Topics

Wednesday, April 2, 2025 12:48 AM

- Introduction to working with Captchas
- Handling Captchas
- Mini Project - Working with Captchas
- Best Practices when dealing with Captchas

Intro

Tuesday, April 1, 2025 11:22 PM

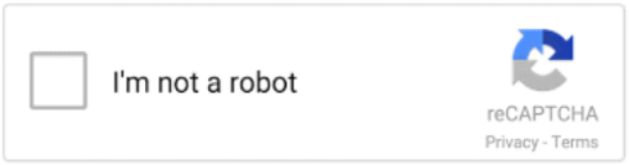
- CAPTCHAs (Completely Automated Public Turing tests to tell Computers and Humans Apart) are designed to prevent automated scripts from accessing websites

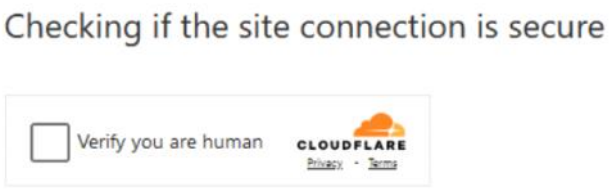
- Bypassing them is legally and ethically tricky, but understanding how to deal with them in a responsible way is crucial for web scraping

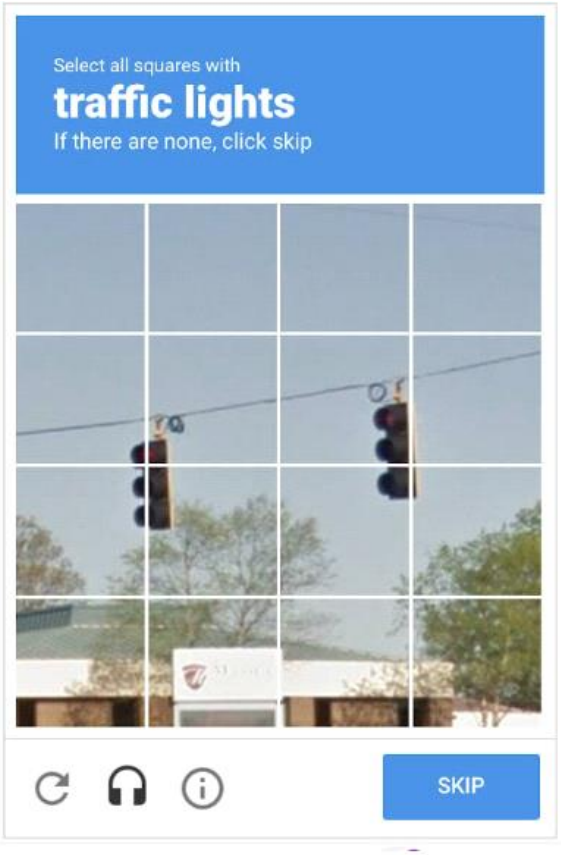
Understanding Captcha

Tuesday, April 1, 2025 11:29 PM

- **What is a CAPTCHA?**
  - A security mechanism to differentiate bots from humans
  - Examples: Text-based, Image-based (reCAPTCHA), Checkbox-based, Audio CAPTCHA, and Captcha







- **Why do websites use CAPTCHAs?**
    - Prevent spam, bot scraping, and DDoS attacks
    - Protect sensitive user data
    - Data Collection

Identification

Tuesday, April 1, 2025 11:47 PM

- Inspect network requests in **Developer Tools (F12) → Network Tab**
- Look for JavaScript files from **Google reCAPTCHA (gstatic.com, recaptcha.net)**
- Check response status codes (**403, 429 = CAPTCHA or rate limiting**)
- Simulate multiple requests and check for anomalies
- Check HTML for **form** elements like:

```
<div class="g-recaptcha" data-sitekey="your-site-key"></div>
```

Ethical & Legal Considerations

Tuesday, April 1, 2025 11:49 PM

Ethical Considerations:

- Always check the website's robots.txt file
- Do not overload a server with requests
- Scraping should be for legitimate and ethical purposes

Legal Risks:

- Violating Terms of Service could lead to legal action
- Unauthorized CAPTCHA bypassing may be considered hacking under CFAA (Computer Fraud and Abuse Act)
- Using third-party CAPTCHA-solving services may breach site policies

1. Prevent Triggering CAPTCHA

Tuesday, April 1, 2025 11:52 PM

- Slow down your requests using time.sleep(), random.uniform(), and request delays

- Use rotating User-Agents to simulate different browsers:

```
headers = {
    "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64)"
}
```

- Use Proxies and VPNs to change IP addresses

- Rotate Cookies and Sessions to prevent detection:

```
session = requests.Session()
session.get(url, headers=headers)
```

