

.Net i Java 01

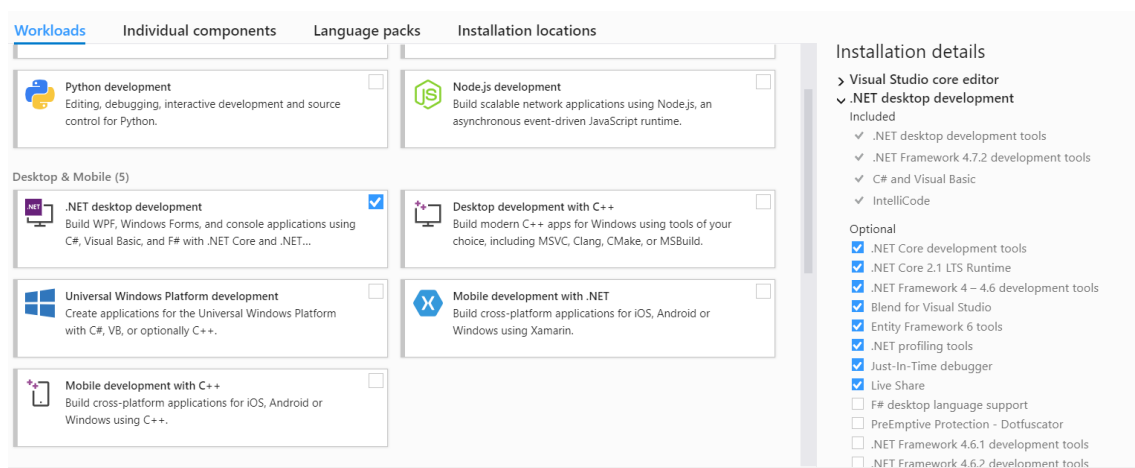
Mgr inż. Teodor Niżyński

Marzec 2020

1 Co potrzeba

1.1 Visual Studio 2019 Community

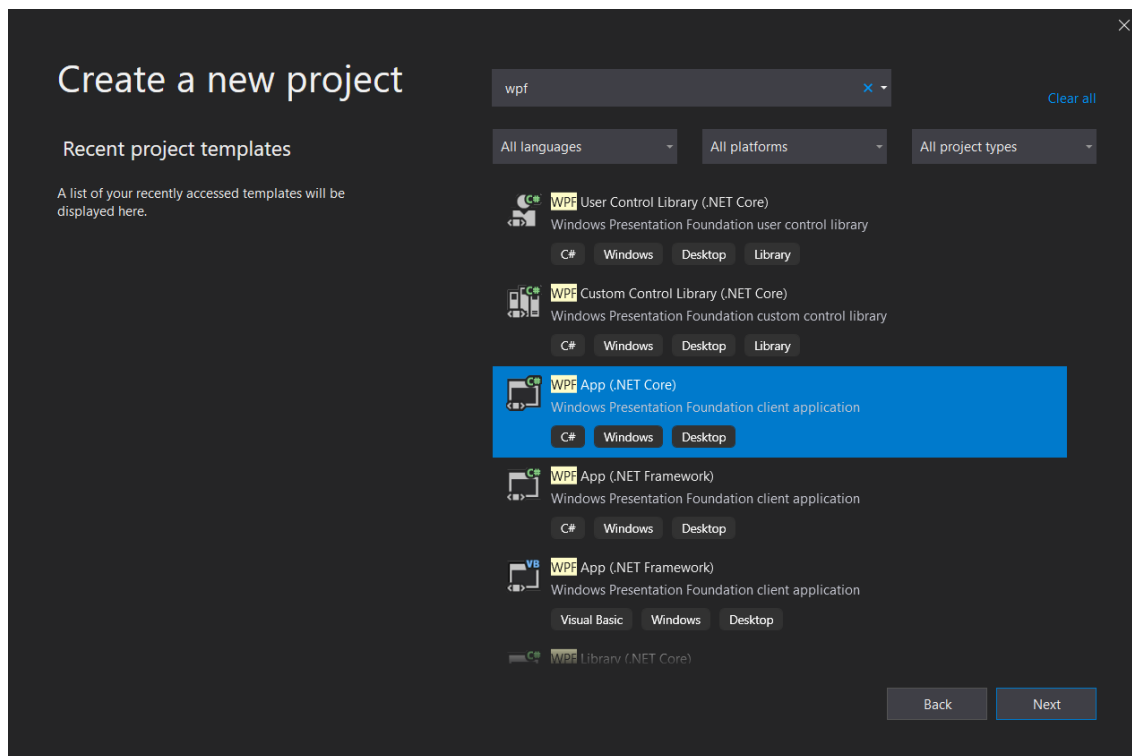
Do pierwszych laboratorium będzie potrzebna najnowsza (niekoniecznie najnowsza ale na tej wersji będę pokazywać) wersja Visual Studio 2019 Community, którą można pobrać ze strony: <https://visualstudio.microsoft.com/vs/>. Przy instalacji proszę zaznaczyć konieczne komponenty konieczne do tworzenia aplikacji desktopowych - jak na rysunku 1 niżej:



