3.0, tar@1.0.2, plist@1.1.0, npm@1.3.4, cordova-js@3.8.0)
```

Joonis 19 – Cordova installatsioon

Järgmisena, et installida PhoneGap, tuleb samas promptis sisestada käsk npm install –g phonegap ning vajutada enter. Lõpptulemuseks saame järgmise tulemuse, mis on kujutatud joonisel 20.

```
C:\Users\Toomas\AppData\Roaming\npm\phonegap -> C:\Users\Toomas\AppData\Roaming\npm\node_modules\pho
negap\bin\phonegap.js
phonegap@4.2.0-0.24.2 C:\Users\Toomas\AppData\Roaming\npm\node_modules\phonegap
├── pluralize@0.0.4
├── colors@0.6.0-1
├── semver@1.1.0
├── minimist@0.1.0
├── qrcode-terminal@0.9.4
├── shelljs@0.1.4
├── phonegap-build@0.9.1 (colors@0.6.2, qrcode-terminal@0.8.0, shelljs@0.0.9, optimist@0.3.7, phoneg
ap-build-api@0.3.3)
├── prompt@0.2.11 (revalidator@0.1.8, pkginfo@0.3.0, read@1.0.5, winston@0.6.2, utile@0.2.1)
├── cordova@4.2.0 (underscore@1.7.0, q@1.0.1, nopt@3.0.1, cordova-lib@4.2.0)
└── connect-phonegap@0.14.8 (home-dir@0.1.2, connect-inject@0.3.2, ip@0.3.1, ncp@0.6.0, findit@2.0.0
, request-progress@0.3.1, shelljs@0.2.6, http-proxy@1.8.1, request@2.33.0, gaze@0.4.3, tar@0.1.19, n
ode-static@0.7.0, archiver@0.10.1, localtunnel@1.3.0, useragent@2.0.8, connect@2.12.0, socket.io@1.0
.4)
```

Joonis 20 – PhoneGap installatsioon

3 Esmase PhoneGap rakenduse loomine

Peale eelmises peatükis tehtud installimisi on arvuti seadistused valmis looma esimest PhoneGap rakendust. Loome kettale uue kausta, näiteks nimetusega PG rakendused, ning jätkates tegevust Node.js-ga, tuleb suunduda äsjaloodud uude kausta. Selleks sisestame käsu cd ja uue kasuta nimetus. (Joonis 21)

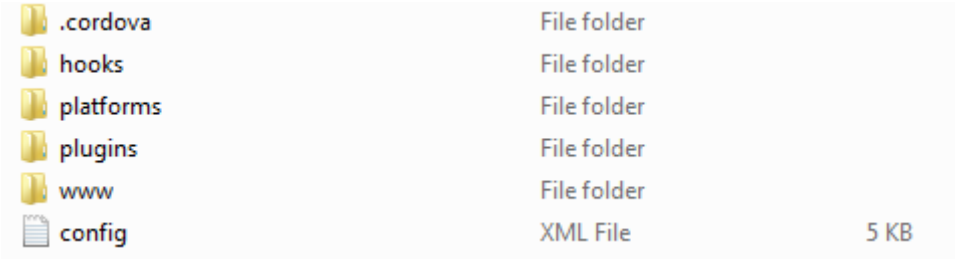
```
C:\Users\Toomas>cd "PG rakendused"

