Universität Leipzig

Fakultät für Mathematik und Informatik
Institut für Informatik

– Bachelorarbeit –

Programmierung einer Browser-Extension zur Anzeige von Datenschutzinformationen im PlayStore sowie Evaluation der Extension-Performance

Author
Alexander Prull
ap62puny@studserv.uni-leipzig.de
Institut für Informatik

First Supervisor	**Second Supervisor**	**External Supervisor**
Prof. Nummer 1	Prof. Nummer 2	Extern Nummer 1
ggg@informatik.uni-leipzig.de	ttt@uni-leipzig.de	rrr.eee@uuu.com
Fancy Computer Science	Institute of Rocket Science	Something AG

5. März 2019

Abstract

In dieser Arbeit wird sich mit den Eigenheiten der Browser Extension Programmierung auseinander gesetzt. Speziell geht es um Extension die Webseiten um bestimmte Informationen erweitern. Diese werden von einem Backend empfangen und zur Ladezeit der Seite eingespeist. Dabei setzt sich die Arbeit mit zwei Punkten auseinander. In erster Linie geht es darum die Ladezeit der Webseite durch das Anfordern von Informationen so wenig wie möglich zu beeinflussen. Also die Performance der Extension zu maximieren. Auf der anderen Seite wird durch die Nutzung der Extension von einer steigenden Nutzerzahl der Backend-Server mit einer steigenden Anzahl von Anfragen belastet. Um diese Probleme zu lösen werden in der Arbeit verschiedene Möglichkeiten zur Speicherung von Daten betrachtet und eine Auswahl der Methoden auf ihre Performance hin getestet.

Inhaltsverzeichnis

Kapitel 1

Einleitung

Im Zeitalter der Digitalisierung werden große Mengen Informationen immer schneller und detaillierter verarbeitet. -?- Jeder, der im Internet unterwegs ist, hinterlässt dabei wertvolle, persönliche Daten. Dabei spielt der Datenschutz eine wichtige Rolle, denn nicht immer werden diese Daten freiwillig preisgegeben. So reguliert die Datenschutzerklärung, welche Nutzerdaten verarbeitet werden. Denn in dieser Erklärung muss jeder Dienstanbieter dem Nutzer zu Beginn des Nutzungsvorgangs über Art, Umfang und Zwecke der Erhebung und Verwendung personenbezogener Daten sowie über die Verarbeitung seiner Daten in Staaten außerhalb des Anwendungsbereichs [...] in allgemein verständlicher Form zu unterrichten"(Telemediengesetz Paragraph 13 1 1)

Das Projekt Privacy Guard hat sich damit beschäftigt, inwiefern diese Datenschutzerklärungen den Vorgaben entsprechen und im Speziellen analysiert, welche Applikationen unvollständige oder mangelhafte DSEs vorweisen. Im Rahmen dieses Projektes entstand die Idee, bereits vor der Installation von Anwendungen, deren Datenschutzerklärungen zu untersuchen und den Nutzer auf mögliche Bedenken hinzuweisen.

1.1 Aufgabenstellung

Die Arbeit befasst sich mit den folgenden Aufgaben:

1. **"Programmierung einer Browser-Extension zur Anzeige von Datenschutzinformationen im PlayStore"**
2. **"Evaluierung von Caching Methoden einer Browser Extension"**

Hauptaugenmerk ist die Erläuterung von Browser-Extensions, Umsetzung eines Beispiels und Limitationen. Welche Arten von Speicher stehen einer Extension zur Verfügung und welche Performance-Ersparnisse kann durch Abspeichern von

Daten die die Extension wiederholt benötigt eingespart werden. Welche Entlastung erfährt der Server mit Backend. Aufbau und Einbindung des ausgewählten Kandidaten

1.2 Aufbau der Arbeit

Zu Beginn werden Recherche Ergebnisse vorgestellt und ausgewertet. Aus den dadurch gewonnenen Resultaten die Aufgaben genauer Definiert. Auf Basis der Recherche entsteht im 1. Teil eine Extension wobei der Fokus darauf liegt, dass diese möglichst übersichtlich bleibt und zur Evaluierung von Speichermethoden dient. Anschließend werden verschiedene Testläufe präsentiert bei denen bestimmte Methoden zur lokalen Speicherung von Daten unter den gleichen Rahmenbedingungen verwendet werden. Die Ergebnisse werden verglichen und den Erwartungen gegenübergestellt. Zuletzt wird ein Fazit gezogen.

Kapitel 2

Vorarbeit

2.1 Recherche zu Browser Extensions

2.1.1 Extension-Programmierung allgemein

Unter einer Extension versteht man ein Programm, welches den Browser um neue Funktionen ergänzt. Durch eigene Oberflächen oder Manipulation der Website erleichtern diese Erweiterungen das Nutzen des Browser. Im Gegensatz zu Plug-Ins haben Extensions Zugriff auf Browser-spezielle Funktionen und sind in der Lage über die Webseite hinaus zu agieren. Plug-Ins werden direkt in eine Webseite eingebettet und sind auf diese beschränkt. Der Oberbegriff „Add-on“ wird heutzutage Hauptsächlich als Synonym für Extension verwendet. Jeder größere Browser stellt eine Plattform zur Verfügung auf denen Extensions angeboten und installiert werden können. In der Regel sind Diese kostenlos. Können auch von außerhalb installiert werden zu Entwicklungszwecken oder wenn nicht auf der Plattform angeboten. Extensions werden in HTML, JavaScript und CSS implementiert. Dabei können alle Bibliotheken verwendet werden, welche den Browserstandards für Extensions entsprechen. Kapitel 2.3.2 befasst sich genauer damit, welche Bedingungen für diese Bibliotheken in Google Chrome gelten. Bekannte Beispiele sind Werbeblock wie UBlock Origin und VPN-Anwendungen wie Hola.

2.1.2 Existieren bereits vergleichbare Extensions

Gesucht wurde nach einer Extension die auf der Play Store Seite den Nutzer datenschutzrelevante Informationen zu den angebotenen Apps liefert, Eine Datenschutzwertung im Playstore vergibt oder den Nutzer Apps nach Berechtigungen die Apps vorschlägt. Extensions werden nach ihrer Kurzbeschreibung in den Suchergebnissen überprüft und bei nicht eindeutig Aufgabenbeschreibung die Infoseite aufgerufen(Bsp. Safe.ad im Web Store „ecosystem“).Nur deutsche und englische Ergebnisse werden berücksichtigt.

Die Recherche hat ergeben, dass unter den genannten Suchkriterien keine Chrome oder Firefox Extension gefunden wurde die Aufgabenbereich der geplanten Extension abdeckt. Einige aufgeführte Beispiele implementieren einen Teil der geplanten Funktion (Umsortierung, Tracker checken) , aber keine Extension erfüllt alle gewünschten Aufgaben.

2.1.3 Vergleich führender Browser als Plattform für die Extension

Die getroffene Auswahl des Browsers als Plattform für die Entwicklung der Extension basiert hauptsächlich auf den aktuellen Marktanteilen. Google Chrome führt mit ca. 71%, gefolgt von Mozilla Firefox mit 9,5%, Microsoft Internet Explorer mit ca. 5,7%, Apple Safari mit ca. 5%, Microsoft Edge mit 4,4% und Opera mit ca. 2,4%.

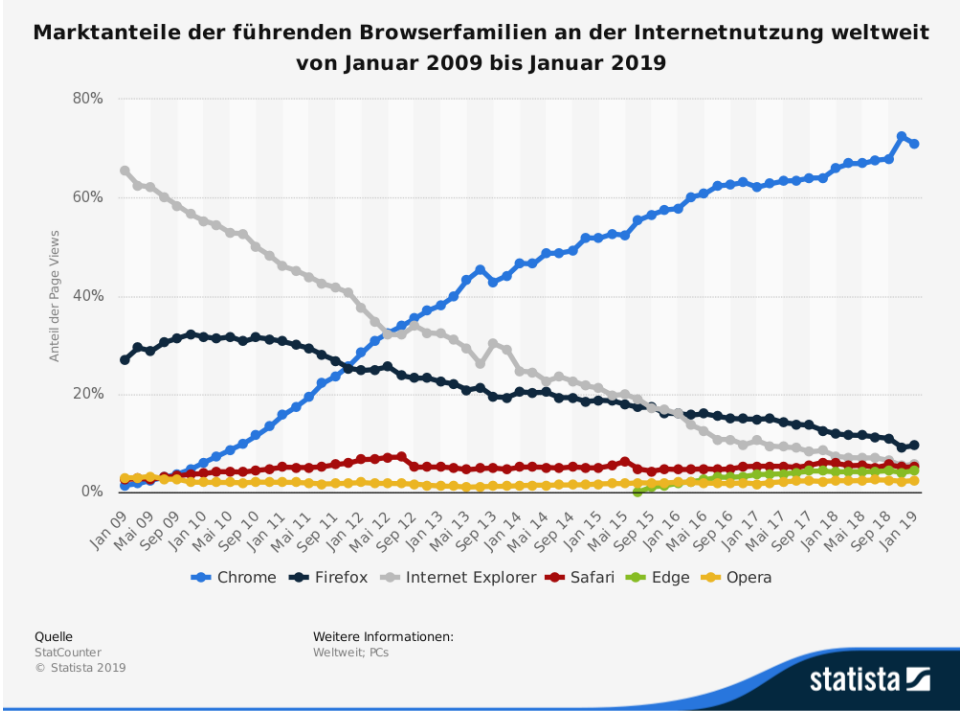


Abbildung 2.1: StatCounter. n.d. Marktanteile der führenden Browserfamilien an der Internetnutzung weltweit von Januar 2009 bis Januar 2019. Statista. Zugriff am 4. März 2019. Verfügbar unter https://de.statista.com/statistik/daten/studie/157944/umfrage/marktanteile-der-browser-bei-der-internetnutzung-weltweit-seit-2009/.

