TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond
Tarkvarateaduse instituut

Raiko Limmart 143043IABB

REKLAAMIBLOKEERIJATE TÕKESTAMINE VEEBILEHTEDEL

Bakalaureusetöö

Juhendaja: Tanel Tammet
Professor

Tallinn 2017

Autorideklaratsioon

Kinnitan, et olen koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

Autor: Raiko Limmart

21.05.2017

Annotatsioon

Lõputöö eesmärgiks on luua reklaame blokeerivaid tarkvarasid tõkestav programm, mis suudaks reklaamiblokeerijate poolt blokeeritud elemente taastada. Selleks uuritakse täpsemalt internetireklaame, reklaamiblokeerijate tööpõhimõtteid ning nende tõkestamise meetodeid erinevatel veebilehtedel.

Töö tulemuseks on lihtsasti seadistatav ning erinevatel veebilehtedel kasutatav programm, mis taastab reklaamiblokeerija poolt blokeeritud elemente ning kasutab ka teisi meetodeid reklaamiblokeerijate tõkestamiseks. Loodud programm on igale veebilehele vastavalt vajadustele seadistatav. Veebilehtedel on võimalik arendust kasutada, kui neil on probleeme külaliste poolt kasutatavate reklaamiblokeerijatega. Arendus on loodud ning testitud Eesti veebilehtede põhjal.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 33 leheküljel, 6 peatükki, 35 joonist, 1 tabelit.

Abstract

Preventing advertisement blockers on websites

The usage of advertisement blockers is an on-going issue as most websites use online advertising to cover their expenses. Many websites are trying to prevent ad blockers from working on their site as their revenue is dependant on the advertisements they display.

The purpose of this thesis is to develop a program that is capable of preventing advertisement blockers from blocking advertisements. This is done by studying online advertising, researching and analysing how advertisement blockers work and how websites have prevented them so far.

In research, different advertisement blockers and their methods of advertisement blocking are studied. Different methods of preventing ad blockers are analysed on well-known Estonian websites. Acquired knowledge is applied to develop a program that is able to prevent ad blocking.

The main result of this work is a working program, which is easily configurable and can be used by different websites. The program uses various methods of preventing usage of ad blockers and its primary function is to detect advertisements that are blocked by an ad blocker and restore them. The developed program is configurable in compliance with website's requirements. Websites can take advantage of this program if they are having problems with their guests using advertisement blockers. The program has been developed and tested on Estonian websites as they are less likely to be affected by popular ad blockers such as Adblock Plus.

Further analysis on the results of developed program usage is accomplished and problems during development have been addressed. Further development is optimal as not all advertisements were possible to restore.

The thesis is in Estonian and contains 33 pages of text, 6 chapters, 35 figures, 1 tables.

Lühendite ja mõistete sõnastik

Acceptable Ad	Reklaam, mida reklaamiblokeerija tahtlikult ei blokeeri
AdMob	Mobiilirakendustele mõeldud reklaamisüsteem
Brauser	Veebilehitseja ehk veebilehtede külastamiseks mõeldud programm
Brauserilaiend	Veebilehitsejas kasutatav tarkvara
Child	Element, mis on käsitletava elemendi sees
Children	Elemendid, mis on käsitletava elemendi sees
Class	Klass ehk HTML elemendi stiili nimetus, mis on defineeritud CSS-is.
CPM	*Cost per thousand impression* ehk hind tuhande näitamise eest
CSS	*Cascading Style Sheets* ehk kaskaadilaadistik on keel veebilehtede kujundamiseks
DART	*Dynamic Advertising Reporting and Targeting*
DMCA	*Digital Millennium Copyright Act*
Domeen	Veebilehe identifikaator internetis
For loop	Sama koodi käivitamine teatud arv kordi
HTML	*HyperText Markup Language* ehk hüperteksti märgistuskeel on keel veebilehtede loomiseks
IAB	*Interactive Advertising Bureau*
Iframe	HTML elemendi tüüp ehk selle elemendi *node*
JavaScript	Programmeerimiskeel veebilehtedele
jQuery	JavaScript funktsioonide kogumik
Node	HTML elemendi või objekti tüüp
Parent	Suurem element, mille sees on käsitletav element
PHP	*Hypertext Preprocessor* on keel veebilehtede serveripoolsete lahenduste loomiseks
PPC	*Pay per click* ehk hind kliki eest
Skript	Arvutiprogramm või programmi osa
Stylesheet	Fail CSS reeglitega lehekülje kujundamiseks
Tag	HTML komponent
URL	*Uniform Resource Locator* ehk internetiaadress
Wildcard	Täht või sõna, mille asemel võib olla ükskõik mis teine täht või sõna, tähistatakse tärniga (*)

Sisukord

Jooniste loetelu

Tabelite loetelu

1 Sissejuhatus

Bakalaureusetöö uurib internetireklaamide blokeerijaid, nende tööpõhimõtteid ning blokeerijate tõkestamise meetodeid. Töö põhiülesanne seisneb reklaamiblokeerijaid tõkestava programmi loomises ning selle optimeerimises suurtemate Eesti veebilehtede jaoks.

1.1 Taust ja probleem

Paljud veebilehed kasutavad tänapäeval reklaame, et oma lehte majanduslikult toetada. Tihti on reklaamid aga kasutajatele tüütud ja seega on nende blokeerimine pidevalt kasvav trend. Reklaamide blokeerimisel on otsene mõju veebilehtedele ning seepärast on reklaamiblokeerijate kasutust viimastel aastatel proovitud piirata või tõkestada.

Töö on ettenähtud reklaame kasutavate veebilehtede väljaandjatele, kes on raskustes külastajate poolt kasutatavate reklaamiblokeerijatega. Veebilehtede omanikel on võimalik vähendada reklaamiblokeerijate kasutust nende lehel ning säilitada seal olevaid reklaame.

1.2 Ülesande püstitus

Eesmärgiks on uurida reklaamiblokeerijate tööpõhimõtete kohta ning neid teadmisi rakendada tõkestaja arendamiseks. Arendatav programm oleks kasutatav erinevatel veebilehtedel ning see oleks lihtsasti seadistatav. Programm peaks olema suuteline leidma veebilehelt reklaamiblokeerija poolt blokeeritud elemendid, ning leitud elemendid osaliselt või täielikult taastama.

1.3 Metoodika

Eesmärkideni jõudmiseks analüüsitakse reklaamiblokeerijaid ning olemasolevaid tõkestamise meetodeid. Töös arendatava programmi programmeerimiseks kasutatakse

veebiarenduse jaoks loodud tarkvara Brackets ning programmeerimiskeeleks on JavaScript. Arendamine toimub Eesti tuntumate veebilehtede põhjal.

1.4 Ülevaade tööst

Kõik töös kasutatavad viitamised on 2017 aasta seisuga. Töö koosneb neljast suuremast sisupeatükist. Peatükis number 2 räägitakse lühidalt internetireklaamide ajaloost ning nende liikidest.

Peatükis number 3 käsitletakse erinevaid reklaame blokeerivaid tarkvarasid. Uuritakse reklaamiblokeerijate tööpõhimõtete, statistika ning mõjude kohta.

Peatükis number 4 on uuritud reklaamiblokeerijate tõkestamise kohta. Käsitletakse erinevaid meetodeid tõkestamiseks ning uuritakse Eesti veebilehtede kohta, kus tõkestamine kasutust on leidnud.

Peatükis number 5 toimub reklaamiblokeerija tõkestaja arendamine ning selle tulemuste analüüsimine.

2 Reklaamid veebilehtedel

Interneti algusaegadel kasutati reklaame suhteliselt vähe ning paljud veebilehed olid tasupõhised. Kasutajad pidid tellima veebilehele õigused ligipääsuks või maksma iga kasutuse kohta kindlat tasu, et saada ligipääs sisule. Interneti massilise leviku järel mõistis reklaamitööstus potentsiaali kasutada internetti klientideni jõudmiseks. Sellest tulenevalt on kulud internetis reklaamimiseks jõudnud väga suurte summadeni. Praegusel hetkel toetub suur osa interneti sisust reklaamidele, ja kui jätta välja mõningad erandid, siis paljud veebilehed tänapäeval põhinevad just sellisel ärimudelil. Kui veebilehe eesmärk on ilma kasutustasu küsimata levitada informatsiooni, siis võetakse tihti veebilehe kulude katmiseks kasutusele reklaamid [1].

2.1 Ajalugu

Esimene reklaam, mis paigutati veebilehele, müüdi aastal 1993 adokvaadibüroole Heller Ehrman White & McAuliffe. See paigaldati lehele nimega Global Network Navigator (GNN), mis oli informatsiooni ja uudiste portaal. Aastaks 1995 maksid suurfirmad nagu Mastercard ja Zima portaalile kuni 11000 dollarit nädalas reklaami koha eest. Samal aastal leidis ka esimest korda kasutust võtmesõna-põhine reklaam, mille võttis kasutusele Yahoo [2].

Aastal 1995 leiutati hüpikaknad, mis kasutasid JavaScript funktsiooni avada brauseris uus aken. Kuna hüpikaknad muutusid tüütuks, siis loodi ka sellised hüpikaknad, mis avanesid veebilehe põhiakna taga, mitte ees. Kuidki sellised reklaamid, mis olid hüpikakendes, olid väga ärritavad ja pealetükkivad, siis need täitsid siiski eesmärgi, mida tavalised bänner-reklaamid ei suutnud – juhtida kasutaja tähelepanu reklaamile. Üsna varsti aga loodi hüpikakna blokeerijaid ning need ei leia enam tänapäeval palju kasutust [2].

Aastal 1996 loodi reklaamiagentuur DoubleClick, mis lähenes internetipõhistele reklaamidele uutmoodi. Kui vanasti ei olnud reklaamid organiseeritud ja veebilehtedel võis olla keeruline leida kliente, kes nende veebilehtedele reklaami ostaksid, siis nüüd leidis DoubleClick sellele lahenduse. Nad võtsid kasutusele kasutajate käitumispõhised

reklaamid ning suutsid reklaame paremini jälgida. Nad arendasid ka süsteemi DART, läbi mille oli reklaamijatel võimalik jälgida nende reklaamidel tehtud klikke ning optimiseerida oma reklaame. Kui klient nägi, et kuskil veebilehel reklaam väga palju klikke ei kogunud, siis see oli võimalik sealt maha võtta ja näiteks kuskile mujale panna, kus see tulusam oleks [2].

Google tutvustas oma reklaamisüsteemi, Adwordsi, aastal 2000, mis kasutas CPM mudelit. 2002. aastal võtsid nad kasutusele PPC mudeli ning nad täiendasid seda võttes kasutusele klikkide määra, mis tähendas seda, et kui odav reklaam sai rohkem klikke kui kallis, siis selle hind tõusis. Tänaseks tuleb 96% Google tulust reklaamide pealt [2].

2.2 Reklaamide liigid

- Google otsingumootori reklaamid on reklaamid, mis kuvatakse vastavalt otsitud võtmesõnale. Need reklaamid on PPC ehk igal klikil on hind. Kõiki Google reklaame on võimalik hallata Adwords reklaamisüsteemist, läbi mille luuakse ka muid internetireklaame veebilehtedel [3].

