

Dyslexo

A simple Text To Speech launcher for dyslexic people



Stefano Castiglia

I.I.S. Benedetto Castelli, Brescia

5^a AI

A.S. 2016 - 2017

Premessa	4
Introduzione	5
Descrizione	5
Istruzioni	6
Requisiti	6
Installazione	6
Abilitare le funzioni	8
Play, stop	8
Pausa, play	8
Salvataggio su file	8
Tecnologie impiegate	9
Java	9
Maven	9
Git	10
Principali librerie impiegate	11
MaryTTS	11
JNativeHook	11
Optimaize Language Detector	11
Apache Commons	11
Classi principali	13
Settings	13
Translator	13
Classi principali - Gestione delle chiamate dell'utente	13
GlobalKeyListener	13
Classi principali - Gestione dell'audio	13
TextProcessor	13
Player	15
Converter	15
Sviluppi futuri	16
Commenti nel codice e JavaDoc	16
Aggiornamenti	16
Rilascio repository pubblica su GitHub	16
Interfaccia impostazioni	16
Miglioramento lettura	17
Sito web	17
Conclusioni	18

Premessa

Questo progetto nasce dalla mia passione per l'informatica e dalla metodologia della programmazione impartitami in questi anni dal professor Bugatti Alessandro, il quale mi ha portato ad apprezzare anche questo campo, ma soprattutto dal bisogno di trovare un'alternativa software alla sintesi vocale, di cui io faccio molto uso in quanto dislessico.

Intendendo per sintesi vocale la tecnica per la riproduzione della voce umana attraverso un computer tramite hardware o software, il calcolatore produce in autonomia segnali uditivi, utili per l'interazione tra macchina e persona. Gli utenti della sintesi sono principalmente ciechi, ipovedenti e dislessici; ultimamente essa si sta diffondendo anche tra gli assistenti vocali, con i quali si è in grado di interagire tramite il riconoscimento vocale, che non deve essere confuso con la sintesi, in quanto nel secondo caso è il computer che riconosce ed elabora la voce umana.

Il mio progetto non vuole fermarsi a questo esame di maturità, ma continuare ad essere migliorato nel tempo e ad offrire un'alternativa opensource fruibile dalla comunità di dislessici.

Tengo a precisare che conosco già il programma "LeggiXme" di Giuliano Serena e, se devo essere sincero, il suo software, per quanto possa avere degli ottimi ideali, non mi ha mai completamente soddisfatto, perché ritengo che l'esperienza utente non sia ottimale. Senza tener conto che un progetto così ampio, gestito da una sola persona, che non ha rilasciato nemmeno il codice sorgente all'AID (Associazione Italiana Dislessia), e lo ha distribuito soltanto come software freeware, mi induce non tanto a sostituire LeggiXme, ma ad offrire una valida alternativa cross platform per Windows, Linux e macOS.

Introduzione

Questo documento descrive tutte le funzioni di Dyslexo e le parti principali del codice che sono state implementate per ovviare ai problemi delle librerie utilizzate per la realizzazione di questo progetto.

Descrizione

Dyslexo è un software di sintesi vocale, ovvero permette la lettura di un testo in maniera automatica, attraverso l'utilizzo della libreria MaryTTS. Esso è indirizzato ad un'utenza dislessica, che ha già appreso le basi della lingua, quindi a partire dalla scuola secondaria di primo grado.

Quanto io vengo offrendo ad oggi nella versione 0.1B è un prototipo sul quale è possibile intervenire, trasformando o arricchendo alcune sue componenti.

In essa sono supportate una serie di lingue: inglese, sia americano sia britannico; italiano, francese e tedesco e permette di eseguire la sintesi vocale su di un testo selezionato, di mettere in pausa la medesima in riproduzione o di riprenderla dal punto sulla quale era stata fermata e di salvarla come file audio in maniera tale da riascoltare il testo anche in assenza del programma.

A differenza di altri software già esistenti, Dyslexo è in grado di offrire un'esperienza prevalentemente priva d'interfaccia grafica; la sintesi vocale è esclusivamente attivata dalla tastiera tramite un'apposita combinazione di tasti.