Rysunek 1: Screen z procesu instalacji/modyfikacji z zaznaczonymi komponentami do tworzenia aplikacji desktopowych.

1.2 WPF App (.NET Core)

Dla testów proszę sprawdzić czy możecie Państwo stworzyć nowy projekt WPF App (.NET Core) (rysunek 2) a następnie go z sukcesem skompilować i odpalić tym samym upewniając się, że wszystko jest zainstalowane poprawnie.



Rysunek 2: Screen z procesu tworzenia nowej aplikacji WPF App (.NET Core) - prawidłowa opcja zaznaczona.

1.3 System kontroli wersji

Dodatkowo proszę założyć repozytorium na grupie (2-3 osoby) specjalnie na ten przedmiot w serwisie <https://github.com/>. Prosiłbym aby kolejne projekty/programy dodawać jako ponumerowane podfoldery z nazwą projektu/programu. Dla przykładu program z pierwszych laboratoriów proszę umieścić na repozytorium w folderze: 01_FizzBuzz. Jeśli Państwo wcześniej nie korzystali z systemu kontroli wersji, zapraszam do mojej mikro instrukcji: http://teodornizynski.com/new_git.pdf.

1.4 Zapoznanie się z pojęciami

1.4.1 Driven Development

Na pierwszych prawdziwych laboratoriach zajmiemy się podejściem Test Driven Development (TDD) - można poczytać czym to jest, choćby tutaj: <https://www.samouczekprogramisty.pl/test-driven-development-na-przykladzie/>.

1.5 Testy jednostkowe

Polecam przeczytać:
<https://devstyle.pl/2011/08/11/ut-1-co-to-sa-testy-i-po-co-sa-testy-jednostkowe/>

1.5.1 Kata

Dodatkowo w ramach szlifowania języka C# będę próbować zachęcić Państwa do przerobienia kilku zadań - kat (czemu taka nazwa: [https://en.wikipedia.org/wiki/Kata_\(programming\)](https://en.wikipedia.org/wiki/Kata_(programming))) w serwisie: <https://www.codewars.com/>.

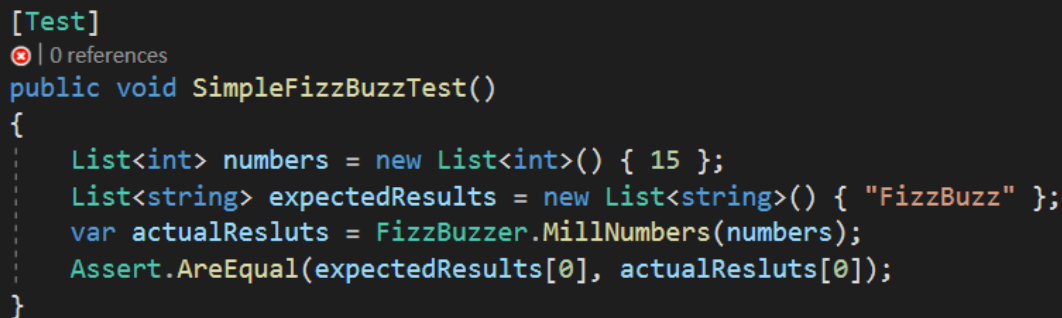
2 Zadanie

2.1 Etap 1

Przygotuj klasę z metodą która przyjmuje listę liczb całkowitych (int) a następnie zwraca listę ciągów znaków (string) oraz spełnia następujące założenia:

