УДК 004

Разработка генератора паролей на C#

Ульянов Егор Андреевич
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
В данной статье рассматривается и описывается разработка генератора паролей. Приложение разрабатываться на языке программирования C# с помощью IDE Visual Studio. Практическим результатом является разработанный генератор паролей.
Ключевые слова: генератор паролей, C#, интернет безопасность

Developing a password generator on C

Ulianov Egor Andreevich
Sholom-Aleichem Priamursky State University
Student

Abstract
This article discusses and describes the development of a password generator. The application is developed in the C # programming language using the Visual Studio IDE. The practical result is a developed password generator.
Keywords: password generator, C#, internet security

Генератор паролей - специальная программа или интернет-страница, которая случайным образом по заданным параметрам создает пароли для пользователей. Используются для создания наиболее сложных паролей, которые могут включать латинские буквы, цифры и символы. Существует множество интернет-сайтов предлагающих генерацию паролей. Также на просторах сети Интернет можно найти и программы для генерирования паролей. Существует также программы генераторы паролей для мобильных операционных систем Android и iOS. Основным плюсом большинства генераторов является большая сложность паролей, что помогает пользователям не беспокоиться о безопасности своих мобильных устройств, или данных в сети.

Цель данной статьи создать генератор паролей в среде разработки Visual Studio на языке программирования C#.

В своей работе Н.Н.Додобоев, О.И.Кукарцева, Я.А.Тынченко рассмотрели вопросы появления различных языков программирования (в частности C#), определения особенностей этих языков, а также составления основных видов и классификаций языков программирования [1]. Е.В.Карачанская, М.А.Костров представили компьютерную программу -

многофункциональный генератор паролей, реализующий надежные пароли, а также способный к оценке их надежности [2]. В статье Н.А.Картушева, А.А.Морозовой описывается принцип работы генераторов случайных паролей, природа случайных данных, различия между генераторами случайных чисел [3].

Пароль должен соответствовать минимум трем из четырех правил безопасности паролей:

- не менее 1 символа верхнего регистра (A-Z);
- минимум 1 строчный символ (a-z);
- минимум 1 цифра (0-9);
- как минимум 1 специальный символ (пунктуация) - пробел считается специальным символом;
- не менее 8 символов;
- максимум 128 символов;
- не более двух одинаковых символов подряд (например, 111 не допускается).

Создаем проект и называем его. Для начала реализуем единый метод, в который передадим генератора (параметры будут храниться в конструкторе), а на выходе будем получать пароль. Также понадобиться список доступных символов для случайного подбора пароля см. рисунок 1.

```
namespace CodeShare.Library.Passwords
{
    Ссылок: 0
    public static class PasswordGenerator
    {
        /// <summary>
        /// Generates a random password based on the rules passed in the parameters
        /// </summary>
        /// <param name="includeLowercase">Bool to say if lowercase are required</param>
        /// <param name="includeUppercase">Bool to say if uppercase are required</param>
        /// <param name="includeNumeric">Bool to say if numerics are required</param>
        /// <param name="includeSpecial">Bool to say if special characters are required</param>
        /// <param name="includeSpaces">Bool to say if spaces are required</param>
        /// <param name="lengthOfPassword">Length of password required. Should be between 8 and 128</param>
        /// <returns></returns>
        Ссылок: 0
        public static string GeneratePassword(bool includeLowercase, bool includeUppercase, bool includeNumeric, bool includeSpecial, bool includeSpaces, int lengthOfPassword)
        {
            return "not random";
        }

    }
}
```

Рис. 1. Создание метода

Создаем набор строковых переменных с символами на основе параметров в конструкторе см. рисунок 2.

```
{
    const int MAXIMUM_IDENTICAL_CONSECUTIVE_CHARS = 2;
    const string LOWERCASE_CHARACTERS = "abcdefghijklmnopqrstuvwxyz";
    const string UPPERCASE_CHARACTERS = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
    const string NUMERIC_CHARACTERS = "0123456789";
    const string SPECIAL_CHARACTERS = @"!#$%&*@\";
    const string SPACE_CHARACTER = " ";
    const int PASSWORD_LENGTH_MIN = 8;
    const int PASSWORD_LENGTH_MAX = 128;

    if (lengthOfPassword < PASSWORD_LENGTH_MIN || lengthOfPassword > PASSWORD_LENGTH_MAX)
    {
        return "Password length must be between 8 and 128.";
    }

    string characterSet = "";

    if (includeLowercase)
    {
        characterSet += LOWERCASE_CHARACTERS;
    }

    if (includeUppercase)
    {
        characterSet += UPPERCASE_CHARACTERS;
    }

    if (includeNumeric)
    {
        characterSet += NUMERIC_CHARACTERS;
    }

    if (includeSpecial)
    {
        characterSet += SPECIAL_CHARACTERS;
    }

    if (includeSpaces)
    {
        characterSet += SPACE_CHARACTER;
    }

    return "not random";
}
```