Istruzioni

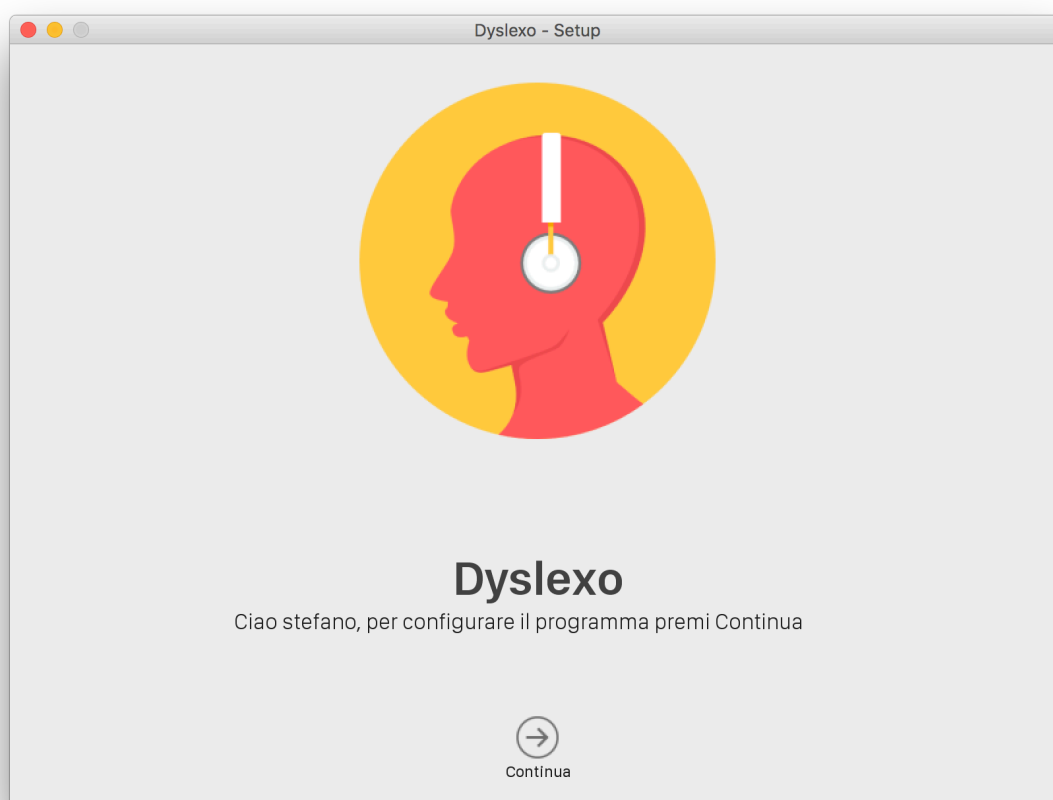
Requisiti

Dyslexo è basato su Java 8, di conseguenza è necessario installare sul proprio sistema Java 1.8, reperibile al seguente indirizzo: <https://www.java.com/it/download/>.

Se il programma è eseguito su macOS, bisogna abilitare Dyslexo al controllo del computer per intercettare le chiamate delle combinazioni di tasti che andranno ad abilitare le funzioni del programma.

Installazione

Una volta soddisfatti i requisiti, non sarà necessaria alcuna installazione aggiuntiva in quanto Dyslexo è completamente **portable**. Solo al primo avvio verrà eseguito un setup che permetterà di scegliere in maniera guidata i settaggi di base utili al corretto funzionamento del programma.

