

## **Projektarbeit**

von

Marian Phil Mutschler

Datagroup Ulm GmbH

für die

Robert-Bosch-Schule Ulm

Thema

**„RSS Feed Reader“**

*GIT HUB:*

*[https://github.com/ElevenSpins/RSS\\_Feed\\_Reader](https://github.com/ElevenSpins/RSS_Feed_Reader)*

# Inhaltsverzeichnis

<b>Eigenständigkeitserklärung .....</b>	<b>3</b>
<b>Projektidee / - ziel .....</b>	<b>4</b>
<i>Idee .....</i>	<i>4</i>
<b>Anforderungsanalyse.....</b>	<b>4</b>
<i>Grundfunktionalität .....</i>	<i>4</i>
<i>Wünschenswerte Features .....</i>	<i>4</i>
<b>Projektverlauf .....</b>	<b>5</b>
<i>Timeline .....</i>	<i>5</i>
<b>Dokumentation .....</b>	<b>6</b>
<i>Implementierung.....</i>	<i>6</i>
<i>Klassendiagramm .....</i>	<i>6</i>
<i>Relevante Klassen.....</i>	<i>7</i>
<i>Versionsverwaltung .....</i>	<i>7</i>
<b>Bedienung .....</b>	<b>8</b>
<i>Grundfunktion .....</i>	<i>8</i>
<i>Übersicht.....</i>	<i>8</i>
<b>Fazit.....</b>	<b>10</b>
<i>SOLL-IST Vergleich.....</i>	<i>10</i>
<i>Ausblick .....</i>	<i>10</i>
<b>Quellen.....</b>	<b>10</b>

# Eigenständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Projektarbeit selbständig verfasst und keine anderen als die angegebenen Hilfsmittel benutzt habe.

Die Stellen der Hausarbeit, die anderen Quellen im Wortlaut oder dem Sinn nach entnommen wurden, sind durch Angaben der Herkunft kenntlich gemacht. Dies gilt auch für Zeichnungen, Skizzen, bildliche Darstellungen sowie für Quellen aus dem Internet.

\_\_\_\_\_  
Unterschrift Ort, Datum

# Projektidee / - ziel

## Idee

Die Idee zum Projekt entstand beim Lesen meiner Lieblingsplattform „Heise.de“.

Neben dem normalen „heise+“ Abo habe ich auch sämtliche RSS Feeds abonniert.

Dies ist auf Dauer im Browser etwas kompliziert zu managen. Deshalb dachte ich an eine Art „Beitragsmanager“, dabei soll es mir möglich sein eine Liste meiner Lieblings Feeds anzulegen und Beiträge welche mir gefallen zu archivieren.

Das Ziel ist es nun ein Tool zu entwickeln, welches die Beiträge mittels Grafischer Oberfläche anzeigt und die Möglichkeit bietet diese zu archivieren.

# Anforderungsanalyse

## Grundfunktionalität

- Eingabemöglichkeit für die „Feed Url“
- Anlegen einer Liste mit den besten Url's
- Anzeigen der aktuellen Topics
- Anzeigen des Inhalts der jeweiligen Topics

## Wünschenswerte Features

- Möglichkeit zum Drucken der Lieblingsbeiträge
- Möglichkeit zum Bewerten der Lieblingstopics
- Möglichkeit deine Lieblingsbeiträge mit Tags zu versehen
- Beiträge nach Tags filtern
- Möglichkeit zur Archivierung der Lieblingstopics

# Projektverlauf

## Timeline

### **02. JUNI 2020**

- Vorstellung der Projektidee anhand eines Pflichtenheftes
- Informationssammlung zu Bibliotheken / Umsetzungsmöglichkeiten

### **10. JUNI 2020**

- Klassenstruktur planen und Erstellen des Grundgerüsts des Projektes
- Beginn mit Erstellung der GUI („MainWindow“)
- Import des der Bibliothek „FeedReader“ und Einbindung in die Anwendung

### **24. JUNI 2020**

- Implementierung der Klasse „FeedContent“
- Erstellen der Events in der GUI
- Erstellen der Methoden