- Reklaamid sotsiaalmeedias (Facebook, Twitter, Tumblr jms) [3].

- Bänner-reklaamid on pildipõhised reklaamid, mis on tavaliselt lehekülje üleval, all või küljel. Neid on erinevates suurustes ja erinevate disainidega ning tavaliselt kasutatakse neid uudisteportaalides ning blogides. Tihti pakuvad veebilehed oma lehel kohta, kuhu klientidel on võimalik oma reklaam osta. Sellist kasutust võib võrrelda reklaamidega ajalehtedes [3].

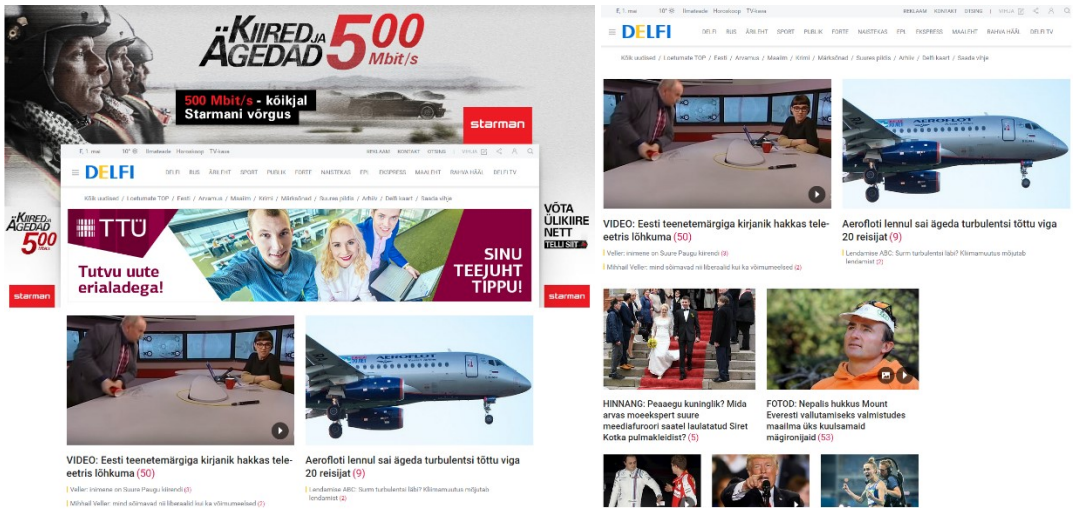
- Edasisuunavad reklaamid ehk ajaloopõhised reklaamid. Veebileht võtab arvesse kasutaja eelmiseid otsinguid ning nende põhjal genereeritakse reklaam toote kohta, mida kasutaja enne otsis [3].

- Mobiilireklaamid on reklaamid, mida kuvatakse nutitelefonides ja tahvelarvutites. Paljud veebilehed kasutavad selliste seadete jaoks eraldi seadistusi. Mobiilireklaamide alla kuuluvad ka AdMob reklaamid, mida näidatakse rakenduste sees. Neid kasutatakse rakenduste arendajate poolt, et luua ja toetada tasuta rakendusi [3].

3 Reklaame blokeerivad tarkvarad

Seoses pidevalt kasvava tasuta sisuga internetis arvavad paljud kasutajad, et sisu peakski olema täiesti tasuta ning reklaamid, mis seda toetavad, on tüütud ja pealetükkivad. Kasutajatele ei meeldi idee, et tasuta sisu jaoks on nad sunnitud reklaame vaatama. Selle tulemusena on kasutajad hakanud reklaame blokeerima (Joonis 1) [1].

Reklaamiblokeerijad on tarkvarad (pluginid või brauserilaiendused), mis eemaldavad veebilehelt reklaami sisaldava sisu või muudavad seda [4]. Blokeerijad on eksisteerinud aastast 1999, kui internetipõhised reklaamid olid veebilehitsemise teinud aeglaseks [5]. Üldjuhul arendatakse selliseid tarkvarasid kui brauserilaiendusi brauseritele nagu Chrome või Firefox. Hiljuti on ka Apple hakanud lubama sisublokeerivaid plugine oma brauseri Safari jaoks, mida kasutavad iOS seadmed. On ka teisi populaarseid tarkvarasi, näiteks Ghostery ja DisconnectMe, aga need on keskendunud pigem kasutaja privaatsusele, mitte reklaamide blokeerimisele. Reklaamiblokeerijad blokeerivad veebilehtedel reklaame ning kaitsevad kasutaja privaatsust filtreerides päringuid. Statistika on näidanud, et blokeerijate kasutus on üle maailma pidevas kasvus [6].



Joonis 1. Delfi.ee ilma reklaamiblokeerijata (vasakul) ning kasutades Adblock Plus'i (paremal).

3.1 Vajadus

Reklaamiblokeerijad on kasulikud mitmel viisil. Kõige levinum on muidugi reklaamide peitmine ja blokeerimine, aga tegelikkuses on sellel ka palju muid positiivseid külgi. Sealhulgas kiiremad veebilehtede laadimisajad, parem aku kasutus nutitelefonides, tahvelarvutites ja sülearvutites ning väiksem andmekasutus, sest ei laeta alla nii palju pilte või videosid [7].

Väga tähtsal kohal on ka kasutaja enda ja tema privaatususe kaitsmine ning otsene kaitse pahavara eest [7]. Näiteks aastal 2007 oli tomshardware.com lehel reklaam, mis viis lehele, kus kasutaja arvutisse laeti automaatselt alla viirus. Oli ka hetk, kus 6.9% reklaamidest viisid kahtlastele lehekülgedele, kust oli võimalik viirusi alla laadida. Seega võib ka mõnes mõttes ütelda, et reklaamiblokeerijate kasutus tuleneb vajadusest, mitte eelistustest [5].

3.2 Populaarsemad tarkvarad

Aastal 2004 loodi veebibrauser Firefox'i jaoks reklaamiblokeerija AdBlock, millest sai Adblock Plus. Aastal 2008, kui Google oma veebibrauseriga Chrome välja tuli, loodi sellele reklaamiblokeerija AdBlock, millel ei ole seost Adblock Plus-ga. Tänaseks on mõlemad brauserilaiendid mõlemale veebibrauserile saadaval. AdBlock ja Adblock Plus on kaks levinumat reklaame blokeerivat tarkvara [8].

AdBlock on tasuta brauserilaiendus Chrome, Safari, Firefox, Edge ja Androidi jaoks. Seda kasutab ligi 40 miljonit seadet ning selle rahastamine käib läbi annetuste. Kuna kõik annetused on valikulised, siis nad nimetavad ennast "maksa palju tahad" tarkvaraks [9].

Adblock Plus on tasuta brauserilaiendus Android, Chrome, Firefox, Internet Explorer, Maxthon, Opera, Safari ja Yandex brauseritele, mida kasutab ligi 100 miljonit seadet. Tegu on avatud lähtekoodiga tarkvaraga ning kasutab ka avatud lähtekoodiga filtreid, mis tähendab, et kasutajatel on ise võimalik neid arendada ning vastavalt vajadusele muuta. Adblock Plus kasutab Acceptable Ad programmi, mis neid majanduslikult toetab [10].

3.3 Statistika

Detsember 2016 seisuga kasutas ca 11% maailma internetikasutajatest reklaame blokeerivaid tarkvarasid (Tabel 1). Kokku oli kastutuses ligi 615 miljonit seadet, millele oli paigaldatud reklaamiblokeerija. Nendest seadmetest 62% olid mobiilsed seadmed. Võrreldes aastaga 2015, on blokeerijate kasutatavus kasvanud 30%. PageFair'i koostatud ja läbiviidud küsitluse põhjal lahkuvad 74% inimestest veebilehtedelt, kus neil palutakse reklaamiblokeerija välja lülitada. 77% reklaamiblokeerijate kasutajatest on nõus kindlaid reklaame mitte blokeerima. Peamine põhjus, miks reklaame blokeeritakse, on segavad reklaamid ning turvalisuse eesmärgil [11].

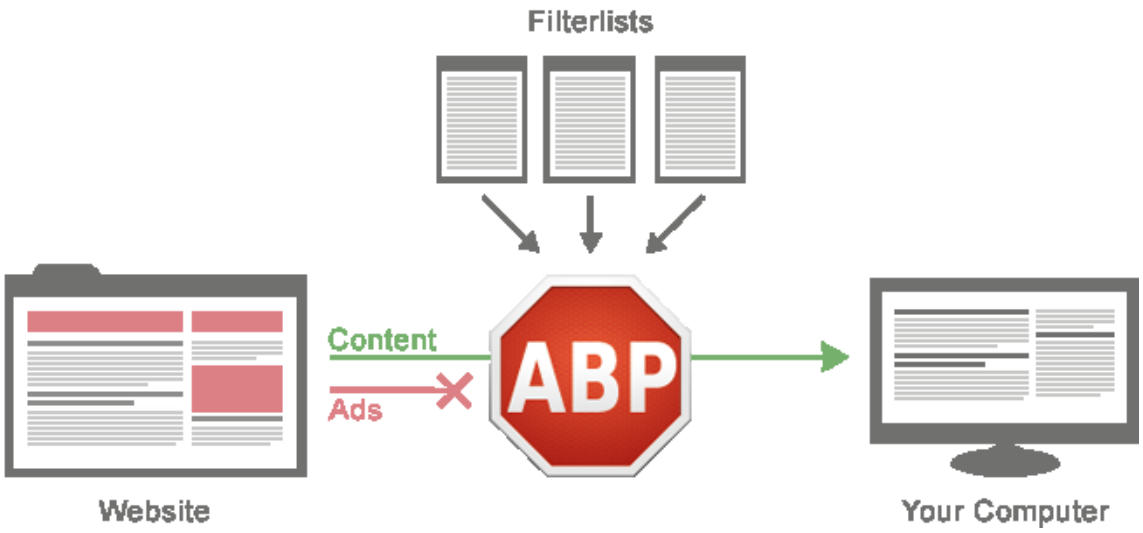
Tabel 1. Reklaame blokeerivate tarkvarade kasutus riikide järgi [11].