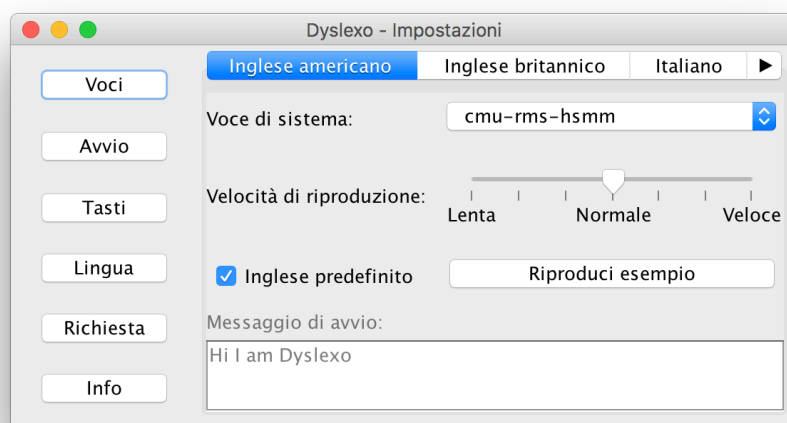


Prima schermata del setup di Dyslexo

Successivamente sarà possibile personalizzare maggiormente o cambiare le impostazioni precedentemente selezionate attraverso le finestre di setup di Dyslexo:

- Voci (inglese americano, inglese britannico, italiano, francese, tedesco):
 - voce da utilizzare come predefinita;
 - velocità di riproduzione;

- ascoltare un esempio della voce;
 - solo per l'inglese, americano e britannico, è possibile scegliere quale delle due usare come predefinita;
 - impostare un messaggio di avvio, che verrà letto all'apertura del programma.
- Avvio:
 - possibilità di aprire Dyslexo all'avvio del sistema (parzialmente implementata);
 - possibilità di riprodurre il messaggio scelto nella sezione "Voci".
- Tasti:
 - scelta della combinazione di tasti per attivare le funzionalità di Dyslexo;
 - scelta del tempo di attivazione della funzione di salvataggio del testo in un file audio;
 - scelta della cartella predefinita dove salvare i propri file audio;
 - possibilità di impostare il salvataggio automatico degli audio senza chiedere tutte le volte la cartella dove salvarlo.
- Lingua:
 - scelta della lingua predefinita di lettura;
 - scelta della lingua predefinita del programma.
- Info, informazioni su:
 - sviluppatori;
 - librerie usate;
 - contenuti visivi impiegati;
 - traduttori delle corrispettive lingue.



Schermata delle impostazioni di Dyslexo

Abilitare le funzioni

Una delle particolarità di Dyslexo è l'assenza dell'interfaccia grafica durante l'utilizzo delle normali funzioni. Sono stati scelti due tasti, "meta" (⌘, ⌘, ⌘) + "esc" e proposti come predefiniti, in maniera tale da rendere più semplice l'utilizzo delle funzioni.

Play, stop

Per avviare la sintesi di un testo selezionato, è necessario tener premuto il tasto meta, tener premuto il tasto esc e successivamente rilasciarli in ordine contrario: prima il tasto esc e di seguito il tasto meta, oppure utilizzare i tasti scelti dall'utente in fase di configurazione.

Se la sintesi è in esecuzione e verranno premuti i tasti secondo quanto descritto sopra, essa verrà interrotta e messa in stop.