- Jeśli liczba jest podzielna zarówno przez 3 jak i przez 5 zwracany jest ciąg znaków: "FizzBuzz"
- Jeśli liczba jest podzielna tylko przez 3 zwracany jest "Fizz"
- Jeśli liczba jest podzielna tylko przez 5 zwracany jest "Buzz"
- Dla pozostałych liczb zwracana jest liczba po konwersji na ciąg znaków (np. 2 → "2")

Konieczne utwórz też projekt z testami jednostkowymi który pomoże przetestować czy klasa działa poprawnie. Idealnie zgodnie z podejściem TDD - najpierw tworzymy testy jednostkowe czyli funkcje które wywołują naszą klasę wraz z jej metodą po czym sprawdzana jest różnica między oczekiwanymi wartościami a wartościami zwróconymi przez klasę którą testujemy.



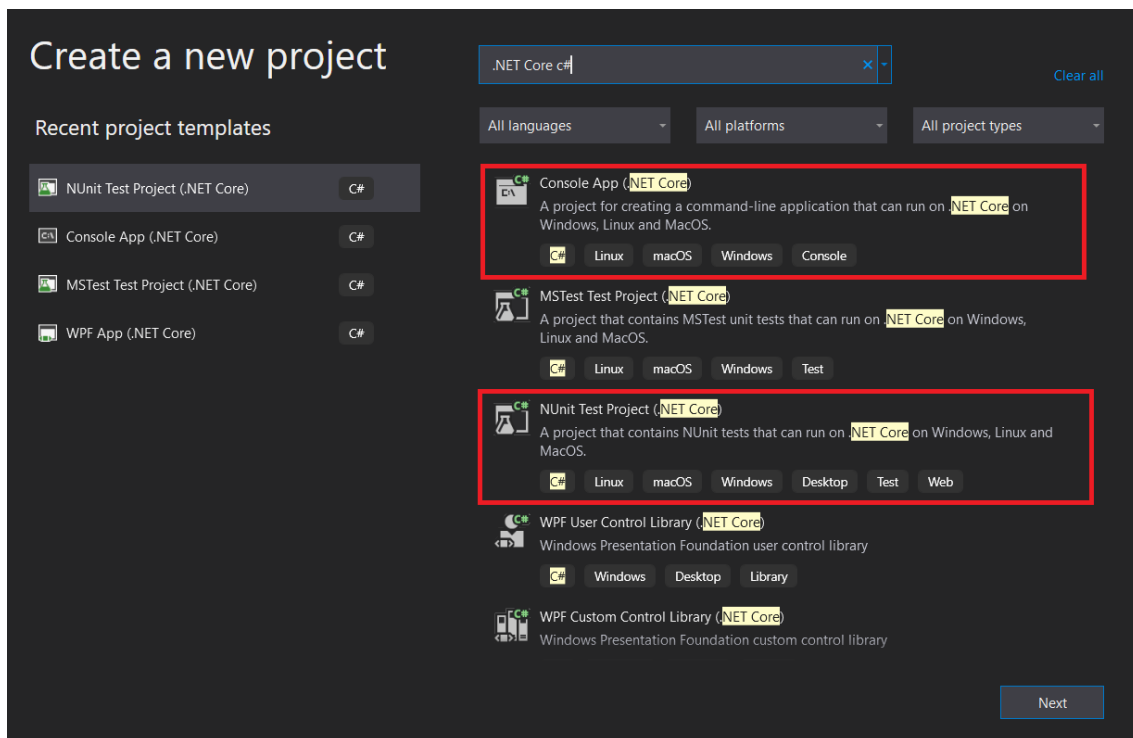
```
[Test]
public void SimpleFizzBuzzTest()
{
    List<int> numbers = new List<int>() { 15 };
    List<string> expectedResults = new List<string>() { "FizzBuzz" };
    var actualResults = FizzBuzzer.MillNumbers(numbers);
    Assert.AreEqual(expectedResults[0], actualResults[0]);
}
```

Rysunek 3: Przykładowy test jednostkowy.

