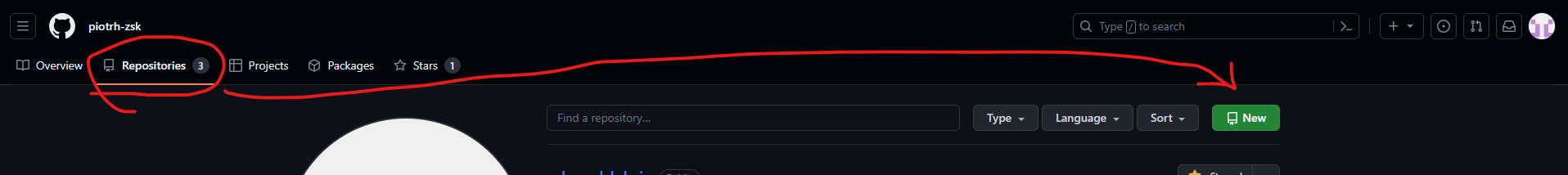
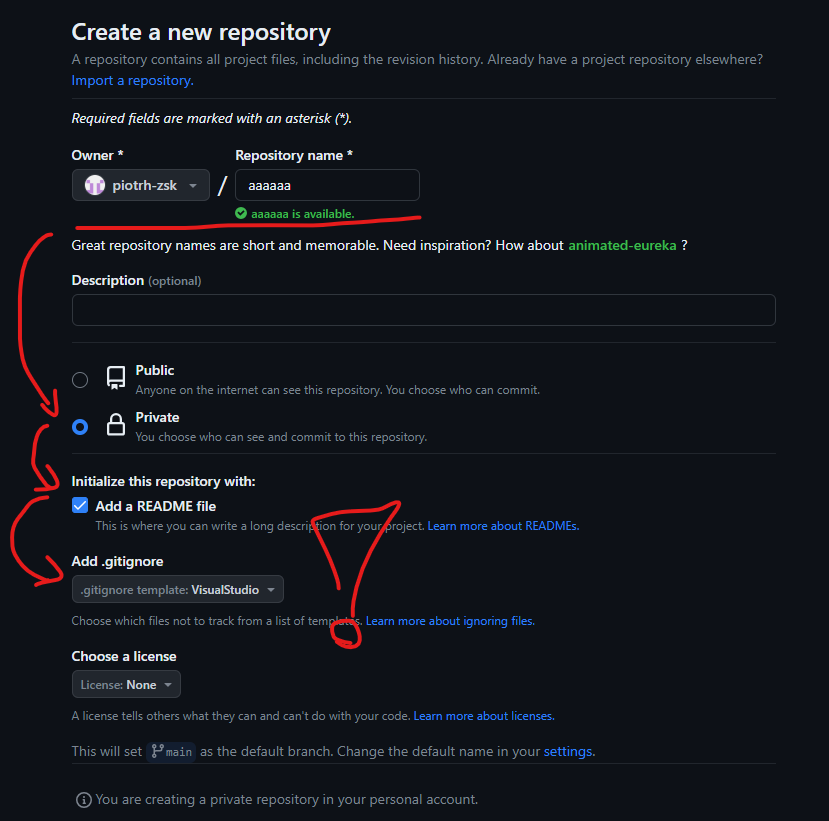
Wchodzimy na portal GitHub i zakładamy na nim konto (lub logujemy się na już posiadane konto):

https://github.com/

Po założeniu konta przechodzimy do zakładki Repositories i wybieramy opcję New:



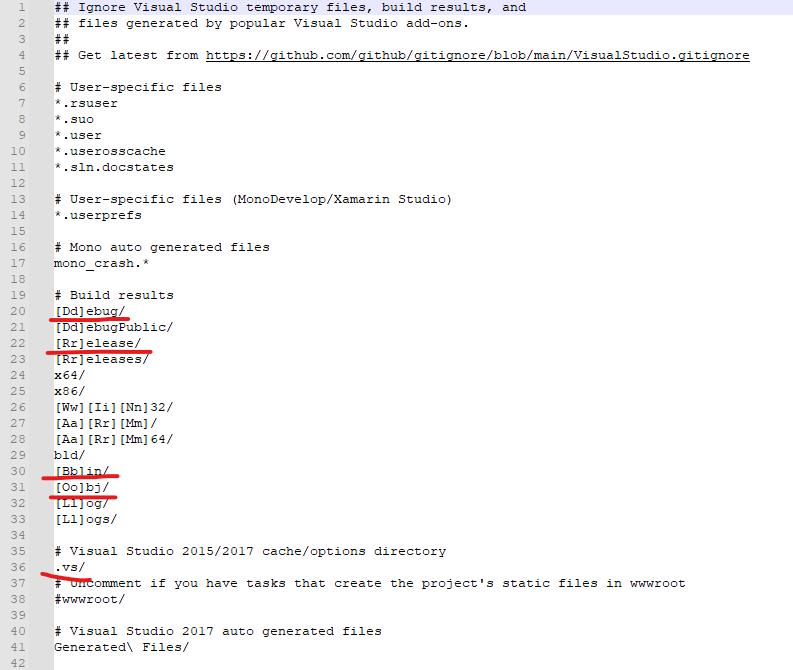
W formularzu tworzenia nowego repo wpisujemy sobie jakąś sensowną nazwę. Oznacz repozytorium jako prywatne, zaznacz sobie opcję dodania pliku Readme (choć prawdopodobnie nie będziemy z niego korzystać), **i super ważna rzecz**: w opcji „Add .gitignore” wybierz opcję Visual Studio.