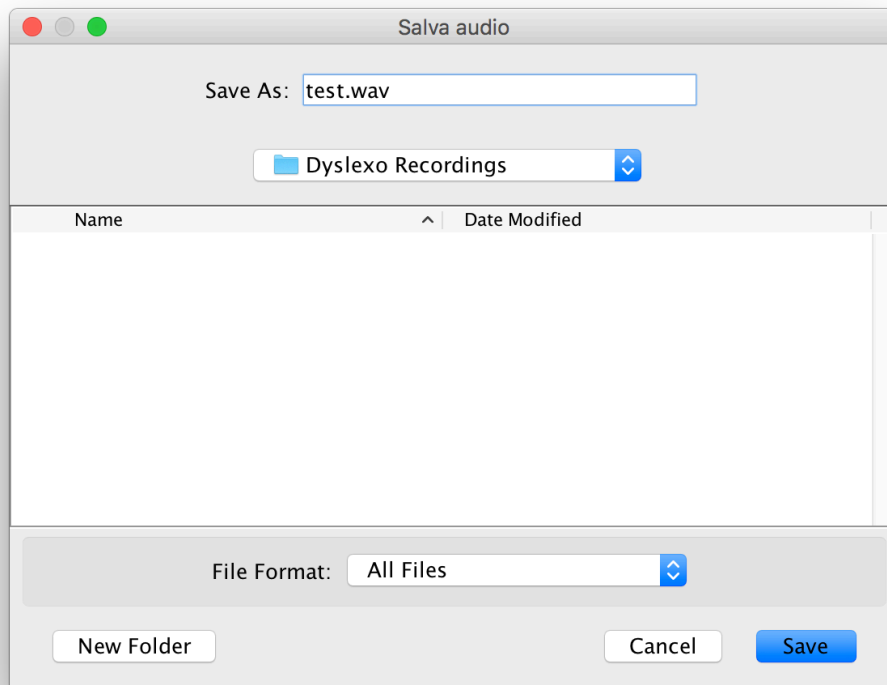
Pausa, play

É possibile anche mettere in pausa o riprenderla, tenendo premuto il tasto meta e premendo due volte consecutivamente il tasto esc o i corrispondenti scelti dall'utente.

Salvataggio su file

Si può salvare il testo selezionato come file audio, premendo il tasto meta e il tasto esc o corrispondenti per almeno 2,5 secondi o per quanto scelto dall'utente nelle impostazioni.

Verrà proposta una schermata del file manager dove sarà possibile scegliere la cartella nella quale salvare il file o, se impostato dall'utente, sarà salvato automaticamente nella cartella predefinita da lui scelta.



Schermata di salvataggio di un file audio

Tecnologie impiegate

Java

Java nasce nel 1995, sviluppato da Sun Microsystems e successivamente acquisito da Oracle Corporation grazie all'acquisto dell'intera azienda. Esso è un linguaggio di programmazione orientato agli oggetti ed è fortemente tipizzato a differenza di altri linguaggi come, ad esempio, PHP.

Java riprende molti aspetti del C# e C++, anche a livello sintattico e strutturale, come il concetto di "classe" e "interfaccia", per facilitare la programmazione ad oggetti; supporta l'overriding, l'ereditarietà e, nell'ultima versione oggi distribuita, Java 8 è stata introdotta una serie di particolari costrutti come "Lambda Expression", che permette di ridurre la lunghezza del codice delle funzioni anonime come, ad esempio, i classici "action listeners" e, privilegiando la facilità di lettura del codice, di seguito si propone un esempio di action listener in notazione normale e in notazione lambda:

```
JButton button = new JButton();
button.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        //write your code here
    }
});
```

```
JButton button = new JButton();
button.addActionListener(e -> {
    //write your code here
});
```

Java organizza il progetto in un "package", ovvero una cartella che contiene le classi e le interfacce del medesimo in modo da semplificare l'organizzazione del codice ed evitare la collisione tra classi aventi lo stesso nome. Un package può contenere a sua volta uno o più package. Solitamente è buona pratica separare la logica dalla grafica dell'applicazione dunque i package in questo caso possono essere d'aiuto per dividere completamente la grafica dalla logica, ad esempio chiamando due classi con lo stesso nome, ma dividendo la grafica dalla logica.

Lo slogan di Java è "Write once, run anywhere", che tradotto in italiano "Scrivilo una sola volta, esegilo ovunque": per questo Java è composto da un interprete chiamato "JVM" (Java Virtual Machine), il quale è dipendente dal software e dalla piattaforma sulla quale è eseguito. Successivamente si preoccuperà di eseguire il codice compilato (bytecode), che a differenza della JVM è indipendente dalla piattaforma sulla quale verrà eseguito.

La scelta di questo linguaggio è quasi obbligata, dato che Dyslexo si prefigge di supportare più piattaforme "out of the box" e la principale libreria utilizzata, ovvero MaryTTS, è scritta in Java.

Maven

Maven è un software opensource, sviluppato da Apache e rilasciato nel 2004, inizialmente ideato per lo sviluppo di applicazioni Java e supporta anche altri linguaggi come, ad esempio, C#, Ruby, Scala, Kotlin e altri ancora.



Esso introduce il concetto di "build life-cycle", ossia una serie di fasi eseguite in un ordine ben preciso e stabilite nel file "POM" (Project Object Model), cioè un semplice file XML in grado di gestire le dipendenze con le relative versioni, le quali verranno scaricate automaticamente

dalla rete e salvate in una cache locale; gestisce inoltre le repository e attraverso i plugin è possibile anche eseguire la compilazione del progetto stesso.