Riik	Kasutus	Riik	Kasutus	Riik	Kasutus
Albania	8%	*Germany*	29%	*Nicaragua*	4%
Algeria	5%	*Greece*	39%	*Nigeria*	2%
Andorra	12%	*Greenland*	7%	*Oman*	2%
Argentina	14%	*Guatemala*	2%	*Pakistan*	32%
Armenia	5%	*Honduras*	4%	*Panama*	5%
Aruba	9%	*Hong Kong*	10%	*Paraguay*	2%
Australia	20%	*Hungary*	26%	*Peru*	10%
Austria	26%	*Iceland*	27%	*Philippines*	7%
Bahamas	4%	*India*	28%	*Poland*	33%
Bahrain	4%	*Indonesia*	58%	*Portugal*	21%
Bangladesh	2%	*Iraq*	8%	*Puerto Rico*	5%
Barbados	9%	*Ireland*	39%	*Qatar*	6%
Belarus	10%	*Israel*	19%	*Romania*	21%
Belgium	12%	*Italy*	17%	*Russia*	6%
Belize	4%	*Jamaica*	5%	*Saudi*	21%
Bermuda	10%	*Japan*	3%	*Serbia*	17%
Bolivia	4%	*Jordan*	3%	*Singapore*	29%
Brazil	6%	*Kazakhstan*	3%	*Slovakia*	9%
Bulgaria	21%	*Kuwait*	4%	*Slovenia*	23%
Cambodia	8%	*Laos*	2%	*Somalia*	2%
Canada	25%	*Latvia*	17%	*South Africa*	2%
Chile	13%	*Lebanon*	2%	*South Korea*	4%
China	13%	*Libya*	2%	*Spain*	19%
Colombia	8%	*Lithuania*	21%	*Sri Lanka*	2%
Costa Rica	6%	*Luxembourg*	15%	*Sweden*	28%
Croatia	22%	*Malaysia*	8%	*Switzerland*	18%
Cyprus	15%	*Malta*	17%	*Thailand*	6%
Czech Rep.	10%	*Mexico*	9%	*Tunisia*	2%
Denmark	25%	*Moldova*	7%	*Turkey*	7%

Riik	Kasutus	Riik	Kasutus	Riik	Kasutus
Dom. Rep.	4%	*Mongolia*	3%	*Ukraine*	13%
Ecuador	9%	*Montenegro*	8%	*UAE*	14%
Egypt	5%	*Morocco*	2%	*United Kingdom*	16%
El Salvador	4%	*Myanmar*	2%	*United States*	18%
Estonia	26%	*Namibia*	3%	*Uruguay*	11%
Finland	23%	*Nepal*	2%	*Venezuela*	3%
France	11%	*Netherlands*	17%	*Vietnam*	4%
Georgia	5%	*New Zealand*	24%		

3.4 Tööpõhimõte

Kuidki reklaamiblokeerijaid on erinevaid, on nende põhifunktsioonid samad. Eesmärgiks on blokeerida või peita ära reklaam vastavalt kindla süntaksi järgi koostatud filtrile. Kui päringu URL vastab filtrile, siis see päring blokeeritakse (Joonis 2). See toimib kasutades sisueeskirju (*content policies*), mida brauser kasutab iga kord, kui see midagi laeb. Brauserilaiendustel on õigus sisueeskirju muuta ning sellest tulenevalt seadistada see nii, et blokeeritaks ära kindlalt veebilehelt sissetulevad pildifailid [8].



Joonis 2. Filtrite kasutamine Adblock Plus reklaamiblokeerijaga [10].

Selline lähenemine aga alati ei tööta, sest mõned veebilehed sisestavad reklaame nii, et kui nende päringuid blokeerida, siis ei lae veebileht üldse. Seega kasutatakse HTML elementide peitmist läbi CSS reeglite muudatuste. Elemendile, mida tahetakse peita, pannakse külge CSS omadus *„display: none!important;“*. Brauserilaiendustel on võimalik lisada kasutajapõhiseid CSS *stylesheete*, mis on suurema osakaaluga, kui

veebilehe enda *stylesheet*. Sellised reklaamid on tegelikkuses küll laetud ja olemas, aga ei ole kasutajale lihtsalt nähtavad. Peidetavaid elemente saab vastavalt vajadusele muuta reklaamiblokeerija filtris [8].

3.4.1 Filtrid

Reklaamiblokeerijatele on võimalik filtreid kasutajate poolt ise lisada. Kõige levinum filtrite loend, mis arendati koos AdBlock-ga aastal 2004, on EasyList. See koosneb umbes 60000-st erinevast reeglist ning on reklaamiblokeerijatel tavaliselt automaatselt küljes [8].

Ka Eesti veebilehtede kohta on kokku pandud filtrite loend. See asub aadressil www.adblock.ee/list.php ning koosneb umbes 10000-st erinevast reeglist. Nendest umbes 99% on delfi kohta ning nii mõnigi neist ei blokeeri konkreetset reklaami, vaid sisu delfi lehel.

Lihtsaim reegel filtris on terve domeeni blokeerimine [8]. Näiteks *'test.ee'* blokeerib:

- http://test.ee
- https://test.ee
- http://reklaamid.test.ee
- http://test.ee/reklaam.js

Reegli täpsustamiseks võib kasutada *wildcarde* [8]. Näiteks, '*test.ee/reklaamid/*'* blokeerib:

- http://test.ee/reklaamid/reklaam123
- https://test.ee/reklaamid/testreklaam/3

Reegleid saab veel täpsemini paika panna, näiteks milliste HTML tag-ide seest tuleks domeen blokeerida või läbi lasta. Täpselt määratud elemente on võimalik peita id või klassi nime järgi. Sellised reeglid ei vaja domeeni, kuid võivad seda sisaldada. Näiteks '*###advert*' blokeerib ära kõik elemendid, mille id on '*advert*', olenemata domeenist [8].

3.4.2 Elementide peitmine

Elementide peitmiseks viiakse kõik filtrisse kirjutatud elementide peitmise reeglid üle CSS-i ja need töötavad kõikide veebilehtede kohta, mida kasutaja külastab. Näiteks on filtrisse kirjutatud reegel *example.com#div(evil_ad)*, mis blokeeriks domeenil example.com elemendi *div*, mille id on *evil_ad*. Selle blokeerimiseks brauseris kasutab Adblock Plus CSS-i (Joonis 3) [12].

```
@-moz-document domain(example.com)
{
	div#evil_ad, div.evil_ad
	{
		display: none !important;
	}
}
```

Joonis 3. Elemendi peitmine kindlal domeenil kasutades Adblock Plus'i [12].

Reeglid, mida ei piirata domeeniga, piiratakse *http://* ning *https://* protokollidega (Joonis 4). Seda tehakse selleks, et mitte peita elemente brauseri enda kasutajaliideses, näiteks lehed protokolliga *chrome://* [12].

```
@-moz-document url-prefix(http://),url-prefix(https://)
{
	div#evil_ad, div.evil_ad
	{
		display: none !important;
	}
}
```

Joonis 4. Elemendi peitmine kõikidel domeenidel kasutades Adblock Plus'i [12].

3.4.3 Päringute blokeerimine

Enamus blokeerimiseks tehtavast tööst tehakse ära kasutaja arvutis. Tehniliselt on veebileht lihtsalt dokument, mis koosneb HTML elementidest. Iga elemendi jaoks peab kliendi arvuti saatma päringu serverisse, kus on sisu või muud ressursid, näiteks pildid ja reklaamid. Seejärel saab klient serverilt vastuse koos päritud sisuga. Enne igat elemendi allalaadimist kontrollib reklaamiblokeerija üle, kas sellist sisu tohib alla laadida või mitte. Seega tehniliselt blokeerija mitte ei blokeeri reklaami, vaid kliendi arvuti lihtsalt ei tee reklaami sisu kohta päringut ja seega ei raiska ka asjatult andmemahtu [5].

Reklaamiblokeerijate filtrite loendis on suuremate reklaamiserverite domeenid, kuid nende populaarsete serverite kõrval on alati ka selliseid, mida ei ole loendisse lisatud. See

on ka üks põhjustest, miks reklaamiblokeerijad kasutavad lisaks päringute blokeerimisele ka teisi blokeerimise meetodeid [13].

3.5 Funktsionaalsed ning majanduslikud mõjud veebilehtedele

Reklaamiblokeerijaid ei blokeeri internetis ainult reklaame, vaid mõningatel juhtudel ka lehekülje sisu ja sellest tulenevalt kaotab veebileht funktsionaalsuse. Näiteks British Airways ning RyanAir lehtedel ei olnud võimalik lendudele registreerida, sest reklaamiblokeerija blokeeris ära võimaluse nõustuda tingimustega. Selle asemel, et kasutajale teada anda, et reklaamiblokeerija tuleks välja lülitada, oli lihtsalt osa lehest kadunud ning külastajatele kuvati veateade [14].

Reklaamiblokeerija ei tuvasta eraldi, kas tegu on reklaamiga või mitte. Blokeerija töötab kasutades sellele ette antud filtrite loendit, milles on defineeritud blokeeritavad elemendid. Probleem peitubki nendes loendites, mida tihti muudetakse ja uuendatakse. Nendes loendites on tihti liiga agressiivsed filtrid, millest tulenevalt võib reklaamiblokeerija blokeerida ära veebilehe sisu ja lõhkuda ära lehe funktsionaalsuse. Näiteks on populaarse blokeerija Adblock Plus ühe filtri loendis reegel, kus blokeeritakse ära *.com* domeenilt tulev pilt, mille nimi on *i.gif*. Sellisel reeglil ning teistel nii laiadel reeglitel on potentsiaal lõhkuda paljusid veebilehti [15].

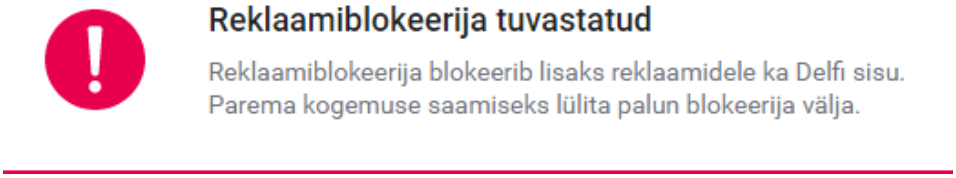
Paljudele veebilehtedele, millel on palju külastajaid, makstakse reklaamide kuvamise eest raha. See summa oleneb sellest, kui palju reklaami on vaadatud, reklaami asukohast veebilehel ning selle suurusest. Külastajad, kes kasutavad reklaamiblokeerijat, ei näe neid reklaame ja seega jääb veebilehel tulu teenimata [5]. Internetipõhised reklaamid on mitmeid aastaid väga suurel määral üleval pidanud kogu internetti. IAB andmetel oli aasta 2014 tulu reklaamidest 49.5 miljardit dollarit [6]. Aasta 2015 aastane tulu ulatus 59.6 miljardi dollarini [16]. Reklaame blokeerivad tarkvarad on igaaastaselt reklaamitööstustelt kaotanud miljardeid dollareid potentsiaalset tulu. Aastal 2015 kaotasid reklaamijad hinnanguliselt 22 miljardit dollarit [7].

4 Reklaamiblokeerijate tõkestamine

2007. aasta sügisel hakkas üks veebileht blokeerima ligipääsu sisule kasutades ühte kindlat veebibrauserit. Selleks leheks oli JackLewis.net ja brauseriks Firefox. Põhjuseks oli see, et veebilehe kasutajad olid hakanud kasutama plugini nimega Adblock Plus. Kasutajad said selle brauserile nii seadistada, et see blokeeriks ära kõik reklaamid. Kuna paljud veebilehed, sealhulgas ka JackLewis.net, teenivad sissetuleku läbi reklaamide, siis on Adblock Plus-il otsene mõju ärimudelile [1].

