TALLINNA ÜLIKOOL

Informaatika instituut

Vidinate tehnoloogia Windows keskkonnas tudengi ajaplaneerimise näitel

Bakalaureusetöö

Koostaja: Alexey Gudz

Juhendaja: Erika Matsak

Autor: ..... ,, ... ,,2011

Juhendaja: ..... ,, ... ,,2011

Instituudi direktor: ..... ,, ... ,,2011

Tallinn 2011

Autorideklaratsioon

Deklareerin, et käesolev bakalaureusetöö 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

Kiiresti arenevas veebimaailmas võib tihti vaadelda, et mingi tehnoloogia, mis kogub populaarsust, hakkab muutuma, et olla kättesaadav laiemale seltskonnale. Sellise nähtuse võib märgata ka vidinate tehnoloogias. Need objektid, mis kannavad erinevaid nimesid nagu "*Widgets*", "*Gadgets*", "*Desklets*", "*Screenlets*", "*Plasmoids*" ja muid, on arenenud nii kõvasti, et nad jõudsid nii Windows, *Mac* ja *Linux* operatsioonisüsteemidele, nii veebisaitidele, kui ka arvuti töölauale, samuti mobiiltelefonidesse, ning isegi meie televiisoritesse. Selles töös aga keskendutakse Windows töölaua vidinatele.

Töölaua vidinad on interaktiivsed virtuaalsed tööriistad, mis pakuvad mingit konkreetset teenust. See teenus võiks olla ilma, aega, kalendri, piltide galerii, meelespea või mingi muu kasuliku või huvitava informatsiooni näitamine. Võimalusi on tõesti palju. Seega autor arvab, et vidinate loomine ja arendamine on praegu väga aktuaalne ja huvitav teema, ning just sellel põhjusel sai valitud bakalaureusetööks.

Töö käigus autor räägib erinevatest vidinate aspektidest, tutvustab lugejatele mõningaid vidinate platvorme, nende eeliseid ja puudusi, puudutab vidinate arendamise teemat, ning ise loob näidisvidinat, kirjeldades selle loomise protsessi.

Selle bakalaureusetöö eesmärgiks on anda põhjalikku ülevaadet vidinatest, ning vidinate platvormidest, analüüsida ja võrrelda erinevaid vidinate programme ehk platvorme, ja, kasutades üht neist, luua kasulikku rakendust, mida saaksid kasutada ka tudengid.

1 Vajaduste väljaselgitamine

Enne vidinate tutvustamise alustamist proovin vastata küsimusele “Miks on see töö vajalik?”.

Vidinad ise on väga populaarne asi - mõnedel vidinate programmidel pakutakse alla laadimiseks ning kasutamiseks tuhandeid neid. Huvitav on aga see, et õppematerjale nende kohta on äärmiselt vähe. Raamatud on täiesti puudu. Paremal juhul võib leida internetis paarkümmend väikest artiklit, aga enamus pakub infot ainult ühe konkreetse vidina programmi kohta. See on, mis puutub inglise keelseid allikaid, - eesti keeles on veel kurvem lugu. Siis tuli mõte, et kuna vidinad on tunduvalt aktuaalne asi, ning vidinate programme on päris mitu tükki, oleks hea idee pakkuda eestikeelset infot vidinate, ning erinevate platvormide kohta, et kasutajad saaksid paremini selles orienteeruda, ning endale õige programmi valida.

Mitte vähem tähtis on ka vidinate arendamise aspekt. Kerge arendamine on see, mis teeb vidinaid nii edukateks. Seetõttu tundus loogilisena tutvustada lugejatele vidina loomist - selleks ma arendan ühe kasulikku vidinat ise, ning kirjeldan selle protsessi.

Alguses oli kavandatud luua Tallinna Ülikooli tudengi vidinate paketti. Plaani järgi oli vaja diskussiooni või küsitluse abil välja uurida, mis oleks kõige kasulikum funktsionaalsus vidinate paketis, mis oleks seotud Tallinna Ülikooliga. Variantidena võiksid olla kalendri ja tunniplaani näitav vidin, TLÜ uudiste või sündmuste voo lugeja, TLÜ posti lugev vidin, akadeemiline kalender ja muud. Kahjuks tuli välja, et sellise funktsionaalsuse realiseerimine on võimatu või väga ebamõistlik. Kuna TLÜ ilmselt ei paku mingit infot *XML* või *JSON* formaadis, siis puudub võimalus vidinatel midagi saidilt kätte saada. Kohe sai eesmärgiks välja mõelda sellise funktsionaalsusega vidinat, mis oleks kasulik TLÜ tudengile, ning ei võta informatsiooni ülikooli saidilt. Vastus tuli, kui leidsin Tallinna Ülikooli *Twitter* lehe, mis oli pidevalt uuendatud, ning kui tutvusin *Google Calender* teenuse võimalustega lähedamini. Kiiresti sai vastu võetud otsus luua sellise vidina, mis näitaks inimese *Google Calender* sündmusi ja Tallinna Ülikooli uudiseid töölaual asuvas kalendris.

2 Vidinad ja mootorid

2.1 Ülevaade vidinatest

Selles peatükis anname ülevaadet vidinatest, siis pühendan teid vidinate ajaloosse, tutvume populaarsemate vidinate kategooriatega, ning lõpuks seletan ära vidinate tehnilist ülesehitust.

2.1.1 Tutvustus vidinatega

Tahaks alustada kõigepealt sellest, et vidinad, nagu eelnevalt mainisin, on virtuaalsed tööriistad, mis pakuvad mingit konkreetset kasulikku funktsiooni ehk teenust. Me võime kohtuda nendega täiesti erinevates kohtades erinevatel platvormidel ja asju teeb veel hullemaks see, et nemad kannavad palju erinevaid nimesid. "*Widgets*", "*Gadgets*", "*Skins*", "*Desklets*", "*Screenlets*", "*Plasmoids*" ja muud - kõik on põhimõtteliselt sama funktsionaalsusega, aga mõeldud erinevatele sihtkasutajatele ja platvormidele objektid. Tundub, et ametlik nimi või liigitamine lihtsalt puudub, kuid "vidin" ehk "*widget*" on kindlasti enim levinud termin.

Tegelikult neid võiks jagada mobiilseteks vidinateks, mida võib näha eelkõige *Android OS* kasutavatel mobiiltelefonidel, veebi vidinateks, mis paiknevad veebisaitidel, töölaua vidinateks, mis elavad *Windows*, *Linux* ja *Mac OS* töölaudadel, ning on veel üheks uueks "*TV widgets*" kategooriaks. Selles töös keskendutakse ainult Windows töölaua vidinatele.

Üks töölaua vidinate aspekt, millest tasub kohe teada, on see, et nad ei saa sõltumatult töölaua peal elada, nad vajavad toetava programmi, mida nimetatakse "platvormiks" või "mootoriks". Sellest aga räägime hiljem.

2.1.2 Ajalugu

Töölaua vidinate vanaisaks võiks pidada "*Desk Ornaments*" projekti, millest 1984. aastal sai „*Mac Desk Ornaments*". Esimene asi, mida võiks nimetada vidinateks Windows keskkonnas, oli 1997. aastal ilmunud „*Active Desktop*" - *Internet Explorer* 4.0 brauseri tunnusjoon, mis võimaldab kasutajaid paigutada HTML koodi abil töölaua peale. Aastal 2000 ilmus *Stardock DesktopX* - väga võimas töölaua välimuse kohandamise programm, mis rakendab selleks vidinaid. Vaieldamatult kõige

populaarsem vidinate programm nimega "*Konfabulator*" sai aluse kõigepealt *Mac* platvormil 2003. aastal ja siis *Windows* operatsioonisüsteemil järgmisel aastal. Hiljem sellest sai "*Yahoo Widgets*", mis on ka tänasel päeval üks populaarsematest vidina programmidest. Vidinate populaarsus kasvas veel kiiremini, kui ilmus *Windows Vista OS,* mille üheks innovatsiooniks oli *Windows Sidebar* – töölaua serval asuv paneel, mille peal istusid vidinad. Windows 7 ilmumisega nimetati selle paneeli *Windows Desktop Gadget'iks* ümber. Veel üks suur mängija selles valdkonnas on Google oma "Google Desktop" programmiga, mis vaieldamatult mõjutas vidinate populaarsust. Olid ka teised, kes proovisid rakendada vidinate tehnoloogiat, kuid mitte iga mängija saab mõjutada mingi tehnoloogia arendamise protsessi.

2.1.3 Vidinate tüübid

Üks vidinate aspekt, millega saab kindlasti praalida, on rikas valik. Vidinate funktsionaalsuse ehk tüüpide mitmesugusus on lihtsalt kolossaalne, ning sama põhifunktsionaalsusega, aga väikeste erinevustega vidinaid, on enamustel platvormidel tavaliselt ka suur hulk. Isegi väga spetsiifiliste vajaduste puhul tihti leidub unikaalne vidin, mis teeb täpselt seda, mis vaja oli. Vaatamata sellele, proovime neid ikkagi funktsionaalselt liigitada:

- Süsteemi infot väljastavad vidinad. Need, mis näitavad meie süsteemist kättesaadavat infot, nagu aega, kuupäeva, *CPU*, *RAM*, *HDD*, aku, prügikasti, võrgustiku ja muud. Arvatavasti kõige populaarsem kategooria.

- *RSS* lugejad – mingi *RSS*-voo loevad vidinad. Allikateks võiksid olla erinevad uudiste kanalid, *Facebook* või *Twitteri* vood, ilma voog - võimalusi on palju. Igal endast lugupidaval vidinate kasutajal on kindlasti paar neist oma töölaual.

- Teised töölauda manipuleerivad vidinad. Siia kuuluvad ülejäänud vidinad, mis ei võta internetist, ega näita süsteemi infot. Need võiksid olla märkmikud, programmide ja kaustade linkide hoiavad konteinerid ehk „*Docks*"' või „*Launchers*"', kalkulaatorid, mängud, piltide „*Slideshow*" näitavad ja muud vidinad.