Рис. 2. Создание набора символов

Это дает строковую переменную под названием «characterSet». Теперь нужно написать часть кода, которая случайным образом выбирает символы из этого набора и создает пароль, для этого создадим переменную «random» см. рисунок 3.

```
        char[] password = new char[lengthOfPassword];
        int characterSetLength = characterSet.Length;

        System.Random random = new System.Random();
        for (int characterPosition = 0; characterPosition < lengthOfPassword; characterPosition++)
        {
            password[characterPosition] = characterSet[random.Next(characterSetLength - 1)];
        }

        return string.Join(null, password);
    }
}
}
```

Рис. 3. Реализация случайного выбора символа

Далее передаем в программу длину набора символов, что бы выбиралась случайная позиция в наборе символов. Одно из правил безопасности состоит в том, что не должно использоваться более двух одинаковых символов в ближайших позициях. Поэтому следующая часть кода состоит в том, чтобы добавить проверку совпадений и сгенерировать другой символ см. рисунок 4.

```
            char[] password = new char[lengthOfPassword];
            int characterSetLength = characterSet.Length;

            System.Random random = new System.Random();
            for (int characterPosition = 0; characterPosition < lengthOfPassword; characterPosition++)
            {
                password[characterPosition] = characterSet[random.Next(characterSetLength - 1)];

                bool moreThanTwoIdenticalInARow =
                    characterPosition > MAXIMUM_IDENTICAL_CONSECUTIVE_CHARS
                    && password[characterPosition] == password[characterPosition - 1]
                    && password[characterPosition - 1] == password[characterPosition - 2];

                if (moreThanTwoIdenticalInARow)
                {
                    characterPosition--;
                }
            }

            return string.Join(null, password);
        }
    }
}
```

Рис. 4. Улучшение кода

Далее можно проверить работу генератора паролей см. рис 5.

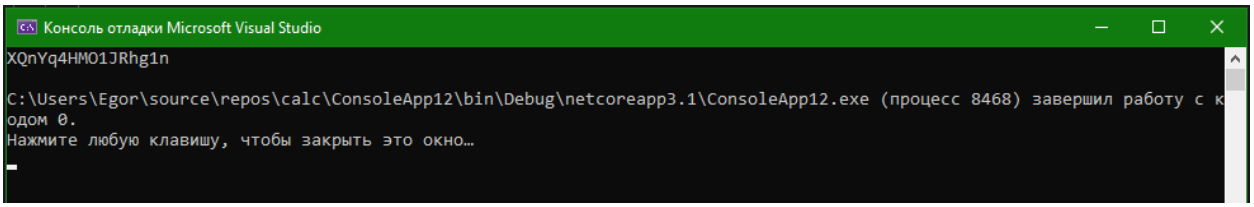

Рис.5. Готовый генератор паролей

В данной статье были проанализированы существующие аналоги и методы разработки, а также выбрана среда разработки. Для реализации поставленной задачи отлично подошла разработка с помощью Visual Studio и языка программирования C#. Такой выбор заметно упростил разработку проекта, так как в интернете имеется достаточное кол-во документации. Во время разработки был полученный ценный опыт работы с этим средством разработки.

В итоге был разработан генератор паролей. Данное консольное приложение имеет потенциал к развитию, например, добавление новых функций.

Библиографический список

1. Додобоев Н. Н., Кукарцева О. И., Тынченко Я. А. Современные языки программирования // Современные технологии: актуальные вопросы, достижения и инновации. 2014. №5. С. 81-85.
2. Карачанская Е. В., Костров М. А. Многофункциональный генератор паролей // Вопросы защиты информации. 2019. №2. С. 43-46.
3. Картушев Н.А., Морозова А.А. Принцип работы генератора случайных паролей // Теория и практика современной науки. 2018. №4. С. 118-123.