JAVA PROJECT ON TYPING SPEED TEST

ABSTRACT

The "Typing Speed Test" Java Project is a software application designed to assess and enhance a person's typing proficiency. The project provides an interactive platform where users can practice their typing skills, measure their typing speed in WPM and assess their accuracy. By providing a diverse range of text and accurately calculating typing speed and accuracy, the project aims to help users improve their keyboarding skills and efficiency. Project's user-friendly interface, real-time performance metrics, and customization options make it a valuable tool for individuals looking to enhance their typing abilities in a variety of personal and professional contexts.

INTRODUCTION

In today's fast-paced digital landscape, proficient typing skills are indispensable for effective communication and efficient task execution. The ability to input information quickly and accurately is a fundamental skill that transcends various professions and personal endeavors. The "Typing Speed Test" Java project addresses this need by offering a user-friendly and engaging platform to assess and improve typing skills.

The project aims to create an interactive environment where users can improve not only their typing speed but also their accuracy and precision. Through a combination of thoughtfully designed user interfaces, real-time performance evaluations and customizable practice exercises, the project intends to empower individuals to enhance their keyboarding abilities and keep up with the demands of modern communication.

DESIGN AND FEATURES

Design Considerations:

- User-Friendly Interface: Create an intuitive and visually appealing graphical user interface (GUI) that accommodates users of all skill levels.
- Text Selection: Offer a variety of texts of varying lengths and complexity for users to practice typing. Implement an option to randomly select texts.
- Real-Time Feedback: Provide instant feedback to users as they type, indicating correctness and errors.
- Performance Metrics: Calculate and display the typing speed in words per minute (WPM) and accuracy percentage based on correctly typed characters.
- Timer: Implement a timer that starts when the user begins typing and stops when the entire text is typed, aiding in speed measurement.

Features:

- Text Display: Present the selected text for typing in a clear and readable format, ensuring comfortable readability.
- Typing Area: Include a designated text entry area where users can type the displayed text.
- Accuracy Assessment: Compare each typed character with the corresponding character in the original text, highlighting errors and correct inputs.
- Typing Speed Calculation: Calculate the typing speed by dividing the total number of words typed by the time taken to complete the text.
- Accuracy Calculation: Determine the accuracy percentage by comparing the number of correctly typed characters to the total characters.
- Random Text Selection: Allow users to select a random text from the available options to keep practice sessions diverse.

Implementation:

The implementation of the Typing Speed Test project involves several key components and features:

1. **User Interface:** The project provides a graphical user interface (GUI) that includes a text box for users to type the given text, a timer to measure typing speed, and an accuracy calculation module.
2. **Text Selection:** Users can choose from a variety of predefined texts of varying lengths and complexities to practice typing. The selected text is displayed in the text box for users to type.
3. **Timer and Speed Calculation:** The timer starts when the user begins typing and stops when the text is completed. The typing speed is calculated by dividing the number of words typed by the time taken and converting it to words per minute (WPM).
4. **Accuracy Calculation:** The accuracy of the typed text is calculated by comparing each typed character with the corresponding character in the original text. The accuracy percentage is then determined based on the number of correctly typed characters.
5. **Progress Tracking:** The project keeps track of users' progress by recording their highest achieved typing speed and accuracy. This encourages users to practice and improve over time.

Advantages:

- **Skill Enhancement:** The project provides an efficient way for users to enhance their typing speed and accuracy, leading to increased productivity in various tasks that involve typing.
- **User Engagement:** The interactive interface and challenges make practicing typing enjoyable and engaging for users of all skill levels.
- **Measurement Metrics:** The project offers objective metrics, such as WPM and accuracy percentage, which allow users to gauge their improvement and set goals.
- **Customizability:** Users can select texts of different lengths and complexities, allowing them to tailor their practice sessions to their specific needs.

Disadvantages:

- **Dependency on Technology:** The project relies on the availability of a computer and the Java runtime environment, limiting access for those who lack these resources.
- **Text Limitation:** The project's effectiveness may be limited by the range of texts available for practice. Diverse content may be needed to cover various typing scenarios.
- **Lack of Real-time Feedback:** The project does not offer real-time feedback during typing, potentially hindering users' ability to correct mistakes as they occur.

