# Практически проект: Генератор на случайни изречения

A picture containing text, outdoor, case, accessory

Description automatically generated

Този генератор е просто за забавление. Изреченията може да бъдат хумористични и да предоставят готин начин да изненадате другите, като споделите отличаващ се текст в социалните мрежи.

Graphical user interface, text

Description automatically generated

Text

Description automatically generated

## Създаване на GitHub репо

Създайте **ново репо** от <https://github.com/new>. Изберете **смислено име**, например "RandomSentencesGeneratorBy{вашето потребителско име}", добавете **кратко описание** и направете репото **публично**. **Добавете** README.md **файл** и .gitignore за **Visual Studio**. **Променете лиценза** на "MIT" и кликнете бутона[Create] за да **създадете репото**.

|  |  |
| --- | --- |
| Icon  Description automatically generated | Моля изберете **оригинално име** за вашия проект!  Вашият GitHub профил трябва да бъде **уникален**, а не същият като тези на съучениците ви.  Можете да следвате този документ, но е позволено да **правите промени** и да **добавяте свои функционалности**. |

Сега вашето **репо е създадено** и трябва да изглежда по подобен начин:

Graphical user interface, text, application, email

Description automatically generated

## Код за генератора на изречения

### Създаване на проект във Visual Studio

Първо, трябва да **стартираме Visual Studio** и да **създадем нов C# console application**. След това трябва да **изберем подходящо име** и **място за съхранение** на проекта. На следващия екран изберете [.NET 6 (Long-term support)] и създайте проекта.

Graphical user interface, text, application, email

Description automatically generated

Before we continue, let's change the **name** of our main class to something more **meaningful**:

Graphical user interface, text, email

Description automatically generated

### Имплементиране на логиката на играта

#### Създаване на модел за изречение

За да създадем нашите **изречения**, ще са ни необходими: **имена**, **места**, **глаголи**, **съществителни имена**, **наречия** и **детайли**. Изречението, което ще създадем, ще е базирано на следния **модел**:

* **[Кой откъде] [действие] [детайл]**
  + Пример за **„Кой и откъде“**: **[име** +от + **място]** ("David from London").
  + Пример за **„Действие“**: **[наречие]** + **[глагол]** + **[съществително]** ("calmly watched the sunset").
  + Пример за **„Детайл“**: "near the river", "at home", "in the park".

#### Добавяне на думи за изреченията

Нека да започнем, като създадем **масиви** с всички **думи**, които ще използваме за нашите **случайни изречения**. **Масивите** се използват, за да **съхраняват няколко стойности** в **една променлива**, вместо да декларираме отделни променливи за всяка стойност.

За да **декларираме масив**, дефинираме **типа на неговите елементи**, последван **квадратни скоби**:



Сега нека да създадем нашия **първи масив** и да го кръстим “**names**”. За да въведем **елементите на масива**, използваме **къдрави скоби**. В тях въвеждаме **имената**, разделени със **запетая**. Ето няколко примера, които можете да използвате:

|  |
| --- |
| "Peter", "Michell", "Jane", "Steve" |

Масивът трябва да изглежда така:



Нека сега да създадем **масиви** с думи за "**places**" (**места**) "**verbs**" (**глаголи**), "**nouns**" (**съществителни имена**), "**adverbs**" (**наречия**) и "**details**" (**детайли**). Ето някои примери, които може да използвате:

* **Места:**

|  |
| --- |
| "Sofia", "Plovdiv", "Varna", "Burgas" |

* **Глаголи:**

|  |
| --- |
| "eats", "holds", "sees", "plays with", "brings" |

* **Съществителни:**

|  |
| --- |
| "stones", "cake", "apple", "laptop", "bikes" |

* **Наречия:**

|  |
| --- |
| "slowly", "diligently", "warmly", "sadly", "rapidly" |

* **Детайли:**

|  |
| --- |
| "near the river", "at home", "in the park" |

Масивите трябва да изглеждат така:

A picture containing text

Description automatically generated

Повече информация за **масивите**: <https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/arrays/>.

#### Създаване на метод за генериране на случайна дума

Сега ще създадем един **метод**. Методите са полезни, за да **подобрят** кода, като **предотвратят повтарянето на код**. Ако имаме **една и съща функционалност** на **няколко места**, тогава можем да създадем един **метод** с нужната функционалност и да го **преизползваме**, когато е необходимо. В нашия случай **методът** ще ни помогне да избираме **случайни думи** за нашите изречения.