Aufgrund mangelnder Relevanz der Extension für Safari-Nutzer, sowie der Obsoleszenz des Internet Explorers, wurden diese Browser nicht weiter berücksichtigt.

Google Chrome ist den verbleibenden Alternativen Mozilla Firefox, Microsoft Edge und Opera im Punkt Marktanteile weit voraus und somit die gewählte Plattform zur Entwicklung der Extension.

Unabhängig der Implementierung bieten sowohl Mozilla[1], als auch Edge[2] eine intuitive Lösung zur Portierung der fertigen Google Chrome-Extension.

2.2 PrivacyGuard

2.2.1 Vorstellung

Das Forschungsprojekt Privacy Guard[3] wurde im Januar 2016 durch das Bundesministerium für Bildung und Forschung ins Leben gerufen. Das Institut für Angewandte Informatik[4], die mediaTest digital GmbH[5], die Quadriga Hochschule Berlin[6] und die selbstregulierung informationswirtschaft e.V.[7] haben gemeinsam Möglichkeiten entwickelt, Verbraucher auf die Verarbeitung ihrer Daten durch Handy-Applikationen aufmerksam zu machen. Dabei werden Vor- und Nachteile einzelner Aspekte der Datenverarbeitung erläutert und, bei Bedarf, Gegenmaßnahmen empfohlen.

„Ziel [...] ist die Erleichterung des Selbstdatenschutzes für Verbraucher auf mobilen Endgeräten.„(https://datenschutz-scanner.de/das-projekt.html,Stand 4.3.2019)

Die Ergebnisse des Projekts teilen sich in 3 Kategorien ein. Im Rahmen der Datenbeschaffung entstanden verschiedene Werkzeuge um Informationen über Apps zu extrahieren. Besonderer Wert wurde hier auf verlinkte Datenschutzerklärungen im Playstore und in der später installierten App auf dem Handy gelegt. Desweiteren wurden die Datenschutzerklärungen mittels eines entwickelten Pre-Tagging Tools, aber auch manuell annotiert, um die Verarbeitung der Texte zu optimieren.

Zur Datenverarbeitung entstand ein Backend[8], welches auf Anfrage der Bundle-ID einer App alle analysierten Daten übergibt. Dieses Backend bildet die Grundlage für die Datenvisualisierung von PGuard und ist die Schnittstelle der Informationen für die Browser-Extension.

Mit dem Projektabschluss im Juni 2018 stellte PrivacyGuard eine Webseite zur Analyse von Datenschutzerklärungen und Apps zur Verfügung[9]. Als Prototypen entstanden zusätzlich eine App zur Analyse aller auf dem Handy installierten Applikationen auf ihren Datenschutz und die in dieser Arbeit behandelte Browser-Extension zur Visualisierung der, durch das Projekt gewonnenen, Informationen

[1] https://developer.mozilla.org/en-US/docs/Mozilla/Add-ons/WebExtensions/Porting_a_Google_Chrome_extension

[2] https://docs.microsoft.com/en-us/microsoft-edge/extensions/guides/porting-chrome-extensions

[3] https://datenschutz-scanner.de/das-projekt.html

[4] https://infai.org/

[5] https://appvisory.com/company

[6] https://www.quadriga-hochschule.com/

[7] https://sriw.de/

[8] https://pgadmin.datenschutz-scanner.de/api/docs.html

[9] https://dseanalyser.pguard-tools.de/

über Apps im PlayStore.

2.2.2 API-Anbindung für die Extension

Sämtliche, von der Extension visualisierte, Daten werden über die Schnittstelle des Backends angefragt. Dazu benötigt die API mindestens eine Bundle-ID der App, Priorität und die Ausführtlichkeit der Antwort.

Je höher die Anfrage priorisiert ist, um so eher wird sie vom Backend verarbeitet. Bei einer hohen Ausführlichkeit umfassen die angeforderten Informationen komplette Datenschutzerklärungen und alle Metainformationen zur Datenbeschaffung. Dagegen beinhaltet eine Antwort mit niedriger Ausführlichkeit lediglich Sprache, Quelle und Extraktionsdatum der Datenschutzerklärung, sowie die Nummer der geltenden Infofelder mit jeweiligen Textpassagen aus der Datenschutzerklärung.

Für den Anwendungsfall der Browser-Extension besteht eine hohe Priorität um Wartezeiten möglichst gering zu halten. Dagegen reicht eine niedrige Ausführlichkeit zur Darstellung der nötigen Informationen für den Verbraucher. Zusätzlich bleibt so der Datenverkehr der Extension eher gering.

Die Infofelder sind der Hauptinformationsträger der Schnittstelle. Sie sind in 30 Eigenschaften unterteilt und können eine sogenannten rote Linießein. Das bedeutet, dass bei Besitz dieser Eigenschaft, die entsprechende App potentiell gegen ein Gesetz verstößt. Folgenden Infofelder sind im Rahmen des PrivacyGuard Forschungsprojekts zur Beurteilung von Apps entstanden:

2.3 Implementierung einer Google Chrome Extension

stsa

2.3.1 Eigenschaften

Die Architektur einer Chrome Extension stellt ein Paket aus mehreren Dateien dar und ist vergleichbar mit anderen Web-Technologien wie zum Beispiel Webseiten. Grundvoraussetzung für eine funktionierende Extension ist die manifest.json, welche grundlegende Informationen für den Browser bereitstellt und festlegt mit welchen Dateien und Rechten die Extension aufgebaut ist. Hinzu kommt mindestens eine HTML-Datei zur Darstellung der Inhalte und mindestens ein Skript zur Umsetzung der Funktionalität. Erweitert werden diese oft durch CSS-Dateien. Externe Bibliotheken wie JQuery können ebenfalls eingebunden werden, müssen aber aufgrund der Policies von Google Chrome vollumfänglich lokal vorliegen. Mehr dazu Im nächsten Abschnitt. Die Manifest-Datei ist im JSON-Format aufgebaut und beinhaltet sämtliche Informationen über die Extension. Wichtige Punkte sind Name der Extension, Beschreibung, Rechte und Aufbau. Unter Rechten oder „permissions“ werden alle APIs aufgelistet, welche die Extension benötigt um ordnungsgemäß zu funktionieren. Bevor ein Nutzer später die Extension installiert,

Property	**Stereo Camera**	**Multi-Mode Radar (near / far)**
meas. principle	CMOS sensor	FMCW
cycle time	60ms	66ms
latency	42ms	198ms
frequency	16fps	76 - 77 Ghz
bandwidth	—	187 Mhz
opening angle	45°	60° / 18°
range	500m (3D-vision: 50m)	60m / 200m
angle accuracy (3σ)	—	$\pm$ 1° / $\pm$ 0.1°
distance accuracy (3σ)	—	$\pm$ 0.25m
velocity accuracy (3σ)	—	$\pm 0.278 \frac{m}{s}$ / $\pm 0.139 \frac{m}{s}$

Tabelle 2.1: Overview of the properties of several sensors

muss er diesen „permissions“ zustimmen. Mehr zu den APIs im entsprechenden Abschnitt. Der Aufbau wird unter „content scripts“ (BILD?) in drei Eigenschaften unterteilt: unter welchem URL sind die Skripte aktiv, welche Skripte sind dort aktiv und welche CSS-Dateien werden dort von der Extension eingesetzt. HTML-Dateien werden als „User-Interface Elemente“ zusammengefasst und beinhalten im Normalfall eine popup.html zur Darstellung des Fensters der Extension in der oberen rechten Ecke des Browser-Fensters (BILD?). Je nach Funktionsumfang können weitere UI-Elemente eingebunden sein, um zum Beispiel die besuchte Webseite zu erweitern.

Die vorhandenen Skripte werden normalerweise in zwei Kategorien eingeteilt. Das sogenannte „Background-Skript“ dient als Event-Handler und kommuniziert zwischen Extension und Browser. Alle restlichen Skripte sind „Content-Skripte“. Sie beinhalten die eigentliche Funktionalität der Extension.