C:\Users\Toomas\PG rakendused>
```

Joonis 21 – loodud PG rakendused kausta suundumine

Edasi sisestame PhoneGap käsu, mis loob esmase valmisrakenduse. Rakenduse käsk on phonegap create my-app com.myapp.myapp MY_APP. Rakendus loob PG rakendused kausta uue kausta, mis saab nimeks my-app (Joonis 22).

Joonis 22 – rakenduse loodud my-app kaust

Selles kaustas on nüüd olemas põhi, mida saab kasutada, et jätkata rakenduse loomise protsessi. My-app kausta loodi juurde 5 alamkausta (Joonis 23), mida hakkavad kasutama PhoneGap ja Cordova, et ehitada rakendust emulaatori või seadme peale. Kaustas www on native rakenduse jaoks vajalikud HTML, CSS ja JS failid.



Joonis 23 – my-app kausta alamkaustad

Alguses ei looda rakendusele ühtegi platvormi, millel rakendus töötaks ehk platforms kaust on tühi. Platvormi lisamiseks tuleb Node.js CMD's minna kasuta cd my-app ning seejärel sisestada käsk phonegap platform add android. Nüüd on my-app rakendusel olemas Android platvorm (Joonis 24). Tulemuseks saame, et Android projekt on edukalt loodud ning failid on lisatud kausta platforms/android.

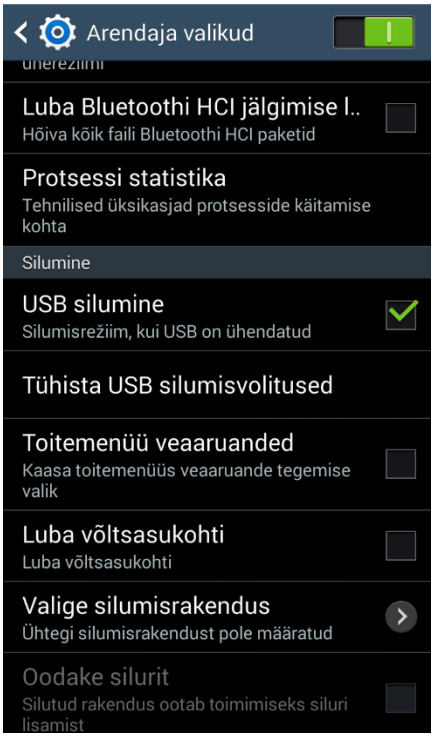
```
C:\Users\Toomas\PG rakendused\my-app>phonegap platform add android
Creating android project...

Creating Cordova project for the Android platform:

        Path: platforms\android
        Package: com.myapp.myapp
        Name: MY_APP
        Android target: android-19
Copying template files...