За да създадете **метод**, трябва да имате предвид следните особености:

* Методът ще връща променлива от **тип стринг**.
* Трябва да имаме **име** на **метода**.
* Трябва да дефинираме **параметри**, които методът ще получи.

Можем да го направим по следния начин:

A picture containing text

Description automatically generated

Сега нека да напишем логиката на метода. Първо, трябва да създадем **променлива** от тип Random – вече знаете как можете да го направите:

A picture containing text

Description automatically generated

Сега трябва да използваме метода Next()на класа Random, за да **изберем случаен индекс**. Индексът обаче **не трябва** да е **по-голям от дължината на масива с думи**, така че трябва да го напишем така:



Нека да продължим, като създадем **променлива** от тип string за нашата **случайно генерирана дума**. Тази дума ще бъде на **случайно генерирания индекс** от **масива с думи**:



Последното нещо, което трябва да направим, е да **върнем** нашата **случайно генерирана** **дума**:

A picture containing icon

Description automatically generated

Сега нашият **метод** GetRandomWord() е готов за използване. Трябва да изглежда така:

Graphical user interface, text, application

Description automatically generated

Сега е време за по-лесната част – да направим генераторът работещ.

#### Имплементиране на логика за генератора

Първо, нека да създадем **безкраен** while-loop. Вече знаете как да го направите:

Graphical user interface, text, application, chat or text message

Description automatically generated

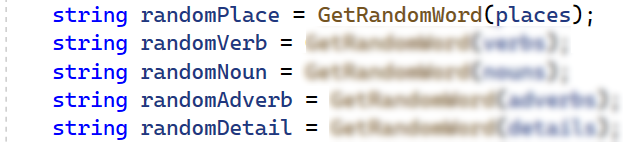
Сега трябва да създадем **променливи** за **всички различни** **случайни думи**. За тази цел ще използваме нашия **метод** GetRandomWord(), който ще свърши тази задача за нас.

Първо, нека да създадем **променлива** от тип string и да я кръстим"randomName". Тази променлива ще съхранява **резултата** от метода GetRandomWord(), когато му **подадем като аргумент масива с думите за имена**:

A picture containing text

Description automatically generated

Сега се опитайте да създадете **променливи** за **останалите думи** сами. Всички трябва да **подават съответния масив** като аргумент и да **съхраняват резултата** от **метода** GetRandomWord():



Следващото нещо, което трябва да направим, е да **структурираме** нашето **случайно изречение**. Спомнете си **модела**, който разгледахме в началото – първо имаме "**Кой и откъде**", след това "**Действие**" и накрая "**Детайл**":

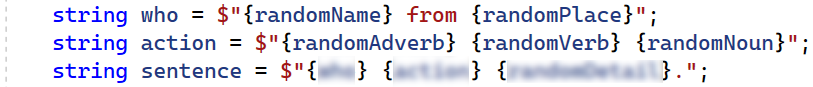
За да конструираме "**Кой и откъде**", трябва да имаме **[име + from + място]**:



За да конструираме "**Действие**", трябва да имаме **[наречие + глагол + съществително]**:



Ние вече имаме **готови детайли**, така че последното нещо, което трябва да направим, е да **комбинираме конструкциите** в **едно** **изречение**. Използвайте **модела** и го направете сами:

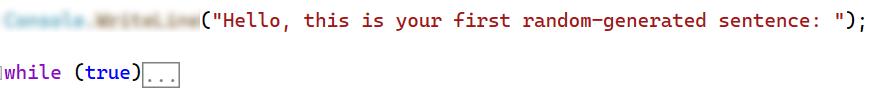


Сега остава да **отпечатаме изречението** на **конзолата**. След това напишете **съобщение** към потребителя да натисне [Enter], за да **генерира ново изречение**:

Logo, company name

Description automatically generated

Също можем да **напишем поздравително съобщение** преди нашия while-цикъл:



Това е всичко, което се изисква, за да **завършим** нашия **проект**. След като го изпълним, генераторът трябва да изглежда така:

Graphical user interface, text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Нека сега да качим проекта в GitHub.

## Качване на проекта в GitHub

Вече знаете как да качите проекта с **Git** **Bash** или **TortoiseGit**.