Hetkel on enamike veebilehtedega nii, et nad lihtsalt lepivad reklaamiblokeerijate negatiivse mõjuga oma sissetulekule. Osa veebilehti aga võitleb selle vastu. Google on näiteks Play poest, mobiilirakenduste allalaadimise keskkonnast, ära võtnud Adblock'i rakenduse, tuues vabanduseks, et see segab teiste rakenduste tööd. Hulu, üks suurimaid veebilehti telesaadete ja filmide vaatamiseks, on pidevalt reklaamiblokeerijate vastu võidelnud. Nad on näiteks blokeeritud reklaamide asemel näidanud kasutajatele 90 sekundit tühja musta ekraani ning sellele järgnedes keelanud kasutajatel sisule ligi pääseda. Nad on leidnud ka viisi, kuidas reklaamiblokeerijad üldse nende videosi ja reklaame ei mõjuta [17].

Reklaamiblokeerijate lai kasutus on viinud kirjastajad ja reklaamiblokeerijad pidevasse võitlusesse. Paljud veebilehed on hakanud tuvastama, kas kasutaja kasutab reklaamiblokeerijat või mitte. Seda tuvastades palutakse kasutajal see välja lülitada (Joonis 5). Sellised teated võivad ulatuda leebetest, mitte pealetükkivatest kirjadest veebilehe sisus, kuni sisu või funktsionaalsuse blokeerimiseni. Strateegiad reklaamiblokeerijate tuvastamiseks on pidevalt arenevad [6].



Joonis 5. Teade reklaamiblokeerija kasutusest delfi.ee veebilehel.

4.1 Blokeerija tuvastamine

Blokeerija tuvastamiseks on üldiselt kaks võimalust. Üks võimalustest on tekitada veebilehele element, mis käitub kui reklaam, kuid tegelikult seda ei ole (Joonis 6). Seejärel võrreldakse selle elemendi suurusi (Joonis 7) ning nähtavust (*display*) oodatavate väärtustega, mis korrektse laadimise korral peavad kattuma [18].

```
<div id="testAd"  class="advertisement1" style="padding:10px">
      <!-- sisu -->
</div>
```

Joonis 6. Veebilehele loodav element, mis käitub blokeerija jaoks reklaamina.

```
if ($('#testAd').height() == 0)
        <!-- kasutatakse reklaamiblokeerijat -->
}
```

Joonis 7. JavaScriptis elemendi suuruse võrdlemine.

Teine võimalus on laadida veebilehele peibutisskript, mis sisaldab muutujat ja selle väärtust (Joonis 8). Seejärel leht kontrollib (Joonis 10), kas muutjal on õige väärtus, et teha kindlaks, kas skript laeti korralikult sisse (Joonis 9). Kui muutuja väärtus on puudulik ehk *undefined*, siis tähendab see seda, et reklaamblokeerija on skripti ära blokeerinud ning veebileht teab, et kasutaja kasutab reklaamiblokeerijat [18].

```
var adBlockOff = true;
```

Joonis 8. Peibutisskripti ads.js sisu blokeerija tuvastamiseks.

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

Joonis 9. Peibutisskripti lisamine veebilehele.

```
<script>
      if( window.adBlockOff === undefined ){
        <!-- kasutatakse reklaamiblokeerijat -->
      }
</script>
```

Joonis 10. Muutuja väärtuse kontrollimine reklaamiblokeerija tuvastamiseks.

4.2 Sisu blokeerimine

Osad veebilehed kasutavad tehnoloogiat, mis piirab või tõkestab ligipääsu oma leheküljele, kui nad kasutavad reklaamiblokeerijaid. Blokeerijad on aga leidnud viise, kuidas sellest olenemata sisule ligi pääseda. See on aga, näiteks USA-s, illegaalne. DMCA-s on sätestatud, et selline tehnoloogia, mis tagab ligipääsu sisule, mis tegelikult peaks olema kaitstud, pole seaduslik. Sellepärast ei kasuta suuremad reklaamiblokeerijad sellist lähenemist ning võimaldavad sellise ülesehitusega veebilehtedel ennast kaitsta [7].

Veebilehed paluvad tihti lisada oma domeeni reklaamiblokeerija valgesse nimekirja ehk lubatud veebilehtede loendisse. Selline funktsionaalsus on paljudel blokeerijatel olemas. Nimekirja lisatud veebilehel pole blokeerija aktiivne ehk kõik kolmandate osapoolte poolt sissetulevad andmed (reklaamid) on blokeerimata. Domeeni saab nimekirja lisada blokeerija kasutajaliideses ning see salvestatakse arvuti mälusse. Sellised veebilehed tuvastavad, kui kasutajal on reklaamiblokeerija aktiivne, ning ei kuva neile lehe sisu, vaid paluvad lehe valgesse nimekirja lisada. Tavaliselt teavitatakse kasutajat ka reklaamiblokeerimise mõjust ja tagajärgedest. Selline käitumine veebilehe poolt viib aga külastajate arvu alla, sest kasutajad otsivad pigem infot kuskilt mujalt, kui näevad vaeva lehe lisamisega valgesse nimekirja [19].

4.3 Lubatud reklaamid

Aastal 2011, koostöös Adblock Plus kasutajatega, otsustati leida kompromiss reklaamipakkujate ning blokeerijate vahel. Kuna enamus kasutajad arvavad, et kõik reklaamid ei ole ühiselt pealetükkivad ning häirivad, siis loodi *Acceptable Ads* ehk lubatud reklaamide programm. See võimaldab reklaamijatel ja kirjastajatel, kes on nõustunud reklaamikriteeriumitega, lisada oma reklaamid valgesse nimekirja ehk võimaldab kuvada reklaame isegi Adblock Plus'i kasutades. Kasutajad saavad toetada selliseid reklaame hoides brauserilaienduse seadistustes seda seadistust sisse lülitatuna. Kuna kõik siiski seda ei soovi, siis on seda võimalik ka välja lülitada [20].

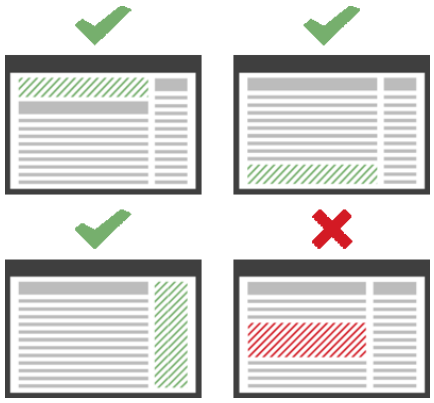
Lubatud reklaamide programm on kasulik, sest see julgustab reklaamitööstust välja töötama reklaame, mis oleksid kindla formaadi järgi ning sellest tulenevalt avaldama positiivset mõju kogu internetile. Lubatud reklaamide programm on kirjastajatele ja

reklaamijatele tasuline ning võimaldab seejuures arendada ning hoida üleval tasuta tarkvara Adblock Plus [20].

4.3.1 Kriteeriumid

Acceptable Ad ehk lubatud reklaam on mitte-animeeritud ning märgistatud reklaam, mis ei sega leheküljel sisu lugemist või vaatamist. Sellistel reklaamidel on paika pandud kindlad reeglid ja kriteeriumid, mis kehtivad kõigile [20].

- Reklaamid ei tohi segada lehekülje sisu lugemist või vaatamist. Reklaamid peavad olema paigutatud sisu ülesse, küljele või alla, mitte sisu keskele (Joonis 11) [20].



Joonis 11. Lubatud reklaamide paigutus veebilehel [20].

- Reklaamid peavad olema eristatavad kui reklaamid ning peavad eristuma sisust. Ei tohi eksitada lugejat arvama, et reklaam on osa lehekülje sisust. Sellised reklaamid peavad olema sildistatud sõnaga "*advertisement*" ehk reklaam (Joonis 12) [20].

Joonis 12. Lubatud reklaamide eristatavus muust sisust [20].

- Reklaamide suurus oleneb lehekülje paigutusest. Esmasel lehekülje laadimisel, kui reklaamid on sisu üleval või kõrval, võivad need võtta ära kuni 15% nähtavast lehe sisust. Juhul, kui reklaame on paigutatud ka lehekülje lõppu, võivad reklaamid võtta kuni 25% nähtavast sisust (Joonis 13) [20].

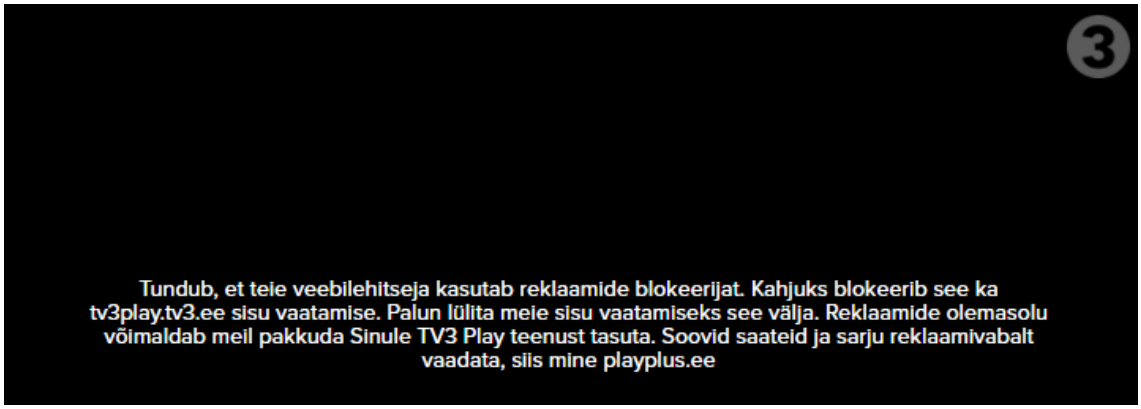
Joonis 13. Lubatud reklaamide suurused veebilehel [20].

- Reklaamis olevad tekstid ei või olla ülamääraselt värvilised või muud moodi sellised, et tähelepanu eemale juhiksid [20].

Mitteaksepteeritavad reklaamid on näiteks liikuvad, ise käivituvad videod ja helid, suuruselt muutuvad, liiga mahukad, sisu katvad või hüpikaknad [20].