Przykładem niech będzie prosty test z rysunku 3 (testuje on pierwsze założenie - Jeśli liczba jest podzielna zarówno przez 3 jak i przez 5 zwracany jest ciąg znaków: "FizzBuzz"). Wewnątrz metody testowej "SimpleFizzBuzzTest" tworzona jest lista liczb (numbers), w której znajduje się wyłącznie jeden element, liczba: 15 - będą to nasze dane testowe. Następnie tworzone są oczekiwane rezultaty jakie powinna dawać poprawnie działająca klasa - jest to lista ciągów znaków z pojedynczym elementem: "FizzBuzz". W trzeciej linii kodu testu uruchamiamy metodą "MillNumbers" z testowanej klasy "FizzBuzzer" dla testowanych danych testowych (numbers) - rezultat trafia do zmiennej actualResults. Test kończy się przez porównanie czy pierwszy element z oczekiwanej listy rezultatów jest równy pierwszemu elementowi z listy faktycznie zwróconej przez metodą testowanej klasy.

2.1.1 Jak ugryźć to zadanie?

Do spełnienia tych wymagań w pełni wystarczającym projektem jest aplikacja konsolowa (WPF jeszcze nie na tych laboratoriach). Dodatkowo ponieważ chcemy zastosować podejście TDD potrzebny będzie dodatkowy projekt z testami jednostkowymi. Ważne aby oba projekty były w ramach jednej solucji i z wykorzystaniem tej samej technologii (polecam oba w .Net Core jak na rysunku 4).



Rysunek 4: Screen z zaznaczonymi rodzajami projektów wystarczających do tego laboratorium

Aby ułatwić Państwu zadanie można wykorzystać projekt który przygotowałem (od razu też obrazuje jaką strukturę repozytorium oczekuje): <https://github.com/TeoDark/dotNetAndJavaPublic>

Jeśli macie Państwo poprawnie zainstalowanego gita to wystarczy odpalić linię komend (cmd) przejść do folderu gdzie chcecie ściągnąć projekt i odpalić komendę:

```
git clone https://github.com/TeoDark/dotNetAndJavaPublic.git
```

Jeśli dostaniecie odpowiedź "git is not recognized as an internal or external command" oznacza to, że git jednak nie jest poprawnie zainstalowany albo przynajmniej nie jest dodany do PATH (Windows path variable). Po tym zabiegu proponuję usunąć ukryty folder .git a następnie stworzyć nowe repozytorium - szczególnie jeśli zamierzacie rozbudowywać ten projekt. Zapobiegnie to próbom wrzucenia zmian na moje repozytorium a nie tak jak trzeba w ramach laboratorium na swoje.

Projekt który umieściłem na repozytorium nie spełnia wszystkich wymagań etapu 1 to już Państwa zadanie aby uzupełnić braki!

2.2 Etap 2

Zmodyfikuj zarówno klasę jak i testy jednostkowe tak aby uwzględniały następujące założenia (w wypadku konfliktu założenie "wyżej" jest ważniejsze):

- Jeśli liczba jest podzielna przez 7 zwracany jest ciąg znaków "Buzzinga" (nawet jeśli liczba jest też podzielna też przez 3 lub 5 lub spełnia inne założenia niżej).
- Jeśli liczba jest podzielna zarówno przez 3 jak i 5 lub jeśli cyfry 3 i 5 znajdują się obok siebie (np. 352 czy 532) - zwracany jest ciąg znaków "FizzBuzz" (nawet jeśli spełnia inne założenia niżej)
- Jeśli liczba jest podzielna tylko przez 5 lub zawiera "5" (np. 51) zwracany jest "Buzz"
- Jeśli liczba jest podzielna tylko przez 3 zwracany jest "Fizz"
- Dla pozostałych liczb zwracana jest liczba po konwersji na ciąg znaków (np. 2 → "2")

2.3 Etap 3

Proszę o rozwiązanie co najmniej 2 treningowe zadania (kata) dla języka C# w serwisie www.codewars.com (najniższy poziom trudności - 8 kyu wystarczy). A następnie udokumentowanie tego na repozytorium (dodatkowy plik kodu samego rozwiązania z linkiem do katu lub screen z wykonanymi katami w pełni wystarczy)

2.4 Zadania na plusa/dodatkowe punkty

Punktowane osobno:

- A) Rozwiązać co najmniej jedno zadanie dla poziomu trudności co najmniej 6 kyu dla języka C# w serwisie www.codewars.com. Udokumentować jak w etapie 3.
- B) Uodpornić metodę z etapu 2 na co najmniej jedną wartość “nie oczywistą” oraz dodać odpowiednie testy jednostkowe. Przykład który się nie będzie liczył jako wykonanie tego zdania to wywołanie metody dla pustej listy liczb. Podpowiedź: czy nasza metoda może przyjąć coś bardziej “pustego” tak aby program się skompilował ale doprowadził do błędu?
<https://dariuszwozniak.net/posts/kurs-tdd-22-pokrycie-kodu-testami-code-coverage/>

3 Na kiedy?

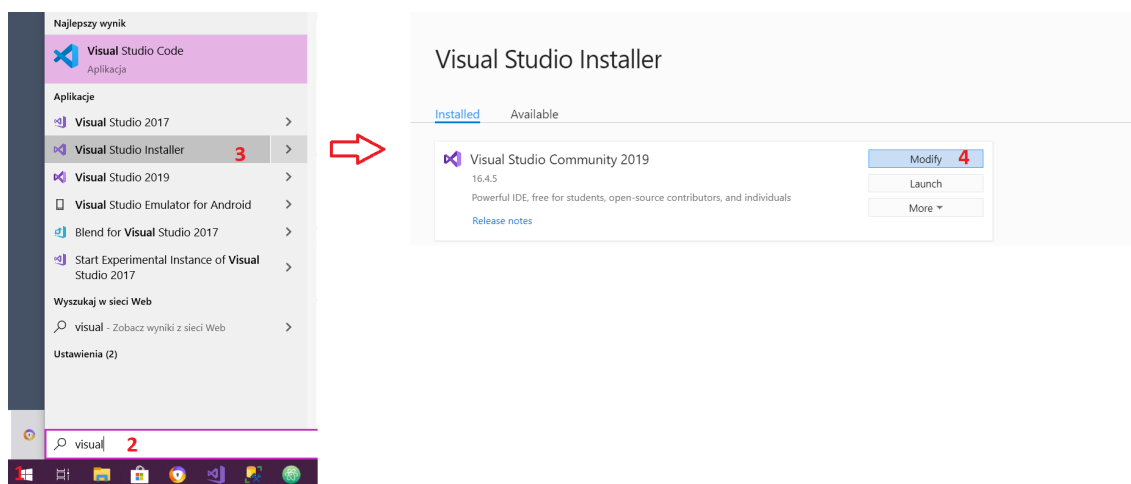
W związku z zamieszczeniem, późną aktualizacją instrukcji oraz brakiem możliwości poprawy/podpowiedzenia Państwu na żywo jak rozwiązać problemy które zapewne się pojawią w trakcie - termin jest znacznie przedłużony. Oczekuję, że rozwiązanie wszystkich etapów pojawi się na Państwa repozytorium (jedno na grupę 2-3 osobową) do:

25.03.2020

4 Pytania i może nawet odpowiedzi

1. Nie wybrałem właściwego komponentu w czasie instalacji Visual Studio 2019! Co robić?!

Nie martwić się, doinstalować - wystarczy odpalić “Visual Studio Installer” a następnie kliknąć “Modify” - rysunek 5



Rysunek 5: Krok po kroku jak modyfikować Visual Studio 2019

2. Czy komputery w sali będą działać jak trzeba?

Pytanie mocno nie aktualne w związku z obecną sytuacją.

Na pewno nie w obecnym tygodniu ale pracuję nad tym – zdecydowanie zalecam własny sprzęt (również ze względu na komfort użytkowania – jeden laptop na grupę zdecydowanie wystarczy).

3. A co z Linuxem?

Visual Studio Code i trochę samozaparcia - .NET Core wygląda bardzo obiecująco (https://en.wikipedia.org/wiki/.NET_Core) ale nie wgrzyzałem się w pracę z .Net na Linuxie i nie będę w stanie super pomóc. “_(~)_/”