2.3.2 Funktionsumfang

2.3.3 Darstellung im Browser

2.4 Caching-Methoden

2.4.1 Welche Rolle spielt Performance?

2.4.2 Verwendete Methoden und deren Eigenschaften

Kapitel 3

Hauptteil

This chapter will first outline the problems that constitutes the main portion of this work. Each problem is described separately starting with the available data sources followed by a detailed description of the proposed solutions. After that, the proposed solutions are evaluated by empirical means and the results are presented. Performance studies are conducted to provide suitable recommendations concerning the real world application.

3.1 Erlaeuterung der Aufgabenstellungen

3.2 Aufgabe 1: Implementierung einer Browser-Extension zur Anzeige von Datenschutzinformationen im Play-Store

3.2.1 Anwendungsszenario

QUELLE STATISTIK?

Während vor einigen Jahren Applikationen hauptsächlich auf eigenen Webseiten zum Download angeboten wurden, haben sich die AppStores mitlerweile durchgesetzt. Vorteile für diese Plattformen sind unter anderem: erleichterter Zugang, Vergleiche mit anderen Applikationen und individuelle Empfehlungen. Bei der Wahl für eine bestimmte Applikation achten Nutzer auf Aspekte, wie Preis, Anzahl der Downloads und Bewertungen von anderen Nutzern.

Immer wichtiger aber auch die Frage: Welche Daten gebe ich der Applikation frei und wie werden diese verarbeitet. Der PlayStore bietet zwar einen groben Überblick, welche Daten eine Applikation von dem Handy ausließt, aber nicht wie diese vom Anbieter verarbeitet werden.

Dadurch entstehen beim Nutzer Fragen, welche der Playstore nicht beantwortet:

1. Handhabung der Daten: Wie werden die Daten verarbeitet und an wen werden diese weitergeleitet? Wird ein Profil anhand der Daten erstellt? Welche Sicherheit besteht bei der Übertragung der Daten?
2. Vor- und Nachteile der Datenverarbeitung: Kann der Anbieter die Applikation dadurch komfortabler gestalten? Wird Werbung in der Applikation personalisiert? Besteht Gefahr vor Missbrauch der Daten?
3. Kontrolle über die Daten: Welche Möglichkeiten stehen zu Verfügung im Falle von Nichteinverständnis? Ist der Umgang mit den Daten nach der Installation noch einschränkbar. Kann der Nutzer die Verwendung der Daten verbieten und trotzdem die App weiterhin nutzen?

3.2.2 Anforderungsanalyse

3.2.2.1 Funktionale Anforderungen

Aus den Fragen die bei dem Anwendungsszenario entstanden sind werden funktionale Anforderungen gebildet um konkrete Aufgaben für die Extension zu schaffen.
-Anforderungen in TEXTFORM-

-Direkt bei Apps in Anforderungen erwähnen

/F10/ Erweiterung der Informationen im PlayStore: Der Nutzer hat die Möglichkeit im Browserfenster per Aktivierung bzw. Deaktivierung der Extension zusätzliche Datenschutzinformationen zu den angezeigten Applikationen ein- bzw. auszublenden.

/F20/ Anzahl von bedenklichen Eigenschaften einer Applikation: Zu jeder Applikation erhält der Nutzer ein Feedback von der Extension, wieviele Bedenken vorliegen.

/F30/ Darstellung von kritischen Eigenschaften einer Applikation: Eigenschaften einer Applikation, welche einen erheblichen Nachteil für den Nutzer darstellen oder einen möglichen Gesetzesverstoß beinhalten werden hervorgehoben.

/F40/ Abrufen von Details zu den Bedenken: Wird ein Bedenken angezeigt, kann der Nutzer direkt Erläuterung, Handlungsempfehlung sowie Vor- und Nachteile zu diesem Bedenken abrufen.

/F50/ Empfehlung bei Suchanfragen: Basierend auf den Bedenken einer Applikation kann der Nutzer die Suchanfrage so anpassen, dass ihm unbedenkliche Applikationen priorisiert angezeigt werden.

3.2.2.2 Nichtfunktionale Anforderungen

Das Programm richtet sich in erster Linie an Nutzer, denen keine besonderen informatischen Kenntnisse abverlangt werden Extensions zeichnen sich durch

ihre Einfachheit aus. Nutzer wissen vor der Installation, welche Funktionen diese Programme haben. Die Extension soll auf den ersten Blick klar machen, welche Komponenten des Browsers erweitert oder verändert wurden. -QUELLE- Was ist eine Nichtfunktionale Anforderung + kurze Auflistung (BUCH)

/NF10/ Darstellung und Einbindung der Informationen Darstellung und Einbindung spielen bei Browser-Extensions eine wichtige Rolle. Hier wird keine grundlegend neue Oberfläche gestaltet sondern eine bereits vorhandene erweitert. Der Fokus fällt darauf, die bestehende Oberfläche so zu verändern, dass alle Elemente der Extension an der richtigen Stelle eingebaut werden. Der Nutzer soll auf den ersten Blick erkennen welche neuen Informationen zu welchen bereits bestehenden Teilen der Website gehören.

/NF20/ Persistenz der Website Im Gegenspiel zu NF10 darf die Website nicht so verändert werden, dass sie in ihrem Aussehen und ihren Funktionen zu stark von ihrem Originalzustand abweicht. Gerade bei Seiten auf denen viele Elemente automatisch generiert werden, verursachen kleine optische Veränderungen schon Probleme beim Aufbau der Website. Entsprechend müssen Informationen so subtil wie möglich eingebettet werden. So wird verhindert, dass der Nutzer die Extension nur aufgrund der Optik wieder deinstalliert.

/NF30/ Handhabung In der Extension werden viele und vor allem auch umfangreiche Informationen angeboten. Diese dürfen den Nutzer nicht überwältigen. Dennoch müssen sämtliche Punkte vgl pguard informationen an der richtigen Stelle zur Verfügung stehen.

/NF40/ Skalierbarkeit def Skalierbarkeit? Hier bezieht sich der Begriff Skalierbarkeit vor allem auf Anfragen an das Backend. Angenommen die Extension erreicht eine hohe Nutzerzahl. Dadurch steigt das Risiko auf Überlastung des Servers. Um das zu verhindern werden bei der Informationsgewinnung zwei Aspekte besonders wichtig. Zum Ersten wie aktuell die Informationen sein sollen. Je aktueller, desto öfter müssen Anfragen gesendet werden. Zum Anderen die Relevanz. Wie schnell müssen welche Informationen vorhanden sein und welche Informationen, die eine Analyse erfordern, werden erst auf spezielle Anfrage des Nutzers angefragt. Diese Anforderung stellt einen zentralen Punkt in der Entwicklung der Extension dar und wird in Aufgabe 2 detailliert behandelt.

/NF50/ Datenschutz Bei allen Webdiensten spielt der Datenschutz eine wichtige Rolle. Auch in diesem Programm sollen Daten gespeichert werden um die Anforderung /NF40/ zu unterstützen. Um Datenschutzbedenken auszuschließen muss das Format der Daten so gewählt werden, dass diese nicht personalisert werden und nach Möglichkeit komplett lokal gespeichert werden.

/NF60/ Korrektheit der Daten Alle Informationen zu Applikationen die dieses Programm darstellt werden extern von einem Server des privacy guard-Projekts

eingespeist. Dieser gewinnt die Daten hauptsächlich auf automatischen Textmining-Verfahren. Ein Problem bei diesen Verfahren ist die fehlenden Validierung der Informationen. Ursachen wie das heterogene Format von Datenschutzerklärungen und Mehrfachverlinkungen von Datenschutzinformationen können zu bei dieser Methode zu Fehlern oder Ungenauigkeiten führen. Aus diesem Grund muss dem Nutzer verdeutlicht werden, dass alle Angaben als Empfehlungen zu betrachten sind und keine verbindlichen Aussagen über Applikationen getätigt werden.

3.2.3 Aufbau der Website

Singlepage, Multipage, Kacheln, APP-ID

3.2.4 Programmaufbau

Die Struktur des Programms orientiert sich an der beschriebenen Architektur in Kapitel 2.3.3. Dieser Abschnitt befasst sich mit der Umsetzung der Anforderungen, welche im vorangegangen Kapitel aufgesetzt wurden. Dabei werden wichtige Eigenschaften aus an Teilen des Quellcodes betrachtet und getroffene Entscheidungen begründet. Außerdem werden kritische Stellen beleuchtet, welche für die Diskussion und fortführende Arbeit relevant sind.

```
{
  "name": "PGuard AppRating",
  "description": "rates PlayStore Applications based on 30 data safety criteria",
  "version": "0.1",
  "page_action":{
    "default_popup": "popup/popup.html"
  },
  "manifest_version": 2,
  "permissions": ["storage", "declarativeContent", "activeTab", "tabs" ],
  "content_scripts": [
    {
      "matches": ["*://play.google.com/store/apps*"],
      "js": ["lib/js/jquery-3.3.1.min.js", "lib/js/popper.js", "lib/js/bootstrap.min.js",
      "lib/js/fontawesome-all.js", "pguard.js", "popup/popup-controller.js"],
      "css": ["lib/css/bootstrap.min.css", "lib/css/multiapp.css"]
    }
  ],
  "background": {
    "scripts": ["background.js"],
    "persistent": false
  },
  "web_accessible_resources":[
    "lib/data/*.json",
    "lib/templates/*"
  ]
}
```

Listing 3.1: Aufbau der manifest.json

Das Manifest stellt die grundlegenden Zusammenhänge der Extension dar. Unter anderem den gewählten Entwicklungsnamen der Extension PGuard AppRatingünd die entsprechende Beschreibung. Weiterhin sind die nötigen Berechtigungen aufgeführt.