4.4 Tõkestamine Eesti veebilehtedel

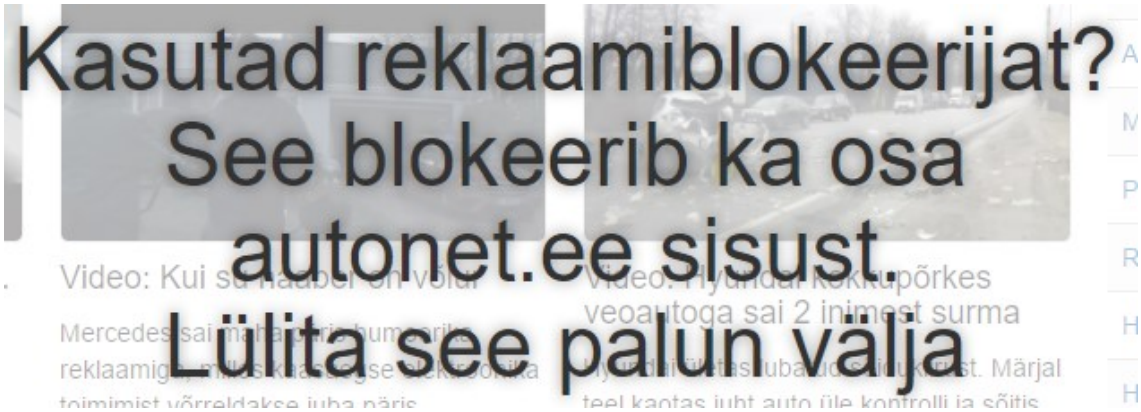
Reklaamiblokeerijate tõkestamist Eesti lehtedel esineb üsna vähe, kuid suuremad portaalid on nii mõndagi juba blokeerimise vastu teinud. Näiteks kasutades lehte tv3play.tv3.ee peab külastaja välja lülitama reklaamiblokeerija, et pääseda ligi videodele (Joonis 14). Veebilehel näidatavad videod kuuluvad tv3-le ja neid on võimalik ainult seal vaadata. Seega on neil õigus ja võimalus sisu blokeerida teades, et külalisel ei ole võimalik kuskile mujale minna ning on sunnitud reklaamiblokeerija välja lülitama. Tv3play kuvab külastajatele kuni 120 sekundilisi reklaame umbes iga kümne minuti tagant ning alati enne saate algust.



Joonis 14. Reklaamiblokeerija tõkestamine tv3play.tv3.ee lehel.

Sarnast lähenemist kasutab ka autonet.ee, kus minnes kindlale lehele kuvatakse teade ning ei lasta külastajal sisu lugeda (Joonis 15). Veebileht kontrollib, kas

reklaamiblokeerija on kasutusel ning vastavalt sellele käivitab skripti. Skript loob uue elemendi, mis katab kogu veebilehe ning mis ei oma klassi ega id-d (Joonis 16). See tagab selle, et teadet ei ole võimalik ära blokeerida. Kuvatud teadet on võimalik küll läbi veebibrauseri ühekordselt kustutada, kuid selline lähenemine ei ole efektiivne ning lihtsam on reklaamiblokeerija välja lülitada.



Joonis 15. Autonet.ee sisu blokeerimine reklaamiblokeerija kasutamisel.

```
if (window.canRunAds === undefined) {
    // adblocker detected, show fallback
    var body = $('body'),
            fullscreen_popup = '<div style="display: block; background-color:
rgba(255, 255, 255, .6); position: fixed; width: 100vw; height: 100vh; top:
0; left: 0; right: 0; bottom: 0; pointer-events: none; z-index: 99999;"><h3
style="position: fixed; top: 50%; left: 50%; -webkit-transform: translate(-
50%, -50%); -moz-transform: translate(-50%, -50%); -o-transform: translate(-
50%, -50%); -ms-transform: translate(-50%, -50%); transform: translate(-50%,
-50%); text-align: center; font-size: 48px; text-shadow: 0 0 10px rgba(0, 0,
0, .4);">Kasutad reklaamiblokeerijat?<br>See blokeerib ka osa autonet.ee
sisust.<br>Lülita see palun välja</h3></div>';
    body.append(fullscreen_popup);
    body.css({
        'overflow': 'hidden',
        'pointer-events': 'none'
    });
    $(document).bind("contextmenu", function (event) {
        event.preventDefault();
    });
    $('.container-fluid.menu').css({
        'z-index': 100000,
        'position': 'fixed',
        'top': 0,
        'left': 0,
        'right': 0,
        'width': '100%',
        'pointer-events': 'all'
    })
}
```

Joonis 16. Autonet.ee skript sisu blokeerimiseks reklaamiblokeerija kasutamisel.

5 Loodud reklaamiblokeerijate tõkestaja

Töö praktilises osas keskendutakse reklaamiblokeerija tõkestaja arendamisele ning selle tulemuste ja probleemide analüüsimisele. Käsitletakse arendatud programmi kasulikkust, programmi koodi, arenduse käigus tekkivaid probleeme ning tulemusi erinevate Eesti veebilehtede näitel.

5.1 Arenduse eesmärk

Eesmärgiks on luua igale veebilehele lihtsasti integreeritav programm, mis aitaks tõkestada elementide peitmist, sealhulgas reklaame blokeerivate tarkvarade kasutust sellel veebilehel. Programm oleks lihtsasti seadistatav vastavalt kasutaja vajadustele ning soovidele.

Programm kirjutatakse kasutades JavaScript programmeerimiskeelt ning peale seadistamist oleks see võimalik veebilehele integreerida kasutades koodi (Joonis 17) või kopeerida kogu kood ning kleepida see veebilehe failide sisse. Kopeerimine on optimaalne valik, sest arendatav programm on koodiridade poolest lühike (Lisa 1).

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

Joonis 17. JavaScript skripti sisestamine HTML lehele.

5.2 Kasulikkus

Arendatav programm aitab veebilehtedel võidelda reklaamiblokeerijate kasutusega. Kuna nende kasutus on pidevalt kasvav, siis peaksid veebilehed nendega pidevalt võitlema, et tulu ei jääks teenimata.

Programm suudab osaliselt või täielikult vältida reklaamiblokeerijate poolt tekitatud lehekülje lõhkumist. Tühjasid, katkiseid või peidetud leheküljeosasi on võimalik tagasi välja tuua või asendada muu sisuga. See parandaks lehekülje visuaalset poolt ning samal

ajal oleks võimalik kasutajale teada anda, et sellel veebileheküljel võiks reklaamiblokeerija välja lülitada.

5.3 Metoodika

Programm loodi kasutades JavaScripti ning see koosneb neljast põhilisest funktsioonist ning seitsmest muutujast, mis on mõeldud kasutajale muutmiseks. Arendamiseks kasutati veebibrauserit Google Chrome, millele oli reklaamide blokeerimiseks installitud Adblock Plus. Valitud veebibrauser ja reklaamiblokeerija on mõlemad enimkasutatud ning enamus arendusest ning testimisest viidi läbi neid kasutades. Testimist tehti ka veebibrauseris Mozilla Firefox ning mõlemas brauseris testiti nii Adblock Plus kui ka AdBlock brauerilaiendeid.

Suurem osa arendusest toimus veebilehe delfi.ee põhjal. Valik tehti selline, sest Delfi on Eesti üks enimkülastatuim veebileht ning Delfi on üks väheseid, kus on tegeletud reklaamiblokeerijate tõkestamisega. Programmi arendus toimus ka teiste Eesti veebilehtede põhjal. Näiteks postimees.ee, city24.ee ning ilm.ee. Arendatud koodi sisestamiseks veebilehtedele kasutati Chrome brauerilaiendit *Styler*, mis võimaldab igale veebilehele JavaScript koodi sisestada ilma, et oleks vaja ligi pääseda veebilehe failidele (Joonis 18).

Joonis 18. JavaScript sisestamine delfi.ee lehele arenduse eesmärgil.

5.3.1 Kasutaja valikud

Kood on programmeeritud nii, et seda oleks võimalik kasutada kõikidel veebilehtedel. Selleks on koodis olemas seadistused (Joonis 19), mida kasutaja vastavalt soovidele ja vajadustele paika saab panna.

```
var enableAntiAdblock = true;
var disableContent = true;
var enableScrolling = true;
var scrollingDistance = 500;
var enableTextFill = true;
var textToDisplay = "Siin oleks osa delfi sisust. Parema
kogemuse saamiseks lülita palun reklaamiblokeerija välja.";
var blockedList = [".dado-bnr", "test", "test*", ".notification-
ribbon", "ab--notification-ribbon"];
```

Joonis 19. Kasutaja seadistatavad muutujad.

Kokku on 7 muutujat, millest neli on programmi mingi osa sisse- ja väljalülitamine, üks on elementide loend, üks on tekst ning üks on arv.

- enableAntiAdblock – lülitab sisse kogu programmi (*true* või *false*).
- disableContent – blokeerib ära veebilehe sisu reklaamiblokeerija tuvastamisel.

- enableScrolling – võimaldab funktsioonide pidevat käivitamist, kui veebilehel alla kerida. See oleks vajalik lehtedel, kus uute elementide genereerimine toimub veebilehe kerimisel (näiteks delfi.ee, postimees.ee).
- scollingDistance – pikslite arv, mille kerimisel funktsioon käivitatakse ning elemente peidetakse. Mida suurem on muutuja, seda rohkem tuleb kerida, et funktsioon käivituks. Leiab kasutust ainult juhul, kui muutuja *enableScrolling* on *true*.
- enableTextFill – elementide, mida ei suudetud taastada, sisu asendamine tekstiga.
- textToDisplay – tekst, mida kuvada sellisel elemendil, mida programmil ei olnud võimalik taastada. Kuvatakse ainult juhul, kui muutuja *enableTextFill* on *true*.
- blockedList – loend, mis sisaldab kasutaja poolt sisestatud klasse, elementide id-sid või elemendi id *wildcarde*. Loendisse peaks lisanduma ka reklaamiblokeerijate poolt kasutatava filtrite loendi sisu.

5.3.2 Reklaamiblokeerija tuvastamine ning sisu blokeerimine

Selleks, et programm saaks blokeeritud reklaamidega tegeleda, kontrollib see enne seda, kas veebilehe külastaja kasutab reklaamiblokeerijat (Joonis 20). Selleks luuakse muutuja *abEnabled*, mille vaikimisiväärtus on *false* ehk reklaamiblokeerija pole sisse lülitatud. Seejärel, kui veebileht on lõpuni laadinud, luuakse lehele element. Elemendile pannakse külge klass nimega "advertisement1", mis on vaikimisi reklaamiblokeerijate filtrite hulgas. Selline element blokeeritakse automaatselt ära.

Reklaamiblokeerija tuvastamist tehakse läbi elemendi suuruse võrdluse. Kuna elemendile on kirjutatud ka sisu, mis programmis on "adblock test", siis peab elemendi kõrgus olema suurem kui 0. Kasutades funktsiooni *offsetHeight* elemendi peal, saab kätte selle elemendi kõrguse. Kui elemendi kõrgus on 0 ehk *offsetHeight === 0*, siis on element peidetud, mis tähendab, et külastaja kasutab reklaamiblokeerijat. Seda tuvastades muudetakse muutuja *abEnabled* väärtus *true* peale ning käivitatakse funktsioon reklaamiblokeerija tõkestamiseks. Kõige lõpuks kustutatakse loodud element ära.