4. Co z następnymi zadaniami? Jak będzie to wyglądać?

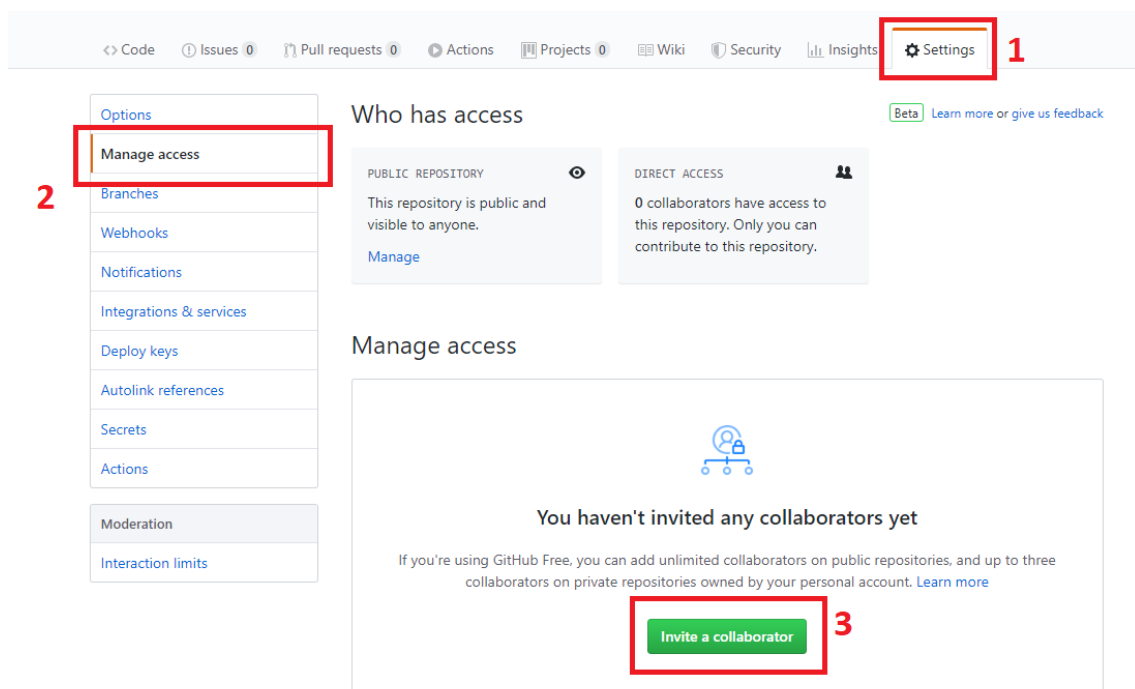
Jeszcze nie wiem, jak tylko się dowiem pojawi się dodatkowa instrukcja na mojej stronie: <http://teodornizynski.com/>.

5. Za dużo nowych rzeczy! Nie wiem co klikać!

Obecnie dokładne pokazanie co klikać jest dla mnie mocno utrudnione. Bardzo przydatne mogą okazać się poradniki. Polecam całą playlistę a w szczególności film o testowaniu: <https://youtu.be/QBiBZ8bsfcU?list=PLdo4f0cmZ0oWoazjhXQzBKMrFuArxpW80> (Mogą pojawić się pewne różnice w porównaniu do projektu który umieściłem na repozytorium wynika to z tego, że na repozytorium projekt testowy to NUnit). Jeśli nie pracowaliście Państwo z Visual Studio przydatne może być też: <https://youtu.be/1CgsMtUmVgs>

6. Czy repozytorium gdzie będą oddawane programy musi być publiczne? Jeśli nie to jak dodać Pana do repozytorium?

Repozytorium nie musi być publiczne, może być jak najbardziej prywatne - nie mam preferencji, zostawiam decyzję Państwu. Aby dodać współpracownika należy w opcjach repozytorium wybrać “Manage access” - kroki zaznaczone na rysunku 6. Najłatwiej mnie znaleźć przez wpisanie mojego prywatnego adresu: teodor.nizynski[małpka]gmail.com ale można też poprzez konto powiązane z repozytorium, gdzie będę zostawiać Państwu kod związany z kursem: <https://github.com/TeoDark/dotNetAndJavaPublic>



Rysunek 6: Screen z procesu instalacji/modyfikacji z zaznaczonymi komponentami do tworzenia aplikacji desktopowych.

5 Inne

Jeśli znają Państwo jakiś błąd/niejasności w instrukcji - proszę o kontakt mailowy - przy czym w pierwszej kolejności będę uaktualniał instrukcję a dopiero później odpisywał.