- Teised internetipõhised vidinad. Heaks näiteks on raadio vidin. Sellesse kategooriasse võiks panna ka sõnastikud ja tõlkimise vidinad, „*Flickr*" või sarnase piltide galerii vidinad. On ka teised, unikaalsemad vidinad, näiteks need, mis laadivad ülesse nende peale visatud faile mingile *FTP* serverile.

2.2 Ülevaade vidinmootoritest

Selles peatükis kõigepealt proovime aru saada, mis asjad on vidina platvormid, milleks nad üldse vaja on, ja kuidas töötavad. Siis tutvume mõnede konkreetsemate programmidega lähedamini.

2.2.1 Vidina platvormid ehk mootorid

Tuli aeg tutvuda vidinate südamega – programmiga, ilma milleta nad lihtsalt ei saa eksisteerida. Vidina platvorm, „*Widget Platform*", tihti nimetatud ka vidinmootor, „*Widget Engine*", on baas, mille peal istuvad ja mille abil elavad vidinad. Miks on neid üldse vaja, miks vidinad ei saa iseseisvalt elada? See on tingitud sellest, et vidinad ise pole tegelikud programmid - see on nende puudus ja eelis. Nad on ehitatud niimoodi, et meie operatsioonisüsteem lihtsalt ei saa aru, kuidas neid opereerida. Sarnaselt veebibrauseriga, mida on vaja, et veebilehtede *HTML* koodi lugeda ja meile resultaadi näidata, töötavad vidinmootorid. See näide ei olnud juhuslik - enamus mootoreid rakendab veebi tehnoloogiaid, nagu *XML* ja *Javascript,* oma vidinates, aga sellega tutvume lähedamini "Vidinate arendamine" peatükis.

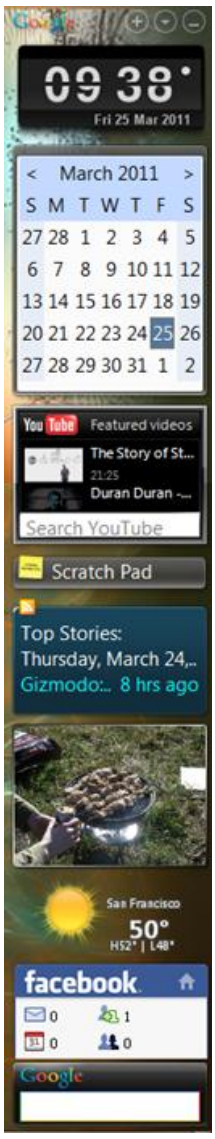
Tasub märkida, et siin on ka erandid, - on olemas sellised vidinad, mille kasutamiseks ei ole vaja mingit platvormi. Üheks näiteks on "*Zinc*" - *Flash* ja *Flex* arenduskeskkond, mille abil saab vidinaid luua, konverteerides *Flash* rakendusi ".exe" programmidesse. Vidinaid saab kirjutada ka Javas näiteks. Tegelikult võimalusi on palju, sest vidinate maailmas ranged standardid või spetsifikatsioonid puuduvad. Igal juhul, platvormipõhised vidinad on vaieldamatult domineerivad, kuna neid on tunduvalt lihtsam arendada. See ongi vidina platvormide ehk mootorite kõige suurem eelis.

2.2.2 Yahoo Widgets

Joonis 1: Yahoo Widgets mootor

Yahoo Widgets on kindlasti üks populaarsemaid ning parimaid vidinmootoreid. Alguses oli tuntud nimega "*Konfabulator*", mille arendas *Pixoria*, kuid oli 2005. aastal ostetud, ning edasi arendatud *Yahoo* poolt. Mis teeb *Yahoo Widget Engine* edukaks, on esteetiliselt meeldiv vidinate välimus ja hämmastav vidinate valik. Erinevalt "*Vista Sidebar Gadgets*", *Yahoo* laseb kasutajatel paigutada vidinaid, mistahes kohale töölaual. Ta pakub ka oma vidinate jälgimiseks ning kontrollimiseks ühe universaalse paneeli. Kõige unikaalsemaks *Yahoo Widgets Engine* tunnusjooneks on aga võimalus nuppu vajutamisel kõik vidinad kohe esiplaanile tõsta. *Windows OS* vidinmootoritest on see platvorm ainuke, mis pakub sellist, minu arvamusel väga kasulikku, funktsionaalsust.

2.2.3 Google Desktop



Joonis 2: Google Desktop mootor

Kuigi *Google Desktop* on peamiselt mõeldud töölaua otsingu programmina, mis indekseerib ja laseb kasutajatel kiiresti leida failisüsteemis asuvaid faile, oma e-maili kontol asuvaid kirju, külastatud saite ja muis asju, on tema ka hästi tuntud oma "*Sidebar*" paneeli tõttu.

Google Sidebar on töölaua serval asuv paneel, mis hoiab *Google* vidinaid. Tegelikult ka *Google* pakub võimalust paigutada neid suvalistele kohtadele - paneel on mõeldud nende kompaktseks organiseerimiseks. Vaikimisi on paneelis juba olemas uudiste ja ilma lugeja, kell, piltide galerii, märkmik ning *Google* otsingu vidinad. Üks *Google Desktop* vidinate platvormi tugev tunnusjoon on võimalus neid kiiresti ja mugavalt lisada, selleks pakub ta oma programmi sisse ehitatud „*Add widgets*" liidest. *Google Sidebar* reserveerib vajamineva ekraani osa, aga seda ja palju teisi seadeid võib kergesti muuta. Näiteks, paneeli võib üldse eemaldada, kasutades "*Deskbar*"

režiimi, milles *Google Search* akent pannakse Windows *Taskbar* sisse.

Üldiselt on *Google Desktop* väga korralik platvorm, mis pakub piisavalt palju vidinaid. Probleem on aga selles, et vidinatega tulevad ka töölaua otsimise funktsionaalsused, mida ilmselgelt mitte iga inimene tahab näha.

2.2.4 Rainmeter Skins

Rainmeter on selline programm, mis laseb teil paigutada "nahad" ehk "*Skins*" töölaua peale. Need "nahad" võivad mõõta ja näidata suurt hulga erinevat informatsiooni. Vaatamata sõnale "*skin*", jutt on samasugustest vidinatest, nagu me oleme siiamaani harjunud nähda. "Vidin" asendamine sõnaga "*skin*" on õigustatud, *Rainmeter* struktuur on täiesti erinev, võrreldes teiste platvormidega. Programm kasutab ".ini" faile, mis koosnevad *Rainmeter* käskudest ning parameetridest. *Rainmeter* kasutab "*themes*" ehk „teemasid", mis on põhimõtteliselt vidinate komplektid samasuguse disainiga, mille tihti tegi üks ja sama inimene või grupp. Selliseid komplekte on internetis hästi palju, mis annab võimalust valida just selliseid, mis teile kõige rohkem meeldivad. Soovi korral võib erinevaid komplekte omavahel segada. Üks *Rainmeter* tugev külg on kohandatavus ja seadistamise võimalus - isegi kui "teema" ei paku seadeid, asju võib suhteliselt kergesti muuta ise, kohandades vastavat „.ini" faili. Tänu optsioonidele (valikutele) nagu "*draggable*", "*click through*", "*position*", "*transparency*", võib nimetada *Rainmeter* kõige paindlikumaks vidina mootoriks. See, koos suure vidinate ning vidinate komplektide valikuga, teeb *Rainmeter* väga heaks kandidaadiks töölaua kohandamise tarkvara otsijatele.

Joonis 3: Rainmeter mootor, APB skins

2.2.5 Windows Desktop Gadgets

Alustame vidina mootorite tutvustamist Windows süsteemis oleva platvormiga – Windows *Desktop Gadgets*. Kõigepealt tasub mainida, et enne *Windows 7* ilmumist oli see nimetatud *Vista Sidebar* - paneel ekraani serval Vista operatsioonisüsteemis, kus paiknesid Vista vidinad. *Windows 7* süsteemis tutvustati võimalust paigutada

neid väljaspool seda paneeli, mistahes kohta, ning nimetati selle “Windows *Desktop Gadget*”*iks* ümber.

Microsoft vidinad on nagu kõik teised tüüpilised vidinad, ainuke unikaalne tunnusjoon on võimalus enamus neid suurendada või vähendada. Uusi võib ülelihtsalt lisada, kasutades “*Gadgets*” optsiooni töölaua kontekstimenüüs ja valides soovitud vidinat ilmunud ekraanil. Vaikimisi valik seal on väga väike, aga kasutades “get more gadgets online” linki saabume *Microsoft* saidile, kus neid saab leida rohkem. Igale vidinale saab määrata läbipaistmatuse taset, aga seda pakub enamus mootoreid. Tegelikult kohandamise võimalused on väga nõrgad - kui mingis vajaliku funktsionaalsusega vidinas midagi ei meeldi, tuleb lihtsalt otsida uut sarnast.

Joonis 4: Windows Desktop Gadgets mootor

Peame aga silmas pidama seda, et *Windows Desktop Gadgets* on operatsioonisüsteemi sisseehitatud funktsionaalsus, s.t. ei ole vaja mingit tarkvara alla laadida ning installeerida. See on kindlasti tema tugevaim külg.

2.3 Vidinate arendamine

See peatükk on pühendatud vidinate struktuurile ning nende arendamisele. Tuletan meelde, et vidinad on, põhimõtteliselt, minirakendused ja, erinevalt tavalisest programmidest, neid on suhteliselt lihtne ise arendada. Kuna vidinmootorid enamuselt rakendavad veebi tehnoloogiaid nende loomiseks, arendusprotsess on väga sarnane veebirakenduste arendamisega. Vidina liides tavaliselt koosneb *HTML* või *XML* ja *CSS* koodist koos teiste ressurssidega, näiteks piltidega ja helidega. Interaktiivsus ja funktsionaalsus tüüpiliselt implementeeritakse *Javascript* abil.

Enamus vidinaid võtavad mingeid infosid ehk veebiressurse internetist. Selleks nad kasutavad *XMLHttpRequest* objekti. Saadud informatsiooni, mis on tavaliselt *XML* formaadis, näiteks ilma või *RSS* uudise vood, töödeldakse ja vidina liidest uuendatakse *Javascript* ning "*Document Object Model*" abil . Lisaks igal vidinal on "manifest" fail, kus hoitakse "meta" informatsiooni, näiteks vidina ja autori nimi ning versiooni. *Yahoo* platvormil, näiteks, selline fail on "widget.xml". Lõpuks kõik vidinaga seotud failid pakitakse ühe faili mis saab laiendust mida sellel platvormil kasutatakse. Sellised vidina struktuurid ning oma laiendused annavad võimalust platvormile neid kergesti installida.

