Анализ методов формирования и проверки штрих-кодов

К.С. Шарая, e-mail: kristina.alexa.sh@gmail.com
С.Н. Девицына

ФГАОУ ВО «Севастопольский государственный университет»

Аннотация. *Проведен анализ существующих решений для работы со штрих-кодами, выявлены сильные и слабые стороны выбранных проектов, определено время, затрачиваемое на их выполнение, сделаны выводы об эффективности каждого метода.*

Ключевые слова: *информационная система, технологии штрихового кодирования, штрих-коды, системы автоматической идентификации.*

Введение

Штрих–код представляет собой графическое изображение, наносящееся на какую-либо поверхность. У каждого кода — свой уникальный рисунок, в котором зашифрованы числовые и текстовые данные. Проще и надежнее считывать данные, используя их двоичного кодирование. Считывая информацию слева направо, сканер присваивает 1 первой встреченной черной полоске и 0 — первому промежутку. Следующие промежутки и штрихи считываются как последовательности нулей или единиц, в зависимости от их ширины. Следовательно, все изображение может быть представлено как последовательность битов [1]. Другими словами, штрих-код — это метод представления данных в машиночитаемой форме [2], символьный ключ к информации в базах данных [3].

Выбор используемого штрих-кода зависит от вида данных, которые необходимо записать — цифры или буквы. Исходя от этих критериев одномерные и двухмерные штрих-коды используют в разных сферах, а для их считывания необходимо разное оборудование.

Развитие систем штрихового кодирования даёт возможность решить одну из самых сложных проблем компьютерной техники – ввод данных, - почти полностью исключив ошибки, т.к. ЭВМ легче считывает широкие и узкие штрихи и промежутки между ними, чем буквы и цифры.

Цель данного исследования заключается в поиске не ресурсоемкого, но эффективного программного обеспечения, которое подойдет для самостоятельной разработки штрих-кодов для внутренних

© Шарая К.С., Девицына С.Н., 2022

целей организации. Тем не менее, рассмотренные программы лежат в основе более сложных и профессиональных, используемых крупными фирмами, специализирующимися на их создании и распространении.

1. Методы обработки и формирования штрих-кодов

Появление систем автоматической идентификации значительно увеличило скорость, эффективность и точность обработки и сбора данных. Для генерации и печати штрих-кодов используют специальные принтеры, а для чтения — сканеры, что значительно снижает вероятность человеческой ошибки. Человек тратит около шести минут на ввод двенадцати позиций, при этом в среднем будет одна ошибка на каждые 300 позиций, машина тратит около 300 миллисекунд на считывание двенадцати позиций, при этом может допустить одну ошибку на миллион. Ознакомившись с теоретическими основами штрихового кодирования, его видами и применением, рассмотрим методы и способы формирования и сканирования штрих-кодов.

Большинство решений, существующих на рынке, позволяет получить растровое изображение штрих-кода (BMP, JPG, TIFF), в то время как оптимальным является векторное (EPS, AI) [4]. Продукты, предлагаемые на рынке, делятся на бесплатные и платные. К бесплатным относятся веб-ресурсы, позволяющие в режиме реального времени получить файл. Минусом веб-решений является необходимость постоянного доступа к Интернету, с одной стороны, и работы самого сайта, с другой.

Платное решение — это, как правило, DLL-библиотеки. Данная реализация имеет два недостатка: во-первых, необходимо заплатить некую сумму денег при покупке продукта; во-вторых, почти все библиотеки генерируют код только в растровом формате.

Но самым лучшим решением для обработки штриховых кодов стали библиотеки SDK, которых существует огромное количество для разных языков программирования, после установки они не зависят от Интернет-подключения и в большинстве своем бесплатны.

Проанализированные программные продукты в значительной степени упрощают понимание обработки штриховых кодов, работы с ними, их генерации, считывания и декодирования. Работа этих проектов основана на использовании технологий SDK штрих-кодов.

Потребность в SDK значительно возросла. Сочетание технологии штрихового кодирования и библиотеки штрих-кодов позволяет с низкими затратами реализовать любое приложение автоматической обработки любого вида такого кодирования информации, больше нет

необходимости в приобретении дорогостоящего специализированного однофункционального устройства для этих целей.

Проведя исследование и анализ проектов в веб-сервисе GitHub, можно отметить достаточное количество программного обеспечения на данную тематику на разных языках программирования и для разных целей применения. Каждый язык имеет свои методы, способы, ресурсы и возможности для реализации штрихового кодирования, которые зачастую являются дополнительными модулями, требующими предварительной установки.

2. Практическая реализация формирования и обработки штрих-кодов

Для исследования практической реализации работы со штрих-кодами на веб-сервисах для размещения IT-проектов — GitLab и GitHub была произведена фильтрация по запросу «barcode», в результате которой на первом сервисе было выявлено не так много соответствующих проектов, на GitHub же на 12.12.2021 был получен результат из 19797 репозиториев и 393333 коммитов. Самыми популярными языками программирования для кодов на данную тематику оказались: Java — 3974 проектов, JavaScript — 3028 проектов, Python — 1908 проектов. Именно из последних было выделено 3 простых кода, которые являются основой и базой для многих представленных на языке Python программ.