- storage: Berechtigung zum Zugriff auf Speicherplatz, um Informationen aus Backend-Anfragen zu speichern. Details dazu in Kapitel 3.3.1.

- declarativeContent: Bereitstellung von Events, wie Seitenaufruf oder -änderung und damit zusammenhängende Regeln, wie das Ausführen von Content-Skripten. Diese API wird vom Background-Skript genutzt,welches die genannten Aufgaben umsetzt.
- activeTab und tab: Gibt an, ob sich der Nutzer gerade in einem Tab befinden auf dem die Extension aktiv ist.

Außerdem werden alle Dateien ihren Rollen zugewiesen.

- Content-Skript: Unter dem Punkte content scripts"wird festgelegt, welche Skripte unter welchem URL aktiv sind. Der Ausdruck "*://play.google.com/store/apps*"bedeutet, dass die Extension auf jeder Playstore-Seite der Kategorie Apps und deren Unterverzeichnis aktiv ist. Da es sich um eine page action Extension handelt, wird lediglich eine Website als matchäufgeführt. Die beiden wichtigen Dateien hier sind pguard.jsäls das Content Skript für sämtliche Funktionen die die Erweiterung der Website betreffen und popup-controller.jsfür alle Funktionen des Popups. Hinzu kommen sämtliche Bibliotheken, welche von den Content-Skripten benötigt werden.
- Background-Skript Wie bereits erwähnt fungiert das Background-Skript "background.jsäls Eventhandler der Extension ist deshalb separat im Manifest aufgeführt.
- web_accessible_resources Diese Ressourcen sind Dateien welche der Extension zur Verfügung stehen aber selber keine Skripte sind. Sie beinhalten ausgelagerte Informationen wie Fließtexte und Templates zum Bauen von HTML-Elementen. Die popup.htmlïst hier ein Sonderfall und wird direkt dem Popup zugewiesen.

Die Background.js besteht lediglich aus Callback-Funktionen der declarativeContent API. Hier wird zur Installation der Extension ein Listener eingebunden. Dieser funktioniert mit Regeln nach dem Konditionen-Aktionen-Prinzip. Zum Start des Aufrufs werden alle bereits vorhandenen Regeln des Listeners entfernt und anschließend die übergebenen Regeln installiert. Hier benötigt das Programm eine Regel. Die Kondition prüft ob der passende URL aufgerufen wurde. Dieser stimmt mit dem String aus der Manifest-Datei überein. Ist die Kondition erfüllt, aktiviert sich das Popup.

```
1  /**
2   * Created by Alexander on 04.04.2018.
3   */
4  chrome.runtime.onInstalled.addListener(function() {
5      //removeRules, braucht declarativeContent permission, 1 = liste an regeln(undefined = keine
       Eingabe), 2= function nach loeschen
6      chrome.declarativeContent.onPageChanged.removeRules(undefined, function() {
7          //siehe removeRules, wird am ende von remove rules aufgerufen.
8          chrome.declarativeContent.onPageChanged.addRules([{
9              conditions: [new chrome.declarativeContent.PageStateMatcher({
10                 pageUrl: { schemes: ['https'], hostContains: 'play.google.com',
       pathContains:"/store/apps"}
11             })
12             ],
13             actions: [new chrome.declarativeContent.ShowPageAction()]
14         }]);
```

```
15      });
16  });
```

Listing 3.2: background.js

Das Contentskript pguard.js bildet den Hauptteil der Extension und dient zur Erfassung aller, auf der Webseite dargestellten Apps, der Einbettung von zusätzlichen Informationen durch das PGuard-Backend und optischen Anpassung der Webseite, um die neuen Inhalte ordentlich einzubinden.

Bei Skript-Aufruf werden zuerst die lokalen Bibliotheken der Extension geladen. Dazu gehören die IB texte.json sowie die HTML-Templates. Außerdem überprüft das Skript, ob lokaler Speicher zur Verfügung steht. (???)

Anschließend prüft die Funktion fillApps", ob die aktuelle Seite eine Single-App-Page oder Mutli-App-Page ist und ermittelt sämtliche Kandidaten, welche für die Einbettung der Informationen in Frage kommen. Dabei liest der JQuery-Selektor alle Elemente mit dem entsprechenden Klassnamen aus.

Mit der Funktion loadInfoPanels"wird jeder so gefundene Kandidat auf seine APP-ID überprüft. Diese befindet sich entweder in dem Attribut "data-docidöder "href". Mithilfe dieser ID durchsucht die Funktion "getStorageItem"den lokalen Speicher auf Einträge. Der Eintrag ist valide, falls er nicht leer ist und vor weniger als 3 Tagen angelegt wurde. Findet die Funktion keinen validen Eintrag im lokalen Speicher, wird eine neue Anfrage an das Backend, für die entsprechende APP-ID, erstellt.

BEISPIEL ANFRAGE

Liefert das Backend eine Antwort mit mindestens einem Ergebnis, speichert die Funktion die Informationen im lokalen Speicher ab. Bei mehr als einem Ergebnis, entscheidet eine Prioritätenliste abhängig von der zuverlässigsten Quelle, welcher Datensatz genutzt wird.

Der gewählte Datensatz wird der Funktion createPanel"übergeben. Handelt es sich bei dem Kandidaten um eine App-Page(Single-App), wird das Popover aus den Informationen direkt erstellt. Dazu wird der Datensatz mit Hilfe der IB texte.json in den entsprechenden Text umgewandelt und in die Html-Template einfügt. Bei App-Kacheln(Multi-app) baut die Funktion einen Banner in die Kachel ein. Auf diesem Banner wird mittels JQuery ein Event geladen, welches bei einem Klick auf den Banner das Popover erstellt.

