Международная конференция
«Информатика: проблемы, методы, технологии»
Секция: Технологии обработки и защиты информации

АНАЛИЗ МЕТОДОВ ФОРМИРОВАНИЯ И ПРОВЕРКИ ШТРИХ-КОДОВ

Шарая Кристина Сергеевна, студент (2 курс)
Девицына Светлана Николаевна, к.т.н., доцент

ФГАОУ ВО «Севастопольский государственный университет»
Институт радиоэлектроники и информационной безопасности,
кафедра «Информационная безопасность»

Воронеж, 2022

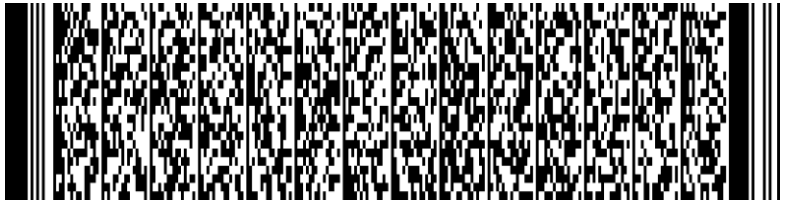
Актуальность

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 целей организации.

Рассмотренные программы лежат в основе более сложных и профессиональных, используемых крупными фирмами, специализирующимися на их создании и распространении.

Применение штрихового кодирования



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

Отдельные стандарты шифрования связаны с определенными областями:

- международный стандарт кодирования *EAN* применяется во всех сферах торговли;
- разновидности *EAN*-кодов используются для маркировки в конкретной сфере: *ISBN* предназначен для книг, *ISSN* — для периодической печатной продукции;
- *DataMatrix* используется для маркировки компьютерной техники, а также в авто- и авиа-промышленности;
- *QR*-код распространен широко, но основная сфера использования — маркетинг;
- код *Aztec* распространен на почте и в логистике, в особенности — в сфере авиационных и железнодорожных перевозок.

Существующие методы получения штрих-кода

Три основных метода формирования штрих-кода:

1) создание растровой картинки (BMP или PNG);

2) создание векторной картинки (WMF);

3) непосредственно внедрение в документ набора прямоугольников.

Большинство решений, существующих на рынке, позволяет получить растровое изображение штрих-кода (BMP, JPG), в то время как оптимальным является векторное (EPS, AI).

Практическая реализация формирования и обработки штрих-кодов

Barcode-Generation-using-Python. Данная программа предназначена для генерирования EAN13 штрих-кодов. Это реализуется подключением библиотеки python-barcode, а именно ее элементов EAN13 из barcode.

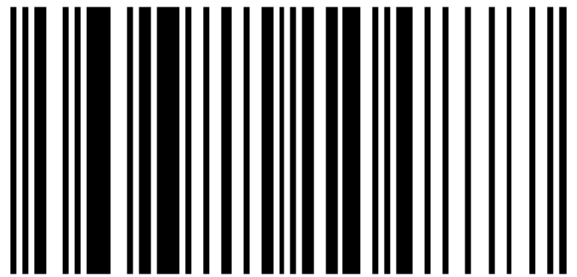
Текст программы *Barcode-Generation-using-Python*

```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:'))
```

Результат выполнения программы *Barcode-Generation-using-Python* (*EAN13* штрих-код)



Практическая реализация формирования и обработки штрих-кодов

igu-odoo — создает *EAN13* код, но уже с использованием других средств *Python,* библиотеки *reportlab* и её модуля *reportlab.graphics.barcode.*

Результат выполнения программы igu-odoo


```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()
```

Текст программы igu-odoo

Практическая реализация формирования и обработки штрих-кодов

- Barcode-Reader — представляет собой реализацию алгоритма считывания штрих-кода с изображения.

Текст программы Barcode-Reader

```
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)
```

Результат выполнения программы *Barcode-Reader* (считывание *EAN13* штрих-кода)

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

Полученные результаты

Для исследования основных принципов работы библиотек штрих-кодов выбраны три проекта на одном из самых распространенных и быстро развивающихся языков программирования Python. Эти программы отразили три разных библиотеки, обеспечивающих штриховое кодирование.

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.

БЛАГОДАРЮ ЗА ВНИМАНИЕ!