### **08. JULI 2020**

- Dokumentation erstellen
- GitHub Repo erstellt
- Initial Commit
- Erstellen der Readme.md

### **10. JULI 2020**

- Projekt überarbeiten für Abgabe vorbereiten
- Letzter Commit in den Master

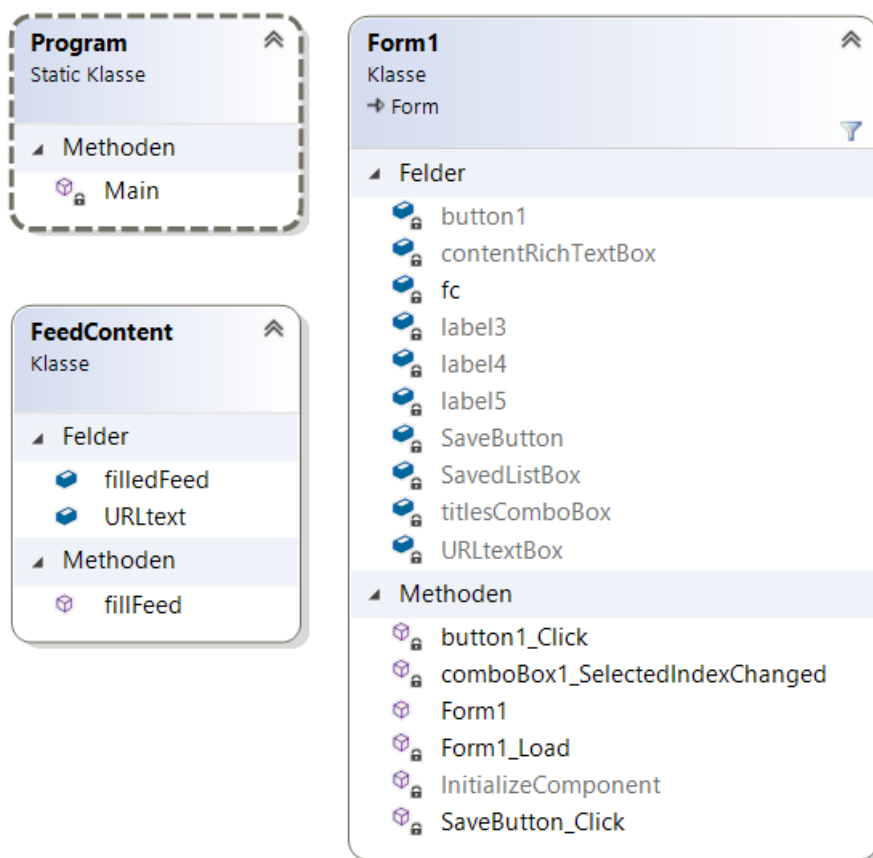
# Dokumentation

## Implementierung

Die Implementierung habe ich mit dem Designen und Erstellen meines Hauptfensters begonnen, so habe ich ein Gefühl bekommen wie die Ausgangssituation aussehen soll und wusste welche Events & Methoden ich benötige, um dies umzusetzen. Daraufhin folgte die Implementierung des der Bibliothek „FeedReader“ und weiter die Implementierung der Logik, um die Funktionalität der Anwendung bereitstellen zu können. Die Klassenbibliothek wurde über den NuGet Packet Manager installiert. Zuletzt kann dann die Verknüpfung der GUI und der Logik anhand von Eventhandlern.

## Klassendiagramm

Das Programm setzt sich aus einer Klasse zusammen, welche die Logik bildet und einem Fenster für die Oberfläche.



Erstellt mit VisualStudio 2019 – Klassen Designer

## Relevante Klassen

### „FeedContent“

Besitzt die Methode “fillFeed” welche mit der Klassenbibliothek „FeedReader“ die Informationen der XML Datei zieht.

### „Form1“

Hier ist das Design / Events integriert

Die kompletten Funktionen werden auf der GUI reflektiert.

## Versionsverwaltung

GitHub ist ein netzbasierter Dienst zur Versionsverwaltung für Software-Entwicklungsprojekte. Dieser Dienst wurde genutzt um das Projekt sauber zu Verwalten und um einen Verlust des Projekts vorzubeugen. Ebenfalls kann somit jederzeit einfach.