Peame meeles pidama seda, et kõik vidinmootorid on unikaalsed, mõned kasutavad täiesti erinevat struktuuri, aga kõikidel on eesmärgiks lihtsus - vidinad peavad olema kergesti arendatavad. Võtame mõned konkreetse näite.

Yahoo Widgets on väga standardne selles mõttes. Seal kasutatakse *XML*, *Javascript* ja vajalikke ressursse vidina loomiseks ning *AJAX* tehnoloogiat veebisaitidest informatsiooni saamiseks. Valmis vidin koosneb ".kon" failist, mis sisaldab vidina *XML* koodi, "Resources" kaustast mis hoiab kõike ressursifaile ning eelnevalt mainitud "widget.xml" failist, kus asub kõik "*meta*" informatsioon. Pakitud vidin saab „.widget" laiendust ja on valmis installimiseks. Need kellele veebi arendamine pole võõras saavad kiiresti *Yahoo Widget* platvormil alustada.

Detailsemalt vaatame kuidas toimub vidinate loomine konkreetsel näitel – ajaplaneerimise ja uudiste pakkuva vidinal, mida ma ise arendan selle diplomitöö raames.

3 Vidinmootorite võrdlus ning analüüs

Selles peatükis teeme midagi kasulikumat - proovime ise mitut platvormi katsetada, võrrelda ning analüüsi teostada. Võrdlemiseks võtame 4 populaarset Windows töölaual töötavat vidinate programmi: *Yahoo Widgets*, *Google Desktop Gadgets*, Windows *Desktop Gadgets* ja *Rainmeter Skins*. Kriteeriumitena võtame pakutud vidinate valiku, ressursinõudluse, välimuse ning funktsionaalsuse ja arendajasõbralikkuse.

Kuna iga platvorm ise kasutab arvuti ressursse ka, ning tema vidinad näevad välja teistmoodi võrreldes teiste platvormiga, pole mõistlik segada vidinaid, ehk kasutada mitut vidinate programme samal ajal. Seetõttu vidinate armastajad valivad tavaliselt ainult üht ja need kriteeriumid on tihti otsustavad, millist programmi valida.

3.1 Vidinate valik

Alustame kõige lihtsama ja arvatavasti kõige olulisema kriteeriumiga - vidinate arvuga, mida platvormid, ning nende kogukonnad pakkuvad. Lähtudes sellest, et inimesed tavaliselt kasutavad ainult üht platvormi korraga, on mõistlik oletada, et see platvorm peab kõigepealt pakkuma sellise vidinate valiku, et kasutaja saaks leida seal kõiki temale vajaliku funktsionaalsusega vidinaid. Minu meeles, ükskõik kui ilus, kiir ja mugav vidinmootor on, kui seal ei ole selliseid vidinaid mis mul vaja on, siis ma seda kindlasti ei valiks.

Vaatame kõigepealt, kui rikas on vidinate maailm *Yahoo Widgets* platvormil, mis hakkas kogunema nii kasutajaid, kui ka arendajaid kõige varem. Õnneks, *Yahoo* pakub ühe korraliku vidinate galerii, kus jälgitakse ka nende arvu . Praegusel momendil on seal 6044 vidinat olemas, mis on tegelikult päris suur arv.

Yahoo konkurent, *Google*, mis samuti omab korraliku vidinate galerii nii veebisaidil, kui ka programmi sees, oma arvu ei näita . See aga pole probleem kuna neid on lihtne ise kokku lugeda - saame lõpus 1952 tüki. *Google Desktop Gadgets* platvormil vidinate valik on tunduvalt nõrgem, aga see on loogiline, kuna see mootor ei ole pühendatud ainult vidinatele.

Rainmeter platvorm on mõnevõrra ebastandardne ka selles suhtes. Tavaliselt vidinaid saab kätte "*theme*" komplektidena, mis koosneb kuni 40 vidinatest. Ametlik veebisait pakub alla laadimiseks ainult kolm sellist - enamus laaditakse alla teistest allikatest, näiteks "*DeviantArt*" , "Customize.org" või ametlik *Rainmeter* foorum . Kuigi neid on väga raske kokku lugeda, on ilmselgelt näha, et nende arv on päris suur - paar tuhat on kindlasti olemas, ning kogukond loob pidevalt hea tempoga uusi.

Vaatame nüüd, millise vidinate valiku pakub *Windows*. Kui paar aastat tagasi oli *Windows Live* galeriis üsna väike number, siis praegu pakutakse seal 6059 vidinat ja paistab, et see arv koguaeg kasvab. Võib eeldada, et kasv on tingitud *Windows 7* ilmumisega ja tänu sellele on *Windows Desktop Gadgets* platvorm liider pakkuvate vidinate arvu poolest .

3.2 Arendajasõbralikkus

Vaatamata sellele, et vidinaid on internetis palju (tuhandeid) ja enamusjuhtudel me saame leida just sellise funktsionaalsusega vidinat, mis meil vaja on, mõnikord ikka tuleb ette vajadus ise vidinat luua. Näiteks, meil võiks minna vaja väga spetsiifilist funktsionaalsust, või meil on vidin, mis meeldib väga, aga on soov midagi selles vidinas muuta või parandada. Võib olla selline vidin on olemas ühel vidinplatvormil, aga me kasutame teist ja vahetada ei soovi. Stsenaariume on palju. Õnneks, vidina mootorid on loodud niimoodi, et vidinate arendamine oleks kiir ja kerge. Seetõttu, valides endale vidina mootori, tuleb uurida, kui raske on sellel platvormil ise vidinaid arendada.

Proovin nüüd ise luua lihtsat vidinat erinevatel platvormidel, ning võrdlen, kui sõbralikud on nemad selles suhtes. Vidinaks loon sellist, mis konverteerib raha kroonidest eurodesse ning vastupidi. Kirjutan seda *Yahoo Widgets*, *Google Desktop*, *Windows Desktop Gadgets* ning *Rainmeter* platvormidel.

Alustan siis *Yahoo Widgets* mootoriga. Protsess kulges niimoodi:

- Laadisin alla, ning installisin *Yahoo Widgets SDK*.
- Lugesin "Yahoo Widget engine: How to build a widget" *SDK* sees oleva juhendi, ning kasutasin "Konfabulator reference" juhendi.
- Kirjutasin peamist lähtekoodi ja valmistasin ressursse, ning teisi faile, nimelt pilte ja "*meta*" faili.

- Pakkisin oma vidina kausta ".widget" failisse kasutades selleks ettenähtud "Widget Converter" vidinat *Yahoo* platvormil.

- Topelt-klõps ning vidin on tööle pandud.

Aega selleks läks umbes 2 tundi. Optimeerisin, ning tegin kompaktseks lähtekoodi - lõppuks koosnes see ainult 37 ridadest. Lõpptulemus näeb välja niimoodi:

Joonis 5: Euro kalkulaator Yahoo Widgets vidinana

Vaatame nüüd *Google Gadgets* platvormi:

- Laadisin alla *Google Desktop SDK*.

- Vaatasin läbi "Google Desktop Gadget API" .

- Kasutasin "Google Desktop Gadget Designer" programmi – *Google* poolt pakutud *SDK* sisse kuuluv *IDE* "*gadget*"ite kiireks ja mugavaks loomiseks, mis sobib nii programmeerijatele kui ka tavainimestele.

- Kirjutasin valmis „main.xml" faili, mis hoiab XML koodiosa, "main.js" faili, mis hoiab Javascript koodi, ning täitsin "meta" informatsiooni.

- Pakkisin projekti „.gg" ehk „google gadget" faili sisse, kasutades *IDE's* oleva "Build package" funktsiooni.

- Samamoodi nagu Yahoo platvormil, topelt klõps ".gg" faili peale, ning vidin installib ennast ja hakkab töötama.

Tänu *Google* "Gadget Designer" *IDE* programmile, milles disaini osad saab teha visuaalses ehk "*WYSIWYG*" režiimis, aega arendamiseks läks tunduvalt vähem - natuke üle 1 tundi. Tegin koodi kompaktsemaks, ning tulemusena sain 8 *XML* ning 23 *Javascript* koodi ridu. *Google Gadgets* platvormil näeb see vidin välja niimoodi:

Joonis 6: Euro kalkulaator Google Desktop vidinana

Vaatame siis, kuidas vidina arendamine toimub *Windows Desktop Gadgets* platvormil:

- Kuna ei õnnestunud leida mingit *Vista Sidebar* või *Windows Desktop Gadgets SDK,* hakkasin kohe tutvuma *Microsoft* vidinate arendamise juhendiga .
- Valmistasin ".xml" *meta* faili ja ressursse, ning kirjutasin terve koodi ".html" faili.
- Juhendi jälgides panin oma vidina kausta "C:\Program Files\Windows Sidebar\Gadgets" kausta, ning valisin oma vidinat "Gadgets" ekraanil. Vidin läks tööle edukalt. Pakkides kausta ".gadget" faili sisse automaatne installimine mingil põhjusel ei õnnestunud.

Kokku läks 2.5 tundi, ning lähtekood koosneb 70-st reast. Tulemust võib näha järgmisel pildil:

Joonis 7: Euro kalkulaator Windows Desktop vidinana

Viimaseks on meil *Rainmeter* platvorm jäänud. Kahjuks, asi ei läinud nii edukalt sellega:

- *Rainmeter* mingit *SDK* ka ei pakku, alustasin "Skin Tutorials" juhendiga tutvumisest.

- Kuna juhend osutus raskeks ja ebamugavaks *Rainmeter* programmiga alustamiseks, proovisin õppida näidetel.

- Platvorm kasutab täiesti erinevat, unikaalset struktuuri. Kõik kood istub ühes või mitmes ".ini" failis, mis koosnevad suurest hulgast võrranditest. See tähendab, et õppimine sellel platvormil oma vidinaid arendada algab nullist.

