Содержание

[Введение 2](#_Toc470207415)

[Теоретическая часть 3](#_Toc470207416)

[Выбор фреймворка 3](#_Toc470207417)

[JavaFX 4](#_Toc470207418)

[Выбор IDE 6](#_Toc470207419)

[Формат обмена данными JSON 11](#_Toc470207420)

[API для работы с файлами в формате JSON 13](#_Toc470207421)

[Практическая часть 15](#_Toc470207422)

[Описание интерфейса 15](#_Toc470207423)

[Структура классов 16](#_Toc470207424)

[Словари 17](#_Toc470207425)

[Анимация 17](#_Toc470207426)

[Перехват клавиш 18](#_Toc470207427)

[Таблица стилей приложения 19](#_Toc470207428)

[Список литературы 21](#_Toc470207429)

# Введение

Скорость выполнения поставленной задачи на компьютере может зависеть не только от очевидных факторов: производительность процессора или, грубо говоря, сообразительность человека. Порой наиболее сдерживающим фактором является сугубо механическая работа, а именно набор текста на клавиатуре. Мысли заходят далеко вперед, а пальцы не успевают фиксировать текст. С такой ситуацией, очевидно, сталкиваются не только писатели, журналисты, программисты, но и все те, кому приходится набирать текст двумя-тремя пальцами, не отрывая взгляд от клавиатуры.

Удивительно, что слепой десятипальцевый метод печати, открытый еще в XIX веке, сейчас многократно более актуален для каждого активного пользователя компьютера. Его умелое применение позволяет существенно сократить затрачиваемое на ввод текста время.

Клавиату́рный тренажёр — вид компьютерных программ или онлайн-сервисов, предназначенных для обучения набору на компьютерной клавиатуре.

Обычно целями тренажёров являются:

* научить слепому методу печати, в частности, задействовать для набора все десять пальцев рук
* увеличить скорость набора
* уменьшить количество опечаток
* улучшить ритмичность набора (что позволяет уменьшить усталость при наборе)

# Теоретическая часть

## Выбор фреймворка

Фреймворк (англ. framework — каркас, структура) — программная платформа, определяющая структуру программной системы; программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта. Употребляется также слово «каркас», а некоторые авторы используют его в качестве основного, в том числе, не базируясь вообще на англоязычном аналоге.

В языке Java существует несколько фреймворков для работы с графический интерфейсом пользователя (GUI).

AWT (Abstract Window Toolkit) был первым GUI фреймворком. Идея была правильная — AWT использует нативные контролы, то есть, они выглядят и физически являются родными, независимо от того, где вы запускаете свое приложение. К сожалению, оказалось, что общих для различных окружений контролов мало и писать кроссплатформенные нативные интерфейсы так, чтобы ничего не поползло и не разъехалось, очень сложно.

Поэтому на смену AWT пришел Swing. Swing использует формы, создаваемые AWT, на которых он своими силами рисует контролы. Работает это хозяйство, понятно дело, медленнее, но зато UI становится намного более портабельным. Swing предлагает на выбор программисту множество Look&Feel, благодаря которым можно сделать либо так, чтобы приложение выглядело и вело себя одинаково как под Windows, так и под Linux, либо чтобы приложение было очень похоже на нативное независимо от того, где его запускают. В первом случае приложение проще отлаживать, во втором — становятся счастливее пользователи.

SWT (Standard Widget Toolkit) — фреймворк, написанный в IBM и используемый в Eclipse. Как и в AWT, используются нативные контролы. SWT не входит в JDK и использует JNI, поэтому не очень соответствует идеологии Java «написано однажды, работает везде». Вроде как при очень сильном желании можно запаковать в пакет реализацию SWT для всех-всех-всех платформ, и тогда приложение вроде как даже станет портабельным, но только до тех пор, пока не появится какая-нибудь новая операционная система или архитектура процессора;

JavaFX активно разрабатывается компанией Oracle и позиционируется, как скорая замена Swing. Идеологически JavaFX похож на Swing, то есть, контролы не нативные. Среди интересных особенностей JavaFX следует отметить аппаратное ускорение, создание GUI при помощи CSS и XML (FXML), возможность использовать контролы JavaFX’а в Swing’е, а также кучу новых красивых контролов, в том числе для рисования диаграмм и 3D. Начиная с Java 7, JavaFX является частью JRE/JDK;

Был выбран фреймворк JavaFX т.к. является самым новым из представленных фреймворков.