### Използвайки TortoiseGit (Опция 1)

Използвайте **Git clone** за клониране с TortoiseGit. Отидете в директорията, в която искате да го съхраните, натиснете **десен бутон** в папката и изберете click [Git Clone**]**. Отидете във вашия **GitHub** профил, отворете новосъздаденото **репо** и копирайте неговия **repo URL**.

Отворете TortoiseGit, поставете **URL**-а и кликнете [OK].

Graphical user interface, application

Description automatically generated

Файловете от вашето GitHub репо ще бъдат свалени в **подпапка** с името на проекта в GitHub, в случая "**RandomSentencesGeneratorByPeter**".

Следващото нещо, което трябва да направите, е да **добавите проекта** към **клонираното репо**. Можете да използвате “**Cut & Paste**”. Трябва да изглежда така:

Graphical user interface, application, PowerPoint

Description automatically generated

Сега нека **качим** промените на проекта в GitHub.

Можем да използваме [GitCommit…]от TortoiseGit. Отидете в папката на проекта, натиснете **десен бутон** и кликнете[Git Commit -> "main"…].

Добавете **подходящо** съобщение и кликнете [Add], за да не пропуснете някои файлове, и след това изберете [Commit].

Graphical user interface, application

Description automatically generated

After that click [Push] and then [OK]:

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Това е всичко, от което се нуждаете, за да **качите вашия проект** във вашето **GitHub репо**, използвайки TortoiseGit.

### Използвайки Git Bash (Опция 2)

Като **алтернатива на предишния вариант**, ако нямате "**TortoiseGit**", можете да използвате"**Git Bash**", за да качите проекта си във вашето GitHub репо.

Първо, ако нямата Git на своя **компютър**, го **инсталирайте** от <https://git-scm.com/downloads>.

Отидете в **директорията** с проекта, натиснете **десен бутон** в папката и изберете "**Git Bash Here**", за да отворите конзолата на Git.

Напишете командата **"**gitclone**"**, последвана от линка на **репото**:

|  |
| --- |
| git clone |

Поставете **URL**-а към вашето репо след командата.

Text

Description automatically generated

Файловете от вашето репо ще бъдат свалени в **подпапка** с името на самия проект в GitHub, в нашия случай "**RandomSentencesGeneratorByPeter**".

Следващото нещо, което трябва да направите, е да **добавите файловете от проекта** в **клонираната папка**. Трябва да изглежда така:

Graphical user interface, application, PowerPoint

Description automatically generated

Сега сме готови да качим нашите промени. Отидете в папката на репото, натиснете **десен бутон**, изберете"GitBashHere" и изпълнете следните **команди**.

|  |
| --- |
| git status |

**Git status** показва състоянието на директорията.

A screenshot of a computer

Description automatically generated

Сега напишете:

|  |
| --- |
| git add . |

Горната команда **добавя** всички нови/модифицирани файлове към вашето локално **Git repo**.

След това напишете:

|  |
| --- |
| git commit -m "Your message here" |

Тази команда **commit**-ва вашите промени във вашето локално **Git repo**. Също трябва да **добавите** подходящо **съобщение**.

Имаме да изпълним **още две** **команди**. Напишете:

|  |
| --- |
| git pull |

Тази команда **обновява** вашето локално **репо** от GitHub. Сваля последната версия на проекта от GitHub и я слива с вашето локално копие.

|  |
| --- |
| git push |

Последното нещо, което трябва да направим, е да **push**-нем промените чрез следната команда:

Text

Description automatically generated

Това е всичко, от което се нуждаете, за да **обновите** вашето **репо**, използвайки **Git Bash**.

## \* Модифициране на кода, добавяне на нови функционалности

Нека да си **поиграем с кода** и **да го модифицираме**.

|  |  |
| --- | --- |
| Icon  Description automatically generated | Това е ваш собствен проект. **Бъдете уникални**.   * Имплементирайте **свои собствени функционалности**. * **Имплементирайте кода сами**, добавете ваш стил, форматиране, коментари и т.н. |

### Добавете още думи

Може да добавите **повече думи**, които да се включат в изреченията, за да станат по-интересни и забавни.