- Mingil põhjusel valuuta konverteri vidinat leida ei õnnestunud. Proovides juhendi järgi ise kirjutada sattusin kohe ummikusse - programm ei pakkunud silmanähtavat viisi, kuidas informatsiooni sisestamise funktsiooni lisada.

- Pärast *Rainmeter* foorumil otsimist tulin järeldusele, et *Rainmeter* lihtsalt ei ole mõeldud informatsiooni küsimiseks, ainult näitamiseks. Tegelikult on platvormil "userInput" *plugin* olemas, kuid sellega isegi lihtsat kalkulaatorit kirjutamiseks võtaks ebamõistlikult palju aega.

Rainmeter katse võttis vähemalt 3 tundi, ning mingit tulemust ei andnud. Tundub, et *Rainmeter* on uutele arendajatele raske, ning piiratud funktsionaalsusega platvorm.

3.3 Ressursinõudlus

Võib tekkida küsimus, et kui vidinad on nii ilusad, huvitavad ning kasulikud, miks siis iga inimene ei kasuta neid? Üks põhjus on see, et nemad on ka programmid ja, loomulikult, nad söövad meie arvuti ressursse. Kuna nad töötavad meie süsteemis pidevalt, võib see süsteemi pidurdada, mis paljude inimeste jaoks on tõsine probleem. Järelekult, kui valime endale vidina programmi, peame arvestama sellega, kui palju ressursse see ning selle vidinad nõuavad. Proovisin välja uurida, palju *RAM* ning *CPU* võtavad meie valitud platvormid ise, ilma vidinateta, ning viie standardse vidinaga, mida saab leida igal platvormil, - kalender, kell, ilma näitaja, piltide *Slideshow* ning ressursside näitaja. Ressursside kasutamise mõõtmiseks kasutasin „Process explorer" programmi ning Windows "Resource Manager".

Nagu alati, alustame *Yahoo Widget* mootorist. See platvorm on väga palju tagasisidet saanud, et ta kasutab liiga palju ressursse. *Yahoo* väidab, et viimastel versioonidel oli see parandatud. Tegelikult, tühi, ilma vidinateta, *Yahoo Widgets* programm kasutas 5.4 megabaiti mälust ja keskmine *CPU* utilisatsioon oli ainult

0.05%. Meie viie valitud vidinatega aga tõuseb kasutatud mälu 121 megabaitini, keskmine *CPU* - 1.38 protsendini.

Google platvormi, kahjuks, on raske õiglaselt selles suhtes hinnata, kuna programm tegeleb teiste asjadega peale vidinaid, aga proovime ikka. Iseseisvalt võtab see mootor 15 megabaiti ning 0.10% *CPU*. Koos meie vidinatega - ainult 27 megabaiti mälust ning umbes 0.35% *CPU*. Üllatavalt, *Google Desktop* platvormil on väga madalad nõudmised.

Nüüd vaatame palju nõuab *Vista Sidebar* ehk *Windows Desktop Gadgets*. Temal on üks suur eelis - ei ole vaja mingit mootorit installida, kuna selleks on Windows süsteem ise. Seetõttu võime öelda, et ilma vidinateta platvorm ei kasuta mingeid ressursse. Koos meie vidinatega kasutab „Sidebar" protsess umbes 61.5 megabaiti *RAM* ning 0.27% *CPU*.

Viimaseks on *Rainmeter* mootor jäänud. See unikaalne platvorm on tegelikult väga korralik ning minimalistlik - ise võtab ainult 2.7MB meie mälust ning 0.01% *CPU*. Isegi koos vidinate komplektiga tema nõuab ainult 7.4MB mälust ja 0.09% *CPU*, mis on tõepoolest „kerge" programm.

Lõpuks, peame meeles pidama seda, et kui palju ressursse nõuab iga konkreetne vidin sõltub kõigepealt tema raskusest. Seetõttu vidinatega platvormide ressursside tarbimise andmetest ei ole päris õige järeldusi teha. Need arvud ei ole täpsed ja usaldusväärsed, aga nende abil saab ikkagi ettekujutust, kui palju iga vidinmootor koormab meie süsteemi.

3.4 Välimus ja funktsionaalsus

Need, kellele on ainult paar tüüpilist vidinat vaja on, võibolla pole huvitatud platvormi vidina valiku suurusest. Need, kes ei soovi oma vidinaid luua või teisi modifitseerida, arvatavasti ei ole huvitatud platvormi arendajasõbralikkuses. Siis neid, kellel on kaasaegne ning võimakas arvuti, võib olla ei huvita platvormi ressursinõudlus. Kindlasti on aga see, et ilusad vidinad ning head platvormi võimalused võivad kergesti teha selle platvormi lemmikuks. Proovime siis hinnata need platvormid visuaalselt ning funktsionaalselt. Peame aga peas hoidma seda, et sama funktsionaalsusega aga erineva disainiga vidinat saab leida mitu tüki igal platvormil, ja veel ka seda et inimestel on erinev arusaamine sellest, mis on ilus. Mugava

visuaalse võrdluse jaoks võib kasutada ettevalmistatud pildi, mis näitab meie eelmises katses mainitud vidinaid igal platvormil.

Joonis 8: Windows Desktop Gadgets, Google Desktop Gadgets, Yahoo Widgets, Rainmeter skins. Võrdlus.

Windows'i vidinad tunduvad kõige ilusamad, kuid Yahoo on hea kandidaat ka. Funktsionaalsusest pakub Windows võimalust läbipaistmatust seadistada, rakendada "always on top" optsiooni, mis tähendab, et vidin on alati nähtaval, mõnda vidinat suuremaks teha, ning kutsuda kõike vidinaid esiplaanile, kasutades "ctrl+g" klahvide kombinatsiooni *Windows 7* süsteemis. Uusi vidinaid saab lisada *Windows* süsteemi "Gadget Gallery" sees, aga seal on ainult paar tükki, tuleb ikkagi *online* galeriisse minna.

Google Desktop platvorm, mis näeb suhteliselt korralikult välja, pakub võimsamaid võimalusi. Kõigepealt on temal väga võimekad arvuti otsingu võimalused. Vidinaid võib hoida paneelis või paigutada ise töölaua peal ja peita paneeli. Saab ka paljude vidinate suurendust muuta ning rakendada sama "always on top" optsiooni. Kutsuda

neid esiplaanile saab ka mugava “topelt SHIFT” kombinatsiooniga. Erinevalt *Windows Gadgets* platvormist, programmis sisseehitatud vidinate galeriist saab kohe leida kõik olemasolevad vidinad.

Yahoo platvormi vidinatel on kindlasti ka väga meeldiv välimus. Tema paneeli võib paigutada ükskõik mis serval, isegi ekraani allosas ja ülaosas või üldse peita. Igal vidinal saab muuta tema akna taset, ehk valida kas vidin peab alati istuma ekraani esiplaanil või tagaplaanil. Võib öelda vidinale, et ta üldse ignoreeriks hiire vajutamisi või lihtsalt eemaldada tassimise võimalust. Saab ka läbipaistmatust muuta. Kahjuks, uute vidinate alla laadimiseks tuleb ametlikule saidile minna, aga pakub *Yahoo* kõige ilusama vidinate esiplaanile kutsumise funktsiooni - “F8” klahvi vajutamisel ekraan on kaetud sinise kihiga mille peal istuvad meie vidinad.

Rainmeter vidinate välimus tundub liiga lihtne ning ebameeldiv. Kuna sellel platvormil on täiesti erinev struktuur, on olemas ka teised, unikaalsed funktsioonid. Kõigepealt, vidinatega võib tegeleda, installida ning salvestada komplektide kaupa. Igal vidinal on kõik *Yahoo* vidinate optsioonid ja lisaks teised, nagu võimalus peita vidina, kui hiir on tema peal, või kleepida vidinat erinevatele servadele. Ainult *Rainmeter* platvormil on võimalus kontrollida, millisel monitoril vidinat näidata. Lõpuks on veel väga kasulik funktsioon, “edit skin”, mille abil saab kiiresti mingi vidina muuta.

3.5 Võrdleva analüüsi kokkuvõte

Proovime siin teha mõni järeldus. Pakutava vidinate valiku järgi on kindlasti Windows *Desktop Gadgets* kuningas. *Yahoo* jääb maha ainult natuke praegu, aga vaatlus näitab, et iga mööduva päevaga kasvab Windows “*gadget*”ite arv 5-10 tükiga, kusjuures *Yahoo* vidinate arv ei muutu. Teised platvormid jäävad veelgi rohkem maha.

Arendajasõbralikkuse poolest on *Google* vaidlemata tšempion. Tänu pakutud *SDK* sees oleva *IDE* programmile saab uusi vidinaid luua väga mugavalt ja kiiresti, kasutatud *XML* kood on lihtne ning kompaktne. *Yahoo* platvormil on koodi kirjutamine ka lihtne, aga käsitsi see loomulikult võtab rohkem aega. Windows vidinate jaoks *HTML* koodi kirjutamine pole raske, aga ei ole nii kiir ja kompaktne, nagu eelnimetatud platvormidel. *Rainmeter* on kõige keerulisem ning ebamugavam mootor selles suhtes, kuna tuleb nullist õppida, ning sellel programmil on oma piirangud.

Inimestele, kellele on arvutiressurssidest iga megabait vaja, on ilmtingimata *Rainmeter* parim valik. See kiir ning minimalistlik programm näitab, et ilusad ning kasulikud vidinad ei pruugi meie süsteemi pidurdada. Windows ning *Google* mootoritel on täitsa keskmised ressursside nõuded, *Yahoo* platvormil, kahjuks, on need ülisuured.

Välimuse järgi on raske valida. Mõnedele võivad meeldida *Windows* vidinad rohkem, teistele - *Yahoo* vidinad. Funktsionaalsuse poolest aga *Rainmeter* kindlasti pakub kõige laiemaid võimalusi.