Code:

Main.java

```
public class Main
{
public static void main(String args[])
{
new Frame();
}
}
```

Frame.java

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.time.LocalTime;
import java.util.ArrayList;
import java.util.Random;
//author@TechVidvan
@SuppressWarnings("serial")
public class Frame extends JFrame implements ActionListener, KeyListener
{
String passage="";
String typedPass="";
String message="";
//author@TechVidvan
int typed=0; //typed stores till which character the user has typed
```

```
int count=0;
int WPM;
//author@TechVidvan
double start;
double end;
double elapsed;
double seconds;
boolean running;
boolean ended; //Whether the typing test has ended or not
final int SCREEN_WIDTH;
final int SCREEN_HEIGHT;
final int DELAY=100;
JButton button;
Timer timer;
JLabel label;
public Frame()//author@TechVidvan
{
this.setLayout(new BorderLayout());
this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
SCREEN_WIDTH=720;
SCREEN_HEIGHT=400;
this.setSize(SCREEN_WIDTH,SCREEN_HEIGHT);
this.setVisible(true);
this.setLocationRelativeTo(null);
button=new JButton("Start");
button.setFont(new Font("Georgia",Font.BOLD,30));
button.setForeground(Color.BLUE);
button.setVisible(true);
button.addActionListener(this);
```

```
button.setFocusable(false);
label=new JLabel();
label.setText("Click the Start Button to begin the test");
label.setFont(new Font("Georgia",Font.BOLD,30));
label.setVisible(true);
label.setOpaque(true);
label.setHorizontalAlignment(JLabel.CENTER);
label.setBackground(Color.lightGray);
this.add(button, BorderLayout.SOUTH);
this.add(label, BorderLayout.NORTH);
this.getContentPane().setBackground(Color.WHITE);
this.addKeyListener(this);
this.setFocusable(true);
this.setResizable(false);
this.setTitle("Typing Test");
this.revalidate();
}
@Override
public void paint(Graphics g)
{
super.paint(g);
draw(g);
}
public void draw(Graphics g)
{
g.setFont(new Font("Georgia", Font.BOLD, 25));

if(running)
{
```

```
//This will put our passage on the screen
if(passage.length()>1)
{
g.drawString(passage.substring(0,50), g.getFont().getSize(),
g.getFont().getSize()*5);
g.drawString(passage.substring(50,100), g.getFont().getSize(),
g.getFont().getSize()*7);
g.drawString(passage.substring(100,150), g.getFont().getSize(),
g.getFont().getSize()*9);
g.drawString(passage.substring(150,200), g.getFont().getSize(),
g.getFont().getSize()*11);
}
//Displaying correctly typed passage in White
g.setColor(Color.WHITE);
if(typedPass.length()>0)
{
if(typed<50) //if the typed index is in the first line
g.drawString(typedPass.substring(0,typed),
g.getFont().getSize(),g.getFont().getSize()*5); //From the first letter to the
currently typed one in green
else
g.drawString(typedPass.substring(0,50),
g.getFont().getSize(),g.getFont().getSize()*5); //If the typed character exceeds
50 we can directly show the whole line in green
}
if(typedPass.length()>50)
{
if(typed<100)
g.drawString(typedPass.substring(50,typed),
g.getFont().getSize(),g.getFont().getSize()*7);
else
```

```
g.drawString(typedPass.substring(50,100),
g.getFont().getSize(),g.getFont().getSize()*7);
}
if(typedPass.length()>100)
{
if(typed<150)
g.drawString(typedPass.substring(100,typed),
g.getFont().getSize(),g.getFont().getSize()*9);
else
g.drawString(typedPass.substring(100,150),
g.getFont().getSize(),g.getFont().getSize()*9);
}
if(typedPass.length()>150)
g.drawString(typedPass.substring(150,typed),
g.getFont().getSize(),g.getFont().getSize()*11);
running=false;
}
if(ended)
{
if(WPM<=40)
message="You are an Average Typist";
else if(WPM>40 && WPM<=60)
message="You are a Good Typist";
else if(WPM>60 && WPM<=100)
message="You are an Excellent Typist";
else
message="You are an Elite Typist";
FontMetrics metrics=getFontMetrics(g.getFont());
g.setColor(Color.BLUE);
```

