Ostfalia Hochschule

EKdI-Projekt

Pflichtenheft

Martin Krause 70478294, Tom Strunz 70476813, Patricia Weber 70477439, Kristin Altmann 70476503, Marcel Pulst 70477125

supervised by
Prof. Jensen

19. Januar 2022

Inhaltsverzeichnis

1 Zweck und Ziel

Es soll ein Programm in Java ¿= 11, mit Open JDK, entwickelt werden, welches mit dem Aufrufen des „Lies.java“Java-Programms den Inhalt einer beliebigen Webseite vorlesen soll. Dabei muss beachtet werden das die nicht dargestellten Bestandteile, zum Beispiel Tags und Metadaten, nicht vorgelesen werden sollen. Dieses Programm soll entwickelt werden um zum Beispiel Personen mit Sehbehinderung Webseiten hinter URLs inhaltlich zeigen zu können. Dieses Pflichtenheft basiert auf dem dazugehörigem Lastenheft und beinhaltet alle Zwecke und Ziele des Lastenhefts außer sie werden ausgeschlossen.

2 Abgrenzung

Das Programm soll kein Graphic User Interface (GUI) enthalten und dem Programm wird es ebenfalls fast nur möglich sein Deutsche Webseiten vorzulesen, aufgrund von den auf Deutsch eingelesenen Silben. Das Proramm wird unter Windows geschrieben und getestet.

3 Begriffe

- 'URL': Adressierung der vorzulesenden Website
- 'vorlesen': Abspielen von wav-Dateien, die den Worten des Texts entsprechen
- 'Text': der Haupttext auf der Website
- 'Start': Beginn/ Fortsetzen des Vorlesens
- 'Stopp': Unterbrechen des Vorlesens
- 'parsen': Zerlegen und umwandeln für eine Weiterverarbeitung
- 'Silbe': eine Buchstabenkombination einer Silbe im grammatikalischen Sinne zuweisbar ist, gilt dies als Silbe. Da nicht alle Silben der deutschen Sprache eingesprochen werden können, werden Buchstaben, die nicht einer Silbe zugeordnet werden können, als Silbe aus nur einem Buchstaben betrachtet.

4 Soll-Stand

Was soll erreicht werden? Das Programm soll einen Text von einer Website, deren URL dem Programm übergeben wurde, einlesen und dann vorlesen. Wie soll das erreicht werden? Das Programm besteht im Wesentlichen aus zwei Teilen: dem Einlesen und dem Vorlesen. Beim Einlesen werden zuerst Daten von der Website abgegriffen und der HTML-Code geparst. Für das Vorlesen müssen Start, Stopp, das Trennen des Texts in

Silben und das Wandeln der Silben in gesprochene Sprache. Die Silben müssen dementsprechend noch eingesprochen werden. Was kann noch erreicht werden? Es könnte eine Warnungen ausgegeben werden, wenn der Text wahrscheinlich nicht auf Deutsch ist, zu lang ist oder die wav-Dateien, die zum Vorlesen nötig sind, nicht gefunden werden. Außerdem könnte der vorgelesene Text ausgegeben werden.

4.1 Funktionen

LF1: Beim Programmstart, wird eine URL übergeben.
– java Lies www.ostfalia.de

```
public static void main (String[] args) {
        // Prüft, ob eine URL am Anfang übergeben wurde
        try {
                String url = "https://" + args[0];
                lies(url);
        } catch (Exception e) {
                System.out.println("Es muss eine URL übergeben werden.");
                System.exit(0);
        }
}
```

Abbildung 1: LF1

LF2: Das Vorlesen kann gestartet werden und beliebig pausiert und wieder weitergespielt werden. Außerdem kann das Programm beliebig beendet werden.

```
public static void startProcess(String url) {
        // Filtert den Text der website und spielt ihn ab. Man hat die Möglichkeit zu pausieren,
        // wieder zu starten, oder das Programm zu beenden.
        String opt = "";
        String pagetext = WebpageReader.getTextFromURL(url);
        String[] sylList = sd.getSyllables(pagetext);

        AudioPlayer.syls = new String[sylList.length];
        for (int i = 0; i < sylList.length; i++) {
                AudioPlayer.syls[i] = sylList[i];
        }

        // Startet einen separaten eigenen Thread, der sich um das vorlesen kümmert.
        // Dadurch blockiert die Eingabe für Pause und exit nicht das Vorlesen.
        ReadLoop rl = new ReadLoop();

        boolean loop = true;
        boolean shouldStop = true;
        while (loop) {
                opt = sc.nextLine();
                switch (opt) {
                case "s":
                        if (shouldStop) {
                                AudioPlayer.play = false;
                                shouldStop = !shouldStop;
                        } else if (!shouldStop) {
                                AudioPlayer.play = true;
                                shouldStop = !shouldStop;
                        }
                        break;
                case "e":
                        System.exit(0);
                }
        }
}
```