BEISPIEL IB texte

```
1  /**
2   * Created by Alexander on 03.05.2018.
3   */
4  var today = new Date();
5  var anfragencounter = 0;
6  var ibJson = [];
7  var useLocalStorage = true;
8  var db;
9  //TODO set Storage by toggle in popup
```

```
var usedStorage = "indexedDB";
var ibCardTemplate = document.createElement("div");
var innerCollapseTemplate = document.createElement("div");
var trennZeichen = "|";
var dseOrigins = ["lcm", "playstore", "link_guesser", "app", "mtd", "manuall"];
var route = "get_infai_dataset_by_bundle_id";
var token = "uni_leipzig_ba_prull_2018_01_17_Aihungaem5ie7opheeme";
var verbosity = "3";
var priority = "5";
var trgDsePlay = "false";
var trgLinkGuess = "false";
var trgDseLcm = "false";
var trgDseMtd = "false";
var trgDseInApp = "false";
//var forceExec = "&force_execution=true";
var urlWholeDataSetNoRequest = "https://infaibackend.pguard-tools.de/"+ route +"?api_token="+ token
    +"&verbosity="+ verbosity +"&priority="+ priority + "&trigger_dse_playstore_download="+
      trgDsePlay
    +"&trigger_link_guesser="+ trgLinkGuess + "&trigger_dse_lcm_download="+ trgDseLcm
    +"&trigger_dse_mtd_download="+ trgDseMtd +"&trigger_dse_inapp_search="+ trgDseInApp
      +"&bundle_id=";
var urlTriggerNewAnalysis = "https://infaibackend.pguard-tools.de/"+ route +"?api_token="+ token
    +"&verbosity="+ "3" +"&priority="+ priority + "&trigger_dse_playstore_download="+ "true"
    +"&trigger_link_guesser="+ "true" + "&trigger_dse_lcm_download="+ "true"
    +"&trigger_dse_mtd_download="+ "true" +"&trigger_dse_inapp_search="+ "true"+ "&bundle_id=";

//Funktion von Mozilla die Browser auf localStorage-Verfuegbarkeit ueberprueft.
function storageAvailable(type) {
    if(type === "indexedDB" && type in window){
        return true;
    }
    try {
        var storage = window[type],
            x = '__storage_test__';
        storage.setItem(x, x);
        storage.removeItem(x);
        return true;
    }
    catch(e) {
        return e instanceof DOMException && (
                // everything except Firefox
            e.code === 22 ||
            // Firefox
            e.code === 1014 ||
            // test name field too, because code might not be present
            // everything except Firefox
            e.name === 'QuotaExceededError' ||
            // Firefox
            e.name === 'NS_ERROR_DOM_QUOTA_REACHED') &&
            // acknowledge QuotaExceededError only if there's something already stored
            storage.length !== 0;
    }
}

function getStorageItem(itemKey){
    var appInfo;

    switch (usedStorage){
        case "none":
            break;
        case "localStorage":
            appInfo = localStorage.getItem(itemKey);
            break;
        case "indexedDB":
            var transaction = db.transaction(["apps"], "readonly");
            var store = transaction.objectStore("apps");
            appInfo = store.get(itemKey);
            break;

        default:
            console.log("No storage method selected.");
    }
    return appInfo;
}

function setStorageItem(itemKey, itemValue){
    switch (usedStorage){
        case "none":
            break;
        case "localStorage":
            localStorage.setItem(itemKey, itemValue);
            break;
        case "indexedDB":
            var transaction = db.transaction(["apps"], "readwrite");
```

```
92              var store = transaction.objectStore("apps");
93              var entry = {
94                  appID: itemKey,
95                  data: itemValue
96              };
97              var request = store.put(entry);
98              request.onerror = function (ev) {
99                  console.log("Error", ev.target.error);
100                 console.dir(ev.target);
101             };
102             request.onsuccess = function (ev) {
103                 console.log("IndexedDB funkt");
104             };
105             break;
106         default:
107             console.log("No storage method selected.");
108     }
109 }
110
111 function deleteStorageItem(itemKey) {
112     switch (usedStorage){
113         case "none":
114             break;
115         case "localStorage":
116             localStorage.removeItem(itemKey);
117             break;
118         case "indexedDB":
119             break;
120         default:
121             console.log("No storage method selected.");
122     }
123 }
124
125 //Laedt neugewonnene Daten in den localStorage des Browsers
126 function castDseToStorageString(dse){
127     var extractionDate;
128     var ibString = "";
129     var freqCount = "1";
130     var ibs = [];
131     //Auf Tage runden.
132     extractionDate = Math.floor(today.getTime() /86400000);
133
134     console.log("Waehle: ", dse.origin , dse.date_infobox_calculation_finished);
135     for(var o = 0; o < dse.infoboxes.length; o++){
136         // 20 und 28 werden ignoriert
137         if(dse.infoboxes[o].id !== 28 && dse.infoboxes[o].id !== 20){
138             // 13 und 21 setzten mehrere Elemente bei den jeweiligen Eigenschaften voraus
139             //TODO aktuell werden 13 und 21 ignoriert, da bei verbosity 3 die Information fehlt.
140             if((dse.infoboxes[o].id !== 13 && dse.infoboxes[o].id !== 21)){
141                 ibs.push("" + dse.infoboxes[o].id);
142             }
143         }
144     }
145
146     for(var i = 0; i < ibs.length; i++){
147         if(i === (ibs.length - 1)){
148             ibString += ibs[i];
149         } else {
150             ibString += ibs[i] + trennZeichen;
151         }
152     }
153
154     return "" + extractionDate + trennZeichen + freqCount + trennZeichen + ibString;
155 }
156 function getInfoForIB(attr, ib){
157         for(var i = 0; i < ibJson.length; i++){
158             if(ib === ibJson[i].id){
159                 return ibJson[i][attr];
160             }
161         }
162         return "";
163 }
164 function createPopover(parentElement, ibArray){
165     var eleToRemove;
166     $(parentElement).attr({"data-toggle": "popover","data-trigger": "focus"});
167     var cardContainer = document.createElement("div");
168     cardContainer.id = "accordion";
169     for(var i = 0; i < ibArray.length; i++){
170
171         //Template
172         var cardTemplate = ibCardTemplate.cloneNode(true);
173
174         //Red-Line Markierung
175         for(j = 0; j < ibJson.length; j++){
```

```
176             if(ibArray[i] === ibJson[j].id && ibJson[j]["is_red_line"] === "true"){
177                 var cardHeader = cardTemplate.getElementsByClassName("card-header")[0];
178                 cardHeader.style.backgroundColor = "#ff8c8c";
179                 break;
180             }
181         }
182 
183         //Titel
184         var titel = cardTemplate.getElementsByClassName("ibtitel")[0];
185         titel.innerText = getInfoForIB("titel", ibArray[i]);
186 
187 
188         //Einzigartige collapse-id
189         $(titel).attr("href", "#collapse" + i);
190         $(titel.parentNode.parentNode.children[1]).attr("id", "collapse" + i);
191 
192         //Beschreibung
193         if (getInfoForIB("description", ibArray[i]) === "") {
194             eleToRemove = cardTemplate.getElementsByClassName("theader")[0];
195             eleToRemove.parentNode.removeChild(eleToRemove);
196             eleToRemove = cardTemplate.getElementsByClassName("description")[0];
197             eleToRemove.parentNode.removeChild(eleToRemove);
198         } else {
199             cardTemplate.getElementsByClassName("description")[0].children[0].innerText =
    getInfoForIB("description", ibArray[i]);
200         }
201 
202 
203         //Pro
204         if (getInfoForIB("pros", ibArray[i]).length > 0) {
205             var pros = getInfoForIB("pros", ibArray[i]);
206             var proElement = cardTemplate.getElementsByClassName("pro")[0];
207             for (var j = 0; j < pros.length; j++) {
208                 var row = document.createElement("tr");
209                 var tData = document.createElement("td");
210                 if (pros[j]["second_layer"] !== "") {
211                     tData.appendChild(innerCollapseTemplate.cloneNode(true));
212                     var firstLayerElement = tData.getElementsByClassName("firstlayer")[0];
213                     $(firstLayerElement).attr({"href": ("#collapseSecondPro" + j + "_" + i)});
214                     firstLayerElement.innerText = pros[j]["first_layer"];
215 
    $(firstLayerElement.parentNode.parentNode.parentNode).attr("id","accordionSecondPro" + j +
    "_" + i);
216                     var secondLayerElement = firstLayerElement.parentNode.nextElementSibling;
217                     $(secondLayerElement).attr({
218                         "data-parent": ("#accordionSecondPro" + j + "_" + i),
219                         "id": ("collapseSecondPro" + j + "_" + i)
220                     });
221                     secondLayerElement.children[0].innerText = pros[j]["second_layer"];
222                 } else {
223                     tData.innerText = pros[j]["first_layer"];
224                 }
225 
226                 row.appendChild(tData);
227                 //$(row).insertAfter(proElement.children[proElement.children.length - 1]);
228                 $(row).insertAfter(proElement);
229             }
230 
231         } else {
232             eleToRemove = cardTemplate.getElementsByClassName("pro")[0];
233             eleToRemove.parentNode.removeChild(eleToRemove);
234         }
235 
236 
237         //Contra
238         if (getInfoForIB("cons", ibArray[i]).length > 0) {
239             var cons = getInfoForIB("cons", ibArray[i]);
240             var contraElement = cardTemplate.getElementsByClassName("contra")[0];
241             for (j = 0; j < cons.length; j++) {
242                 var rowCon = document.createElement("tr");
243                 var tDataCon = document.createElement("td");
244                 if (cons[j]["second_layer"] !== "") {
245                     tDataCon.appendChild(innerCollapseTemplate.cloneNode(true));
246                     firstLayerElement = tDataCon.getElementsByClassName("firstlayer")[0];
247                     $(firstLayerElement).attr({"href": ("#collapseSecondCon" + j + "_" + i)});
248                     firstLayerElement.innerText = cons[j]["first_layer"];
249 
    $(firstLayerElement.parentNode.parentNode.parentNode).attr("id","accordionSecondCon" + j +
    "_" + i);
250                     secondLayerElement = firstLayerElement.parentNode.nextElementSibling;
251                     $(secondLayerElement).attr({
252                         "data-parent": ("#accordionSecondCon" + j + "_" + i),
253                         "id": ("collapseSecondCon" + j + "_" + i)
254                     });
```

```
255                     secondLayerElement.children[0].innerText = cons[j]["second_layer"];
256
257                 } else {
258                     tDataCon.innerText = cons[j]["first_layer"];
259                 }
260
261                 rowCon.appendChild(tDataCon);
262                 $(rowCon).insertAfter(contraElement);
263             }
264
265         } else {
266             eleToRemove = cardTemplate.getElementsByClassName("contra")[0];
267             eleToRemove.parentNode.removeChild(eleToRemove);
268         }
269
270         //Empfehlung
271         if (getInfoForIB("recommendations", ibArray[i]) === "") {
272             eleToRemove = cardTemplate.getElementsByClassName("recommendations")[0];
273             eleToRemove.parentNode.removeChild(eleToRemove.nextElementSibling);
274             eleToRemove.parentNode.removeChild(eleToRemove);
275         } else {
276
      cardTemplate.getElementsByClassName("recommendations")[0].nextElementSibling.children[0].innerText
      = getInfoForIB("recommendations", ibArray[i]);
277         }
278         cardContainer.appendChild(cardTemplate);
279     }
280     console.log("Karte erstellt: ", cardContainer);
281     return cardContainer;
282
283 }
284 //Baut den entsprechenden Banner fuer die Multiapp-Ansicht
285 function createPanel(parentNode, appDataArray, hasResults, isSinglePage){
286     var ibArray;
287     var popover;
288     //TODO parentNode umbenennen
289     if(isSinglePage){
290         if(hasResults){
291             ibArray = appDataArray.slice(2,appDataArray.length + 1);
292             var infoCard = document.createElement("div");
293             popover = createPopover(infoCard,ibArray);
294             infoCard.appendChild(popover);
295             parentNode.appendChild(infoCard);
296         }
297     } else {
298         var redLine = false;
299         var banner = document.createElement("div");
300         banner.classList.add("pguard");
301         var funde = document.createElement("span");
302
303         if(hasResults){
304             ibArray = appDataArray.slice(2,appDataArray.length + 1);
305             funde.innerText = "Funde: ";
306             var badge = document.createElement("span");
307             badge.classList.add("badge", "badge-secondary", "float-right");
308             badge.innerText = "" + ibArray.length;
309             funde.appendChild(badge);
310             for(var j = 0; j < ibArray.length; j++){
311                 for(var i = 0; i < ibJson.length; i++){
312                     if(ibArray[j] === ibJson[i].id && ibJson[i]["is_red_line"] === "true"){
313                         redLine = true;
314                         break;
315                     }
316                 }
317             }
318             if(redLine){
319                 banner.style.backgroundColor = "#FF3333";
320                 banner.style.color = "white";
321             }else{
322                 banner.style.backgroundColor = "#3BCCFF";
323                 banner.style.color = "white";
324             }
325
326             $(banner).popover({
327                 title: "Details",
328                 html: true,
329                 trigger: "click",
330                 content: function(){
331                     return createPopover(banner, ibArray);
332                 }
333             });
334         } else {
335             funde.innerText = "Keine Ergebnisse";
336             banner.style.backgroundColor = "#d1d1d1";
```