```
g.drawString("Typing Test Completed!", (SCREEN_WIDTH-metrics.stringWidth("Typing Test Completed!"))/2, g.getFont().getSize()*6);

g.setColor(Color.BLACK);

g.drawString("Typing Speed: "+WPM+" Words Per Minute", (SCREEN_WIDTH-metrics.stringWidth("Typing Speed: "+WPM+" Words Per Minute"))/2, g.getFont().getSize()*9);

g.drawString(message, (SCREEN_WIDTH-metrics.stringWidth(message))/2, g.getFont().getSize()*11);


timer.stop();

ended=false;

}

}


@Override

public void keyTyped(KeyEvent e) //keyTyped uses the key Character which can identify capital and lowercase difference in keyPressed it takes unicode so it also considers shift which creates a problem

{

if(passage.length()>1)

{

if(count==0)

start=LocalTime.now().toNanoOfDay();

else if(count==200) //Once all 200 characters are typed we will end the time and calculate time elapsed

{

end=LocalTime.now().toNanoOfDay();

elapsed=end-start;

seconds=elapsed/1000000000.0;

WPM=(int)(((200.0/5)/seconds)*60); //number of character by 5 is one word by seconds is words per second * 60 WPM
```

```
ended=true;
running=false;
count++;
}
char[] pass=passage.toCharArray();
if(typed<200)
{
running=true;
if(e.getKeyChar()==pass[typed])
{
typedPass=typedPass+pass[typed]; //To the typed Passage we are adding what is currently typed
typed++;
count++;
} //If the typed character is not equal to the current position it will not add it to the checked
}
}
}
//author@TechVidvan
@Override
public void keyPressed(KeyEvent e)
{
}
//author@TechVidvan
@Override
public void keyReleased(KeyEvent e)
{
}
```

```
//author@TechVidvan
@Override
public void actionPerformed(ActionEvent e)
{
if(e.getSource()==button)
{
passage=getPassage();
timer=new Timer(DELAY,this);
timer.start();
running=true;
ended=false;
typedPass="";
message="";
typed=0;
count=0;
}
if(running)
repaint();
if(ended)
repaint();
}
public static String getPassage()
{
ArrayList<String> Passages=new ArrayList<String>();
String pas1="There are usually about 200 words in a paragraph, but this can vary widely. Most paragraphs focus on a single idea that's expressed with an introductory sentence, then followed by two or more supporting sentences about the idea.";
```

String pas2="Your only chance of survival, if you are sincerely smitten, lies in hiding this fact from the woman you love, of feigning a casual detachment under all circumstances.";

String pas3="Pennies saved one and two at a time by bulldozing the grocer and the vegetable man and the butcher until one’s cheeks burned with the silent imputation of parsimony that such close dealing implied. ";

String pas4="At that moment he had a thought that he'd never imagine he'd consider. \"I could just cheat,\" he thought, \"and that would solve the problem.\" He tried to move on from the thought but it was persistent.";

String pas5="Out of another, I get a lovely view of the bay and a little private wharf belonging to the estate. There is a beautiful shaded lane that runs down there from the house. ";

String pas6="What have you noticed today? I noticed that if you outline the eyes, nose, and mouth on your face with your finger, you make an \"I\" which makes perfect sense, but is something I never noticed before. What have you noticed today?";

String pas7="Balloons are pretty and come in different colors, different shapes, different sizes, and they can even adjust sizes as needed. But don't make them too big or they might just pop, and then bye-bye balloon. It'll be gone and lost for the rest of mankind.";

String pas8="He picked up the burnt end of the branch and made a mark on the stone. Day 52 if the marks on the stone were accurate. He couldn't be sure. Day and nights had begun to blend together creating confusion, but he knew it was a long time. Much too long.";

String pas9="You know that tingly feeling you get on the back of your neck sometimes? I just got that feeling when talking with her. You know I don't believe in sixth senses, but there is something not right with her. I don't know how I know, but I just do.";

String pas10="It seemed like it should have been so simple. There was nothing inherently difficult with getting the project done. It was simple and straightforward enough that even a child should have been able to complete it on time, but that wasn't the case.";

Passages.add(pas1);

Passages.add(pas2);

Passages.add(pas3);

Passages.add(pas4);