Samas funktsioonis on käsitletud ka sisu blokeerimist. Kui kasutaja on valinud muutuja *disableContent* väärtuseks *true*, siis reklaamiblokeerija tuvastamisel blokeeritakse lehekülje sisu ning kuvatakse teade (Joonis 21), milles palutakse blokeerija välja lülitada. Selleks tekitatakse veebilehele uus *div* element, millele määratakse kõrgus ja laius 100% ning omadus *position:fixed*. Elemendile määratakse omadus *z-index:1000*, mis tõstab elemendi kõikide teiste elementide peale. Kuvatakse ka nupp, millele vajutamisel kogu leht uuesti laetakse. Kui külastaja on reklaamiblokeerija välja lülitanud, siis enam teadet ei kuvata.

```
var abEnabled = false;
$(document).ready(function(){
    var abTest = document.createElement('div');
    abTest.innerHTML = 'adblock test';
    abTest.className = 'advertisement1';
    document.body.appendChild(abTest);
    if (abTest.offsetHeight === 0 && enableAntiAdblock) {
        abEnabled = true;
        fixAds();
    }
    abTest.remove();

    if (disableContent && abEnabled) {
        $('body').prepend('<div style="background-
color:rgba(105, 105, 105, 0.7);position:fixed!important;
width:100%; height:100%; z-index:1000;"><div style="width:
300px; height: 200px; postition: absolute; margin: 20% auto;
background-color:#F2F2F2;border-radius: 5px; border-width:2px;
border-style:solid; border-color: black; text-align:
center;"><br><br><br>Kasutad reklaamiblokeerijat? Sisule ligi
pääsemiseks lülita see palun välja.<p><button
onClick="window.location.reload()" style="">Lülitasin
välja</button></div></div>');
    }
});
```

Joonis 20. Reklaamiblokeerija tuvastamine ning sisu blokeerimine.

Joonis 21. Lehekülje sisu blokeerimisel kuvatav teade.

5.3.3 Parandatavate elementide leidmine

Programm leiab elemente (Joonis 22) kolmel viisil: läbi klassi nimede, läbi elemendi id ning kasutades elemendi id *wildcarde*. Seejärel alustatakse blokeeritud elementide taastamist.

Esimesena leitakse, mitu erinevat väärtust kasutaja poolt seadistatud loendis *blockedList* on ning kasutades *for loopi* käiakse iga väärtus läbi. Kui väärtus loendis algab punktiga, siis on tegu klassiga. Arvestades, et ühe ja sama klassi nimega võib olla mitu elementi,

siis otsitakse kõigepealt kõik need elemendid üles. Selleks kasutatakse funktsiooni *document.querySelectorAll(classElement)*, kus *classElement* on klassi nimi. Juhul kui leitakse rohkem kui 0 elementi, ehk veebilehel eksisteerib üks või rohkem sellise klassiga elementi, siis kävitatakse iga sellise elemendi kohta funktsioon, mis selle blokeerimist tõkestab ja seda taastada üritab.

Kui loendis olev väärtus sisaldab tärni, siis on tegemist *wildcardiga*. Selle tuvastamiseks on kasutatud funktsiooni *.includes("*")*. Seejärel eemaldatakse loendis oleva väärtuse küljest tärn kasutades *.split* funktsiooni ning moodustatakse loend, mille esimene väärtus on otsitava id *wildcard*. Kuna *wildcardiga* võib leida mitu elementi, siis leitakse kõik sellised elemendid kasutades funktsiooni *document.querySelectorAll("[id^=" + elemWildcard[0] + "]")*, kus *elemWildcard[0]* on otsitava id *wildcard*. Lõpuks, juhul kui leitakse rohkem kui 0 elementi, käiakse kõik leitud elemendid *for loopiga* läbi ning iga elemendi kohta kävitatakse blokeerimist tõkestav funktsioon.

Kui väärtus ei alanud punktiga ega sisaldanud tärni, siis programm järeldab, et tegu on id-ga. Kuna korrektselt kirjutatud veebilehe elemendid omavad unikaalseid id-si, siis saab otsitava elemendi üles leida kasutades funktsiooni *document.getElementById(idElement)*, kus *idElement* on otsitava elemendi id. Juhul, kui selline element leitakse ehk ei tagastata *null*, käivitatakse blokeerimist tõkestav funktsioon.

```
function fixAds() {
    for (var o = 0; o < blockedList.length; o++) {
        if (blockedList[o].charAt(0) === ".") {
            var classElement = blockedList[o];
            var x = document.querySelectorAll(classElement);
            if (x.length > 0 ) {
                    for (var i = 0; i < x.length; i++) {
                    fixElement(x[i]);
                }
            }
        } else if (blockedList[o].includes("*")) {
            var elemWildcard = blockedList[o].split("*");
            var elements = document.querySelectorAll("[id^=" +
elemWildcard[0] + "]");
            if (elements.length > 0 ) {
                for (var i = 0; i < elements.length; i++) {
                    fixElement(elements[i]);
                }
            }
        } else {
            var idElement = blockedList[o];
            idElement = document.getElementById(idElement);
            if (idElement != null) {
                fixElement(idElement);
            }
        }
    }
}
```

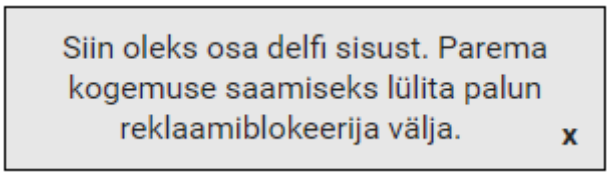
Joonis 22. Blokeeritud elementide leidmine.

5.3.4 Blokeeritud elementide väljatoomine

Kui programm on leidnud loendi *blockedList* järgi elemendi, siis käivitatakse selle elemendi kohta tõkestamise funktsioon *fixElement* (Joonis 25). Programmi efektiivsuse ja jõudluse parandamiseks lisatakse kõik elemendid, millega programm ühe korra juba tegelenud on, loendisse. See tagab selle, et programm ühte ja sama elementi mitu korda välja ei prooviks tuua. Kõik käsitletud elemendid lisatakse *unblockedList* loendisse, mis luuakse kohe programmi lugemisel veebilehe poolt. Esimese asjana kontrollibki funktsioon, kas käsitletav element on juba korra selle funktsiooniga käivitatud. Selleks kasutatakse *jQuery* funktsiooni *$.inArray(element, unblockedList)*, kus *element* on funktsiooni sisestatud element ning *unblockedList* on eelnevalt kirjeldatud loend. Juhul kui element selles loendis ei ole ehk *$.inArray(element, unblockedList) == -1*, siis alustatakse elemendi blokeerimise tõkestamisega.

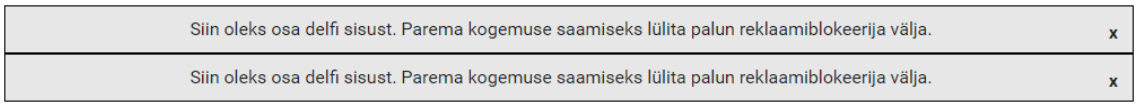
Funktsioon paneb paika muutuja kõikide CSS omaduste kohta kasutades funktsiooni *element.style.cssText* ning lisab sinna seejärel otsa omaduse *display: block!important,* mis kirjutab üle eelnevalt reklaamiblokeerija poolt pandud omaduse *display: none!important*. See tagab programmi põhilise tööülesande – elemendi peidust väljatoomine veebilehel. Seejärel toimub sellise elemendi sisu asendamine ning külastaja teavitamine reklaamiblokeerija kasutusest (Joonis 23). Seda tehakse elementidele, mida tõkestaja ei suutnud välja tuua ja taastada. See eeldab kolme asja:

- Elemendi *parent* element ega *children* elemendid ei tohi olla juba korra funktsiooni poolt käsitletud.
- Kasutaja peab olema muutuja *enableTextFill* väärtuse seadistanud *true* peale.
- Elemendi kõrgus peab olema 0 või elemendi esimene *child* element peab olema *iframe node*.



Joonis 23. Külastaja teavitamine reklaamiblokeerija kasutusest.

Esmalt kontrollitakse *children* elemente. Selleks leitakse elemendi kõik *children* elemendid kasutades funktsiooni *element.children* ning kontrollitakse, kas mõni neist on lisatud loendisse *unblockedList*. Juhul, kui mõni on lisatud, lisatakse ka element ise sinna. See tagab selle, et elemendi sisu ei hakata asendama ning et ei tekiks külastajale kahekordset teavitust (Joonis 24). Samasugune kontroll samal eesmärgil tehakse ka *parent* elemendi kohta.



Joonis 24. Kahekordne teavitus reklaamiblokeerija tuvastamisest

Järgmisena kontrollitakse *enableTextFill* väärtust ning siis seda, et elemendi kõrgus oleks 0 või elemendi *child node* oleks *iframe*. Kui elemendi kõrgus on 0, siis tähendab see seda, et element küll suudeti peidust välja tuua, kuid sisu ei suudetud taastada. *Iframe* elemendi puhul ei kontrollita kõrgust, sest element, mille sees *iframe* on, võib juba omada mingit kõrgust, aga *iframe* ise on blokeeritud. Üldjuhul blokeerivad reklaamiblokeerijad

kolmandate osapoolte poolt kuvatava sisu ehk *iframe* ära, aga on olukordi, kus sisu võib ka olla mitteblokeeritud. Sellepärast toimub programmis sisu asendamine nii, et kirjutatakse vana sisu järele, mitte selle asemele. Kasutatakse funktsiooni *element.innerHTML* +=, kus *element* on element, kuhu sisse kirjutatakse.

```
var unblockedList = [];
function fixElement(element) {
    if ($.inArray(element, unblockedList) == -1) {
        var elementCss = element.style.cssText;
        element.style.cssText = elementCss + "display: block!important";

        if (element.children.length > 0 ) {
            for (var p = 0; p < element.children.length; p++) {
                if ($.inArray(element.children[p], unblockedList) != -1) {
                    unblockedList.push(element);
                }
            }
        }

        if (($.inArray(element, unblockedList) == -1 &&
$.inArray(element.parentElement, unblockedList) == -1) && enableTextFill &&
(element.offsetHeight == 0 || (element.firstChild &&
element.firstChild.nodeName == "IFRAME"))) {
            element.innerHTML += "<div  style='border-width: 1px; border-
style: solid; border-color: black; padding: 10px; background-color: rgba(221,
221, 221, 0.70); font-size: 16px;'>" + textToDisplay + "<span
onclick='$(this).parent().hide();' style='cursor:pointer; font-
weight:bold;float:right; padding:3px;'>x</span></div>";
        }
        unblockedList.push(element);
    }
}
```

Joonis 25. Elemendi blokeerimise tõkestamine.

5.3.5 Tõkestamine veebilehe kerimisel

Paljud lehed, näiteks delfi.ee ja postimees.ee, genereerivad veebilehele pidevalt sisu juurde, kui külastaja lehel alla poole kerib. See tähendab seda, et tõkestaja peab pidevalt oma funktsioone kerimisel käivitama (Joonis 26). Selleks peab kasutaja olema muutujate *enableScrolling* ning *enableAntiAdblock* väärtuse seadistanud *true* peale. Algselt paneb programm paika muutuja *scrolled*, mille algne väärtus on 0 ehk keritud on 0 pikslit. Seejärel kirjutab funktsioon iga kerimisega muutuja *distance* väärtuse, milleks on vahemaa lehekülje praegusest hetkest ning algsest hetkest ehk kõige ülevalt. Kui sellest vahemaast lahutada juba eelnevalt defineeritud väärtus *scrolled* ning see on suurem kui

kasutaja defineeritud muutuja *scrollingDistance* väärtus, siis käivitab funktsioon tõkestamise funktsiooni *fixAds* ning muudab ära muutuja *scrolled* väärtuse ehk nullib keritud vahemaa.

```
var scrolled = 0;
$(document).scroll(function() {
    if (abEnabled && enableScrolling && enableAntiAdblock) {
        var distance = $(window).scrollTop();
        if(distance - scrolled > scrollingDistance){
            fixAds();
            scrolled = distance;
        }
    }
});
```

Joonis 26. Tõkestamise funktsiooni käivitamine lehekülje kerimisel.