## JavaFX

JavaFX можно без преувеличения назвать прорывом. Для отрисовки используется графический конвейер, что значительно ускоряет работу приложения. Набор встроенных компонентов обширен, есть даже отдельные компоненты для отрисовки графиков. Реализована поддержка мультимедийного контента, множества эффектов отображения, анимации и даже мультитач. Внешний вид всех компонентов можно легко изменить с помощью CSS-стилей. И самое прекрасное — в JavaFX входит набор утилит, которые позволяют сделать родной инсталлятор для самых популярных платформ: exe или msi для Windows, deb или rpm для Linux, dmg для Mac. На сайте Oracle можно найти подробную документацию и огромное количество готовых примеров. Это превращает программирование с JavaFX в легкое и приятное занятие.

**Достоинства:**

* быстрая работа за счет графического конвейера;
* множество различных компонентов;
* поддержка стилей;
* утилиты для создания установщика программы;
* приложение можно запускать как десктопное и в браузере как часть страницы.

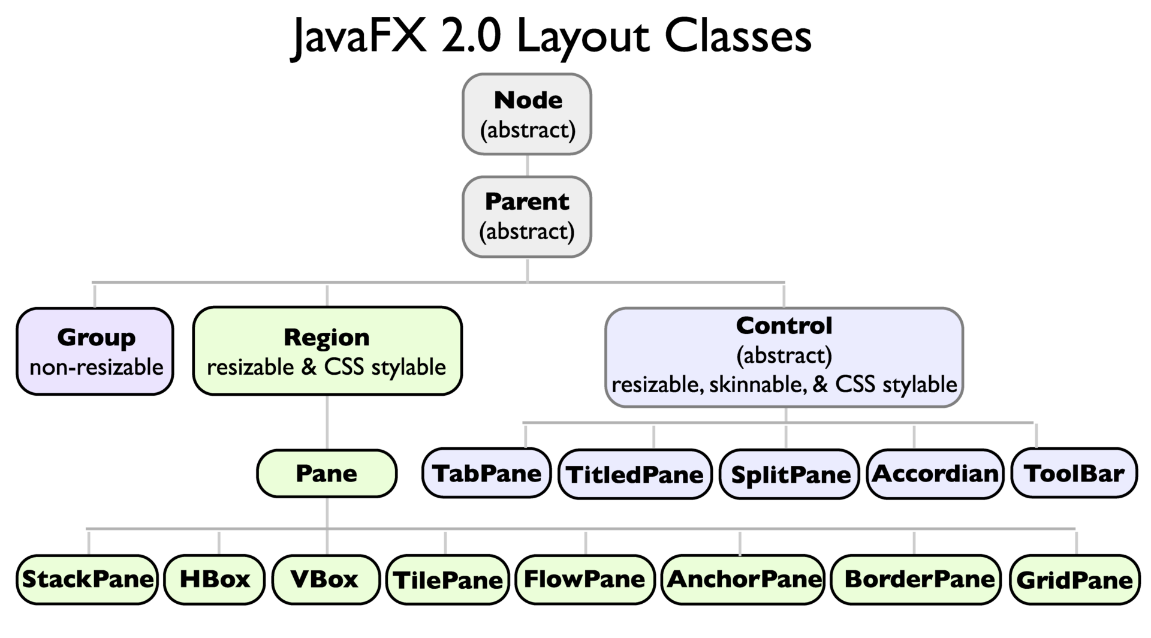
**Недостатки:**

* фреймворк еще разрабатывается, поэтому случаются и падения и некоторые глюки;
* JavaFX пока не получил широкого распространения.

JavaFX похож на веб-страницу: структура хранится в xml формате, для задания стилей используется CSS. Приложение для каждого окна использует контроллер – особый класс, хранящий ссылки на именованные элементы интерфейса и обеспечивающий их взаимодействие. Этот же подход используется и в фреймворках, например, в MVC ASP.NET и NET WPF.