Project successfully created.
```

Joonis 24 – Android platvormi lisamine my-app rakendusele

Veendumaks, et platvorm on lisatud, tuleb kontrollida platforms kaustas alamkausta android olemasolu (Joonis 25).

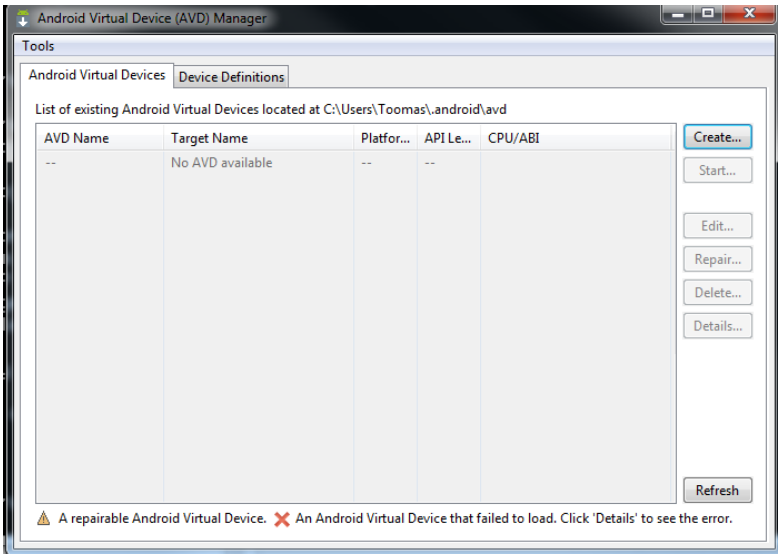
Joonis 25

Järgmise etapina tuleb kasutusele võtta emulaator või seade, millel proovida my-app rakendust. Seadmel peab olema avatud arendaja valik ning sisse lülitatud USB silumisvõimalus (Joonis 26), sest see võimaldab rakenduse installimist seadmesse.



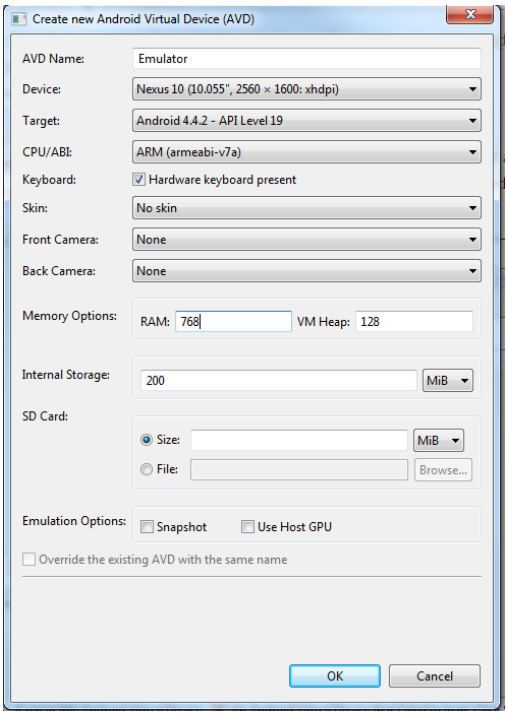
Joonis 26 – Android silumise lisamine telefonile

Emulaatoril rakenduse käima panemiseks tuleb esmalt üles seada virtuaalne android seade. Node.js'i tuleb sisestada käsk android avd, mis avab Android stuudios Android Virtual Device (AVD) Manager'i (Joonis 27). Uue seadme loomiseks tuleb vajutada Create nuppu.

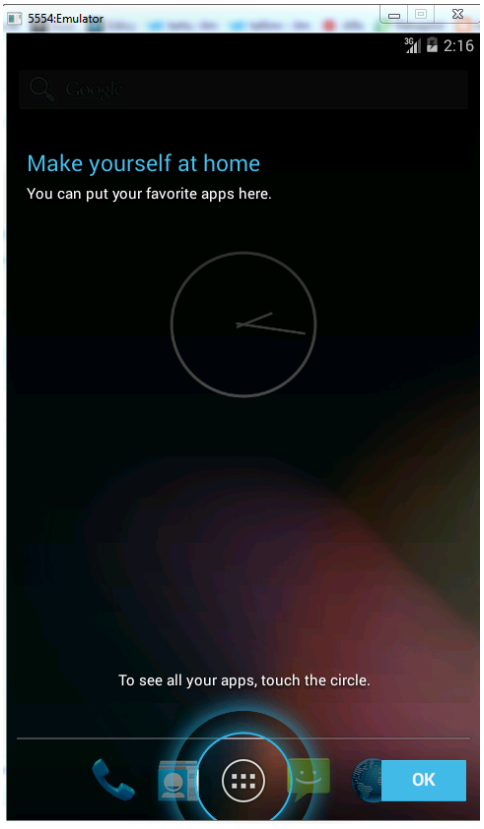


Joonis 27 – Android Virtual Device aken

Avanenud aknast (Joonis 28) tuleb anda seadmele nimi ning valida seade, kus hakatakse rakendust vaatama. Seejärel tuleb valida seadme Android versioon ning CPU andmed. Viimaseks nõudeks võib valida Skin asemel No Skin, sest selles kontekstis ei ole rakenduse loomisel vaja üleliigseid protsessori ressursi nõudvaid lisandeid. Soovituslikult on hea valida RAM suuruseks 768M. Lõpuks, vajutada nuppu OK ning uus seade on valmis. Peale seadme loomist tuleb emulaator käivitada, vajutades nuppu Start (Joonis 29).



Joonis 28 - Emulaatori loomine



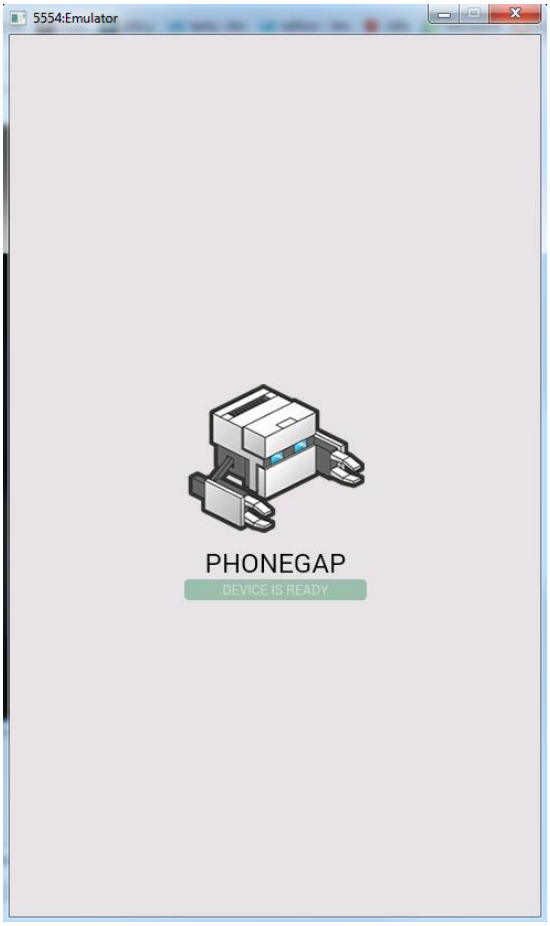
Joonis 29 – Emulaator käivitamisel

Seadme käivitamisel anname Node.js's käsu, mis installeeriks rakenduse emulaator/seadme peale. Emulaatori jaoks on käsk phonegap run --emulator (Joonis 30) ning seadme jaoks käsk phonegap run --device. Kuna PhoneGap töötab Cordova peal, siis PhoneGap teeb ise teise käsu ning kutsub välja Cordova samanimelise käsu PhoneGap asemel.