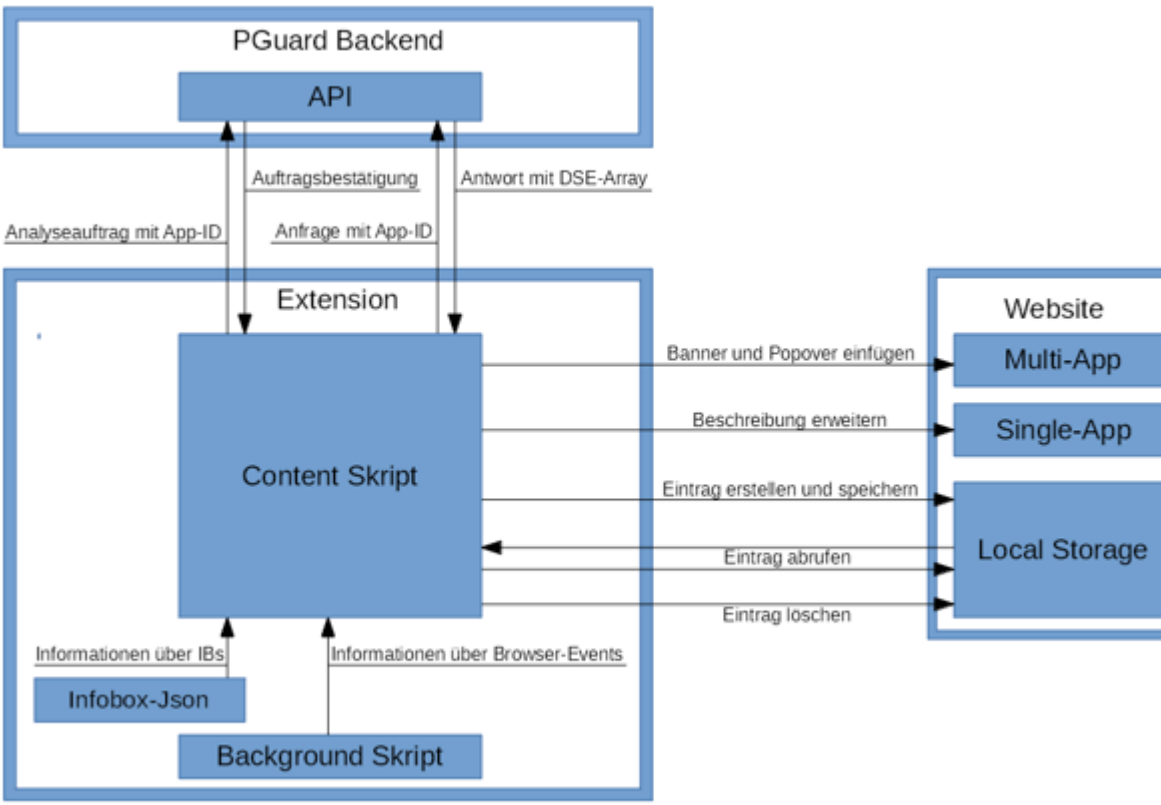
```
337         }
338
339         banner.appendChild(funde);
340         if($(parentNode).hasClass("card")){
341             banner.classList.add("multiapp");
342             parentNode.insertBefore(banner,parentNode.children[0]);
343         } else {
344             banner.classList.add("similarapp");
345             $(banner).insertBefore(parentNode);
346         }
347     }
348
349 }
350
351 function getNewestDseFromData(data){
352     var newestDse = null;
353     //Dummy-DSEs haben Vorang. Ansonsten wird die neuste DSE verwendet. Gibt es 2 DSE zum gleichen
        Analyse-Datum wird die aus der bevorzugten Quelle gewaehlt.
354     for (var i = 0; i < data.dses.length; i++) {
355         //Dummy
356         if(data.dses[i].origin === "dummy"){
357             newestDse = data.dses[i];
358             break;
359         }
360         else{
361             //Erste gefundene
362             if(!newestDse){
363                 newestDse = data.dses[i];
364             } else {
365                 //Vergleich mit vorher gefundener DSE
366                 var candidateTimeStamp = new Date(data.dses[i].date_infobox_calculation_finished);
367                 var dseTimeStamp = new Date(newestDse.date_infobox_calculation_finished);
368                 if(candidateTimeStamp.getTime() > dseTimeStamp.getTime()){
369                     newestDse = data.dses[i];
370                 } else if (candidateTimeStamp.getTime() === dseTimeStamp.getTime()){
371                     if(dseOrigins.indexOf(data.dses[i].origin) <
        dseOrigins.indexOf(newestDse.origin)){
372                         newestDse = data.dses[i];
373                     }
374                 }
375             }
376         }
377     }
378     return newestDse;
379 }
380
381 //Erstellt die Header fuer die Multiapp-Ansicht
382 function loadInfoPanels(parentNode, isSinglePage) {
383     var appID;
384     if(isSinglePage){
385         var currentURL = new URL(location.href);
386         appID = currentURL.searchParams.get("id");
387     } else {
388         if($(parentNode).attr("data-docid")){
389             appID = $(parentNode).attr("data-docid");
390             console.log("appID gefunden",appID);
391         }else {
392             appID =
        $(parentNode.children[0].children[0].children[3].children[0]).attr("href").split("id=")[1];
393         }
394     }
395     //Wurde eine App-ID gefunden?
396     if (appID) {
397         var appDataString = getStorageItem(appID);
398
399         var future = function(param) {
400
401             var appDataArray = [];
402             //Prueft ob bereits Daten im localStorage vorhanden und aktuell sind.
403             if(useLocalStorage && param && param.split(trennZeichen)[0]){
404                 var lastUpdate = new Date(param.split(trennZeichen)[0] * 86400000);
405                 if((lastUpdate.getTime() + 259200000) >= today.getTime()){
406                     appDataArray = param.split(trennZeichen);
407                 } else {
408                     console.log("Aktuallisiere Daten fuer " + appID);
409                 }
410             }
411             //Falls Daten vorhanden baue das Element darauss
412             if(useLocalStorage && appDataArray.length === 1){
413                 console.log("STORAGE OHNE ERGEBNISSE GEFUNDEN: ", appID, appDataArray);
414                 createPanel(parentNode, appDataArray, false , isSinglePage);
415             } else if(useLocalStorage && appDataArray.length > 1){
416                 console.log("STORAGE GEFUNDEN: ", appID, appDataArray);
417                 createPanel(parentNode, appDataArray, true , isSinglePage);
```

```
                //Ansonsten lade die Informationen aus dem Backend
            } else {
                console.log("Frage (erstmalig) Daten ab fuer " + appID);
                anfragencounter ++;
                $.ajax({
                    url: urlWholeDataSetNoRequest + "" +appID,
                    method: "POST",
                    success: function (response) {
                        var data = response.data;
                        console.log(data);
                        //Wurden bereits DSEs fuer die App gefunden?
                        if (data.dses && data.dses.length > 0) {
                            var storageString = castDseToStorageString(getNewestDseFromData(data));
                            setStorageItem(appID, storageString);
                            console.log("Neuer Eintrag angelegt: ", appID, storageString);
                            appDataArray = storageString.split(trennZeichen);
                            createPanel(parentNode, appDataArray, true, isSinglePage);
                        } else {
                            console.log("KEINE DSE VORHANDEN", appID);
                            setStorageItem(appID,""+ Math.floor(today.getTime() /86400000));
                            $.ajax({
                                url: urlTriggerNewAnalysis + appID,
                                method: "POST",
                                success: function(){
                                    console.log("Anfrage erfolgreich: ");
                                },
                                dataType: "json"
                            });
                            createPanel(parentNode, [], false, isSinglePage);
                        }
                    },
                    dataType: "json"
                });
                console.log("Anfragen bisher: ",anfragencounter);
            }
        };

        // Promise
        if (typeof appDataString === 'object') {

            appDataString.onsuccess = function() {
                if(appDataString.result){
                    future(appDataString.result.data);
                } else {
                    future();
                }
            };

        } else {
            future(appDataString);
        }
    }
}

function fillApps(){
    //Prueft, ob auf Single-App-Page oder Multi-App-Page
    if(document.getElementsByClassName("card")[0]){
        $(".card").each(function () {
            loadInfoPanels(this, false);
            //createPanel(this, [], false, false)
        });
    } else {
        if(document.getElementsByClassName("JHTxhe")[0]){
            loadInfoPanels(document.getElementsByClassName("JHTxhe")[0], true);
        }
        $(".Vpfmgd").each(function (){
            loadInfoPanels(this, false);
        });
    }
}

function testStorageCap(){
    localStorage.clear();
    localStorage.setItem("1","test");
    var counter = 2;
    while(localStorage.getItem("1")){
        console.log(counter);
        localStorage.setItem(""+counter,
  "ABCDEFGHIJABCDEFGHIJABCDEFGHIJABCDEFGHIJABCDEFGHIJABCDEFGHIJABCDEFGHIJABCDEFGHIJABCDEFGHIJABCDEFGHIJABCDEFGHIJABCDEFGHIJABCDEFGHIJ
        counter++;
    }
}
//Laedt lokale Json-Bibliothek fuer
$.getJSON(chrome.extension.getURL("lib/data/IB_texte.json"), function (input) {
```

```
501     ibJson = input;
502     $(ibCardTemplate).load(chrome.extension.getURL("lib/templates/ibCardTemplate.html"), function
          (data) {
503
          $(innerCollapseTemplate).load(chrome.extension.getURL("lib/templates/innerCollapseTemplate.html"),
          function(){
504             if((usedStorage === "localStorage" || usedStorage === "indexedDB") &&
        storageAvailable(usedStorage)){
505                 // for(var z = 0; z < localStorage.length; z++){
506                 //     console.log("Key " + z +" :",localStorage.key(z), " Value: ",
        localStorage.getItem(localStorage.key(z)));
507                 // }
508                 //testStorageCap();
509                 //console.log(localStorage.length);
510                 if(usedStorage === "indexedDB"){
511                     var openRequest = indexedDB.open("pguardExt",1);
512
513                     openRequest.onupgradeneeded = function(e){
514                         var thisDB = e.target.result;
515                         console.log("running onupgradeneeded");
516                         if(!thisDB.objectStoreNames.contains("apps")){
517                             var appsOS = thisDB.createObjectStore("apps",{keyPath: "appID"});
518                         }
519                     };
520
521                     openRequest.onsuccess = function (ev) {
522                         console.log("DB bereit");
523                         db = ev.target.result;
524                         fillApps();
525                     };
526
527                     openRequest.onerror = function (ev) {
528                         console.log("onerror!");
529                         console.dir(ev);
530                     }
531                 }
532             } else {
533                 console.log("kein Storage verfuegbar.");
534             }
535         });
536     });
537     console.log("Lokale Json geladen.");
538 });
```