### Експериментирайте с различна структура на изреченията

Можете да **промените вашите изречения** и да ги направите по-сложни

You can **change your sentence** and make it more complex:

* Можете да превърнете **изречението във** **въпрос**: ["Кой" дума/фраза за въпрос] + [Глагол] + [Подлог] + [Основен глагол] + [Допълнение].
* Можете да добавите **още части на изречението** на правилните места или да **промените текущите**.

### Качване в GitHub

**Сега качете промените в кода си** във вашето GitHub репо!

|  |  |
| --- | --- |
| Icon  Description automatically generated | Много е важно да **commit-вате често** вашия код в GitHub. По този начин създавате **богата commit история** за вашия проект:  A picture containing chart  Description automatically generated |

## Добавете README.md файл

Силно препоръчително е да напишете **документация за вашия проект в GitHub**, за да опишете за какво е предназначен. Започнете като редактирате файла README.md от вашето репо:

Graphical user interface, text, application, email

Description automatically generated

### Секции на документацията

Добавете **информация** за вашия проект във вашия README.md файл. Обикновено документацията съдържа следните **секции**:

* **Заглавие на проекта** (трябва да отговаря на въпроса "Какво има в проекта?")
* **Цели на проекта**  (какъв проблем решаваме, напр. имплементация на игра)
* **Решение**  (трябва да описва как решаваме проблема 🡪 **алгоритми**, **технологии**, **библиотеки** и т.н.)
* **Линк към source код**
* **Скрийншоти** (скрийншоти към различните функционалности на проекта)
* **Демо наживо** (линк към директно демо на проекта)

### Използвайте markdown

Забележете, че README.md файла се пише на **Markdown език**. Markdown комбинира текст и специално форматирани тагове.

### Цели на проекта

Започнете документацията, като опишете **целите на проекта**. Какъв проблем решава проекта?

### Примерна документация

Това е **пример** как може да документирате проекта си. Съветваме ви да не го преписвате, а сами да структурирате документацията си!

Graphical user interface, text, application, email

Description automatically generated

Можете да добавите **подходящи изображения**, за да подобрите документацията. Можете да добавите изображение по следния начин:



### Вашето решение

Опишете как **решавате** проблема: **алгоритми, технологии, библиотеки и т.н.**

Table

Description automatically generated with medium confidence

Можете да използвате [**backtick**](https://en.wikipedia.org/wiki/Backtick)(`) в **началото** и в **края** на **думата**, за да я направите **сива**:



Можете да използватe **двойна звездичка** (\*\*) в **началото** и в **края** на думата, за да я **удебелите**:



### Линк към сорс код

Добавете **линк** към вашия **сорс код**, както следва:



### Скрийншоти

Добавете **скрийншоти** на вашия проект:

1. **Направете скрийншот** с предназначен за това инструмент (например [Snipping Tool](https://support.microsoft.com/en-us/windows/open-snipping-tool-and-take-a-screenshot-a35ac9ff-4a58-24c9-3253-f12bac9f9d44) за Windows).
2. **Поставете** скрийншота във файла за документация, използвайки [Ctrl+V]:

Примерни скрийншоти за "Random Sentences Generator":

Text

Description automatically generated

## Качване на приложението си в Replit

Вече би трябвало да имате **Replit** профил. Добавете вашия **проект**, за да можете да показвате **проектите си** на вашите приятели и да сложите линка в секцията "**live demo links**" във вашата GitHub документация.

Създайте **нов проект** в Replit от **менюто** в **горния** **ляв ъгъл**.

Кликнете [Create], след това избере **езика**, на който е **написан** вашият проект, изберете **име** и **създайте** проекта. Изберете "Mono C#".

Graphical user interface, text, application, chat or text message

Description automatically generated

**Поставете вашия код** във файла "Main.cs":

Graphical user interface, text

Description automatically generated

**Кликнете** [Run] и ще изпълните приложението си директно в браузъра:

A picture containing graphical user interface

Description automatically generated

Сега можете да **споделяте** **приложението си** с ваши приятели.

## Добавете Replit линк към вашия README.md файл

Сега добавете линк към демо от вашия проект към вашата GitHub документацията:

Text

Description automatically generated

Можете да направите **скрийншот** от Replit.com и **да го поставите** в GitHub документацията директно чрез **[Ctrl+V]**.

След промените, документацията трябва да изглежда така:

Graphical user interface, text, application

Description automatically generated

Поздравления! Успешно завършихте вашия **четвърти проект** в **GitHub портфолиото ви**.