JavaFX содержит обширный набор элементов интерфейса, таких как кнопки, текстовые поля, текстовые редакторы, флажки, панели, сетка, индикатор прогресса и другие.

Иерархия классов в JavaFX 2.0



Благодаря использования технологии CSS и наличию аппаратного ускорения появляется возможность реализовывать красочные и высоко производительные приложения. Так же данный фреймворк содержит мощные инструменты для создания анимации.

Например, этот плеер использует фреймворк JavaFX



## Выбор IDE

**IDE** (Integrated Development Environment) – это среда программирования, используемая программистами для разработки программного обеспечения.

Среда разработки содержит:

* Текстовый редактор
* Компилятор и/или интерпретатор
* Средства автоматизации сборки
* Отладчик

IDE обычно является единой программой, которая включает в себя множество функций для создания, редактирования, компилирования и отладки ПО. Основной задачей IDE является объединение различных утилит в одном модуле, который позволит программисту выполнять решение, избегая потери времени при выполнении типичных технических действий. Таким образом, разработчик повышает свою производительность.

Перед написанием приложения необходимо выбрать подходящее IDE. Правильный выбор среды программирования может упростить работу программиста и сократить время разработки программного обеспечения, что немаловажно для больших проектов. Например, при написании GUI-приложения немаловажным, является удобный и быстрый способ создания графического интерфейса.

В настоящее время насчитывается около шестидесяти сред программирования на Java. IDE можно разделить на две большие группы по способу распространения: платные, такие как JBuilder, Visual Cafe, а так же бесплатные (или свободно распространяемые), среди них IntelliJ IDEA, NetBeans, Eclipse, Gel.

Ещё один критерий деления – наличие визуального средства для разработки GUI. IDE, в которых отсутствуют данные средства, называют «простыми», например: Gel, JCreator. Наличие средств возводит среду программирования в ранг «сложных» приложений, например: JBuilder, Idea. Достоинством простых IDE является то, что при работе они используют меньший объем памяти (30Мб ОЗУ). Сложные IDE используют более 100 Мб ОЗУ.

С одной стороны, сложные IDE должны быть более полезными, нежели простые, так как они облегчают разработку программного обеспечения, но с другой стороны, для начинающих программистов затрудняют понимание базовых вещей библиотеки Swing (менеджеры размещения, обработка событий и пр.).

При выборе IDE обращают внимание на три основных компонента:

1. Интерфейс.

«Встречают по одежке…» Первое впечатление о программе складывается при столкновении пользователя с интерфейсом программы. Здесь оценивается не только общий дизайн, но и расположение различных компонентов, таких как окно исходного кода, окно проекта, различные навигаторы, меню и пр.

1. Настройка.

После установки, пользователю необходимо указать настройки для правильной работы IDE, т.е. указание путей, где установлены нужные компоненты. Такие настройки, как правило, сильно не различаются. Более того, современные IDE используют автоматическую настройку и сами определяют нужные компоненты.

1. Редактор кода

Настройка отображения исходного кода обычно не сильно отличается. В большинстве случаев можно установить нужные настройки самостоятельно, изменив размер текста, а так же цвет и шрифт. Особо важным преимуществом является наличие помощника, который при наведении мыши на метод или переменную вызывает контекстное меню, в котором подробно описаны все параметры объекта. Помимо этого существует масса разных мелочей, которые помогают программисту во время разработки приложения.

По одной из статистик чётко показано, что чаще всего выбор останавливается на какой-то из трёх следующих сред:

1. Eclipse

Eclipse – бесплатная интегрированная среда разработки, поддерживаемая Eclipse Foundation. Eclipse разработана на языке программирования Java, поэтому является платформо-независимым продуктом. Она содержит в себе множество различных модулей, диспетчерами для работы с базой данных, серверами приложений и др. Изначально среда была разработана для работы с языком Java, но в данный момент существует масса расширений для поддержки других языков: C/C++, Fortan, Perl, PHP, JavaScript, Python, Ruby и др. GUI в Eclipse содержит библиотеку SWT, которая, в отличии от Swing, использует графические компоненты данной операционной системы. Swing самостоятельно эмулирует графические элементы. Благодаря подключаемым модулям обеспечивается гибкость среды, что позволяет работать не только на Java, но и на других языках.