Listing 3.3: Aufbau der pguard.js

3.2.5 Ergebnis



3.2.6 Diskussion

3.3 Aufgabe 2: Evaluierung von Caching Methoden einer Browser Extension

3.3.1 Anforderungen

Extension erweitert die Landingpage. Diese in Kategorien unterteilt. Kategorien werden bei jedem Besuch wieder aufgerufen SSpiele mit Vorregistrierungßtellt über längere Zeit gleiche Apps dar. New + Updated Gamesünd Top-Bewertung: Spielelllieferen jedes Mal ähnliche Ergebnisse =¿ überlappende Information Ëmpfehlungen für dichünd "Das könnte dir Gefallenpassen sich vorherigen Suchen an und liefern daher auch redudante Ergebnisse Selbe App oft mehrmals in der Übersicht vertreten Konklusion: viel Redundanz. Ergänzende Information werden mehrfach benötigt

Verarbeitet Diese nur zu benutzerfreundlichen Format Informationen müssen von

externen Quelle entnommen werden Dadurch entstehen Anfragen an einen Server mit Antworten Antworten und Anfragen durch // redundant.

Thesen: Nutzung der Extension nach "Veröffentlichung"würde hohen Traffic verursachen mit vielen wiederholten Anfragen. Anfragen könnten ab einer bestimmten Nutzerzahl Server überlasten Performance der Extension leidet unter dieser Art der Infortionsbeschaffung Bei Ausfall der Quelle, bietet Extension keinen Mehrwert für den Nutzer mehr.

Lösung: Einrichtung von unabhängigen Speicher zur Aufbewahrung der gewonnen Informationen. Insbesondere viele/wiederholt genutzt Informationen sollen ohne erneute Anfrage zur Verfügung stehen. Neue Anfragen nur bei Veraltung der Informationen bzw. nur von neu aufgetauchten Apps Aufbau einer Struktur zur Abspeicherung der wichtigen Informationen

Verwendung des Speichers:

Welche Informationen stehen pro App zur Verfügung?(1) Welche Informationen werden pro App benötigt? (2) Wie wird der Speicher gepflegt?(3)

(1)

Wird bei Aufbau der App schon Beschrieben? Aynchron!

(2) Alter der Information: Ist die Information auf dem Aktuellen Stand wie die der Quelle? Ist die Information der Quelle veraltet? Wie oft wird so eine Information erneuert? (Neue DSE o.ä) =¿ Abspeichern des Analysedatums. Regelmäßige Überprüfungen(3 Tage), ob neue Information bei der Quelle vorhanden

Aufruf der App: Wie oft wird diese App aufgerufen? Informationen über die App im Speicher können nach gewisser Zeit gelöscht werden, wenn sie nicht erneut aufgerufen wurde. =¿ Frequency Count: Zähler im Speicher der bei jedem Aufruf erhöht wird. Regelmäßig wird der Speicher nach niedrigen Zählern durchsucht und diese Einträge gelöscht.

Informationen zur App: Inhalt entspricht Kennnummern der Infoboxen. Also Abspeichern der jeweilig vorhandenen Eigenschaft/Infobox

(3) LRU Konzept oder Fehler falls Speicher voll. Aufbau des Speichers von Speichermethode abhängig (Chrome =¿ key, value String) Möglichst kurze Zusammenfassung der benötigten Informationen: Alter der Information als Tageszähler seit festem Datum, da Tagesvergleich stattfindet Frequency Counter als einzelner Integer (ÜB: casten auf einstellige Zahl ProtectFaktor?) Abfolge von Kennnummer mit Infoboxen Trennzeichen zum korrekten Auslesen der Informationen. ÄLTERTAGE(TRENNZEICHEN)FREQ(TRENNZEICHEN)IB(TRENNZEICHEN)IB ..."Bsp: "17500—3—1—2—3—4"

FREQ FÄLLT WEG???

Worst Case: 99999—1—2—3—4—5—6—7—8—9—10—11—12—13—14—15—16—17—18—19—20—21—22—23—2
86 Character 86Bytes

Limit = 5MB = 5242880 Bytes

5242880 / 86 = 60963 Einträge

Appanfragen pro Initialladen 70

Nach Antwort von Server werden Informationen im Speicher abgelegt mit Startcounter. Bei jedem erneuten Abrufen wird der Speicher abgefragt. Falls Information vorhanden und aktuell wird Counter angepasst und Informationen lokal ausgelesen. Ansonsten neue Anfrage an Server. Ist Antwort mit neuer Information vorhanden wird die alte gelöscht und durch die neue ersetzt.

Extension prüft in regelmäßigen Abständen den Counter-Status. Ist dieser zu niedrig wird die Information aus dem Speicher entfernt.

Speicher wird auf Maximalkapazität geprüft und eine Approximation vom FFüllstand"wird errechnet. Ist der Speicher voll, werden Einträge mit dem niedrigsten Counter gelöscht.

3.3.2 Vorauswahl

Welche Arten von Speicher stehen einer Extension für Browser (hier Chrome) zur Verfügung?