Gefunden wird das Projekt unter: [https://github.com/ElevenSpins/RSS\\_Feed\\_Reader](https://github.com/ElevenSpins/RSS_Feed_Reader)

# Bedienung

## Grundfunktion

Die Grundfunktion besteht darin eine URL eines RSS-Feeds in die oberste TextBox einzufügen. Durch anschließendes drücken des Buttons „Laden“ werden durch die Klasse „fillFeed“ die Daten des Feeds geladen und in der ComboBox aufgelistet. Anschließend kann der Beitrag gelesen werden & die Feed URL in der ListBox abgelegt werden.

Mögliche RSS-Feeds (jeder Feed möglich – nur Beispiele):

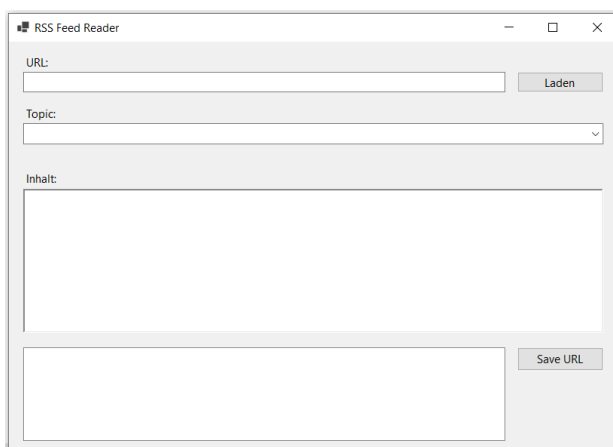
- *Alle News von heise online*  
[www.heise.de/rss/heise-atom.xml](http://www.heise.de/rss/heise-atom.xml)  
[www.heise.de/rss/heise.rdf](http://www.heise.de/rss/heise.rdf)
- *Top-News von heise online*  
[www.heise.de/rss/heise-top-atom.xml](http://www.heise.de/rss/heise-top-atom.xml)
- *Netzwerk-Tools – iMonitor – Internet-Störungen*  
[www.heise.de/netze/netzwerk-tools/imonitor-internet-stoerungen/feed/aktuelle-meldungen/](http://www.heise.de/netze/netzwerk-tools/imonitor-internet-stoerungen/feed/aktuelle-meldungen/)

## Übersicht

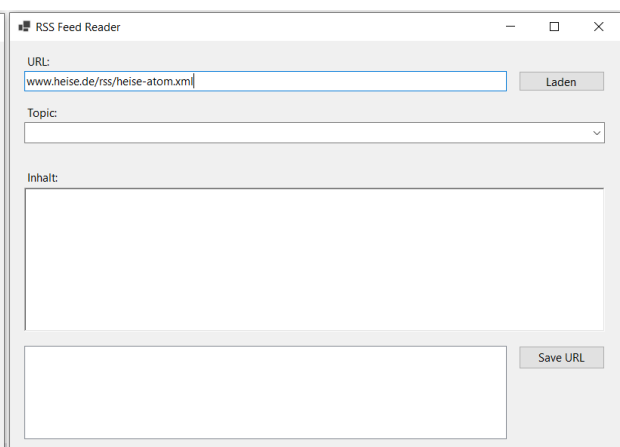
UI-Elemente: URL-TextBox; Topic-ComboBox; Inhalt-RichTextBox; URL-ListBox; 2x Button

Oben genannte UI- Elemente wurden verwendet um dem User eine möglichst einfache UI zu bieten um somit das Usability der Anwendung zu erhöhen.