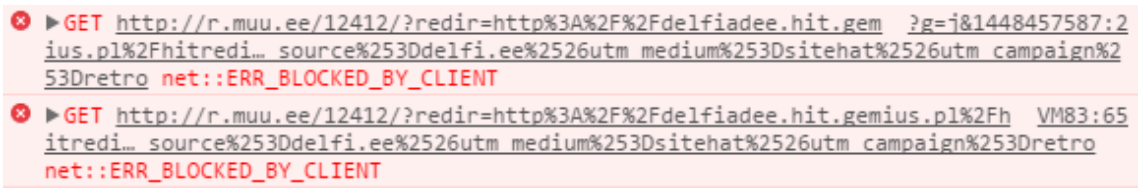
5.4 Probleemid ning takistused

Suurimaks probleemiks oli efektiivse ning korrektselt töötava programmi arendamine, mis tagaks veebilehe funktsionaalsuse ning samal ajal reklaamiblokeerija tõkestamise. Algselt läheneti tõkestamisele kasutades juhuslikku elementide nimetamist (Joonis 27). Kõik klassi ning elementide id nimed muudeti ära juhuslikult genereeritud nime vastu, milleks kasutati funktsiooni *random*. Selle tulemusena kadusid ümbernimetatud elementidelt ära CSS reeglid ning lõhkus seda tehes veebilehte, muutes visuaalselt sisu. Lahendamiseks kasutati funktsiooni *getComputedStyle*, mis võimaldas leida kõik elemendi CSS reeglid. Seda kasutades lisati uutele elementide nimedele nende vanad reeglid, eemaldades samal ajal ära *display:none* reegli, mille reklaamiblokeerija külge pani. Selline lähenemine aga ei töötanud, kui elemendil genereeriti nii uus klass kui ka uus elemendi id. Sellest tulenevalt jõuti lahenduseni, kus elemendi ümbernimetamise asemel kirjutati üle reklaamiblokeerija poolt kirjutatav reegel *display:none*.

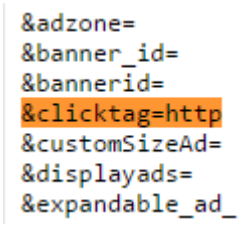
```
        var rndName = (Math.random() + 1).toString(36).substring(7);
        var elementStyle = document.defaultView.getComputedStyle(idElement,
null).cssText;
        elementStyle = elementStyle.split("display: none;");
        elementStyle = elementStyle.join();
        var style = document.createElement('style');
        style.type = 'text/css';
        style.innerHTML += '#' + rndName + ' { ' + elementStyle + ' } ';
        document.getElementsByTagName('head')[0].appendChild(style);
```

Joonis 27. Blokeerija tõkestamine kasutades elemendi id ümbernimetamist.

Takistusena võib välja tuua sellised reklaamiblokeerija poolt blokeeritud reklaamid, mille URL-ide osad olid reklaamiblokeerija filtrite loendis defineeritud. Sellest tulenevalt ei lasknud veebibrauser päringuid nendele URL-idele teha ning reklaamide kuvamine polnud võimalik. Näiteks veebilehel delfi.ee on sellise sisuga *iframe* elemendid (Joonis 28). Täpsemalt uurides selgus, et reklaam blokeeritakse, kui *iframe* atribuut *src* URL sisaldab teksti *&clicktag=http*. Võttes lahti AdBlockPlus kasutatava filtrite loendi *EasyList* leiabki sealt selle sama filtri (Joonis 29).



Joonis 28. Blokeeritud päringud delfi.ee lehel.



Joonis 29. Lõik Adblock Plus'i kasutatava EasyList filtrite loendist.

Sellele prooviti läheneda võttes *src* atribuutilt küljest *&clicktag=http*, kuid tulemuseks olid katkised reklaamid. Visuaalselt oli reklaam küll olemas, kuid kõrgused ja laiused olid paigast ning osade reklaamide asetus oli vigane või tekkis tühi valge kast. Selline lähenemine eemaldas ka reklaamilt edasisuunava lingi, mida hiljem juurde ei olnud võimalik lisada, sest olemasolev *iframe* suunas juba *blank* ehk tühjale lehele.

Eelnevale iframe sisuga elementide taastamisele prooviti leida lahendus kasutades URL-i saatmist kolmandale osapoolele ning selle uuesti kuvamist. Prooviti edasisuunamist kasutades PHP funktsiooni *header('Location: URL')*, mis ebaõnnestus, sest reklaamiblokeerija suutis siiski laadimise blokeerida. Lisaks sellele prooviti kogu reklaami sisu ümber kirjutada uuele lehele, kuid see ebaõnnestus, sest sellisel juhul puudusid lehel vastavad JavaScript skriptid, mis reklaami genereerimiseks vajalikud olid.

5.5 Tulemused

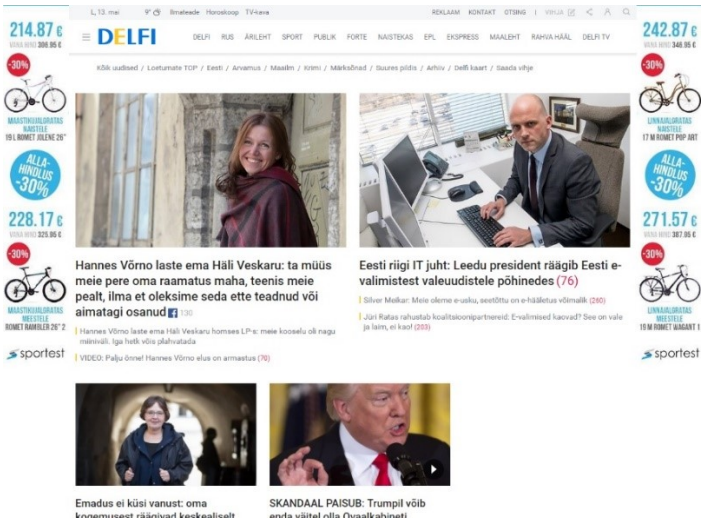
Kõige paremaid tulemusi on programmi kasutades näha delfi.ee veebilehel, sest sellel lehel pole kasutuses Google'i või teiste suuremate reklaamifirmade reklaamide genereerimine. Enamus delfi.ee reklaamidest asuvad nende enda serveris ning sellel põhjusel pole nende päringud reklaamiblokeerijate poolt blokeeritud. Ilma reklaamiblokeerijata (Joonis 30) on esmasel laadimisel delfi.ee lehel kaetud ligi 52% sisust reklaamidega. Selle arvutamiseks leiti lehel olevate reklaamide suurused pikslites, milleks oli kokku 619400 pikslit, ning jagati see kogu lehe pikslite arvuga, milleks oli 1260*950 ehk 1197000. Saadud tulemus korrutati 100-ga, et viia arv protsendilisele väärtusele.

Joonis 30. Delfi.ee sisu ilma reklaamiblokeerijata.

Kasutades brauserit Chrome ning blokeerijat Adblock Plus ilma lisaseadistusi tegemata, oli veebilehel kaetud ligi 35% sisust reklaamidega (Joonis 31).

Joonis 31. Delfi.ee kasutades Adblock Plus'i ilma lisaseadistusteta.

Lisades reklaamiblokeerijale seadistuse HTML klassi *.dado-bnr* blokeerimiseks, saadi tulemuseks, et reklaamidega kaetud sisu oli ligi 9% (Joonis 32). Klassi blokeerimiseks valiti Adblock Plus menüüst valik *Block element* ning vajutati suvalisele reklaamile.



Joonis 32. Delfi.ee kasutades Adblock Plus'i ning klassi *.dado-bnr* blokeerimist.

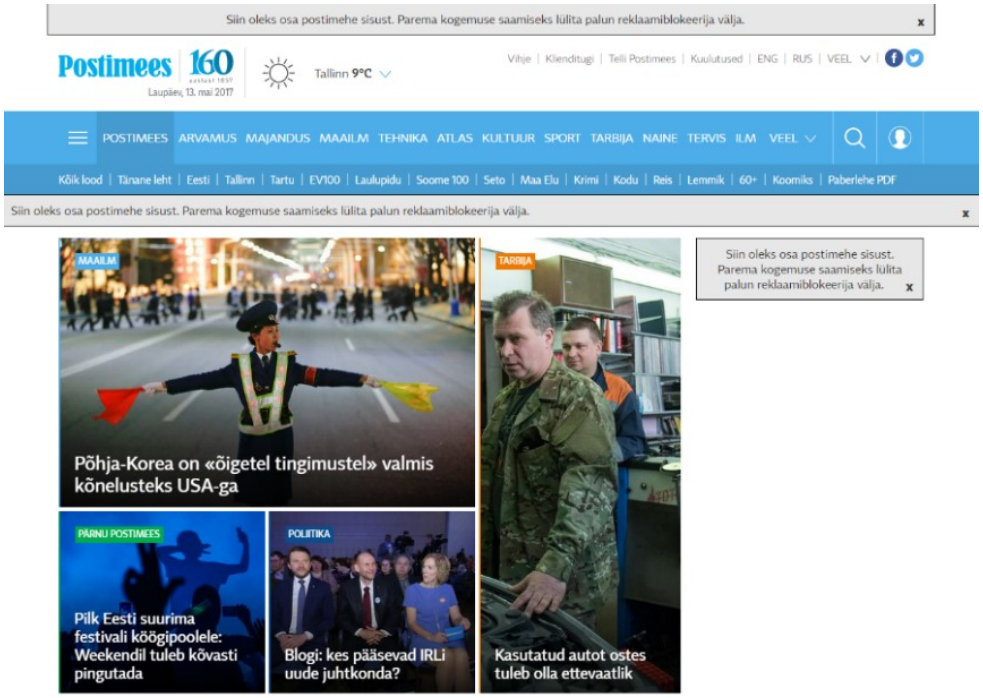
Võttes kasutusele tõkestamiseks arendatud programmi ning lisades selle muutuja *blockedList* loendisse väärtuse *.dado-bnr* (Joonis 33), oli reklaamidega kaetud 52% veebilehe sisust ehk veebilehe reklaamid suudeti viia reklaamiblokeerija eelsesse seisu (Joonis 34). Kogu lehe laadimisel leiti kokku 34 elementi, mille reklaamiblokeerija oli blokeerinud, ning nendest 25 suudeti täielikult taastada.

```
var blockedList = [".dado-bnr"];
```

Joonis 33. Arendatud programmi muutuja *blockedList* ning selle väärtus *.dado-bnr* klassi blokeerimise tõkestamiseks.

Joonis 34. Delfi.ee reklaamiblokeerija tõkestamine Adblock Plus'i kasutamisel ning klassi *.dado-bnr* blokeerimisel.