Integriert: Session Storage und lokal Storage Session Storage: Daten bleiben nur während der Sitzung erhalten. Das Schließen des Browsers bzw. das Öffnen der Website in einem anderen Tab/Browserfenster beendet die Sitzung

Local Storage: Daten bleiben über eine Sitzung hinaus erhalten und verfallen erst durch Überschreiben oder Löschen

Kapazität: 5MB

Aufbau: String-Tupel nach dem (Key, Value)-Prinzip

Zugriff: getItem(key), setItem(key,value) und removeItem(key)

Serverseitiger Speicher: Identifizierung notwendig. Aufwendig in Pflege und Wartung. Kein Mehrwert zu Anfragen an Informationsquelle

Serverseitiger Speicher fällt vor vorne herein weg aus oben genannten Grund und datenschutzrechtlichen Bedenken.

Datenbanken: Verfügbar: IndexedDB und WebSQL

WebSQL seit November 2010 von W3C nicht mehr empfohlen (veraltet).

IndexedDB: API in allen modernen Browser zur Speicherung von Daten und Dateien in einer object-orientierten Datenbank. synchron und asynchron möglich. Funk-

tioniert nach key, value prinzip Alle Datentypen von JavaScript werden unterstützt. Kann indexiert werden um Suchen effizient zu machen. Verwendet Prinzip von Transaktionen Anfragen mit Rückgabewerten als Basis aller Operationen Verfolgt den NoSQL-Ansatz

Speicherlimit nach global Limit (1/2 Festplatte) und Gruppenlimit (1/5 von global Limit, min 10MB max. 2GB) Gruppenlimit voll = voll (Fehler) Global Limit voll = löschen bis wieder frei (Quellenabhängig komplette Elemente gelöscht)

Warum nicht IndexedDB?

Vorteile von IndexedDB: Abspeicherung von großen strukturierten Datenmengen. Nachteile: hoher Aufwand bei Implementierung.Overhead lohnt nicht bei kleinen Datenmengen. Transaktionen blockieren bei Fehlern eventuell den Datenabruf bzw. die Aktualisierung

Storage API von Chrome ausreichend Speicher und geringer aufwand bei der Implementierung. Lediglich Strings benötigt. Indices bei gewählen value-Struktur nicht notwendig.

Vorteil von Session Storage: Speicherpflege nicht notwendig, da 5MB groß genug für Anzahl(?) an App-Informationen während einer Session im PlayStore. Informationen immer auf Stand der Quelle Nachteil: Bei erstmaligen Öffnen des Stores in neuer Browsersession werden viele Anfragen losgeschickt für Apps die bereits in der letzten Session schon angefragt wurden. Bei Serverausfällen fehlen die Informationen Lediglich in einer Session mehrfach aufgerufene Apps ersparen erneute Anfragen. =¿ Speicherpflege fällt weg, dafür kaum Mehrwert bei Anfragen.

Vorteil von Lokal Storage: Apps werden einmal abgefragt und sind anschließend abgespeichert. Fällt der Server aus können die lokalen Informationen genutzt werden. Daten auch aus letzter Session bleiben vorhanden. Neue Anfragen werden nur dann geschickt wenn aktuelle Daten über 3 Tage alt sind. Nachteile: Speicherpflege notwendig. Dadurch wird die Information länger (Counter und Tag). Zusätzliche Rechenzeit für das Löschen von alten Informationen notwendig. Dadurch wird sichergestellt dass die 5MB nicht überschritten werden und somit Informationen ungewollt verloren gehen. Für Informationen mit hohem Counter muss regelmäßig überprüft werden, ob die Information noch aktuell ist, weil diese in der Regel lange im Speicher verweilt. =¿ Hohe Einsparung bei Anfragen an den Server möglich. Dafür müssen zusätzliche Operationen zur Speicherpflege und Prüfung der Informationen ausgeführt werden.

3.3.3 Rahmenbedingungen

Plattform: Windows 10 Rechner Build, Specs Chrome Details App Details Was wird gemessen? Limitierungen

Getestet auf:

Windows 10 Education 64 Bit Build 10.0.17134 Prozessor i7-6700K RAM: 16GB GPU: Nvidia GTX 1070

Chrome 67.0.3396.99 64 Bit

3.3.4 Vorgehensweise

3.3.5 Ergebnisse

Storage: none Ladezeit: 1435ms Start der Extension-Funktionsaufrufe: 944ms Dauer: 491ms Anzahl der Anfragen an das Backend: 0

Storage: Local Storage Ladezeit: 1711ms Start der Extension-Funktionsaufrufe: 892ms Dauer: 819ms Anzahl der Anfragen an das Backend: 125

Storage: none Ladezeit: 1761ms Start der Extension-Funktionsaufrufe: 883ms Dauer: 878ms Anzahl der Anfragen an das Backend: 131

3.3.6 Diskussion

Kapitel 4

Abschließende Diskussion

4.1 Konklusion

4.2 Fortsetzung der Forschung

Kapitel 5

Appendix

5.1 Derivations

5.1.1 Example Matlab Code

Add a sourcefile directly into LaTeX

```
% simple Kalman Filter example:
% state "x" consists of position and velocity
% system model "F" is a cinematic model of constant velocity
% only the position is measured

clear     % clear all matlab variables

%%% declare matlab variables and asign default (randomly chosen) value
% simulation specifications
T = 1;                % make a measurement every T steps. also called \Delta t
                      % i.e. every 1, 2, 3, ... seconds
real_x = [0; 10];     % "real world" state, only needed in simulation context
                      % also called ground truth. start: position=0, velocity=10

% model specifications
model_F = [1, T ;     % the model we have about the real world
           0, 1];     % here: cinematic model of constant velocity
q = 9;                % controls the amount of process noise. is usually unknown
model_Q = [T^4/4, T^3/2;       % process noise
           T^3/2, T^2 ]*q;     % arises from the cinematic model

% estimations specifications
esti_x = [0; 10];     % estimated state: position and velocity
esti_P = [1, 0 ;      % estimated covariance of esti_x. refelcts
          0, 2];      % the uncertainty about the estimated state esti_x

esti_z = 0;       % estimated measured value. here: just depicting position-entry
                  % of esti_x since we are only iterested in the position. Or
                  % maybe it is only possible to measure position, but not velocity
esti_S = [0,0 ; % estimated covariance of esti_z. Will consist of process noise
          0,0]; % with added measurement noise

H = [1, 0;        % observation matrix. we only measure position values
     0, 0];       % this row could be left out, but then also modify R to 1x1
R = [1, 0 ;       % measurement noise. is usually unknown. reflects the
     0, 1];       % inaccuracy of the sensors
K = [0, 0];       % Kalman gain vector


%%% Initialization
esti_x = [0; 10];
esti_P = [1, 0 ;
          0, 2];

for step = 1:1000          % simulate for 1000 steps (simulate continuous time)
  if mod(step, T) == 0     % if it is time to take a new measurement
    % update the "real data". For simplicity: take the model F. But could be any
    % other function, possibly non-linear.
    % mvnrnd = multi variate normal random numbers
    real_x = model_F * real_x + transpose(mvnrnd([0,0], model_Q));

    %%% Step 1: Prediction Step
    esti_x = model_F * esti_x;  % estimate the new state according to the
                                % system model since we do not have any
                                % control inputs, this term is left out
    esti_P = model_F * esti_P * transpose(model_F) + model_Q;  % update the
                                % covariance of estimated state esti_x
    esti_z = H * esti_x;        % depic position value from estimated state
    esti_S = H * esti_P * transpose(H) + R;  % estimation of the covariance of
                                % the estimated measured value. inherits model
                                % noise and measurement noise

    %%% make a measurement z
    z = H * real_x + transpose(mvnrnd([0,0], R)); % make a noisy measurement

    % Step 2: Innovation Step
    K = esti_P * transpose(H) * esti_S^-1;  % calculate Kalman gain vector by
                                            % comparing model and measurement
                                            % noise
    esti_x = esti_x + K * (z - esti_z);     % update the estimated state by an
                                            % weighted sum of the measurement
                                            % and the model-estimation
    esti_P = esti_P - K * esti_S * transpose(K);  % update covariance of
                                                  % estimated state
  end
end
```

Listing 5.1: Simple example of a Kalman Filter in Matlab

Acknowledgement

First of all, I would like to express my gratitude to … for the aspiring guidance, useful comments and invaluably support throughout the whole process of this Master Thesis. Furthermore, I would like to thanks …, … and the other members of the research team for helpful discussions and constructive criticism. In addition, I would like to thank my University supervisor … for all the helpful remarks, advises and discussions.

Also, I would like to thank my parents and … who have supported me throughout the entire process by keeping me harmonious and motivated.

Last but not least, I like to thank … for funding my research and providing me with the facilities being required.

Abbildungsverzeichnis

Literaturverzeichnis

Proclamation

Hereby I confirm that I wrote this thesis independently and that I have not made use of any other resources or means than those indicated.

Forname Surname, Place, 5. März 2019