# Playwright Java + Tesseract OCR – Keypad Automation

## ✅ Scenario Overview

You need to automate a keypad where digits are images rendered dynamically using base64-encoded <img> tags. The digits change positions and cannot be read from the DOM directly. OCR (Optical Character Recognition) using Tesseract is used to extract digits from images.

## 🧠 High-Level Approach

1. Extract all <img> elements for keypad buttons.  
2. Decode base64 image data.  
3. Save each image and run it through Tesseract OCR to extract the digit.  
4. Map digit → Locator (UI element).  
5. Click the appropriate buttons to enter the desired PIN.

## ✅ Sample Code – Java (Playwright + Tess4J)

import com.microsoft.playwright.\*;  
import net.sourceforge.tess4j.ITesseract;  
import net.sourceforge.tess4j.Tesseract;  
import org.apache.commons.io.FileUtils;  
  
import java.io.File;  
import java.util.\*;  
  
public class OCRKeypadAutomation {  
 public static void main(String[] args) throws Exception {  
 try (Playwright playwright = Playwright.create()) {  
 Browser browser = playwright.chromium().launch(new BrowserType.LaunchOptions().setHeadless(false));  
 Page page = browser.newPage();  
 page.navigate("https://your-app.com/keypad");  
  
 ITesseract tesseract = new Tesseract();  
 tesseract.setDatapath("C:/Program Files/Tesseract-OCR/tessdata");  
 tesseract.setLanguage("eng");  
 tesseract.setTessVariable("tessedit\_char\_whitelist", "0123456789");  
  
 String pin = "4932";  
 Locator allButtons = page.locator("img[class^='uia-pin-']");  
 int count = allButtons.count();  
  
 Map<String, Locator> digitToLocatorMap = new HashMap<>();  
  
 for (int i = 0; i < count; i++) {  
 Locator button = allButtons.nth(i);  
 String base64Src = button.getAttribute("src");  
 String base64 = base64Src.split(",")[1];  
 byte[] imageBytes = Base64.getDecoder().decode(base64);  
 File imageFile = new File("digit\_" + i + ".png");  
 FileUtils.writeByteArrayToFile(imageFile, imageBytes);  
  
 String detectedDigit = "";  
 try {  
 detectedDigit = tesseract.doOCR(imageFile).replaceAll("[^0-9]", "").trim();  
 } catch (Exception e) {  
 System.out.println("OCR failed for image " + i);  
 }  
  
 if (!detectedDigit.isEmpty() && !digitToLocatorMap.containsKey(detectedDigit)) {  
 digitToLocatorMap.put(detectedDigit, button);  
 System.out.println("Mapped digit '" + detectedDigit + "'");  
 }  
 imageFile.delete();  
 }  
  
 for (char ch : pin.toCharArray()) {  
 String digit = String.valueOf(ch);  
 Locator button = digitToLocatorMap.get(digit);  
 if (button != null) {  
 button.click();  
 System.out.println("Clicked digit: " + digit);  
 } else {  
 System.err.println("Digit " + digit + " not found via OCR!");  
 }  
 }  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
}

## 📦 Required Dependencies (Maven)

<dependency>  
 <groupId>net.sourceforge.tess4j</groupId>  
 <artifactId>tess4j</artifactId>  
 <version>5.4.0</version>  
</dependency>  
<dependency>  
 <groupId>commons-io</groupId>  
 <artifactId>commons-io</artifactId>  
 <version>2.11.0</version>  
</dependency>

## 🛠 Notes

- Ensure Tesseract is installed and path is set correctly.  
- You may need image preprocessing (grayscale, resize) for better OCR results.  
- PIN entry is simulated by matching extracted digits to image buttons.