Kasutades arendatud programmi leheküljel postimees.ee, leiti lehelt kokku 49 reklaami. Kuna postimees.ee kasutab Google'i reklaamide genereerimist AdSense, siis ei olnud võimalik reklaame täielikult taastada, sest reklaamide päringud olid reklaamiblokeerija poolt blokeeritud ning elementide uuesti väljatoomisest ei piisanud. Kõik sellised reklaamid suudeti üles leida ning külastajale vastav teade kuvada (Joonis 35).



Joonis 35. Tõkestaja kasutamine veebilehel postimees.ee.

6 Kokkuvõte

Töö eesmärgiks oli tutvuda reklaamiblokeerijate tööpõhimõtetega ning nende teadmiste põhjal luua reklaame blokeerivaid tarkvarasid tõkestav programm, mis suudaks blokeeritud elemendid veebilehelt üles leida ning neid osaliselt või täielikult taastada. Arendus toimus JavaScript programmeerimiskeeles ning arenduseks kasutati vabavara Brackets.

Tulemusena loodi programm, mis on igale veebilehele lihtsasti seadistatav ning täidab ka seatud eesmärke. Programmi on võimalik seadistada vastavalt veebilehe omadustele ning kasutaja soovidele. Arendus on veebilehele lihtsasti integreeritav eraldi failina või otse veebilehe koodi kopeerituna. Eesmärkide täitmiseks prooviti erinevaid lahendusi efektiivsuse ja võimekuse saavutamiseks.

Jõuti järelduseni, et reklaamiblokeerijate tõkestamine on võimalik, kuid mitte täielikus ulatuses. Veebilehtedelt on võimalik leida blokeeritud reklaame ning neid taastada või nende sisu millegi muuga asendada. Reklaame, mis on genereeritud kasutades tuntumaid reklaamigenereerijaid, ei olnud võimalik taastada, kuid suudeti veebilehelt üles leida. Alternatiivselt genereeritud reklaame suudeti taastada ning veebilehel uuesti kuvada.

Edasises arenduses tuleks leida lahendus sellistele reklaamidele, mille sisu arendatud programm praeguse seisuga taastada ei suuda. Hetkel on programm kõige efektiivsem veebilehtedel, kus pole reklaamide genereerimiseks kasutatud kolmandate osapoolte poolt pakutavat reklaamide genereerimist, näiteks Google AdSense'i. Tuleks leida ka lahendus töös käsitletud probleemile, kus reklaamiblokeerijad blokeerivad internetiaadresside ehk URL-ide kindlad osad.

Kasutatud kirjandus

[1] J. Vallade, „ADBLOCK PLUS AND THE LEGAL IMPLICATIONS OF ONLINE COMMERCIAL-SKIPPING,“ 2009. [Võrgumaterjal]. Available: http://heinonline.org/HOL/Page?handle=hein.journals/rutlr61&div=43&g_sent=1&collection=journals. [Kasutatud 2017].

[2] A. Oberoi, „The History of Online Advertising,“ 3 juuli 2013. [Võrgumaterjal]. Available: https://www.adpushup.com/blog/the-history-of-online-advertising/. [Kasutatud 2017].

[3] „Online Ads: A Guide to Online Ad Types and Formats,“ WordStream, [Võrgumaterjal]. Available: http://www.wordstream.com/online-ads. [Kasutatud 2017].

[4] L. Kolowich, „How Ad Blocking Works: Everything You Need to Know,“ 1 oktoober 2015. [Võrgumaterjal]. Available: https://blog.hubspot.com/marketing/how-ad-blocking-works#sm.0000k4p7tsakrerkrmf1jzoa19xs1. [Kasutatud 2017].

[5] J. L. Walbesser, „Blocking Advertisement Blocking: The War over Internet Advertising and the Effect on Intellectual Property,“ jaanuar 2011. [Võrgumaterjal]. Available: http://search.proquest.com/docview/838991935?pq-origsite=gscholar. [Kasutatud 2017].

[6] M. H. Mughees, Z. Qian, Z. Shafiq, K. Dash ja P. Hui, „A First Look at Ad-block Detection – A New Arms Race on the Web,“ 19 mai 2016. [Võrgumaterjal]. Available: https://arxiv.org/abs/1605.05841. [Kasutatud 2017].

[7] T. Barbacovi, „BLOCKING AD BLOCKERS,“ jaanuar 2017. [Võrgumaterjal]. Available: http://repository.jmls.edu/ripl/vol16/iss2/5/. [Kasutatud 2017].

[8] W. Rens, „Browser forensics: adblocker extensions,“ 12 veebruar 2017. [Võrgumaterjal]. Available: http://work.delaat.net/rp/2016-2017/p67/report.pdf. [Kasutatud 2017].

[9] „AdBlock,“ [Võrgumaterjal]. Available: https://getadblock.com/. [Kasutatud 2017].

[10] „About Adblock Plus,“ Eyeo GmbH, [Võrgumaterjal]. Available: https://adblockplus.org/en/about. [Kasutatud 2017].

[11] S. Blanchfield, „The state of the blocked web 2017 Global Adblock Report,“ veebruar 2017. [Võrgumaterjal]. Available: https://pagefair.com/downloads/2017/01/PageFair-2017-Adblock-Report.pdf. [Kasutatud 2017].

[12] „Adblock Plus internals,“ Eyeo GmbH, [Võrgumaterjal]. Available: https://adblockplus.org/faq_internal. [Kasutatud 2017].

[13] „How does an ad blocker work?,“ 19 oktoober 2016. [Võrgumaterjal]. Available: https://www.adspeed.com/Blog/howto-ad-blocker-work-1918.html. [Kasutatud 2017].

[14] M. Murgia, „Ad blockers are breaking the internet, study finds,“ 22 aprill 2016. [Võrgumaterjal]. Available: http://www.telegraph.co.uk/technology/2016/04/19/ad-blockers-are-breaking-the-internet-study-finds/. [Kasutatud 2017].

[15] J. Avery, „Website Breakdown: When Ad Blocking Goes Wrong,“ AdExchanger, 13 oktoober 2015. [Võrgumaterjal]. Available: https://adexchanger.com/data-driven-thinking/website-breakdown-when-ad-blocking-goes-wrong/. [Kasutatud 2017].

[16] Interactive Advertising Bureau, „IAB internet advertising,“ aprill 2016. [Võrgumaterjal]. Available: http://www.iab.com/wp-content/uploads/2016/04/IAB_Internet_Advertising_Revenue_Report_FY_2015-final.pdf. [Kasutatud 2017].

[17] Josiah Bubna, „The Ethics of Adblock,“ 6 mai 2013. [Võrgumaterjal]. Available: http://bubnaphotography.com/josiah/writing/TheEthicsofAdblock.pdf. [Kasutatud 2017].

[18] H. Haddadi, R. Nithyanand, S. Khattak, M. Javed, N. Vallina-Rodriguez, M. Falahrastegar, J. E. Poweles, E. d. Cristofaro ja S. J. Murdoch, „The Adblocking Tug-of-War,“ 2016. [Võrgumaterjal]. Available: http://eprints.networks.imdea.org/1540/. [Kasutatud 2017].

[19] I. Jalbă, A.-C. Olteanu ja A. Drăghici, „Customized Ad Blocking,“ september 2016. [Võrgumaterjal]. Available: http://ieeexplore.ieee.org/abstract/document/7753245/. [Kasutatud 2017].

[20] „Allowing acceptable ads in Adblock Plus,“ Eyeo GmbH, [Võrgumaterjal]. Available: https://adblockplus.org/acceptable-ads. [Kasutatud 2017].

Lisa 1 – Arendatud reklaamiblokeerija tõkestaja

```
var enableAntiAdblock = true;
var disableContent = true;
var enableScrolling = true;
var scrollingDistance = 500;
var enableTextFill = true;
var textToDisplay = "Siin oleks osa delfi sisust. Parema kogemuse saamiseks
lülita palun reklaamiblokeerija välja.";
var blockedList = [".col-has-ad", ".dado-bnr", "test", "test*",
".notification-ribbon", "ab--notification-ribbon"];

var abEnabled = false;
$(document).ready(function(){
    var abTest = document.createElement('div');
    abTest.innerHTML = 'adblock test';
    abTest.className = 'advertisement1';
    document.body.appendChild(abTest);
    if (abTest.offsetHeight === 0 && enableAntiAdblock) {
        abEnabled = true;
        fixAds();
    }
    abTest.remove();

    if (disableContent && abEnabled) {
        $('body').prepend('<div style="background-color:rgba(105, 105, 105,
0.7);position:fixed!important; width:100%; height:100%; z-index:1000;"><div
style="width: 300px; height: 200px; postition: absolute; margin: 20% auto;
background-color:#F2F2F2;border-radius: 5px; border-width:2px; border-
style:solid; border-color: black; text-align: center;"><br><br><br>Kasutad
reklaamiblokeerijat? Sisule ligi pääsemiseks lülita see palun
välja.<p><button onClick="window.location.reload()" style="">Lülitasin
välja</button></div></div>');
    }

});

function fixAds() {
    for (var o = 0; o < blockedList.length; o++) {
        if (blockedList[o].charAt(0) === ".") {
            var classElement = blockedList[o];
            var x = document.querySelectorAll(classElement);
            if (x.length > 0 ) {
                    for (var i = 0; i < x.length; i++) {
                    fixElement(x[i]);
                }
```

```
            }
        } else if (blockedList[o].includes("*")) {
            var elemWildcard = blockedList[o].split("*");
            var elements = document.querySelectorAll("[id^=" +
elemWildcard[0] + "]");
            if (elements.length > 0 ) {
                for (var i = 0; i < elements.length; i++) {
                    fixElement(elements[i]);
                }
            }
        } else {
            var idElement = blockedList[o];
            idElement = document.getElementById(idElement);
            if (idElement != null) {
                fixElement(idElement);
            }
        }
    }
}

var unblockedList = [];
function fixElement(element) {
    if ($.inArray(element, unblockedList) == -1) {
        var elementCss = element.style.cssText;
        element.style.cssText = elementCss + "display: block!important";

        if (element.children.length > 0 ) {
            for (var p = 0; p < element.children.length; p++) {
                if ($.inArray(element.children[p], unblockedList) != -1) {
                    unblockedList.push(element);
                }
            }
        }

        if (($.inArray(element, unblockedList) == -1 &&
$.inArray(element.parentElement, unblockedList) == -1) && enableTextFill &&
(element.offsetHeight == 0 || (element.firstChild &&
element.firstChild.nodeName == "IFRAME"))) {
            element.innerHTML += "<div style='border-width: 1px; border-
style: solid; border-color: black; padding: 10px; background-color: rgba(221,
221, 221, 0.70); font-size: 16px;'>" + textToDisplay + "<span
onclick='$(this).parent().hide();' style='cursor:pointer; font-
weight:bold;float:right; padding:3px;'>x</span></div>";
        }
        unblockedList.push(element);
    }
}

var scrolled = 0;
$(document).scroll(function() {
    if (abEnabled && enableScrolling && enableAntiAdblock) {
        var distance = $(window).scrollTop();
```

```
        if(distance - scrolled > scrollingDistance){
            fixAds();
            scrolled = distance;
        }
    }
});
```