	Yahoo Widgets	Google Desktop	Windows Gadgets	Rainmeter skins
Vidinate valik	Väga suur. 6044tk	Keskmine. 1952tk	**Parim. 6059 ja veel kasvab**	Keskmine. Oletatavasti paar tuhat
Arendaja-sõbralikkus	Hea. Kergesti arusaadav keel ning struktuur	**Parim. Kerge keel, struktuur ning väga mugav IDE**	Keskmine. Tuttav HTML keel, aga ei ole nii mugav vidinate loomiseks.	Halvim. Lihtne struktuur aga uus ja väga raske keel ning piirangud.
Ressursside-nõudlus	Halvim. Nõuab liiga palju CPU ning RAM	Keskmine. Kaasaegsel arvutil probleemi ei tekki	Keskmine. Kaasaegsel arvutil probleemi ei tekki	Parim. Õhuke programm mis ei pidurda süsteemi
Välimus	Hea. Täitsa korralikud vidinad	Keskmine.	**Parim. Kõige ilusad vidinad**	Halvim. Liiga lihtsa disainiga
Funktsionaalsus	Väga hea. Head vidinate optsioonid ja esiplaanile	Hea. Mõned kasulikud funktsioonid puuduvad	Keskmine. Puudub palju teistel platvormidel	**Parim. Suuremad vidinate manipuleerimis-**

	kutsumise funktsioon		olevaid funktsioone.	**võimalused**

Tabel 1: Platvormide võrdluse kokkuvõte

Nagu näha, on väga raske öelda, mis on kõige parem vidinmootor - igal on oma eelised ning puudused. Igal inimesel on oma vajadused ning prioriteedid, seetõttu, erinevatele kasutajatele on erinevad platvormid paremad.

4 Oma rakenduse loomine

Tuletan meelde, et rakenduse eesmärgiks oli seatud luua sellist vidinat, mis kõigepealt näitaks kasutajale Tallinna Ülikooli *Twitter* voo uudiseid ning oma *Google Calender* konto sündmusi. Selle elluviimiseks kasutan *Windows Desktop Gadgets*. Selle platvormi valimist võib kergelt põhjendada - tema on juba *Windows 7* ja *Windows Vista* peal olemas, ning arendamine toimub tuttaval, *Javascript* põhisel viisil.

Kuna platvorm on valitud, lähme edasi vidina kavandamise etapiga.

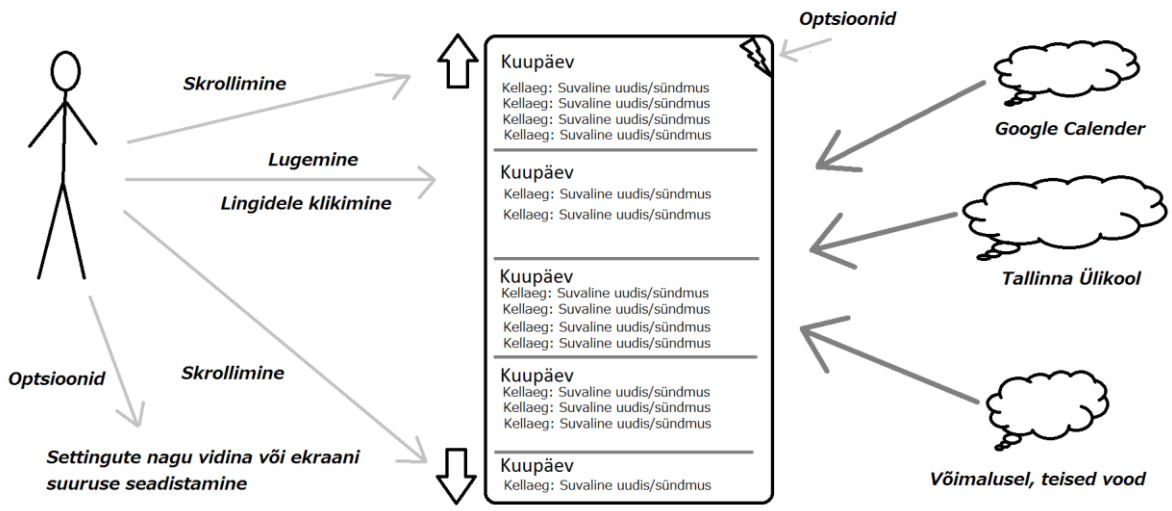
4.1 Kavandamine

Analüüsides vajadusi, selgub, et me peame võtma informatsiooni kasutaja *Google Calender* ning TLÜ *Twitter* voo teenustest, ning väljastama selle mingil mugaval viisil. Funktsionaalsuse poolest kohe paistab, et mõlemal juhul on tegemist *RSS* voo lugemisega.

Saadud informatsiooni tuleb vastavalt töödelda. Informatsiooni all mõeldakse uudise või sündmuse nimetus, kuupäev ja kellaeg. Info kättesaamine peaks toimuma *AJAX* tehnoloogia abil, ning töötlemine tavalise *Javascript'iga*.

Väljastamise viisiga võime olla väga paindlikud, seda võib teha täiesti erinevalt. Alguses oli kavandatud implementeerida seda tüüpilise kalendri vormis, kus iga päeva lahtris pidid olema vastavad infoobjektid. Kuna vidin ise ei saa olla liiga suur, lahtrid on ilmselgelt liiga väiksed pikema teksti hoidmiseks. Seoses sellega, tuli hiljem idee ning otsus kasutada ajaplaneerimise ehk teavitaja või "*agenda*" implementatsiooni. See tähendab, et korraga näeme mitu päevad, tüüpiliselt käesolevat nädalat. Igal päeval on sellel päeval toimuvad sündmused või postitud uudised, sorteeritud kellaegade järgi. Õpilase päevik on siin sobilik näide.

Tervikpilt on juba enam-vähem väljakujunenud ja võib proovida stsenaariumi joonistada. Vaatame, mis tuleb välja:



Joonis 9: Meie "tudengi agenda" vidina stsenaarium

Selle järgi saab nüüd hakata asju programmeerida.

4.2 Loomine

Alustame selle vidina projekti tehnilist realiseerimist. Aluseks võtame eelnevalt tehtud stsenaariumi. Valmistame ka *HTML* šabloonid - üks meie põhikoodi jaoks ja teine optsioonide seadistamiseks. Need ja kõik teised projektiga seotud failid lähevad "C:\Program Files\Windows Sidebar\Gadgets\meie_projekti_nimi" kausta. Kui valmistused on tehtud, alustame põhikoodi kirjutamist.

Esialgselt lisame vidina "baasi", s.t. tausta pildi, mille peal istuks meie vidin. Selleks on oma eripärane *Windows Desktop Gadgets* „tag":

```
<g:background id="" src = "">
```

Tuleb *CSS* abil määrata meie "body" suurust:

```
body{ width: 255;  height:400; }
```

Põhikonteiner on nüüd valmis - liigume edasi funktsionaalsuse poole.

Kavandamise etapil saime teada, et meil on vaja infoallikat, nimelt *RSS* kanalit. Kuna lõpptulemus on suunatud tudengile, tundub hea mõte kasutada järgmisi allikaid:

- "*Google Calender*" teenust, kust saaksime inimese planeeringuid.
- Tallinna Ülikooli "*Twitter*" vood, kust võtaksime ülikooli uudiseid.
- "CVKeskuse" tööportaali, saades sealt mõne valdkonna tööpakkumisi.

Kirjutame siis koodi, mis paluks informatsiooni meie allikatest. Selleks on vaja luua *XMLHttpRequest* objekti. Enamustes veebibrauserites töötab lihtne "new XMLHttpRequest()" käsk, *Internet Explorer* jaoks on eraldi kood:

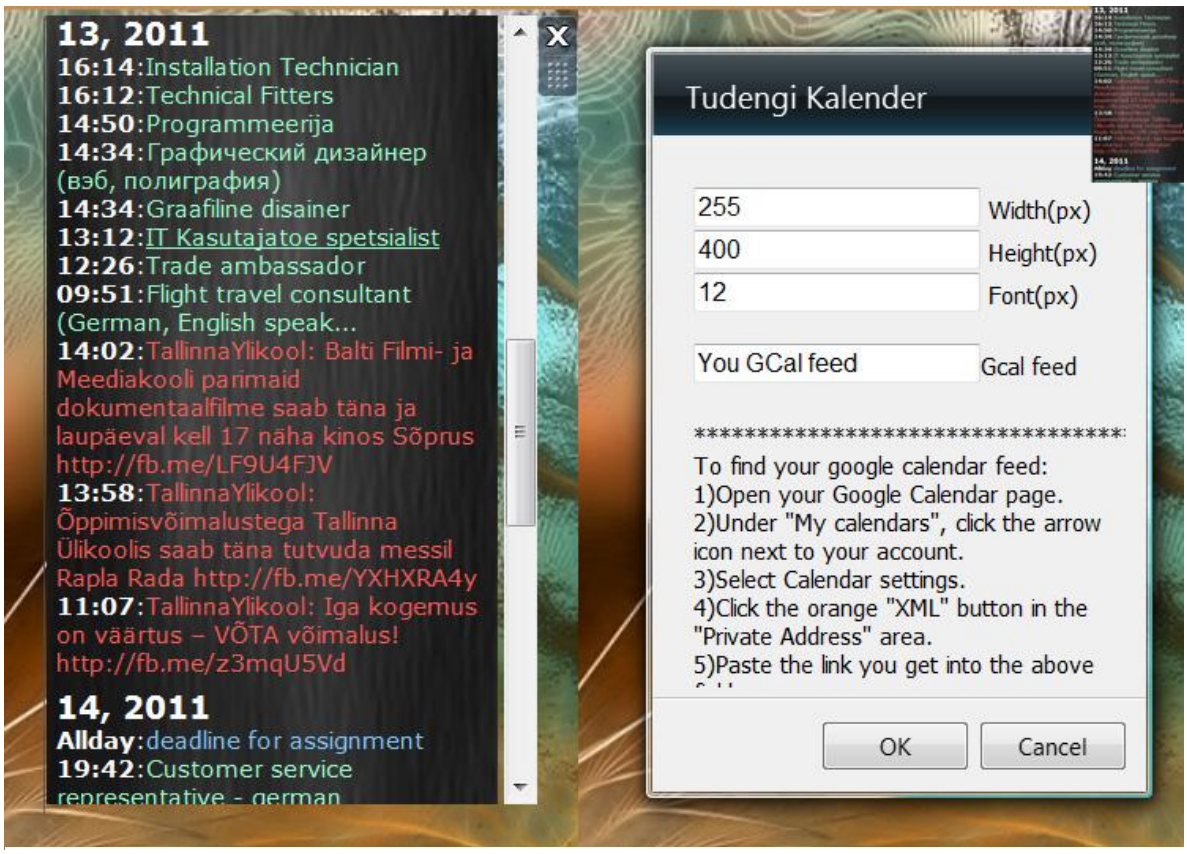
```
new ActiveXObject("MSXML2.XMLHTTP.3.0")
```

Kuigi meile sobib teine variant, parim viis on leida internetist kooditükki, mis arvestaks erinevate brauserite erinevate versioonidega. Tasub märkida, et voogudest saame palju informatsiooni, aga meil on vaja ainult paar asja: iga "item"i ehk allika tiitli või kirjeldust, postitamise kuupäeva ning kellaega, ja *URL* ehk lingi.

