УДК 004

Создание CAPTCHA на языке программирования Python

Кизянов Антон Олегович
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
Создание картинок для борьбы с ботами, другими словами CAPTCHA с помощью языка программирования Python.
Ключевые слова: Python, PIL

Creating a CAPTCHA in the Python programming language

Kizyanov Anton Olegovich
Sholom-Aleichem Priamursky State University
student

Abstract
Create images to combat bots, in other words CAPTCHA using the Python programming language.
Keywords: Python, PIL

CAPTCHA означает полностью автоматизированный публичный тест Тьюринга, чтобы различить компьютер и человека. Этот тест используется для затруднения компьютерных программ (обычно называемых ботами), которые автоматически заполняют различные веб-формы, которые в первую очередь нацелены на людей, и которые не должны быть автоматизированы. Обычными примерами является регистрационные формы, формы входа, опросы и т.д.

Цель исследования – это написание программы для генерации различных изображений в виде CAPTCHA на языке программирования Python.

Ранее этим вопросом интересовались А.В.Яковлев, А.М.Шуваева, М.А.Пеливан с темой «Противодействие атаке brute force на основе графических объектов типа CAPTCHA» [1], а подробнее рассмотрены виды тестов Captcha, их сложность для прохождения при помощи автоматизированных средств, а также метод использования тестов Captcha для увеличения стойкости парольной аутентификации против атаки грубой силы. О.Н. Жданов развивал тему «Распознавание современных CAPTCHA» [2] в которой рассказывается проблема автоматизированного распознавания современной captcha. Для решения этой проблемы рассматриваются три подзадачи - предварительная обработка, сегментация изображения и распознавание отдельных элементов. М.П.Явич, И.З.Пирцхалава

опубликовали статью «Проблемы развития CAPTCHA» [3] рассказали про разные варианты CAPTCHA. В связи с развитием ботов рассматриваются пути усложнения CAPTCHA.

Сама CAPTCHA может принимать различные формы, но наиболее распространенная форма состоит из задачи, когда человек должен прочитать изображение с искаженными символами и цифрами и ввести результат в соответствующее поле ответа.

Мы создадим собственный генератор CAPTCHA следуя следующим шагам.

- Определить размер, текст, размер шрифта, цвет фона и длину CAPTCHA.
- Выбрать случайные символы из английского алфавита.
- Нарисовать их на изображении используя определенные шрифты и цвета.
- Добавить некоторый шум в виде линий и точек.
- Показать сгенерированное изображение пользователю.

Следующий код показывает, как создать персональный и простой генератор CAPTCHA.

```
from PIL import Image as Im, ImageDraw as Img, ImageFont as Imf
from random import randint as r_int
import random as rd
import string as st

class SimpleException(Exception):
   pass

class Captcha():
   def __init__(slf, len=5, sz=(200, 100), fsize=36, rand_text=None,
rand_bgcolor=None):
      slf.sz = sz
      slf.tx = "Капча"
      slf.fsize = fsize
      slf.bgc = 255
      slf.len = len
      slf.img = None
      if rand_text:
         slf.tx = slf._rand_text()
      if not slf.tx:
         raise SimpleException("Поля должны быть не пустыми")
      if not slf.sz:
         raise SimpleException("Размер должен быть не пустым")
      if not slf.fsize:
```

```
            raise SimpleException("Размер шрифта должен быть не
пустым")
        if rand_bgcolor:
            slf.bgc = slf._rand_color()

    def center(slf, drw, ft):
        wid, hei = drw.textsize(slf.tx, ft)
        x_y = (slf.sz[0] - wid) / 2., (slf.sz[1] - hei) / 2.
        return x_y

    def noisedot(slf, drw):
        s_z = slf.image.size
        for _ in range(int(s_z[0] * s_z[1] * 0.1)):
            drw.point((r_int(0, s_z[0]),
                       r_int(0, s_z[1])),
                       fill="white")
        return drw

    def noiselin(slf, drw):
        s_z = slf.image.size
        for _ in range(8):
            wid = r_int(1, 2)
            stat = (0, r_int(0, s_z[1] - 1))
            en = (s_z[0], r_int(0, s_z[1] - 1))
            drw.line([stat, en], fill="white", width=wid)

        for _ in range(8):
            stat = (-50, -50)
            en = (s_z[0] + 10, r_int(0, s_z[1] + 10))
            drw.arc(stat + en, 0, 360, fill="white")
        return drw

    def get(slf, sz=None, tx=None, bgc=None):
        if tx is not None:
            slf.tx = tx
        if sz is not None:
            slf.sz = sz
        if bgc is not None:
            slf.bgc = bgc

        slf.image = Im.new('RGB', slf.sz, slf.bgc)
        fn = Imf.truetype('arial.ttf', slf.fsize)
        drw = Img.Draw(slf.image)
        x_y = slf.center(drw, fn)
        drw.text(xy=x_y, text=slf.tx, font=fn)
```

```
            drw = slf.noisedot(drw)
            drw = slf.noiselin(drw)
            slf.image.show()
            return slf.image, slf.tx

        def _rand_text(slf):
            letter = st.ascii_lowercase + st.ascii_uppercase
            rand_text = ""
            tx = rand_text
            for _ in range(slf.len):
                tx += rd.choice(letter)
            return tx

        def _rand_color(slf):
            r_c = r_int(0, 255)
            g_c = r_int(0, 255)
            b_c = r_int(0, 255)
            return (r_c, g_c, b_c)
    if __name__ == "__main__":
        s_c = Captcha(len=7, fsize=36, rand_text=True, rand_bgcolor=True)
        s_c.get()
```

Этот код дает следующее изображение Рис. 1.

Рис. 1

Вывод

Таким образом, мы можем генерировать неограниченное количество картинок с CAPTCHA’ей для создания преграды для ботов.

Библиографический список

1. Яковлев А.В., Шуваева А.М., Пеливан М.А. Противодействие атаке brute force на основе графических объектов типа CAPTCHA // Приборы и системы. управление, контроль, диагностика. 2015. №7. С. 15-19.
2. Жданов О.Н. Распознавание современных CAPTCHA // Научный вестник Воронежского государственного архитектурно-строительного университета. Серия: студент и наука. 2014. С. 221-224.
3. Явич М.П. Пирцхалава И.З. Проблемы развития CAPTCHA // Современная техника и технологии. 2015. №7 С. 92-93.