1 . Leer Ansicht bei Start des Programms

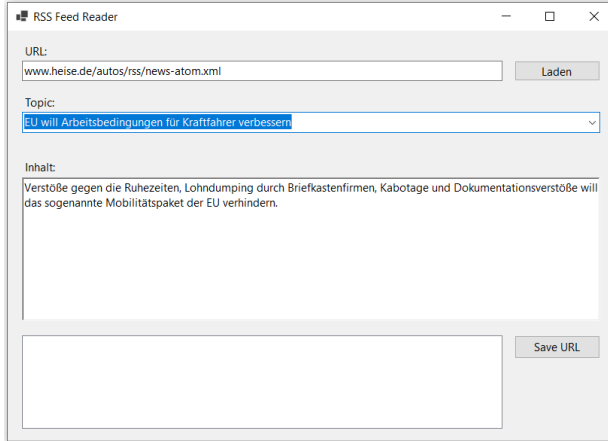


2 . Ansicht nach einfügen der URL



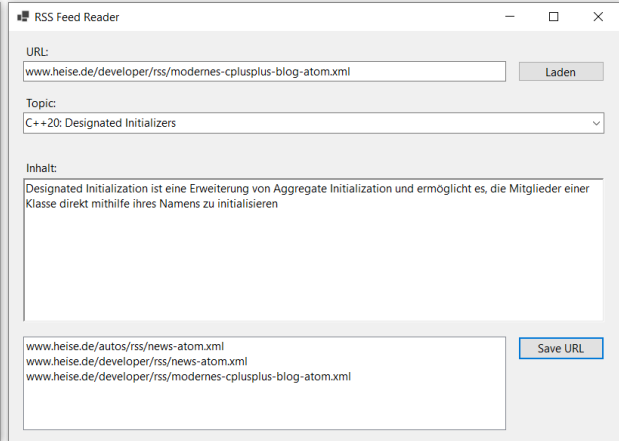


### 3 . Ansicht nach Auswahl des Topics



The screenshot shows the 'RSS Feed Reader' application window. The 'URL' field contains 'www.heise.de/autos/rss/news-atom.xml' and the 'Laden' button is visible. The 'Topic' dropdown menu is open, showing the selected topic 'EU will Arbeitsbedingungen für Kraftfahrer verbessern'. The 'Inhalt' text area displays the corresponding article text: 'Verstöße gegen die Ruhezeiten, Lohndumping durch Briefkastenfirmen, Kabotage und Dokumentationsverstöße will das sogenannte Mobilitätspaket der EU verhindern.' At the bottom, there is an empty text box and a 'Save URL' button.

### 4. Speichern der URL in der ListBox



The screenshot shows the 'RSS Feed Reader' application window. The 'URL' field contains 'www.heise.de/developer/rss/modernes-cplusplus-blog-atom.xml' and the 'Laden' button is visible. The 'Topic' dropdown menu is open, showing the selected topic 'C++ 20: Designated Initializers'. The 'Inhalt' text area displays the corresponding article text: 'Designated Initialization ist eine Erweiterung von Aggregate Initialization und ermöglicht es, die Mitglieder einer Klasse direkt mithilfe ihres Namens zu initialisieren.' At the bottom, the 'Save URL' button is highlighted in blue, and the URL 'www.heise.de/developer/rss/modernes-cplusplus-blog-atom.xml' is visible in the text box below it.

# Fazit

Im Großen und Ganzen ist das Projekt gut verlaufen. Natürlich gab es ein paar kleinere Probleme, diese konnten aber behoben werden.

Mit dem Ergebnis bin ich auch zufrieden in Angesicht der Investierten Zeit. Das Programm erfüllt seinen Zweck und hilft den Benutzer dabei seine Lieblings Feeds zu lesen.

## SOLL-IST Vergleich

Alle Musskriterien des Pflichtenhefts wurden erfolgreich in die Anwendung implementiert (siehe Pflichtenheft).

Die Planung in Anbetracht der Grundfunktionen hat daher gut funktioniert.

Möglichkeit zum wiederverwenden der gespeicherten URL in der ListBox.

## Ausblick

Natürlich kann das Projekt auch noch erweitert werden.

Hierzu kann man dem Pflichtenheft entnehmen welche Features möglich wären, um dem Nutzer ein noch besseres Erlebnis bei der Verwendung der Anwendung zu bieten.

Ebenfalls könnte man andere User befragen was diese sich von einem RSS Feed Reader wünschen würden, um somit den vollen Funktionsumfang der Anwendung zu erreichen.

# Quellen

Bibliotheken:

<https://github.com/codehollow/FeedReader>

Programme/Dienste:

GitHub, VisualStudio 2019, IntelliJ Rider, Word