```
Passages.add(pas5);
Passages.add(pas6);
Passages.add(pas7);
Passages.add(pas8);
Passages.add(pas9);
Passages.add(pas10);
Random rand=new Random();
int place=(rand.nextInt(10));
String toReturn=Passages.get(place).substring(0,200);
if(toReturn.charAt(199)==32)
{
toReturn=toReturn.strip();
toReturn=toReturn+".";
}
return(toReturn);
}
}
```

Output:

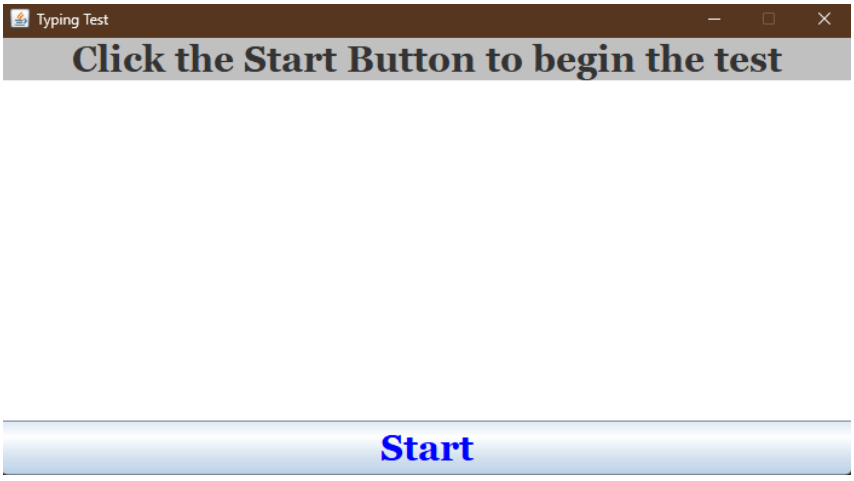

Fig: Start screen

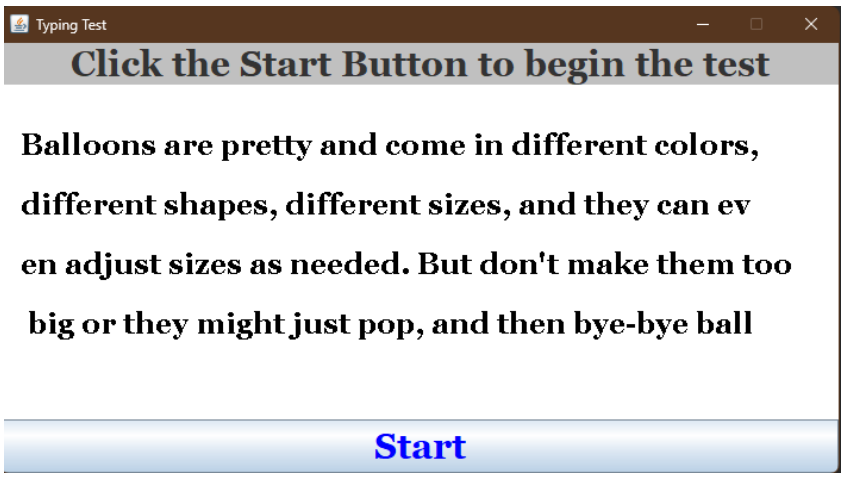

Fig: Paragraph to be written

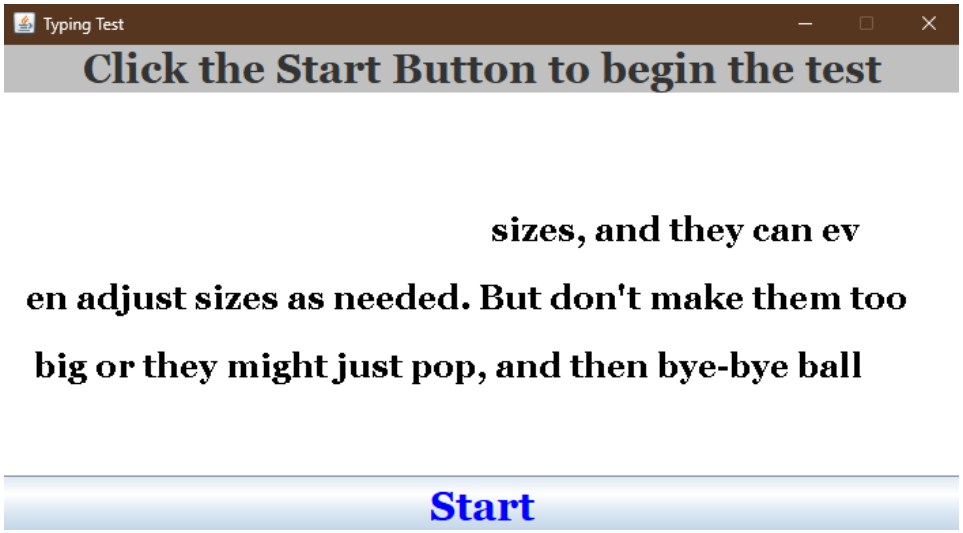

Fig: Test on-going

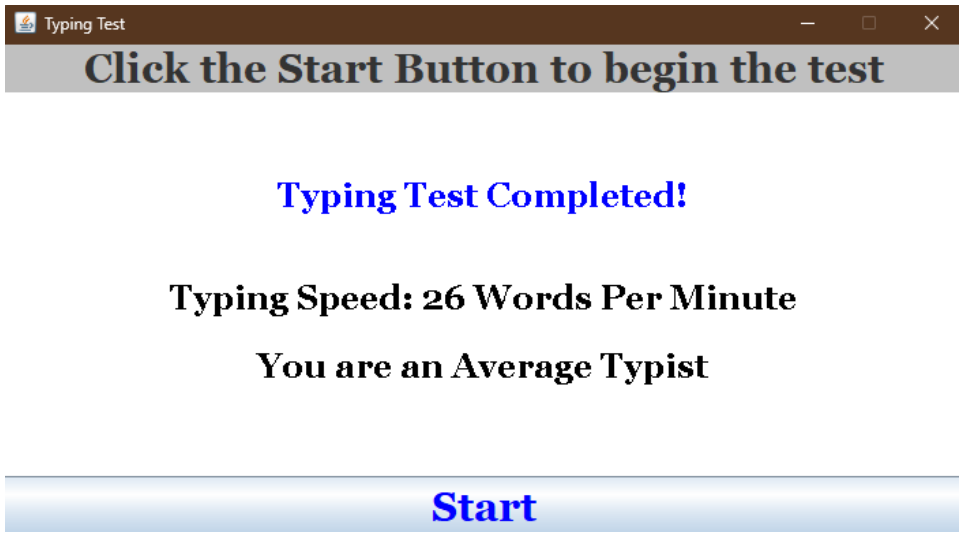

Fig: Test results

Conclusion

The "Typing Speed Test" Java project presents a valuable tool for individuals looking to improve their typing skills. By providing a user-friendly interface, engaging challenges and accurate performance metrics, the project contributes to improving users' typing speed and accuracy. Despite some limitations, the project's advantages outweigh its disadvantages, making it a useful and beneficial tool for individuals aiming to excel in today's digital communications landscape.

References

- **W3Schools Java Tutorial:** W3Schools also provides a Java tutorial section that covers basic and advanced Java concepts.
    - Website: https://www.w3schools.com
- **JavaTpoint:** JavaTpoint offers a comprehensive set of Java tutorials, examples, and exercises for beginners and intermediate learners. Their content is similar to W3Schools.
    - Website: https://www.javatpoint.com/
- **TutorialsPoint Java Tutorial:** TutorialsPoint provides tutorials and resources for various programming languages, including Java. Their Java tutorials cover a wide range of topics.
    - Website: https://www.tutorialspoint.com/java/index.htm
- **GeeksforGeeks Java:** GeeksforGeeks is known for its detailed programming tutorials and coding practice problems. Their Java section covers a broad spectrum of Java programming concepts.
    - Website: https://www.geeksforgeeks.org/java/
- **Programiz Java Tutorial:** Programiz offers tutorials and examples for programming languages, including Java. They provide interactive coding examples to help you understand concepts.
    - Website: https://www.programiz.com/java-programming
- **Java Code Geeks:** Java Code Geeks provides tutorials, articles, and examples related to Java programming and related technologies. The content is contributed by Java developers.
    - Website: https://www.javacodegeeks.com/
- **Baeldung:** Baeldung focuses on in-depth tutorials and articles related to Java, Spring Framework, and related technologies. The content is geared towards more advanced Java developers.
    - Website: https://www.baeldung.com/