Git

Il codice è stato gestito con git su GitHub, un servizio di hosting che fornisce spazio repository. Le principali funzioni utilizzate sono i "branch", ovvero rami della repository in maniera tale da introdurre nuove funzionalità senza intaccare il codice già funzionante.

Il vantaggio di sviluppare su questa piattaforma, dato che Dyslexo è opensource, è il possibile sviluppo del codice con altri membri della comunità.

GitHub



Logo di GitHub

Principali librerie impiegate

MaryTTS

MaryTTS è una libreria per la sintesi vocale opensource multilingue scritta in Java. Sviluppata inizialmente in collaborazione con DFKI Language Technology Lab (Deutsche Forschungszentrum für Künstliche Intelligenz), l'Institute Phonetics presso la Saarland University, la stessa è ora mantenuta principalmente da DFKI e dal Cluster of Excellence MMCI.



Logo di MaryTTS

Nella sua ultima versione 5.2 offre il supporto a varie lingue, tra le quali italiano, tedesco, inglese britannico, inglese americano, russo, svedese, lussemburghese, telugu e turco.

JNativeHook

JNativeHook è una libreria che permette di gestire globalmente gli eventi di mouse e tastiera tramite degli action listener. Essa è stata utilizzata per intercettare la combinazione di tasti desiderata e successivamente chiamare le funzioni di MaryTTS per la sintesi vocale.

Optimaize Language Detector

Optimaize Language Detector è una libreria opensource, che dato in input del testo identifica la lingua. Essa supporta ben settantuno lingue, ma, per aumentare la probabilità di determinare una delle quattro che Dyslexo supporta, sono state rimosse quelle in eccesso, mantenendo solamente quelle desiderate. Di seguito un esempio di codice:

```
List<LdLocale> preferredLang = new ArrayList<>();
preferredLang.add(LdLocale.fromString("en"));
preferredLang.add(LdLocale.fromString("it"));
preferredLang.add(LdLocale.fromString("de"));
preferredLang.add(LdLocale.fromString("fr"));
List<LdLocale> languages = ImmutableList.copyOf(preferredLang);
List<LanguageProfile> languageProfiles = new LanguageProfileReader().readBuiltIn(languages);

LanguageDetector languageDetector =
LanguageDetectorBuilder.create(NgramExtractors.standard()).withProfiles(languageProfiles).build();
```

Apache Commons

Apache Commons è un grande progetto per gestire funzioni basilari all'interno del proprio codice. In questo caso è stato utilizzato per le impostazioni del programma, che, essendo scritte in XML, avrebbero richiesto il DOM Parser. Grazie ad Apache Commons, invece, è possibile accedere ai tag XML come se fossero dei percorsi.

Una volta inizializzato il file XML,

```
private static XMLConfiguration config;

public Settings(File file) throws ConfigurationException {
    try {
        config = new XMLConfiguration(file);
    } catch (ConfigurationException e) {
        e.printStackTrace();
    }
    config.setExpressionEngine(new XPathExpressionEngine());
}
```

ecco un esempio di lettura,

```
public static String get(String path) {  
    return config.getString(path);  
}
```

e un esempio di scrittura

```
public static void set(String path, String value) {  
    config.setProperty(path, value);  
    try {  
        config.save();  
    } catch (ConfigurationException e) {  
        e.printStackTrace();  
    }  
}
```

dove "path" è il percorso del tag richiesto e "value" è il valore da impostare.

Dunque dato un esempio di XML

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>  
<settings>  
    <setting_0 value="true">test</setting_0>  
    <setting_0 value="false">test</setting_0>  
    <setting_1>test</setting_1>  
</settings>
```

è possibile estrarre i valori all'interno dei tag

```
Settings.get("settings/setting_0[@value='true']");  
Settings.get("settings/setting_1");
```

o scriverli