Järgmine samm on saadud informatsiooni töötlemine ja väljastamine, kindlasti kõige suurem ja keerulisem koodiosa. Tahaks siin pöörata tähelepanu sellele, et igale programmile tuleb läbi mõelda, ning välja töötada oma erilähendust. Sellel juhul ma otsustasin korraldada funktsionaalse koodi ülesehitust niimoodi:

- Meil on peatsükkel, mis käib läbi iga allika ehk voo.
- Igast allikast võetakse tiitli, kuupäeva ja kellaega, ning lingi. Neid lisatakse vastavatesse massiividesse.
- Elemendid sorteeritakse kuupäevade ja kellaegade järgi
- Loon 7 päevadest koosnevat massiivi - 3 päeva enne tänast kuupäeva ja 3 järgnevat.
- Iga päeva jaoks võetakse ainult need allikad, ehk tiitlid, kuupäevad, kellaajad ja lingid, mis toimuvad samal päeval.

See on põhikoodi loogika, ülejäänud on disaini ja detailide küsimus. Kuna igat koodirida ei ole võimalik siin arutada, lõpetame siis sellega meie vidina loomise ülevaadet ja vaatame tulemust. Lõpetatud ning tööle pandud vidin tuli välja niimoodi:



Joonis 10: Meie "tudengi agenda" vidina tulemus

Programm oli katsetamiseks andnud mõnedele inimestele Windows 7 operatsioonisüsteemiga. Vidin oli kohe tööle pandud ning mingeid probleeme ei tekkinud.

Tahaks pöörata tähelepanu sellele, milline programm siin täpselt tuli välja, millised probleemid on veel olemas ja millist funktsionaalsust saaks tulevikus lisada.

Nagu „Joonis 10" pildid võib näha, tehtud vidin näitab meile 7 päeva. Igal päeval on sündmused või uudised, seotud just selle kuupäevaga. Kõik nad sisaldavad kellaaja ning tiitli, mis on tegelikult link infoallikale. Paistab välja ka see, et erinevate allikate tiitlid on erinevalt värvitud. Kui informatsiooni on liiga palju, "Scrollbar" annab võimalust kõike läbi vaadata. Lõpuks on veel seadistamise ekraan, kus saab määrata, näiteks, kasutatava *Google Calender* voo.

Kahjuks, kui tegemist on piiratud ajaga, tihti midagi jääb lõpetamata või parandamata. Sellel vidinal on ka omad probleemid. Kasutatavaid voogusid-allikaid ei saa ümberseadistada. Akna suuruse muutumine toimub ebakorrektselt. Mõistlik sorteerimine puudub. Muudetud seadistused kaotatakse ära arvuti või vidina restartimise korral.

Võimalusi selle vidina paremaks muutmiseks või funktsionaalsuse lisamiseks on tõesti palju. Alljärgnevalt on mõned head ideed:

- Pakkuda võimalust uusi allikaid lisada.
- Linkide vajutamisel näidata uudist või sündmust oma väikeses aknas.
- Sorteerida kõike kellaaegade järgi.
- Lisada parameetri, mis määrab, mitu ja milliseid kuupäevi näidata.
- Luua ilusama disaini ehk informatsiooni väljastamise viisi.

Oleme selles peatükis näinud, kuidas toimus selle vidina loomine ja mida saaks veel juurde lisada. Loodan, et sellest oli kasu ja keegi saab selle näitel oma huvitavat vidinat luua.

Kokkuvõte

Vidinad on suhteliselt vana, aga erinevates vormides pidevalt arenev tehnoloogia. Selles töös me vaatasime põhjalikult läbi vidinate tehnoloogiat, mõndasid Windows keskkonna vidinmootoreid, nende plusse ja miinuseid ja vidinate arendamise aspekti. Me tutvusime vidinate ajalooga. Saime teada, et vidinad on kerged, vähe ressursse kasutatavad ja tavaliselt mingi ühe funktsionaalsuse pakutavad programmid. Ka seda, et tavaliselt neid võib jagada nendeks, mis võtavad informatsiooni internetist mingist voost ja nendeks, mis võtavad infot lokaalselt arvutist, ning väljastavad seda.

Analüüsiks võtsime tuntuid platvorme: *Yahoo widgets*, *Google Desktop Gadgets*, *Windows Desktop Gadgets* ning *Rainmeter Skins*. Igaühel olid omad tugevad ja nõrgad küljed – *Yahoo Widgets,* näiteks, laseb oma vidinaid ilusti esiplaanile kutsuda, aga kasutab palju ressursse, *Google Desktop* pakub lisaks lokaalset otsingumootorit, aga vidinate valik on väiksem kui teistel, *Rainmeter* on väge kerge ja kiire, aga väga keerukas oma vidinate kirjutamiseks, ja *Windows Desktop* omab sellist eelist, et puudub vajadus platvormi installida, kuid funktsionaalsus on väga piiratud. Kaaludes neid, oli üks platvorm valitud, *Windows Desktop*, mille peal oli praktiline rakendus loodud. Selleks rakenduseks oli tudengile suunatud ajaplaneerimise ja kasuliku informatsiooni esile toov vidin, mille näitel vaatasime lähedasemalt vidina arendamise protsessi ehk nägime programmi kavandamise ning realiseerimise etapid.

Selle töö põhieesmärgiks oli tutvustada lugejatele vidina ning vidinplatvormide tehnoloogiat, põhjalikult võrrelda mõningaid vidinmootoreid, ning näidata vidina arendamisprotsessi, luues tulemusena tudengile suunatud ajaplaneerimise ning uudistest teavitamise vidinat. Sellega on püstitatud tööeesmärgid saavutatud ja loodan, et sellest tööst oli kellegile kasu.

Summary

It is time to sum up what we've done and learned throughout this work.

First of all, we've discovered that widgets are small, interactive, single-purpose programs with a long history, that offer information in different shapes and form, such as weather, pictures, news, time, resource consumption and much more. They are available on a wide variety of platforms and devices.

We have also learned that the number of widgets available is staggering. From a technical perspective, they can be divided into those that grab some information from our *PC*, and those who gather it from the Internet. Most importantly, however, widgets can only be run with specific widget programs called *Widget Engines* or *Widget Platforms*.

We have had a thorough comparison of 4 popular platforms: *Yahoo Widgets*, *Google Desktop Gadgets*, *Windows Desktop Gadgets* and *Rainmeter Skins*. Among them, *Rainmeter* proved to be the fastest and most functionality offering. *Google's* platform won in developer friendliness aspect. *Windows* gadgets turned out to be the best looking ones and have another advantage, that the platform is embedded into the operating system. *Yahoo Widgets Engine* is a mixed breed – it looks nice, widgets are easy to write and it offers some nice functionality.

To demonstrate the process of writing a widget on the example of a student agenda planner widget, we used Windows *Desktop Gadgets* platform. We started with the planning and a scenario was made. Then the implementing part followed and the widget was created. We've learned that widgets usually retrieve some information from the Internet using *AJAX* technology, pass it to some local *Javascript* code, which parses it and outputs the result. The appearance part is adjusted with the help of *HTML* and *CSS*.

Having said that, the main goals have been achieved, this work comes to its end and hopefully, some readers found this to be entertaining or beneficial.

Kasutatud kirjandus

customize.org. *Browse Rainmeter Skins.* 2011. http://www.customize.org/rainmeter/skins (kasutatud 30. 04 2011. a.).

deviantART. 2011. http://browse.deviantart.com/customization/skins/sysmonitor/rainmeter/?order=9 (kasutatud 30. 04 2011. a.).

Google. *Google Desktop - Features.* 2009. http://desktop.google.com/features.html (kasutatud 25. 3 2011. a.).

—. *Google Desktop Gadget API.* 2007. http://code.google.com/apis/desktop/docs/gadgetapi.html (kasutatud 28. 3 2011. a.).

—. *Google Desktop Gadgets.* 2011. http://desktop.google.com/plugins/ (kasutatud 30. 04 2011. a.).

Hameem, Hamzeen. *Lonely Coder.* 12. 5 2009. a. http://hamzeen.wordpress.com/2009/05/12/developing-a-desktop-widget-in-java/ (kasutatud 22. 3 2011. a.).

Juurikas, Juku. *Tallinna Ülikool.* 1980. www.tlu.ee (kasutatud 15. 03 2011. a.).

Kaar, Christian. *An introduction to Widgets with particular emphasis on Mobile widgets.* Technical report, Hagenberg: University of Applied Sciences, 2007.

Kennedy, Niall. *A brief widget history.* 27. 09 2007. a. http://www.niallkennedy.com/blog/2007/09/widget-timeline.html (kasutatud 03. 03 2011. a.).

Lal, Rajesh. *What is a Web Widget?* 11. 08 2007. a. http://www.widgets-gadgets.com/2007/08/what-is-web-widget.html (kasutatud 29. 04 2011. a.).

Microsoft. *Developing a Gadget for Windows Sidebar Part 1: The Basics.* 24. 2 2010. a. http://msdn.microsoft.com/en-

us/library/ff486361%28v=VS.85%29.aspx#_sidebar_basic_intro (kasutatud 29. 3 2011. a.).