```
C:\Users\Toomas\PG rakendused\my-app>phonegap run --emulator
[phonegap] executing 'cordova run --emulator'...
[phonegap] completed 'cordova run --emulator'
```

Joonis 30 – rakenduse käivitamine run käsuga

Peale completed cordova run –emulator'it on seadmes kuvatud tulemus algsest PhoneGap rakendusest (Joonis 31).



Joonis 31 – esmane töötav rakendus emulaatoris

4 PhoneGap võimalused

PhoneGapil on ulatuslikud native-rakenduste loomise võimalused. Antud peatükis toob autor välja, kuidas on see võimalik ning millised on kasutatavad lisavidinad, mis aitavad arendajal programmi rakendada.

4.1 Java + Parse kasutamine

Parsel on olemas Android dokumentatsioon, mis aitab arendajal leida võimaluse kuidas Java ja andmebaasi Parse ühendada. Dokumentatsioonis on välja toodud näited erinevatest võimalustest.

Parse rakendamiseks tuleb kasutada privaatseid võtmeid, mille Parse genereerib. Parse andmete küsimiseks Android seadmes on lihtne funktsioon (Koodinäide 1):

```
ParseQuery<ParseObject> query = ParseQuery.getQuery("GameScore");
query.whereEqualTo("playerName", "Dan Stemkoski");
query.findInBackground(new FindCallback<ParseObject>() {
   public void done(List<ParseObject> scoreList, ParseException e) {
       if (e == null) {
           Log.d("score", "Retrieved " + scoreList.size() + " scores");
       } else {
           Log.d("score", "Error: " + e.getMessage());
       }
   }
});
```

Koodinäide 1

Koodinäites ParseQuery<ParseObject> luuakse uus objekt, kuhu lisatakse kogu andmestik, mida küsitakse andmebaasi sees olevast tabelist GameScore: ParseQuery.getQuery("GameScore");.

Kui on vaja kätte saada kindla nimega isikut antud tabelist ja teada tema tulemust, siis kasutame koodi query.whereEqualTo(‘tulba nimi’, ‘otsisõna’), mis paneb võrdlusesse antud tabeli andmed vastavast tulbast otsisõna järgi.

`query.findInBackground(new FindCallback<ParseObject>()` annab võimaluse lasta programmil või rakendusel töötada selliselt, et saaks kasutada ka teisi Java funktsioone samal ajal. FindCallback annab vastuse antud soovile objekti kujul, kuid töödeldes andmed ümber,

kuvab saadud tulemust listina. Antud näide oli vaid ühest kutsest Parse andmebaasist, kasutades Java't.

4.2 PhoneGap pluginad

PhoneGapil on sisse kirjutatud 10 pluginat. Autor toob välja nendest kaks, mida kindlasti kasutatakse bakalaureuse töö koostamisel. Lisaks toob autor välja probleemid, mis võivad tekkida antud lahenduste juures, kui rakendust luua esimesel korral. Analüüsis kasutan 11.02.2015 tunnis läbi viidud katsete tulemusi.

4.2.1 Internetiühendus ehk Network Information

Network Information plugin võimaldab rakendusel teada saada, kas seade on ühendatud internetivõrku või mitte. Antud plugin tunnistab seadme poolt tagastatud suurusi: tundmatu ühendus, kaabel ühendus, WiFi ühendus, 2G, 3G, 4G ühendus, WAP ühendus ning ühendus puudub.

Plugina lisamiseks tuleb käsurealt kutsuda välja käsk, mis lisaks antud vidina rakendusele. Käsuks on add funktsioon, mis kutsub välja lisamisvõimaluse rakendusele - cordova plugin add org.apache.cordova.network-information. Antud pugina jaoks on tarvilik sisse lülitada interneti ühendus.

Näide, kuidas saada kätte ning kuvada ekraanile teave, millise ühendusega on tegu (Koodinäide 2).