Abbildung 2: LF2

LF3: Der Text aus einer Webseite wird herauskopiert.

```
public static String getTextFromURL(String url) {
        String text = "";
        try {
                //Instantiating the URL class
              URL url_ = new URL(url);
              //Retrieving the contents of the specified page
              Scanner sc = new Scanner(url_.openStream());
              //Instantiating the StringBuffer class to hold the result
              StringBuffer sb = new StringBuffer();
              while(sc.hasNext()) {
                 sb.append(sc.next());
                 //System.out.println(sc.next());
              }
              //Retrieving the String from the String Buffer object
              text = sb.toString();
              //Removing the HTML tags
              text = text.replaceAll("(?s)<[^>]*>(\\s*<[^>]*>)*", " ");
        }
        catch (Exception e) {

        }
        return text;
}
```

Abbildung 3: LF3

LF4: Der Text wird geparst und die Steuerzeichen werden heraus gefiltert.

```
//Removing the HTML tags
text = text.replaceAll("(?s)<[^>]*>(\\s*<[^>]*>)*", " ");
```

Abbildung 4: LF4

LF5: Der Text wird in Silben bzw. einzelne Buchstaben aufgeteilt.

```
public String[] getSyllables(String s) {
    s = s.toLowerCase();
    s = adjustString(s);
    while (0 < s.length()) {
        if(root.isChild(s.charAt(0))) {
                syllables.add(findSyllables(root.getChild(s.charAt(0)), s, "" + s.charAt(0)));
                s = s.substring(syllables.get(syllables.size() - 1).length());
        }else {
            syllables.add(""  + s.charAt(0));
            s = s.substring(1);
        }
    }
    spaces();
    return syllables.toArray(new String[0]);
}
```

Abbildung 5: LF5

LF6: Die Silben werden nacheinander abgespielt.

```
public static void read() {
        if (play) {
                if (counter < syls.length) {
                        if (syls[counter].equals("LEER")) {
                                // make nothing
                        } else {
                                AudioPlayer aupl = new AudioPlayer("..//res/" + syls[counter] + ".wav");
                                aupl.play();
                        }
                        counter++;
                } else {
                        System.exit(0);
                }
        }
}
```

Abbildung 6: LF6

LF7: Der Thread kümmert sich in einer Endlosschleife um das Vorlesen nach einer bestimmten kurzen Pause.

```
public ReadLoop() {
        this.start();
}

private synchronized void start() {
        if (running) {
                return;
        }
        running = true;
        thread = new Thread(this);
        thread.start();
}

public synchronized void stop() {
        if (!running) {
                return;
        }
        running = false;
        try {
                thread.join();
        } catch (InterruptedException e) {
                e.printStackTrace();
        }
}

@Override
public void run() {
        while(running) {
                long now = System.currentTimeMillis();
                delta += (now - lastTime);
                lastTime = now;
                if (delta >= timeDistance) {
                        AudioPlayer.read();
                        delta = 0;
                }
        }

}
```

Abbildung 7: LF7

4.2 Daten

- LD1: Die ersten Daten mit dem das Programm konfrontiert wird ist die übergebene URL.
- LD2: Mit Tastatureingabe „l“beginnt das Programm mit dem Vorlesen.
- LD3: Mit Tastatureingabe „s“kann das Vorlesen gestoppt und wieder weitergespielt werden.
- LD4: Mit Tastatureingabe „e“kann das Programm beendet werden.
- PD1: Die URL wird als String gespeichert.
- PD2: Der Text der Webseite wird als String gespeichert.
- PD5: Die Silben werden als Strings in einem Array gespeichert.
- PD6: Alle Silben sind als wav-Dateien dem Programm beigelegt, um abgespielt werden zu können.

4.3 Regeln

- PR1: Das Programm soll mit dem Aufruf java Lies „URL“den Inhalt einer beliebigen URL vorlesen. Falls keine URL übergeben wurde, wird eine Exception geworfen und der Nutzer wird darüber informiert.
- LR2: Andere Libraries als die der Oracle JDK oder OpenJDK dürfen nicht verwendet werden.
- LR3: Der Aufruf externer Services oder Programme zum Vorlesen ist nicht erlaubt.
- LR4: Nicht dargestellte Bestandteile einer Website, d.h. Tags oder Metadaten, dürfen nicht vorgelesen werden.
- PR2: Falls eine nicht bekannte Webseite aufgerufen werden soll, wird eine Exception geworfen und der Nutzer wird darüber informiert.
- PR3: Falls eine nicht vorhandene Ton-Datei abgespielt werden soll, wird eine Exception geworfen und der Nutzer wird darüber informiert.

4.4 Nichtfunktionale Anforderungen

- NA1: Das Programm soll von Personen mit Sehbehinderung bedient werden können.

5 Dokumentenhistorie

1. — Martin Krause — Ersterstellung — 06.12.21
2. — Kristin Altmann — Bearbeitung Punkt 1,2 — 06.12.21
3. — Tom Strunz — Begriffe, Soll-Stand überarbeitet — 13.12.21
4. — Patricia Weber — Regeln, Nichfunktionale Anforderungen — 13.12.21
5. — Martin Krause — Deckblatt, Inhaltsverzeichnis, Funktionen bearbeitet — 13.12.21
6. — Martin Krause — Funktionen grafisch eingefügt, Deckblatt bearbeitet — 19.01.22