—. *Windows Live Gallery.* 2011. http://gallery.live.com/results.aspx?bt=1&pl=1&ds=2&la=en&tier=0&st=1&p=1&c=0 (kasutatud 11. 04 2011. a.).

Quezada, Ernesto. *Widgipedia.com.* 27. 8 2007. a. http://www.widgipedia.com/forum/read.php?3,282 (kasutatud 22. 3 2011. a.).

Rainmeter. *Rainmeter 101.* kuupäev puudub. http://rainmeter.net/RainCMS/?q=Rainmeter101 (kasutatud 25. 3 2011. a.).

—. *Share Your Creations.* 2011. http://www.rainmeter.net/forum/viewforum.php?f=27 (kasutatud 11. 04 2011. a.).

—. *Skin tutorials.* kuupäev puudub. http://rainmeter.net/RainCMS/?q=SkinTutorials (kasutatud 30. 3 2011. a.).

Wikipedia. *Widget engine.* kuupäev puudub. http://en.wikipedia.org/wiki/Desktop_widget#Desktop_widgets (kasutatud 13. 04 2011. a.).

X, Avatar. *WidgetsLab.* 29. 2 2009. a. http://www.widgetslab.com/2008/02/29/zinc-30-flash-and-flex-widget-development-software/ (kasutatud 22. 3 2011. a.).

Yahoo. *Konfabulator reference manual.* 2011. http://manual.widgets.yahoo.com/ (kasutatud 11. 04 2011. a.).

—. *Yahoo! widgets.* 2011. http://widgets.yahoo.com/ (kasutatud 30. 04 2011. a.).

Lisad

Yahoo platvormil „EU/EEK" konverteri „widget.xml"

```
<?xml version="1.0" encoding="utf-8"?>
<metadata>
<name>EUEEK konverter</name>
<version>1.0</version>
<image usage="dock" src="Resources/icon.png"/>
<author name="Alexey Gudz" href="http://www.tlu.ee/~alex/"/>
<copyright>(c) 2011 Alexey Gudz</copyright>
<description>EUR/EEK konverter - Yahoo Widget platvormil vidina loomise katsetus</description>
<platform minVersion="4.0" os="windows"/>
</metadata>
```

Yahoo platvormil „EU/EEK" konverteri „EUEEK konverter.kon"

```
<?xml version="1.0" encoding="UTF-8"?>
<widget version="1.0" minimumVersion="4.0">
   <window id="main_window" title="EUR/EEK konverter" width="200" height="100">
                <image src="Resources/bg.png">
                        <name>background</name>
                        <hOffset>0</hOffset>
                        <vOffset>0</vOffset>
                </image>
                <textarea name="money" hOffset="10" vOffset="12" width="100" lines="1" color="#ffffff" data="0.00" style="font-size:24px; font-weight: bold" bgopacity="50" bgcolor="#000000">
                        <onMouseEnter>
                                money.data='';
```

```
                    </onMouseEnter>
                    <onKeyUp>
                        if(currency.data==“EEK”){
                            result.data=money.data/15.6;
                        }else{
                            result.data=money.data*15.6;
                        }
                    </onKeyUp>
                </textarea>
    <text name=“currency” style=“font-size:26px; font-weight: bold” hOffset=“160” vOffset=“37” hAlign=“center” data=“EEK” color=“#CCCCFF”>
                <onMouseUp>
                    if(currency.data==“EEK”){
                        currency.data=“EUR”
                        currency2.data=“EEK”
                        result.data=money.data*15.6;
                    }else{
                        currency.data=“EEK”
                        currency2.data=“EUR”
                        result.data=money.data/15.6;
                    }
                </onMouseUp>
                </text>
                <text name=“result” style=“font-size:24px; font-weight: bold” hOffset=“60” vOffset=“80” hAlign=“center” data=“0.00” color=“#FFFFFF” />
                <text name=“currency2” style=“font-size:24px; font-weight: bold” hOffset=“160” vOffset=“80” hAlign=“center” data=“EUR” color=“#FFFFFF” />
  </window>
</widget>
```

Google Desktop Gadgets platvormi „gadget.gmanifest”

```
<gadget minimumGoogleDesktopVersion=“5.1.0.0”>
        <about>
                <name>EUEEK konverter</name>
                <description>konverter eurodest kroonidesse ning
tagurpidi</description>
                <aboutText>konverter eurodest kroonidesse ning tagurpidi</aboutText>
                <smallIcon>icon_small.png</smallIcon>
                <icon>icon_large.png</icon>
                <version>1.0.0.0</version>
                <author>Alexey Gudz</author>
                <authorWebsite>http;//www.tlu.ee/~alex</authorWebsite>
                <id>49DEF3E7-2CEB-4660-9E93-E76452D965D0</id>
                <copyright>Alexey
Gudz</copyright><authorEmail>it.aleks@gmail.com</authorEmail></about>
</gadget>
```

Google Desktop Gadgets platvormi „main.xml”

```
<view height=“100” width=“200”>
    <img src=“Resources\bg.png”/>
    <label enabled=“true” height=“36” name=“currency” width=“92” x=“107” y=“8”
onclick=“currency_onclick()” align=“center” bold=“true” color=“#CCCCFF” size=“26”
valign=“middle”>EEK</label>
    <label height=“43” name=“currency2” width=“65” x=“121” y=“51” align=“center”
bold=“true” color=“#FFFFFF” size=“24” valign=“middle”>EUR</label>
    <label height=“33” name=“result” width=“100” x=“12” y=“55” align=“center”
bold=“true” color=“#FFFFFF” size=“24” valign=“middle”>0.00</label>
    <edit onclick=“money_onclick()” onchange=“money_onchange()” height=“42”
name=“money” opacity=“50” width=“100” x=“12” y=“7” align=“center” bold=“true”
color=“#FFFFFF” font=““ size=“22” background=“#000000” value=“0.00”/>
    <script src=“main.js” />
</view>
```

Google Desktop Gadgets platvormi „main.js”

```
function currency_onclick() {
	if(currency.innerText==“EEK”){
		currency.innerText = “EUR”;
		currency2.innerText = “EEK”;
		result.innerText = money.value * 15.6;
	}
	else{
		currency.innerText = “EEK”
		currency2.innerText = “EUR”
		result.innerText = Math.round(money.value/15.6*1000)/1000;
	}
}
function money_onchange() {
	if(currency.innerText==“EEK”){
		result.innerText = Math.round(money.value/15.6*1000)/1000;
	}
	else{
		result.innerText = money.value * 15.6;
	}
}
function money_onclick() {
	money.value = ““;
}
```

Windows Desktop Gadgets platvormi „gadget.xml”

```
<?xml version=“1.0” encoding=“UTF-8”?>
<gadget>
 <name>EUR konverter</name>
```

```
 <version>1.0.0.0</version>
 <author name="Alexey Gudz">
  <info url="http://www.tlu.ee/~alex" text="http://www.tlu.ee/~alex"/>
 </author>
 <copyright>© 2011 Alexey Gudz</copyright>
 <description>konverter kroonidest eurodesse ning tagurpidi</description>
 <icons>
  <icon height="64" width="64" src="images/icon.png"/>
 </icons>
 <hosts>
  <host name="sidebar">
   <autoscaleDPI>true</autoscaleDPI>
   <base type="HTML" apiVersion="1.0.0" src="EUR_konverter.html"/>
   <permissions>Full</permissions>
   <platform minPlatformVersion="1.0"/>
  </host>
 </hosts>
</gadget>
```

Windows Desktop Gadgets platvormi „EUR_konverter.html"

```
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <title>Hello World</title>
    <style type="text/css">
                    body{
                        margin: 0;
                        width: 200px;
                        height:100px;
```

```
        font-family: verdana;
        font-weight: bold;
        font-size: 24px;
        color:”#FFFFFF”;
    }
    #money{
        position:absolute;
        left:10px;
        top:10px;
        background-color:”#888888”;
        color:”#FFFFFF”;
        font-size: 24px;
        font-weight: bold;
    }
    #currency{
        position:absolute;
        left:130px;
        top:10px;
        color:”#CCCCFF”;
    }
    #currency2{
        position:absolute;
        left:130px;
        top:60px;
    }
    #result{
        position:absolute;
        left:10px;
```

```
                top:60px;
            }
    </style>
    <script type="text/jscript" language="jscript">
            function swap(){
                if(currency.innerHTML=="EEK"){
                    currency.innerHTML="EUR"
                    currency2.innerHTML="EEK"
                    result.innerHTML=money.value*15.6;
                }else{
                    currency.innerHTML="EEK"
                    currency2.innerHTML="EUR"

      result.innerHTML=Math.round(money.value/15.6*1000)/1000;
                }
            }
            function calculate(){
                if(currency.innerHTML=="EEK"){

      result.innerHTML=Math.round(money.value/15.6*1000)/1000;
                }else{
                    result.innerHTML=money.value*15.6;
                }
            }
    </script>
</head>
<body>
    <g:background src="url(images/bg.png)">
```

```
                    <input name="money" id="money" type="text" value="0.00"
size="3" onclick="this.value='';" onkeyup="calculate()"></input>
        <span name="currency" id="currency" onclick="swap()">EEK</span>
                    <span name="currency2" id="currency2">EUR</span>
                    <span name="result" id="result">0.00</span>
    </g:background>
  </body>
</html>
```

Tudengi kalendri „gadget.xml"

```
<?xml version="1.0" encoding="utf-8"?>
<gadget>
 <name>Tudengi Kalender</name>
 <version>1.0</version>
 <author name="Alexey Gudz">
  <info url="http://www.tlu.ee/~alex" text="http://www.tlu.ee/~alex"/>
 </author>
 <copyright>Alexey Gudz © 2011</copyright>
 <description>Lihtne kalender mis näitab TLÜ uudised ning teie Google Calender
sündmused</description>
 <icons>
  <icon height="48" width="48" src="icon.png"/>
 </icons>
 <hosts>
  <host name="sidebar">
   <autoscaleDPI>true</autoscaleDPI>
   <base type="HTML" apiVersion="1.0.0" src="tudengi_kalender.html"/>
   <permissions>Full</permissions>
   <platform minPlatformVersion="1.0"/>
  </host>
```

```
 </hosts>
</gadget>
```