```
function checkConnection() {
   var networkState = navigator.connection.type;

   var states = {};
   states[Connection.UNKNOWN]  = 'Unknown connection';
   states[Connection.ETHERNET] = 'Ethernet connection';
   states[Connection.WIFI]     = 'WiFi connection';
   states[Connection.CELL_2G]  = 'Cell 2G connection';
   states[Connection.CELL_3G]  = 'Cell 3G connection';
   states[Connection.CELL_4G]  = 'Cell 4G connection';
   states[Connection.CELL]     = 'Cell generic connection';
   states[Connection.NONE]     = 'No network connection';

   alert('Connection type: ' + states[networkState]);
}

checkConnection();
```

Koodinäide 2

navigator.connection.type annab networkState'le väärtuseks objekti.

Funktsiooniga addEventListener'ga saab hoida interneti ühenduse tegevusel silma peal ning plugin omab kahte staatust, mida saab võrrelda – 'online', 'offline'. - ja siis vastavalt olekule välja kutsuda funktsioonid (Koodinäide 3).

```
document.addEventListener("online", onOnline, false);

function onOnline() {
   // Handle the online event
}
document.addEventListener("offline", onOffline, false);

function onOffline() {
   // Handle the offline event
}
```

Koodinäide 3

Kasutame antud võimalust ka näidisrakenduse peal, mis eelnevalt sai loodud ning kuvame tulemused illustreerivate piltidega. Esmalt tuleb rakendusele lisada plugin, kasutades Node.js käsku phonegap plugin add org.apache.cordova.network-information (Joonis 32).

```
C:\Users\Toomas\PG rakendused\my-app>phonegap plugin add org.apache.cordova.network-information
Fetching plugin "org.apache.cordova.network-information" via plugin registry

Installing "org.apache.cordova.network-information" for android
```

Joonis 32 – Interneti ühenduse plugin install

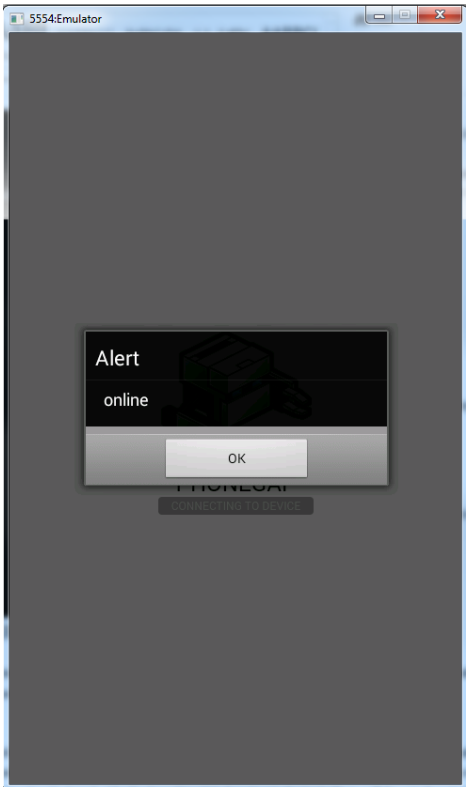
Peale plugina paigaldamist avame rakenduse my-app js kaustast index.js fail. Avatud javascriptile lisame juurde funktsiooni checkConnection().

Peale `checkConnection()` lisame koodile juurde funktsoonid `document.addEventListener( "online", onOnline, false);` ning `document.addEventListener( "offline", onOffline, false);`. Seejärel lisame teised kaks funktsiooni, kus rakenduse käivitamisel antakse teada, kas interneti ühendus seadmes on sees või väljas.