1. NetBeans

NetBeans IDE — свободная интегрированная среда разработки приложений, созданная NetBeans Community. Она написана на языке программирования Java, доступна в готовых дистрибутивах для большинства платформ. Более того, для всех остальных платформ есть возможность самостоятельно скомпилировать NetBeans из исходных кодов. Данная среда поддерживает такие языки программирования, как Java, PHP, C/C++ и др. Это первая бесплатная среда IDE, которая поддерживает JDK 8, JDK 7, Java EE 7, включая усовершенствования HTML5 и JavaFX 2. IDE NetBeans позволяет быстро и легко разрабатывать различные настольные, мобильные и веб-приложения Java.

1. IntelliJ IDEA

IntelliJ IDEA – интегрированная среда разработки, поддерживаемая JetBrains. Начиная с версии 9.0, среда доступна в двух версиях: Community Edition (бесплатная) и Ultimate Edition (платная). Отличительная особенность IDEA – понимание контекста. Именно это имеют в виду сотрудники JetBrains, когда называют её «интеллектуальной». IntelliJ IDEA индексирует весь проект, анализируя его содержимое, и строит синтаксическое дерево. Благодаря этому среда в любой момент «знает», в каком месте программы находится программист, и что возможно там выполнить. Кроме Java поддерживаются и другие языки программирования, среди которых JavaScript, HTML, CSS, XML, Python, Ruby, PHP, C/C++ и др. а так же, через плагины сторонних разработчиков: Go, Fantom, Rust, Perl5 и др.

IntelliJ IDEA имеет интегрированный визуальный редактор GUI для Java FX. Есть удобный редактор json и xml.

Для написания приложения была выбрана IntelliJ IDEA, в связи с удобством использования и возможностями данной IDE.

## **Формат обмена данными** JSON

JSON (JavaScript Object Notation) – текстовый формат обмена, основанный на синтаксисе языка JavaScript. Как и многие другие текстовые форматы, JSON легко читается людьми. JSON сильно похож на формат XML но практический лишен проблемы избыточности XML и требует меньше памяти. Формат представляет собой древовидный набор структур двух видов:

* Набор пар *ключ: значение*.
* Упорядоченный набор *значений*.

В качестве значений в JSON могут быть использованы:

* Объект — это неупорядоченное множество пар ключ:значение, заключённое в фигурные скобки «{ }». Ключ описывается строкой, между ним и значением стоит символ «:». Пары ключ-значение отделяются друг от друга запятыми.
* Массив (одномерный) — это упорядоченное множество значений. Массив заключается в квадратные скобки «[ ]». Значения разделяются запятыми.
* Число.
* Литералы true, false и null.
* Строка — это упорядоченное множество из нуля или более символов юникода, заключенное в двойные кавычки.

За счёт своей лаконичности по сравнению с XML, формат JSON может быть более подходящим для сериализации сложных структур. Если говорить о веб-приложениях, в таком ключе он уместен в задачах обмена данными как между браузером и сервером (AJAX), так и между самими серверами (программные HTTP-сопряжения).

Поскольку формат JSON является подмножеством синтаксиса языка JavaScript, то он может быть быстро десериализован встроенной функцией eval(). Кроме того, возможна вставка вполне работоспособных JavaScript-функций. В языке PHP, начиная с версии 5.2.0, поддержка JSON включена в ядро в виде функций json\_decode() и json\_encode(), которые сами преобразуют типы данных JSON в соответствующие типы PHP и наоборот.

Пример JSON файла:

{

"firstName": "Иван",

"lastName": "Иванов",

"address": {

"streetAddress": "Московское ш., 101, кв.101",

"city": "Ленинград",

"postalCode": 101101

},

"phoneNumbers": [

"812 123-1234",

"916 123-4567"

]

}

В формате XML эта структура может быть записана следующим образом:

**<person** firstName="Иван" lastName="Иванов"**>**

**<address** streetAddress="Московское ш., 101, кв.101" city="Ленинград" postalCode="101101" **/>**

**<phoneNumbers>**

**<phoneNumber>**812 123-1234**</phoneNumber>**

**<phoneNumber>**916 123-4567**</phoneNumber>**

**</phoneNumbers>**

**</person>**

## **API для работы с** файлами в формате JSON

API (application programming interface) – набор готовых классов, функций, процедур, констант и структур, предоставляемый для использования во внешних программных продуктах. Используется программистом для написания приложений. Большинство операционных систем имеют свою API, с помощью которого программисты могут создавать приложения для этой операционной системы. Наиболее известные API:

1. Операционных систем: Cocoa, Windows API, Linux Kernel API и др.
2. Графических интерфейсов: Direct3D (часть DirectX), GDI, OpenGL, OpenVG, SDL, и др.
3. Звуковых интерфейсов: OpenAL, DirectSound, DirectMusic.
4. Аутентификационных систем: BioAPI, PAM.

Для работы с JSON форматом в языке JAVA нет штатного API, однако существует множество библиотек от сторонних разработчиков:

**Alibaba Fastjson** Быстрый JSON обработчик, рейтинг github'а — 4851. User guide и Hello World examples. Лицензия: Apache 2. Лицензия совместима с закрытым ПО

**Gson** — Простая сериализации объектов в JSON и обратно. Хорошая производительность и легкость в использовании, рейтинг github'а — 4120. User guide и Hello World examples. Лицензия: Apache 2. Лицензия совместима с закрытым ПО

**LoganSquare** -Библиотека парсинга и сериализации JSON, основанная на Jackson's streaming API. По словам разработчиков, превосходит по производительности GSON и Jackson библиотеки, рейтинг github'а — 2188. User guide и Hello World examples. Лицензия: Apache 2. Лицензия совместима с закрытым ПО

**JSON java** Реализация работы с JSON в Java от разработчиков JSON стандарта, рейтинг github'а — 1937. User guide и Hello World examples. Лицензия: Crockford's license (MIT License + "Good, not Evil").

**Square Moshi JSON** библиотека для Android и Java, служит для упрощения парсинга Json в объекты Java, рейтинг github'а — 1732. User guide и Hello World examples. Лицензия: Apache 2 Лицензия совместима с закрытым ПО

**Instagram Ig json parser** Быстрый JSON парсер для java проектов, рейтинг github'а — 921. User guide и Hello World examples. Лицензия: BSD 3. Лицензия совместима с закрытым ПО

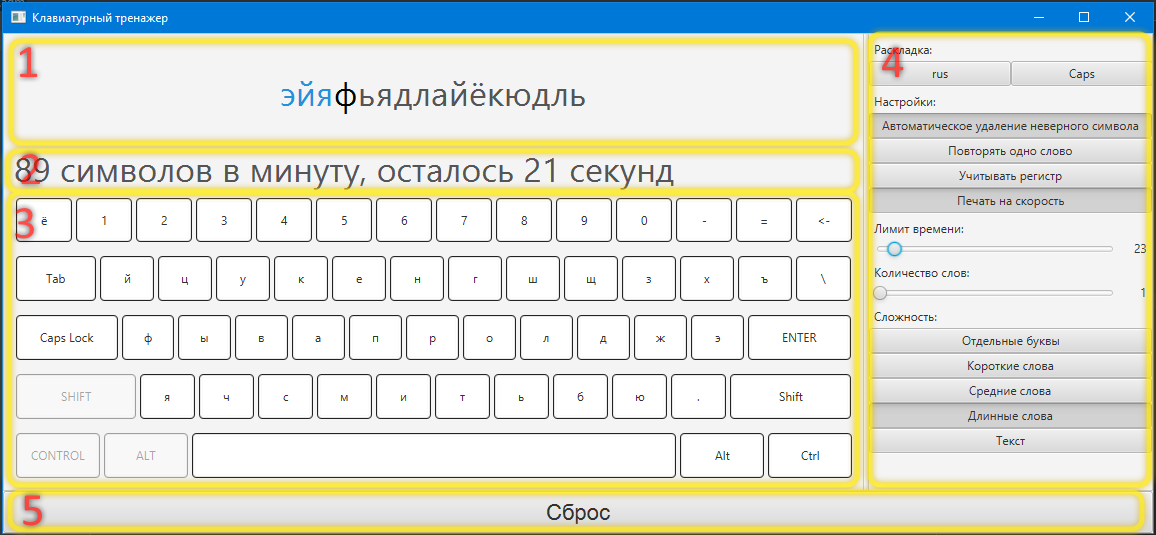
**Jackson** — Похоже на GSON, но более производительна, если вам нужно часто создавать экземпляр библиотеки. Подпроекты: Jackson core Базовая часть функционала, Jackson databind Базовая реализация databind'а, рейтинг github'а — 881. User guide и Hello World examples. Лицензия: Apache 2. Лицензия совместима с закрытым ПО