При анализе было обнаружено, что практически все они основываются на использовании библиотеки python-barcode, которая в своем составе имеет большое количество функций, обрабатывающих разные виды графических меток, и библиотеки OpenCV, которая используется для считывания и обработки изображений. Тем не менее, эти способы не единственные, и существует еще множество программных решений обработки штриховых кодов.

Для исследования практической реализации работы со штрих-кодами с помощью онлайн среды для разработки Google Colab выбранные три проекта были доработаны, исправлены некоторые недочеты, и были получены данные об эффективности их выполнения и сбора.

Barcode-Generation-using-Python [5]. Данная программа предназначена для генерирования штрих-кодов. В ней создается код «EAN13» (European Article Number) — европейский стандарт штрихового кода из 13 знаков. Текст программы Barcode-Generation-using-Python отображен на рис. 1.

```python
# import EAN13 from barcode module
from barcode import EAN13
# import ImageWriter to generate an image file
from barcode.writer import ImageWriter

def generator(number):
    # Let us create an object of EAN13 class and pass the number with the ImageWriter() as the writer
    my_code = EAN13(number,writer=ImageWriter())
    my_code.save("bar_code")

if __name__ == "__main__":
    # Make sure to pass the number as string
    generator(input('Enter 12 Digit Number To Generate Bar Code:'))
```

В программу в качестве строки передаются 12 цифр, вводимых пользователем с клавиатуры, по ним вычисляется контрольная цифра и из которых в дальнейшем и генерируется штрих-код. В процессе ее выполнения создается рисунок на основе заложенной в рассматриваемую функцию таблицы кодирования, в которой приведены цифры, их двоичное представление и соответствующие ему вертикальные линии кодов. Результатом выполнения является изображение с расширением «.png», которое сохраняется на используемое устройство (рис. 2).

Рис. 2. Результат выполнения программы Barcode-Generation-using-Python (EAN13 штрих-код)

В данном случае задача генерации штрих-кода реализуется подключением библиотеки python-barcode, а именно ее элементов EAN13 из barcode и ImageWriter из barcode.writer, который создает png файл с изображением. Этот модуль не является стандартным, и для его использования требуется предварительное подключение.

По времени, оцененному в Colab, с помощью команды %%time исполнение этой программы и генерация такого штрих-кода занимает около 7 миллисекунд.

Таким образом, библиотека Barcode позволяет быстро и без больших системных затрат создавать штриховые коды разных видов и получать их растровые изображения. Тем не менее у нее есть и

недостаток — начальный и конечный символы не прорисовываются, и вид таких штрих-кодов немного отличается от стандартного.

Вторая программа — igu-odoo [6] — по своей сути выполняет идентичную задачу, она создает EAN13 код, но уже с использованием других средств Python. Взятый с GitHub проект igu-odoo предоставляет решения для создания следующих штрих-кодов: Code39, Code93, Code128, EAN8, EAN13, QR; одним кодом, однако, необходимый для рассмотрения фрагмент программы представлен на рис. 3.

```python
from reportlab.graphics.barcode import eanbc
from reportlab.graphics.shapes import Drawing
from reportlab.lib.pagesizes import letter
from reportlab.lib.units import mm
from reportlab.pdfgen import canvas
from reportlab.graphics import renderPDF

def createBarCodes():
    """
    Create barcode examples and embed in a PDF
    """
    c = canvas.Canvas("barcode.pdf", pagesize=letter)
    barcode_value = "1234567890"

    # draw the eanbc13 code
    barcode_eanbc13 = eanbc.Ean13BarcodeWidget(barcode_value)
    bounds = barcode_eanbc13.getBounds()
    width = bounds[2] - bounds[0]
    height = bounds[3] - bounds[1]
    d = Drawing(150, 50)
    d.add(barcode_eanbc13)
    renderPDF.draw(d, c, 30, 700)
    c.save()

if __name__ == "__main__":
    createBarCodes()
```

Рис. 3. Фрагмент текста программы igu-odoo

В данном случае номер штрихового кода не вводится пользователем, а прописан непосредственно в коде, тем не менее при необходимости это легко изменить, не влияя на вычислительный процесс. Вместо библиотеки barcode здесь используется библиотека reportlab, позволяющая создавать документы непосредственно в формате PDF с использованием языка программирования Python, создавать

диаграммы и графику в различных растровых и векторных форматах, и в частности модуль reportlab.graphics.barcode, который работает непосредственно со штрих-кодами. Эта библиотека так же не является стандартной, и для его использования требуется предварительное подключение. Полученным результатом будет файл в формате «pdf» с изображением сгенерированного кода внутри. Он отображен на рис. 4.

Рис. 4. Результат выполнения программы igu-odoo

По времени, оцененному в Colab, исполнение этой программы и генерация такого штрих-кода занимает примерно 6 миллисекунд. Проанализировав работу программы и изучив Reportlab, можно сделать вывод о том, что для генерирования штриховых изображений таким способом необходимо провести большее количество операций и команд, по сравнению с предыдущим, тем не менее скорость выполнения программы, наоборот, немного больше. Такие штрих-коды имеют более стандартный вид. Так же из плюсов можно выделить возможность создания разного количества кодов разных видов в одном файле, а из минусов — необходимость расчета координат расположения изображения внутри документа.