```
function onOnline() {
   alert('online');
}
function onOffline() {
   alert('offline');
}
```

Rakenduse käivitamisel annab seade tagasisidet eelnevalt välja toodud funktsioonidele. Kui rakendus tuvastab internet ühenduvuse, saadab seade välja esimese teate (Joonis 33) ning

seejärel teeb kindlaks, millise ühendusega on tegu (Joonis 34). Emulaatorid kasutavad internetti sisse ehitatud 3G ühendusena.



Joonis 33 – Interneti olemasolu teade

Joonis 34 – Interneti ühenduse liik

Sellist tulemust ei õnnestunud tunni näitel kätte saada, sest seade ei saanud kätte vajalikke andmeid. Javascripti kirjutamisel tekkis probleemseid kohti, näiteks ilmnes koma viga või küsitav funktsioon ei tulnud kohe esile. Selle tulemusena rakendus ei jõudnudki vajaliku koodini ning ei tagastanud soovitavat tulemust. Teisalt juhtus ka seda, et seade ei jõudnud staadiumisse deviceReady ning isegi kui tulemused oleksid olnud kättesaadavad, siis seade ei andnud tagastavat teavet selle kohta. Probleemi lahendamiseks tuli koodi mitmel juhul mitu korda üle lugeda ning vaadata, et seal ei esineks vigu. Teiseks lahenduseks oli rakenduse uuesti loomine ning seejärel eelnevat koodi proovida.

4.2.2 Positsioneerimine ehk Geolocation

PhoneGap arendajate poolt on loodud eraldi plugin, mis võimaldab rakendusel kätte saada seadme positsioon kaheksa erineva muutujaga: laius- ja pikkuskraadid, kõrguse merepinnast, täpsus suuruse koordinaatide kohta, täpsus suuruse kõrguse kohta, suuna, kiiruse ja aja. Plugina installimiseks tuleb käsurealt kutsuda välja käsk, mis automaatselt installiks vastava plugina rakendusele juurde. Käsuks on add funktsioon, mis kutsub välja lisamisvõimaluse

rakendusele - cordova plugin add org.apache.cordova.geolocation. Selleks, et plugin töötaks peab kindlasti olema sisse lülitatud seadme GPS.

Geolocationil on 3 erinevat funktsiooni antud pluginale: getCurrentPosition, watchPosition, clearWatch. getCurrentPosition küsib ühekorra seadmelt selle GPS koordinaadid, näiteks seadme käivitamisel kutsutakse esmakordselt funktsioon välja ja sellega piirdutakse. watchPosition seab seadmele tsüklilise kutsungi, mis peale igat määratud aega nullib ära koordinaadid ning küsib siis seadmelt uuesti. See oleks vajalik, näiteks soovides kuvada visuaalselt kaardile seadme liikumist. Antud võimalusel on võimalik lisada kolm erinevat suurust: maximumAge, timeout ning enableHighAccuracy (Koodinäide 4).

```
var watchID = navigator.geolocation.watchPosition(
    onSuccess, //õnnestunud funktsiooni välja kutsumine
    onError, //ebaõnnestunud funktsiooni välja kutsumine
    {
        maximumAge: 3000, // asukoha punkti maksimum aeg mil seda hoitakse
mälus
        timeout: 5000,  // tsüklite pikkus eelneva kordusega
        enableHighAccuracy: true
    }
);
```

Koodinäide 4

Kasutame antud võimalust ka näidisrakenduse peal. Kuvame tulemused illustreerivate piltidega. Esmalt tuleb rakendusele lisada plugin, kasutades Node.js käsku phonegap plugin add org.apache.cordova.geolocation (Joonis 35).