```
Settings.set("settings/setting_0[@value='true']", "test_mod");  
Settings.set("settings/setting_1", "test_mod");
```

Classi principali

Settings

La classe Settings è stata ideata per gestire le impostazioni di Dyslexo. I principali metodi sono get() e set(), in grado di ritornare o impostare un particolare settaggio, come visto nella spiegazione di Apache Commons; a seguire gli altri, getSpeechLanguage(), ritorna il locale della lingua da utilizzare per la sintesi, getProgramLanguage(), ritorna la lingua predefinita per le traduzioni del programma e genSettingsFile() che dato un Document che genera un file XML per le impostazioni.

Translator

Questa classe è in grado di gestire le lingue e le traduzioni supportate dal programma. Una volta inizializzato l'oggetto statico con la lingua desiderata è possibile usare tutti i suoi metodi. Il principale è getWord(), il quale ritorna come stringa la traduzione presa dai ResourceBundle integrati nel file jar.

Classi principali - Gestione delle chiamate dell'utente

GlobalKeyListener

L'assenza d'interfaccia grafica per la maggior parte dell'utilizzo di Dyslexo è il suo punto di forza, poiché selezionare il testo da leggere e successivamente selezionare un'altra finestra dalla quale selezionare la corretta funzione può essere frustrante. In questo caso la tastiera è l'interfaccia di controllo che permette di attivare le funzioni di sintesi, come già spiegato nelle istruzioni.

La classe GlobalKeyListener sfrutta la libreria JNativeHook, perché Java non ha dei listener per tastiera, ma esistono per componenti grafici o per il terminale della JVM. JNativeHook riprende il concetto di KeyListener disponibile, ad esempio, per un JFrame e ripropone i tre metodi principali: nativeKeyPressed(), nativeKeyReleased() e nativeKeyTyped().

Per realizzare questa classe, sono stati utilizzati i metodi nativeKeyPressed() e nativeKeyReleased(), nei quali vengono controllati dei flag booleani, per verificare che i tasti premuti dall'utente siano quelli scelti nelle impostazioni. Gli stessi chiamano il metodo principale che si occupa di gestire le chiamate delle funzioni richieste in base ai tasti premuti.

Classi principali - Gestione dell'audio

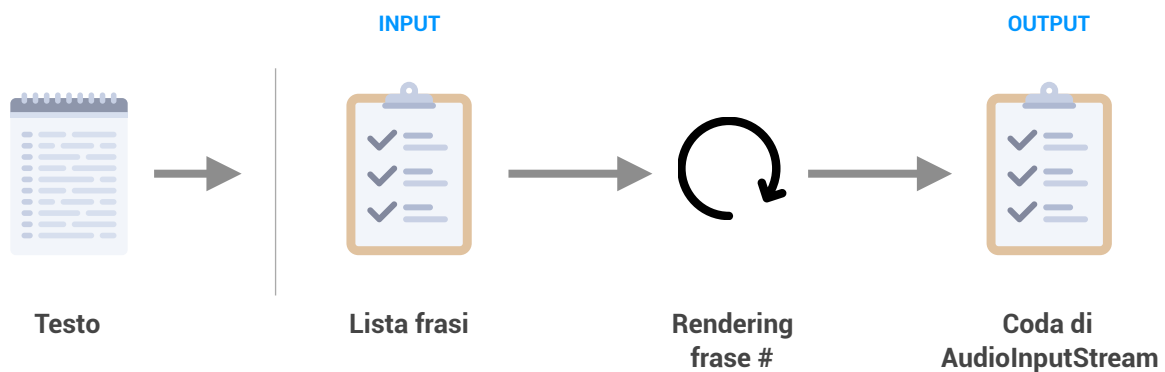
TextProcessor

La classe "TextProcessor" è stata ideata per ovviare ad una serie di problemi di MaryTTS, primo tra tutti la lentezza di elaborazione di testi con qualche migliaio di caratteri: possono sembrare molti, ma in media rappresentano tra le quattro e le cinque frasi consecutive. È possibile risolvere questo problema attivando la cache di MaryTTS, ma questo comporta la necessità di occupare dello spazio aggiuntivo sul pc sul quale viene eseguito Dyslexo, senza tener conto del fatto che agli inizi dell'utilizzo del programma non si trarranno benefici significativi in quanto la cache sarà vuota.

Lo scopo è di leggere sostanzialmente testi di media lunghezza e non semplici parole che appunto sono più veloci da renderizzare, dunque il miglior metodo è di spezzare il testo

sistematicamente all'apparire del punto, per evitare intonazioni errate o spezzettamenti delle parole.

Questo schema rappresenta il funzionamento di TextProcessor:



La classe prende in **input**:

- una lista di tipo "String" delle frasi;
- il locale della lingua che servirà per reperire la voce corretta;