Tudengi kalendri „settings.html”

```
<html xmlns=“http://www.w3.org/1999/xhtml”>
<head>
   <title>Tudengi Kalender</title>
   <meta http-equiv=Content-Type content=“text/html; charset=UTF-8”>
        <META HTTP-EQUIV=“MSThemeCompatible” CONTENT=“yes”>
        <title>Tudengi agenda settings</title>
        <style>
                body{
         width: 230px;
                        height:270px;
      }
   </style>
                <script type=“text/javascript”>
                function settingsClosing(p_event) {
                        if (p_event.closeAction == p_event.Action.commit){
                      System.Gadget.Settings.write(“w_font”,
settingsForm.w_font.value);
                      System.Gadget.Settings.write(“w_height”,
settingsForm.w_height.value);
                      System.Gadget.Settings.write(“w_width”,
settingsForm.w_width.value);
                      System.Gadget.Settings.write(“gFeed”,
settingsForm.gFeed.value);
                        } else{
                                p_event.cancel = false;
                        }
```

```
                System.Gadget.document.parentWindow.settingsChanged();


        }
        function getSettings(){
                settingsForm.w_font.value =
System.Gadget.Settings.read("w_font");
                settingsForm.w_height.value =
System.Gadget.Settings.read("w_height");
                settingsForm.w_width.value =
System.Gadget.Settings.read("w_width");
                settingsForm.gFeed.value =
System.Gadget.Settings.read("gFeed");
                //if not set, use defaults
                if(settingsForm.w_font.value == ""){
                        settingsForm.w_font.value = 12;
                }
                if(settingsForm.w_width.value == ""){
                        settingsForm.w_width.value = 255;
                }
                if(settingsForm.w_height.value == ""){
                        settingsForm.w_height.value = 400;
                }
                if(settingsForm.gFeed.value == ""){
                        settingsForm.gFeed.value = "You GCal feed";
                }
        }
        function runSettings(){
                System.Gadget.onSettingsClosing = settingsClosing;
                getSettings();
        }
```

```
        </script>
</head>
<body onload="runSettings();">
<form name='settingsForm' action="">
        <input id="w_width"type="text" name="w_width" /> Width(px)<br>
        <input id="w_height"type="text" name="w_height" /> Height(px)<br>
        <input id="w_font" type="text" name="w_font" /> Font(px)<br>
        <br>
        <input id="gFeed" type="text" name="gFeed" />Gcal feed<br>
</form>
**********************************************************************
To find your google calendar feed:<br>
1)Open your Google Calendar page.<br>
2)Under "My calendars", click the arrow icon next to your account.<br>
3)Select Calendar settings.<br>
4)Click the orange "XML" button in the "Private Address" area.<br>
5)Paste the link you get into the above field.<br>
**********************************************************************
</body>
</html>
```

Tudengi kalendri peafail „tudengi_kalender.html"

```
<html>
<head>
   <title>Tudengi Kalender</title>
   <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
   <link href="css/calendar.css" rel="stylesheet" type="text/css" />
   <style>
                body{
```

```
                    margin:0px;
                    margin-left:5px;
        width: 255;
                    height:400;
                    font-size:12;
        font-family: verdana;
                    color: white;
                    overflow-y: auto;
                    overflow-x: hidden;
    }
            #days{
                    margin:0px;
                    margin:5px;
                    width: 220;
            }
            #calendarbackground{}
            A{text-decoration:none};
            A:HOVER {text-decoration:underline;}
  </style>
      <script type="text/javascript">
            var feeds = new
Array("https://www.google.com/calendar/feeds/it.aleks%40gmail.com/private-
e6995f478a0de506a103e78678f9412b/basic?&alt=rss",
"http://www.cvkeskus.ee/rss.php?rss=0&cat=8",
"http://twitter.com/statuses/user_timeline/116237829.rss");
            var days;
            var mydiv;
            var colorBase = new Array("#82CAFF", "#99FFCC", "#F75D59",
"#E0E0E0", "#82CAFF");
            var colors = new Array();
```

```
var feed_index = 0;
var fontSize;
var gFeed;

function settingsChanged(){
	mydiv.style.fontSize = System.Gadget.Settings.read("w_font");
	mydiv.style.height = System.Gadget.Settings.read("w_height");
	mydiv.style.width = System.Gadget.Settings.read("w_width");
}
function init(){
	mydiv = document.getElementById("rss");
	System.Gadget.settingsUI = "settings.html";

	var totalDays = 7;
	var today = new Date();
	var half = Math.floor(totalDays/2);
	days = new Array(totalDays);

	for(var i=0;i<totalDays;i++){ // get relevant days
		days[i] = new Date();
		days[i].setDate(today.getDate()-half+i);
	}
	//
	for(var i=0;i<feeds.length;i++){ // Prepare titles, dates and links
from all RSS feeds
		getRSS(feeds[i]);
	}
	drawAgenda();
```

```
        }

        function getRSS(URI){
            var XMLHttpFactories = [
                function () {return new XMLHttpRequest()},
                function () {return new
ActiveXObject("Msxml2.XMLHTTP")},
                function () {return new
ActiveXObject("Msxml3.XMLHTTP")},
                function () {return new
ActiveXObject("Microsoft.XMLHTTP")}
            ];

            function createXMLHTTPObject() { // create ajax request object
                var xmlhttp = false;
                for (var i=0;i<XMLHttpFactories.length;i++) {
                try {
                xmlhttp = XMLHttpFactories[i]();
                }
                catch (e) {
                continue;
                }
                break;
                }
                return xmlhttp;
            }

            var xmlhttp = createXMLHTTPObject();
            xmlhttp.open("GET", URI, false);
```

```
                xmlhttp.onreadystatechange=function() {
                        if (xmlhttp.readyState==4 && xmlhttp.status == 200){
                                if (window.ActiveXObject) {
                                        var doc = new
ActiveXObject("Microsoft.XMLDOM");
                                        doc.async = "false";
                                        doc.loadXML(xmlhttp.responseText);
                                } else {
                                        var parser = new DOMParser();
                                        var doc =
parser.parseFrom(xmlhttp.responseText,"text/xml");
                                }
                                formatRSS(doc);
                        }
                }
                xmlhttp.send(null);


                function formatRSS(items){ // parse response and get dates,
titles and links
                        items_count=items.getElementsByTagName('item').length;
                        if(items_count == 0) { // problematic feed
                                ptag = document.createElement("p");

        ptag.appendChild(document.createTextNode("Empty  RSS feed or
unsupported  feed format (Atom is not supported!)"));
                                mydiv.appendChild(ptag);
                                return;
                        }
                        if(typeof links == "undefined"){ // check arrays
                                links=new Array();
```

```
                titles=new Array();
                pubDates=new Array();
                gTimes=new Array();
            }
            for(var i=0; i<items_count; i++){
                colors.push(colorBase[feed_index]);
                pubDate = null;

    if(items.getElementsByTagName('item')[i].getElementsByTagName('link').length==1)

    links.push(items.getElementsByTagName('item')[i].getElementsByTagName('link')[0].firstChild.nodeValue);

    if(items.getElementsByTagName('item')[i].getElementsByTagName('title').length==1)

    titles.push(items.getElementsByTagName('item')[i].getElementsByTagName('title')[0].firstChild.nodeValue);

    if(items.getElementsByTagName('item')[i].getElementsByTagName('pubDate').length==1){ // in case of other pubDate type
                    pubDate = items.getElementsByTagName('item')[i].getElementsByTagName('pubDate');
                }else if(items.getElementsByTagName('item')[i].getElementsByTagName('dc:date').length==1) pubDate = items.getElementsByTagName('item')[i].getElementsByTagName('dc:date');
                if(pubDate.length==1){

    if(items.getElementsByTagName('category')[0]!=null && items.getElementsByTagName('category')[0].firstChild.nodeValue == 'http://schemas.google.com/g/2005#event'){ // Google calender
                        var date = new Date();
```

```
                                        var summary =
items.getElementsByTagName("atom:summary")[i].firstChild.nodeValue;

                                        summary = summary.split(" ");
                                        date.setDate(parseInt(summary[3]));

        date.setFullYear(parseInt(summary[4]));
                                        time=summary[5]+"-"+summary[7];
                                        time=time.split("&");
                                        time=time[0];
                                        if(summary[5]!="Status:"){
                                                gTimes[i] = time;
                                        }else{
                                                gTimes[i] = "Allday";
                                        }
                                        pubDates.push(date);
                                }else{

        pubDates.push(pubDate[0].firstChild.nodeValue);
                                }
                        }
                }
                feed_index++;
        }
}

        function drawAgenda(){
                for(var i in days){ // for every relevant day
                        var content = document.createElement('div');
```

```
                    content.setAttribute("id", "days");
                    var month = days[i].getMonth(); //+1
                    var day = days[i].getDate();
                    var year = days[i].getFullYear();
                    content.innerHTML += "<font
size=3><b>"+days[i].getDate()+", "+days[i].getFullYear()+"</b></font><br>";
                    for(var j=0;j<titles.length;j++){
                        var date = new Date(Date.parse(pubDates[j]));
                        if(date.getDate()==days[i].getDate()){
                            var now = new Date();
                            if(date.getHours()==now.getHours() &&
date.getMinutes() == now.getMinutes()){
                                content.innerHTML +=
"<b>"+gTimes[j]+"</b>:<font color="+colors[j]+">"+titles[j]+"</font><br>";// current
time means pubDate hasn't been set properly, thus Google - to be fixed
                            }else{
                                var hours = date.getHours().toString();
                                if(hours.length==1) hours="0"+hours;
                                var minutes =
date.getMinutes().toString();
                                if(minutes.length==1)
minutes="0"+minutes;
                                content.innerHTML +=
"<b>"+hours+":"+minutes+"</b>:<font color="+colors[j]+"><a style='color:"+colors[j]+"'
href='"+links[j]+"'>"+titles[j]+"</a></font><br>";
                            }
                        }
                    }
                    mydiv.appendChild(content);
                }
            }
```

```
        </script>
</head>
<body onload=“init()”>
   <g:background id=“calendarbackground”  src = “url(bg2.png)”>
                <div id=“rss”></div>
        <g:background/>
</body>
</html>
```