Последняя программа — Barcode-Reader [7] — представляет собой реализацию алгоритма считывания штрих-кода с изображения. Сам код отображен на рис. 5.

```
from pyzbar import pyzbar
import cv2

image = cv2.imread('/content/bar_code.png')
barcodes = pyzbar.decode(image)
for barcode in barcodes:
    (x, y, w, h) = barcode.rect
    cv2.rectangle(image, (x, y), (x + w, y + h), (255, 0, 0), 2)

    barcodeData = barcode.data.decode('utf-8')
    barcodeType = barcode.type
    text = "{} ( {} )".format(barcodeData, barcodeType)
    cv2.putText(image, text, (x, y - 10), cv2.FONT_HERSHEY_COMPLEX, 0.5, (255, 0, 0), 2)

    print("Information : \n Found Type : {} Barcode : {}".format(barcodeType, barcodeData))

cv2.imshow("Image", image)
cv2.waitKey(0)
```

Рис. 5. Текст программы Barcode-Reader

В нее передается изображение, которое декодируется и обрабатывается попиксельно. При завершении выводится строка, в которой записаны определенные в ходе выполнения параметры: вид штрих-кода и считанные с него символы. Работа программы осуществляется с помощью подключения библиотек pyzbar, которая считывает одномерные штрих-коды и QR-коды, и OpenCV — библиотека компьютерного зрения, которая предназначена для анализа, классификации и обработки изображений. Чтобы получить результаты исполнения данной программы, а также проверить ее на достоверность, код был запущен с результатом работы проекта «Barcode-Generation-using-Python». Результат обработки приведен на рис. 6.

```
Information :
 Found Type : EAN13 Barcode : 0569322104877
```

Рис. 6. Результат выполнения программы Barcode-Reader (считывание EAN13 штрих-кода)

Полученные результаты соответствуют действительности, таким образом можно сделать вывод о том, что библиотека pyzbar позволяет написать компактную по количеству строк, но эффективную программу, занимающую в среднем 5,5 миллисекунд, для считывания графических меток. А полученный номер штрих-кода может использоваться системой для доступа к ячейкам с данными в соответствующих базах данных.

Заключение

Для исследования основных принципов работы библиотек штрих-кодов были выбраны три проекта на одном из самых распространенных

и быстро развивающихся языков программирования Python. Эти программы отразили три разных библиотеки, обеспечивающих штриховое кодирование. Был проведен анализ существующих решений для работы со штрих-кодами, выявлены сильные и слабые стороны выбранных проектов, получено время, затрачиваемое на их выполнение, откуда можно сделать вывод об эффективности каждого из них.

Внедрение штрихового кодирования максимально увеличит скорость документооборота, снизит количество неточностей при вводе и обработке данных и минимизирует ошибки считывания. Плюсы штрих-кодирования неоспоримы, и повсеместное использование штриховых меток и их дальнейшее внедрение во все сферы жизни является подтверждением того, что и программным средствам необходимо быть на одном уровне с растущими потребностями общества: скорость считывания штрих-кодов должна возрастать, как и плотность кодируемой в них информации.

Литература

1. Златопольский, Д.М. Штрих-код / Д.М. Златопольский // Вестник Московского университета. Серия 20: Педагогическое образование. – 2011. – № 3. – С. 114-117.
2. Waksoft: С Python прочитаем любые штрих- и QR□коды [Электронный ресурс] Режим доступа: https://waksoft.susu.ru/2021/05/03/kak-sgenerirovat-i-prochitat-qr-kod-v-python/ (Дата обращения: 26.11.2021).
3. Макшанова, К. А. Технология применения штрих-кода / К. А. Макшанова, В. Г. Ежкова // Студенческая наука Подмосковью: Материалы Международной научной конференции молодых ученых, Орехово-Зуево, 05–06 апреля 2018 года. – Орехово-Зуево: Государственный гуманитарно-технологический университет, 2018. – С. 349-352.
4. Коробко, И. Генерация штрих-кода EAN-13 (ISBN) / И. Коробко // Системный администратор. – 2012. – № 5(114). – С. 74-78.
5. GitHub: Barcode-Generation-using-Python [Электронный ресурс] Режим доступа: Barcode-Generation-using-Python/main.py at main · AbhishekMankame/Barcode-Generation-using-Python · GitHub (Дата обращения: 16.12.2021).
6. GitHub: igu-odoo [Электронный ресурс] Режим доступа: https://github.com/andyut/igu-odoo/blob/8977348026341793a02a4a26ebda59d9a80b4c7d/odoo/testBarcode.py (Дата обращения: 16.12.2021).
7. GitHub: igu-odoo [Электронный ресурс] Режим доступа: https://github.com/andyut/igu-

odoo/blob/8977348026341793a02a4a26ebda59d9a80b4c7d/odoo/testBarcode.py (Дата обращения: 16.12.2021).