```
C:\Users\Toomas\PG rakendused\my-app>phonegap plugin add org.apache.cordova.geolocation
Fetching plugin "org.apache.cordova.geolocation" via plugin registry

Installing "org.apache.cordova.geolocation" for android
```

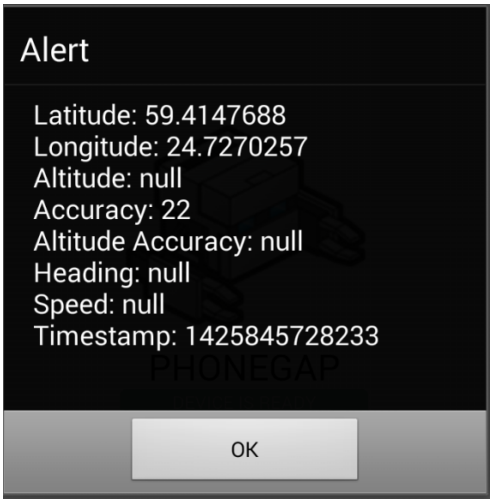
Joonis 35 – Geolocation installatsioon

Peale plugina paigaldamist avame rakenduse my-app js kaustast index.js fail. Avatud javascriptile lisame juurde funktsioonid onSuccess ja onError (Koodinäide 5).

```
var onSuccess = function(position) {
    alert('Latitude: '          + position.coords.latitude          + '\n' +
          'Longitude: '         + position.coords.longitude         + '\n' +
          'Altitude: '          + position.coords.altitude          + '\n' +
          'Accuracy: '          + position.coords.accuracy          + '\n' +
          'Altitude Accuracy: ' + position.coords.altitudeAccuracy  + '\n' +
          'Heading: '           + position.coords.heading           + '\n' +
          'Speed: '             + position.coords.speed             + '\n' +
          'Timestamp: '         + position.timestamp                + '\n');
};

// onError Callback receives a PositionError object
//
function onError(error) {
    alert('code: '    + error.code    + '\n' +
          'message: ' + error.message + '\n');
}

navigator.geolocation.getCurrentPosition(onSuccess, onError);
```

Koodinäide 5

Rakenduse käivitamisel annab seade tagasisidet eelnevalt välja toodud funktsioonidele. Kui rakendusel on GPS sees annab vastuseks teate seadme asukoha kohta (Joonis 36).



Joonis 36 – Geolocationi tagastusteade

4.3 PhoneGap Events

PhoneGap omab 10 erinevat sisse programmeeritud Event tüüpi: deviceready, pause, resume, backbutton, menubutton, searchbutton, startcallbutton, endcallbutton, volumedownbutton, volumeupbutton. Antud rakenduse juures on olulisteks Event tüüpideks deviceready, backbutton, menubutton.

4.3.1 Event - deviceready

Antud sündmus on oluline iga Cordova rakenduse jaoks. See annab märku, et Cordovat toetavas seadmes on rakendusliides laetud ja valmis kasutamiseks.

Cordova koosneb kahest koodi alusest: native ja JavaScript. Senikaua kuni native kood laeb, kuvatakse ekraanil vaikeväärtusega määratud pilti. Kuid JavaScript laeb end vaid korra, kui DOM end seadistab. See tähendab, et veebi rakenduses võib potentsiaalselt kasutada Cordova JavaScript funktsiooni enne kui vastava native kood muutub kättesaadavaks.

Deviceready juhul annab korra teate, kui Cordova on täielikult laetud. Kui event käivitatakse, võib probleemideta teha päringuid Cordova rakendusliidetest. Rakendustele tuleb tavaliselt lisada juurde kuulaja (Listener) - document.addEventListener - üks kord kui HTML'i DOM on laetud.

Deviceready korral käitub kuulaja teistest funktsioonidest erinevalt. Iga event handler, mis on registreeritud pärast deviceready käivitatamist, omab callback funktsiooni, mida kutsutakse välja koheselt (Koodinäide 6).