**Genson** — Мощная и простая в использовании Java библиотека для преобразования в/из JSON, рейтинг github'а — 108. User guide и Hello World examples. Лицензия: Apache 2. Лицензия совместима с закрытым ПО

Все эти библиотеки обеспечивают быструю и надежную работу с JSON. Однако они довольно просты в освоении и представляют собой большие проекты. Для решения задач, необходимых приложению такой функционал излишен. Поэтому была выбрана библиотека Json**-simple** разработанная в компании Google. Это маленькая библиотека с открытым исходным кодом. Она легка в освоении и полностью справляется с поставленными задачами.

# Практическая часть

## **Описание интерфейса**



Окно приложения Рис.1

Программа работает с использованием библиотеке JavaFX, за счет чего удается реализовать гибкий сложный интерфейс, состоящий из таблиц (Grid).

На рис. 1 представлено окно программы, оно состоит из следующих частей:

1. Область вывода текста. Содержит текст, который должен ввести либо уже ввел пользователь.
2. Строка состояния. Отображает текущую скорость ввода текста и оставшееся время (если был выбран режим печати на скорость)
3. Экранная клавиатура. Сетка из клавиш, показывающая на какую клавишу нажал пользователь. Клавиши могут загораться тремя цветами, в зависимости от того, нужно ли было их наживать в данный момент.
   1. Красный  - нажатие клавиши ошибочно
   2. Зеленый  - верно нажатая клавиша
   3. Синий  - нажата клавиша изменения регистра.
4. Список настроек программы. Содержит настройки сложности, активную раскладку клавиатуры и критерии ввода текста.
5. Кнопка сброса. Сбрасывает таймер и генерирует новый текст.

## **Структура классов**

Программа содержит несколько классов:

* Main
* Setting
* MainController
* Task

Main – класс определяющий точку входа в приложение и запускающий основное FX окно приложения из файла MainWindow.fxml с контроллером MainController.

Setting – статический класс, хранящий настройки приложения.

MainController – контроллер главного окна FX приложения. Осуществляет взаимодействие пользователя с экземпляром класса Task и Setting

Task – класс хранящий вводимый текст и позицию в нем. Осуществляет проверку на корректность нажатой клавиши и отображение символов и из состояний.

Так же для хранения уровня сложности был создан новый тип данных Difficulty

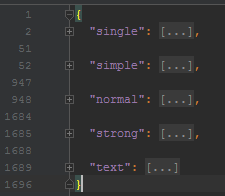
enum Difficulty  
{  
 *single*,  
 *simple*,  
 *normal*,  
 *strong*,  
 *text*}

Настройки приложения, тип и значение по умолчанию:

boolean *useReg* = false;  
boolean *autoBackspace* = true;  
boolean *multiPhrase* = true;  
boolean *useTime* = false;  
int *timeLimit* = 60;  
int *phraseCount* = 5;  
Difficulty *difficulty* = Difficulty.*normal*;

## **Словари**

Словари для каждого из уровней сложности хранятся в одном файле Dictionary.json в формате JSON



Каждому уровню сложности соответствует одноименный словарь. Благодаря возможности преобразования перечислений (enum) в текстовое представление хранение данных в данном формате удобно и не требует дополнительных преобразований.

Метод загрузки словаря из файла:

static void LoadDictionary(String file) throws IOException, ParseException  
{  
 JSONParser parser = new JSONParser();  
  
 InputStream in = Task.class.getResourceAsStream(file);  
 java.util.Scanner scanner = new java.util.Scanner(in,"UTF-8").useDelimiter("\\A");  
  
 *dictionarys* = (JSONObject) parser.parse(scanner.next());  
}

## Анимация

Для отображения нажатия на клавишу используются возможности анимации JavaFX. Основой является класс временной шкалы Timeline, который представляет собой счетчик изменяющий объект на заданном промежутке времени при помощи ключевых точек.