2. Solving CAPTCHA - Theory

Tuesday, April 1, 2025 11:56 PM

- Selenium to open the page and pause execution
  - fake_useragent: For User-agent rotation
  - undetected-chromedriver: Avoid detection by bot filters
- OCR to solve text-based captcha
  - pytesseract
  - EasyOCR
- Google's reCAPTCHA v2
  - SpeechRecognition
  - pydub
- OpenCV: Image matching / slider solving
- Paid tools (*free-tier or trial period*):
  - 2Captcha
  - AntiCaptcha
  - CapSolver
  - DeathByCaptcha

3. Solving CAPTCHA - Practical

Saturday, April 12, 2025 12:46 PM

1. **Using the input() function**

2. **Using pytesseract, OpenCV, PIL and NumPy**
   - Install tesseract-ocr using the link: https://github.com/UB-Mannheim/tesseract/wiki
   - Add the directory of the .exe file to Path variable
   - Install pytesseract, opencv libraries
   - Capture screenshot of webpage
   - Crop the image using PIL
   - Preprocess the cropped image using OpenCV
   - Extract text from pre-processed image using pytesseract
   - Continue interaction using Selenium

1. Prevent Triggering CAPTCHAs in the First Place

Wednesday, April 2, 2025 12:02 AM

1. Use Randomized Delays:

- Instead of bypassing CAPTCHAs, a better approach is to avoid triggering them.
- Sending too many requests too quickly can trigger CAPTCHAs
- Implement randomized delays between requests using Python's time.sleep()
- Use exponential backoff (gradually increasing delays) when facing CAPTCHAs

2. Rotate User-Agents to Mimic Real Browsers:

- Websites detect scrapers by checking the **User-Agent** header
- Use a pool of real browser User-Agent strings and rotate them

```
import random

user_agents = [
    "Mozilla/5.0 (Windows NT 10.0; Win64; x64)",
    "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)",
    "Mozilla/5.0 (X11; Linux x86_64)"
]

headers = {"User-Agent": random.choice(user_agents)}
```

- Avoid using outdated or suspicious user agents (e.g., "Python-requests/2.25.1")

3. Use Session Management for Persistent Requests:

- Create a session object to maintain cookies and headers across requests

```
import requests

session = requests.Session()
```

```
import requests

session = requests.Session()
session.headers.update({"User-Agent": random.choice(user_agents)})
response = session.get("https://example.com")
```

4. Mimic Human Behavior Using Headless Browsers:

- Instead of sending direct requests, simulate human-like interactions using Selenium
- Randomize mouse movements and scrolling to appear more human-like

```
from selenium import webdriver
import time

driver = webdriver.Chrome()
driver.get("https://example.com")

# Scroll the page slightly before interacting
driver.execute_script("window.scrollBy(0, 100)")
time.sleep(2)  # Small delay before next action
```

2. Handling CAPTCHAs the Right Way

Wednesday, April 2, 2025 12:23 AM

1. Manually Solve CAPTCHAs When Necessary:

- If scraping is infrequent, pause execution and solve the CAPTCHA manually
- In Python scripts, use the input function to pause execution; once solved the captcha, resume the execution

```
input("Solve the CAPTCHA manually and press Enter to continue...")
```

2. Use Browser Automation to Bypass Some CAPTCHAs:

- Selenium with undetected-chromedriver can sometimes avoid detection

```
import undetected_chromedriver.v2 as uc

driver = uc.Chrome()
driver.get("https://example.com")
```

3. Consider CAPTCHA Solving Services for Complex Cases:

- Paid CAPTCHA solvers like 2Captcha or Anti-Captcha use human workers or AI

3. Real-world Scraping Strategies

Wednesday, April 2, 2025 12:26 AM

SITUATION	RECOMMENDED APPROACH
Simple website with no CAPTCHA	Direct requests with requests library
CAPTCHA appears after multiple requests	Slow down, rotate IPs and User-Agents
Image/text CAPTCHA	Use OCR (Tesseract) or manual solving
Google reCAPTCHA v2	Use a solving service (e.g., 2Captcha)
Website bans IPs quickly	Use rotating proxies/VPNs
Website provides an API	Use the official API instead of scraping

4. Final Takeaways

Wednesday, April 2, 2025 12:28 AM

- Prevent triggering CAPTCHAs with proper request handling
  - Introduce time.sleep() or randomized delays between requests to avoid being flagged as a bot
- Use ethical techniques (rotating proxies, user-agents, and session management)
  - Use Selenium or Playwright to interact with JavaScript-heavy pages
  - Always wait for elements to load using explicit waits
- Manually solve CAPTCHAs when needed instead of excessive automation
- Use AI-based OCR for simple CAPTCHAs and solving services for complex ones
- Respect website rules (robots.txt) and avoid legal violations
  - Don’t scrape login-protected or private data without explicit permission