Notatki do 2. tygodnia kursu Dotneta

Maciek Mielczarek

5 sierpnia 2020

1 Wstęp

Na ten tydzień zaplanowane są podstawy języka C#. Najprawdopodobniej będę porównywał wprowadzane elementy do tego co znam z C++ albo Javy.

2 Wybór projektu

Nie wiem co wybrać, więc rzucę kostką. Na początek właściwe losowanie przy użyciu fizycznych kości k6 (6 ścian z numerami od 1 do 6). Są różne możliwości wylosowania liczby od 1 do 25 przy pomocy standardowych kostek. Zdecydowałem się na następujący wariant:

1. Z k6 robię k5 przez rzucenie ponownie w przypadku wyniku 6. Dzięki temu wszystkie powtórki rzutów załatwiam od razu.
2. 2 razy rzucam k5 i odejmuję od wyniku 1, aby dostać cyfrę dziesiątek (a właściwie piątek) i cyfrę jedności liczby w systemie piątkowym.
3. Przeliczam tą liczbę do systemu dziesiętnego (pierwsza cyfra razy 5 plus druga cyfra) i dodaję 1, żeby dostać liczbę z przedziału od 1 do 25.

Pierwszy rzut: 5. Drugi rzut: 3. To odpowiada liczbie 23, czyli grze w kółko i krzyżyk. To prawdopodobnie najprostszy temat, ale ponieważ istnieje wiele oczywistych wariantów tej gry, to będę mógł sprawdzić czy zaplanowałem i napisałem kod w taki sposób, żeby aplikację dało się rozwijać.

Skoro już jestem przy temacie rzucania kostką, to sprawdzę jak się losuje liczby w C# i odtworzę powyższą sytuację w kodzie. Pewnie wiele rzeczy jest nie na swoim miejscu lub zrobionych dziwnie, ale kod jest przetestowany i działa. Można go znaleźć tutaj.

3 Start projektu, ekran startowy

Projekt wybrany, czas stworzyć dla niego repozytorium na gicie i zacząć go robić. Do nauki lub przypomnienia sobie gita polecam to miejsce.

Na początku utworzyłem przy użyciu przeglądarki repozytorium do notatek i kodu z całego kursu. Następnie sklonowałem je i przeniosłem do stworzonego w ten sposób folderu już utworzone pliki. Następnie użyłem w tym folderze komendy "dotnet new gitignore" (ktoś chyba o tym wspomniał w okolicach tego kursu) i dodałem do nowego pliku kilka linii odpowiadających plikom pośrednim LaTeX-a. Potem kilka komend żeby wrzucić wszystko na Githuba, w razie błędów git podpowiadał co jest nie tak. To na Linuksie.

Na Windowsie klonowałem z poziomu Visual Studia. Zalogowałem się przy tym do Githuba z poziomu VS, dzięki czemu nie muszę wpisywać loginu i hasła przy każdym commicie. Następnie stworzyłem nowe Rozwiązanie (Solution) w pożądanym miejscu. Gdy już miałem otwarte w VS Rozwiązanie w folderze śledzonym przez Gita, to wszystko co Gitowe znalazłem w VS po środku prawej strony, po kliknięciu w napis "Team Explorer" obok napisu "Solution Explorer".

W pierwszej wersji programu (lekcja 2.) jest tylko ekran startowy z którego można wyjść. Zadbałem o to, żeby były tam już jakieś zmienne i stałe. O, jednak nie muszę ręcznie zamieniać stringów na inty, tak jak to robiłem w zabawie z kostkami powyżej. Tak mi się wydawało, że gdzieś w C# powinno być coś takiego jak Int32.Parse, czy tam TryParse, ale zanim to znalazłem, napisałem już swoją wersję (tylko bez obsługi błędów).

4 Gdzie wsadzić enuma i trochę struktury

W tym projekcie na ekranie głównym nie widzę zapotrzebowania na enumy (kolejny temat), więc potrzebuję już teraz dodać kilka klas.

Typ gracza komputerowego (łatwy/trudny) będzie enumem, więc pododaję klasy odpowiedzialne za typy graczy. Przy definiowaniu zależności między klasami jest parę słów kluczowych, których nie wiem czy dobrze używam, ale sprawdzę to przy okazji jakiegoś tematu o klasach. W VS łatwo tworzy się nowe klasy i interfejsy (kliknięcie na folder w Eksploratorze Rozwiązania -> dodaj nową pustą klasę/interfejs).

Niemal równie łatwo zapomnieć zmienić aktywną gałąź przy zaczynaniu nowego tematu.

5 Typy referencyjne, stos i sterta

Wiedziałem co to są typy referencyjne, ale nie wiedziałem co to stos i sterta. Rzeczy do sprawdzenia za chwilę:

- jak długi string mogę stworzyć
- czy zarówno \r\n jak i \n działają tak samo na Windowsie i Linuksie

Przy próbie stworzenia stringa długości 2^{30} program wywraca się z braku pamięci. W obu przypadkach \r\n i \n przechodzą na początek nowej linii, a samo \r wraca na początek bieżącej linii i kolejny napis nadpisuje początek tej linii. Czas na zadania.