```
document.addEventListener("deviceready", onDeviceReady, false);

function onDeviceReady() {
   // Now safe to use device APIs
}
```

Koodinäide 6

4.3.2 Event - backbutton & menubutton

Backbutton ja menubutton on mõeldud Amazon Fire OS, Android, BlackBerry 10 (backbutton ka Windows Phone 8) platvormidele, kuna nimetatud seadmetel on olemas vastavad nupud.

Backbutton'il on võimalus kutsuda välja funktsioon, näiteks enne rakendusest väljumist, välistades juhtu, et seadmes kogemata suletakse antud rakendus. Funktsiooniks piisab lihtne alert funktsioon (Koodinäide 7).

```
document.addEventListener("backbutton", onBackKeyDown, false);

function onBackKeyDown() {
   alert('Are you sure?');
}
```

Koodinäide 7

Funktsiooni püüab kinni addEventListener. Listeneri esimene väärtus backbutton'iga tuvastab, millist eventi peaks kutsuma. onBackKeyDown on success funktsioon, mis kutsutakse välja peale backbutton'i rakendamist. False listeneri lõpus on return väärtus, et takistada rakenduse edasist tööd peale nupuvajutust.

Menubutton annab võimaluse kasutada eelmainitud platvormidel menu nuppu, et rakenduses oleks võimalik peita tehnilised andmed rakenduse kohta (Koodinäide 8).

```
document.addEventListener("menubutton", onMenuKeyDown, false);

function onMenuKeyDown() {
  //handle menubutton or show the menu function
}
```

Koodinäide 8

Funktsiooni püüab kinni addEventListener. Listeneri esimene väärtus menubutton'iga tuvastab, millist eventi peaks kutsuma. onMenuKeyDown on success funktsioon, mis kutsutakse välja peale menubutton'i rakendamist. False listeneri lõpus on return väärtus, et takistada rakenduse edasist tööd peale nupuvajutust.

Kokkuvõte

Käesolevas seminaritöös tutvustatakse lühidalt bakalaureusetöö eesmärki luua rakendus Rally Estoniale, PhoneGap olemust ning Parse andmebaasi. Seminaritöö eesmärgiks oli teha ära eelinstallatsioonid bakalaureusetööks. Lisaks installatsioonidele tõi autor töös välja ka võimalused, kuidas rakenduses kasutada PhoneGap poolt olemasolevat kahte lisavidinat (plugin) ning kolme sündmust (Event) .

Seminaritöös tegi autor stamprakenduse, milles tõi illustreerivalt välja esmase rakenduse loomise tervikprotsessi algusest rakenduse tööle panemiseni. Vaheetappidena toodi välja kuidas rakendust luua, lisada sellele arendusplatvorm ning panna tööle nii seadmes kui ka emulaatoris.

Seminaritöö põhieesmärk sai täidetud. Installeerides vajalikud programmid ning lisad arvutisse. Antud tegevus annab suure panuse bakalaureusetöö alustamiseks. Seminaritöös tõi autor välja kaks vajalikku lisavidinat, mida PhoneGap võimaldab, ning testis neid erinevatel seadmetel ja emulaatoris, tuues välja saadud tulemused illustreerival kujul.

Saadud informatsioon aitab töö autoril luua esmane prototüüp Rally Estonia piletimüügi rakendusest. Rakendus hõlmab seminaritöös välja toodud funktsioone ning aitab kasutada neid rakenduse loomisel.

Kasutatud kirjandus

Apache Cordova. (2015). *Cordova Geolocation Plugin*. Allikas: http://plugins.cordova.io/#/package/org.apache.cordova.geolocation

Apache Cordova. (2015). *Cordova Network Information Plugin*. Allikas: http://plugins.cordova.io/#/package/org.apache.cordova.network-information

Apache Cordova. (2015). *Cordova Plugins Registry*. Allikas: http://plugins.cordova.io/

Oracle. (2015). *Java SE Downloads*. Allikas: http://www.oracle.com/technetwork/java/javase/downloads/index.html

Parse. (2015). *Parse Core*. Allikas: https://parse.com/products/core

PhoneGap. (2015). *PhoneGap About*. Allikas: http://phonegap.com/about/

PhoneGap. (2015). *PhoneGap Documentation*. Allikas: http://docs.phonegap.com/en/4.0.0/index.html

PhoneGap. (2015). *Supported Features*. Allikas: http://phonegap.com/about/feature/