- la velocità di render delle frasi;
- una coda di tipo "Optional<AudioInputStream>".

Una volta inizializzato un oggetto della classe TextProcessor, essendo un thread, chiamando il metodo run(), il quale istanzierà un oggetto di tipo LocalMaryInterface, itererà la lista delle frasi e genererà l'audio chiamando il metodo generateAudio() sull'oggetto LocalMaryInterface. Questo darà come risultato un AudioInputStream che sarà messo all'interno della coda.

MaryTTS supporta anche una serie di effetti audio applicabili all'**output**. Quello che più si presta a questo progetto è la velocità dell'audio che non funziona con i metodi per impostare gli effetti audio direttamente sull'oggetto LocalMaryInterface; per ovviare a ciò è stato usato un input di tipo "RAWMARYXML". Il sistema MaryXML sfrutta il linguaggio XML per soddisfare le esigenze di sintesi, dunque attraverso una serie di tag e attributi e, ovviamente con il rispettivo testo, si è in grado di specificare anticipatamente le regole che la funzione di sintesi dovrà rispettare per produrre l'audio desiderato.

In questo caso sono stati usati i seguenti tag:

- "maryxml", il quale contiene una serie di attributi, tra cui il più importante, "xml:lang", nel quale è specificata la lingua di lettura;
- "prosody", che contiene due attributi, "rate" dove è specificata la velocità di riproduzione in percentuale e "volume" dove è specificato il volume di riproduzione, selezionabile tra:
 - silent;
 - soft;
 - medium;
 - loud;
 - default.

Questo progetto ha sfruttato al minimo le potenzialità di questo sistema di marcatura, con il quale è possibile perfino specificare gli accenti, forzare completamente la pronuncia e l'intonazione delle parole, cambiare la lingua delle singole parole; in un testo dove è prevalente una lingua è possibile eseguire una corretta pronuncia della parola dalla voce corrispondente alla lingua della parola stessa.

MaryXML sarà sfruttato al meglio nelle prossime versioni di Dyslexo, per migliorare la lettura dei contenuti, rendendo più fluente il discorso.

Player

MaryTTS ha integrata già la classe Player in grado di riprodurre gli `AudioInputStream` generati dalla sintesi, purtroppo, come già spiegato nella classe `TextProcessor`, MaryTTS non è così veloce a renderizzare l'audio, inoltre non supporta la pausa del medesimo in riproduzione, perciò la classe Player è stata ideata per supportare un normale `AudioInputStream` o una coda di `AudioInputStream` e riprodurla mentre il testo viene renderizzato dalla classe `TextProcessor`. L'utente percepisce una riproduzione della sintesi del testo quasi immediata nel mentre `TextProcessor` renderizza gli audio successivi.

Converter

La classe `Converter` è utilizzata per esportare una coda di `AudioInputStream`, vista prima nella classe `TextProcessor`, in un audio in formato wav. In futuro saranno implementati vari formati, tra i quali mp3.

Sviluppi futuri

Commenti nel codice e JavaDoc

Uno dei punti vitali dei progetti opensource è quello di avere un codice ben commentato, in maniera tale da identificare le variabili e le funzioni dello stesso e di permettere a chi modifica o revisiona il codice di agevolarne la comprensione ed il funzionamento.