Метод для анимирования CSS свойства **-background-color** у заданного объекта интерфейса:

private void playClickEffect(Button control, Color color)  
{  
 Timeline timeline = new Timeline();  
 SimpleObjectProperty<Color> op = new SimpleObjectProperty();  
  
 timeline.getKeyFrames().addAll(  
 new KeyFrame(new Duration(0), new KeyValue(op, Color.*WHITE*)),  
 new KeyFrame(new Duration(70), new KeyValue(op, color)),  
 new KeyFrame(new Duration(170), new KeyValue(op, color)),  
 new KeyFrame(new Duration(340), new KeyValue(op, Color.*WHITE*))  
 );  
  
 op.addListener((obs, old, newColor) ->  
 control.setStyle(String.*format*("-background-color: #%02x%02x%02x; ",  
 (int)(newColor.getRed() \* 255),  
 (int)(newColor.getGreen() \* 255),  
 (int)(newColor.getBlue() \* 255))  
 )  
 );  
  
 timeline.play();  
}

## Перехват клавиш

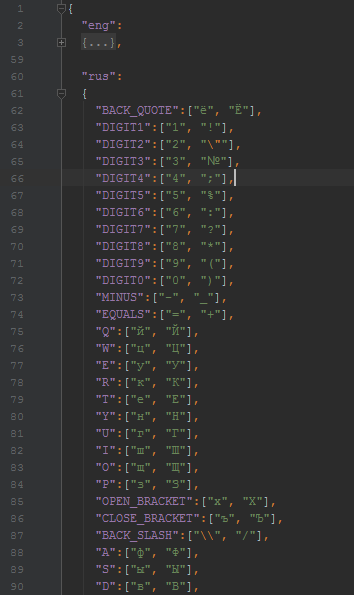
Для того чтобы понять какая клавиша нажата, на главный контейнер формы был установлен обработчик события нажатия клавиши. Так же все кнопки-переключатели лиши возможности получать фокус. Обработчик события возвращает код нажатой клавиши. На форме ищется кнопка, имеющая имя, соответствующее этому коду. Также в словаре раскладки коду находится соответствие.

Словарь раскладки представляет собой таблицу, содержащую строку, соответствующею коду клавиши, языку и регистру.

Метод загрузки словаря из файла:

private void loadLayout()  
{  
 try  
 {  
 JSONParser parser = new JSONParser();  
  
 InputStream in = getClass().getResourceAsStream("Layout.json");  
 java.util.Scanner scanner = new java.util.Scanner(in,"UTF-8").useDelimiter("\\A");  
  
 layout = (JSONObject) parser.parse(scanner.next());  
 }  
 catch(ParseException e)  
 {  
 System.*out*.println("Файл раскладки клавиатуры Layout.json поврежден");  
 System.*exit*(0);  
 }  
}

Данный словарь хранится в JSON файле и имеет следующий вид:



## Таблица стилей приложения

Приложение использует индивидуальные стили для некоторых элементов. Для отображения прогресса ввода текст (Label) имеет 3 визуальных класса: default, next, set. Так же для подсветки ошибки был создан класс error. Кнопки (Button) имеют уникальный дизайн, для схожести с клавишами клавиатуры.

.label {  
 -fx-font-size: 34px;  
 -fx-text-fill: #555555;  
}  
  
.next {  
 -fx-text-fill: #000000;  
}  
  
.set {  
 -fx-text-fill: #2890DC;  
}  
  
.error {  
 -fx-background-color: #fd3f49;  
}  
  
.button {  
 -background-color: #ffffff;  
 -fx-background-color: -background-color;  
 -fx-border-color: #0f0f0f;  
 -fx-border-radius: 4;  
 -fx-padding: 2px;  
 -fx-border-insets: 2px;  
 -fx-background-insets: 2px;  
}

# Список литературы

1. https://ru.wikipedia.org/wiki/
2. https://poi.apache.org/
3. http://www.javaportal.ru/projects/taidej/results.html
4. https://www.jetbrains.com/idea/
5. https://netbeans.org/
6. https://habrahabr.ru/company/luxoft/blog/280782/