In Java esiste il JavaDoc: è un documento html generato automaticamente da una una funzione in grado di interpretare una serie di tag e di compilare il documento. Successivamente sarà possibile consultarlo per capire immediatamente il funzionamento del codice.

Dyslexo al momento, in questa versione, non offre né un JavaDoc, né dei commenti completi.

Aggiornamenti

Mantenere aggiornato il proprio software è molto importante: sia per permettere all'utente di usufruire delle nuove funzionalità che bugfix, sia per proteggerlo da falle di sicurezza se l'applicativo sfrutta la rete.

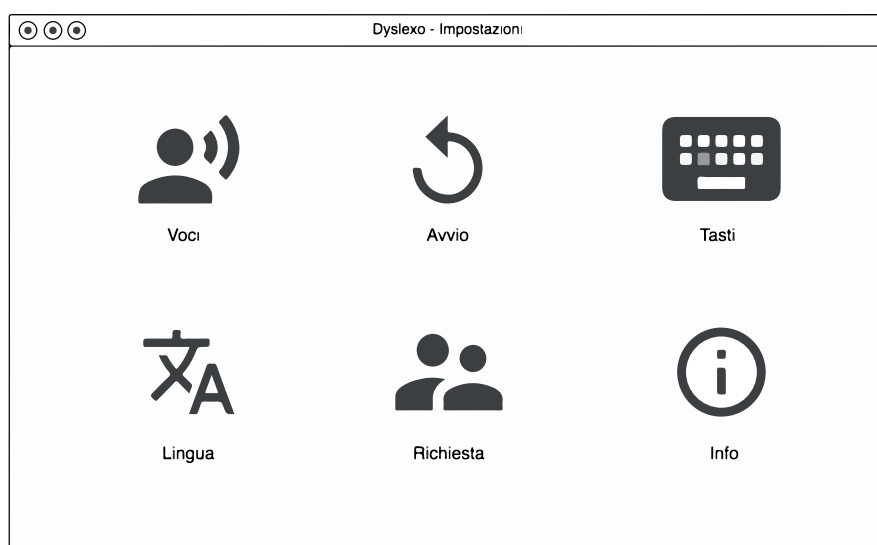
Integrare un metodo che notifichi all'utilizzatore la possibilità di aggiornare automaticamente il programma è di vitale importanza per l'applicativo stesso, anche per facilitare il processo di aggiornamento all'utente inesperto e soprattutto per garantire una buona esperienza a quello finale.

Rilascio repository pubblica su GitHub

Quando il progetto sarà stabile, verrà rilasciato il codice pubblicamente su GitHub e proposto interamente all'AID.

Interfaccia impostazioni

In questa versione di Dyslexo l'interfaccia grafica delle impostazioni non è delle migliori in quanto risulta datata e non è così immediato capire in quale schermata ci si trovi, nonostante alla pressione del tasto del menù corrispondente, appaia attorno al primo un piccolo bordo.



Mockup della nuova grafica delle impostazioni

Miglioramento lettura

Le voci integrate in Dyslexo sono di buona qualità, ma spesso le pronunce non sono ottimali e nei testi multilingue, le parole straniere non sono pronunciate adeguatamente, poiché nel testo una lingua prevale sull'altra e ciò rende sgradevole l'ascolto.

Come già spiegato nella classe TextProcessor, MaryXML è la soluzione al problema. Affiancando a MaryXML un database contenente le parole con la relativa pronuncia e, identificando le medesime di lingua straniera rispetto al testo complessivo, è possibile migliorarne la pronuncia.

Sito web

Presentare il proprio applicativo mediante un sito web è importante per l'utente in maniera tale da offrire un supporto ufficiale.

Conclusioni

Sviluppare questo progetto mi ha portato a vedere in maniera molto più ampia il concetto di programmazione, programmare qualcosa di reale e che ha un fine ben preciso rende tutto il lavoro molto più leggero pur dovendo eseguire continue ricerche; inoltre ho imparato molti nuovi concetti e nozioni riguardanti in generale la programmazione.

A differenza di Dyslexo, LeggiXme si propone come un programma all-in-one, ovvero un programma con all'interno altri programmi, il che è positivo per l'utente inesperto, in quanto non è costretto a scaricare molti programmi e gli facilita il lavoro, evitando di installare materiale non idoneo con virus come malware o ransomware ultimamente diventati popolari. Dyslexo vuole proporsi solo come un programma di sintesi vocale, poiché per tutte le altre funzioni, esistono tantissime alternative rispetto al programma attualmente più usato.

In questi mesi Dyslexo ha raggiunto una dimensione ragguardevole e nel tempo lo diventerà ancora maggiormente, sia in termini di lunghezza che di complessità; credo che nemmeno a tempo pieno sarei in grado di portarlo avanti da solo, questo è un altro dei motivi per il quale è una buona cosa che questo progetto diventi opensource non appena raggiungerà una stabilità accettabile.