Domyślnie mechanizm GIT bierze pod uwagę wszystkie pliki jak leci. Każde środowisko programistyczne tworzy jakieś swoje tymczasowe śmieciowe pliki, lub tworzy pliki automatycznie, które będą się różniły w zależności od miejsca wersji Visual Studio, lokalizacji itp.

Plik .gitignore definiuje nam szereg instrukcji dla repozytorium, które wyłączają poszczególne pliki, foldery lub rozszerzenia, które mają być pomijane przez GITa.

Automatycznie wygenerowany plik .gitignore pod Visual Studio zawiera na przykład:

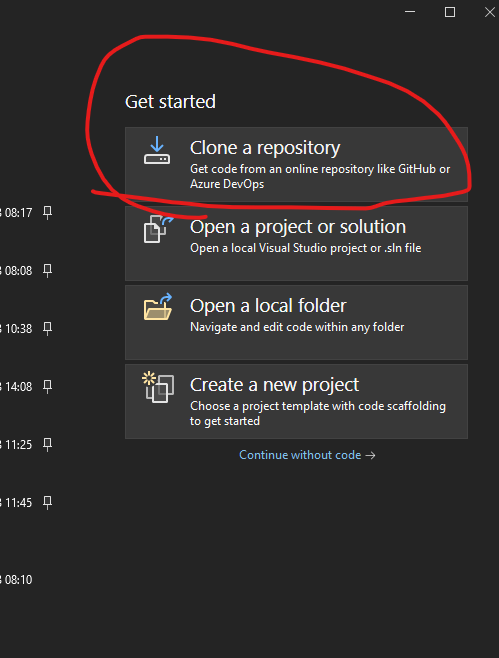


Widzimy tutaj, że ignorowane są całe foldery Debug i Release, są to foldery, w których Visual Studio umieszcza przekompilowany wynikowy program. Foldery Bin/Obj to takie foldery, w których lokalnie przechowywane są zasoby potrzebne do zbudowania projektu, biblioteki DLL itp. Ukryty folder .vs z plikami tworzonymi prze Visual Studio,itd.

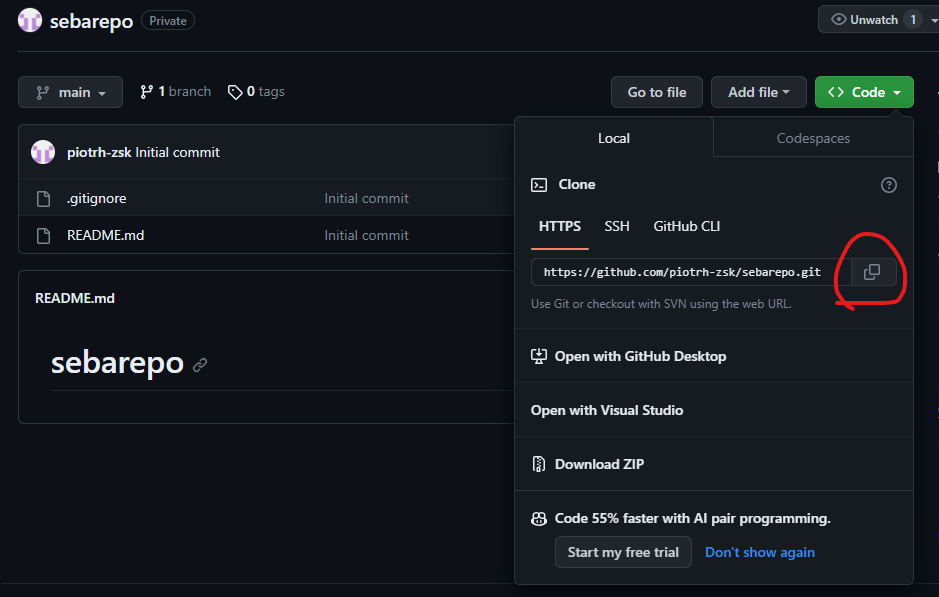
Rzecz w tym, że jeżeli foldery te będą nieobecne, to przy próbie uruchomienia programu Visual Studio tak czy inaczej automatycznie wszystko to utworzy od nowa, dlatego lepiej jest nie wrzucać tego do GITa.

**Podpięcie repozytorium do Visual Studio**

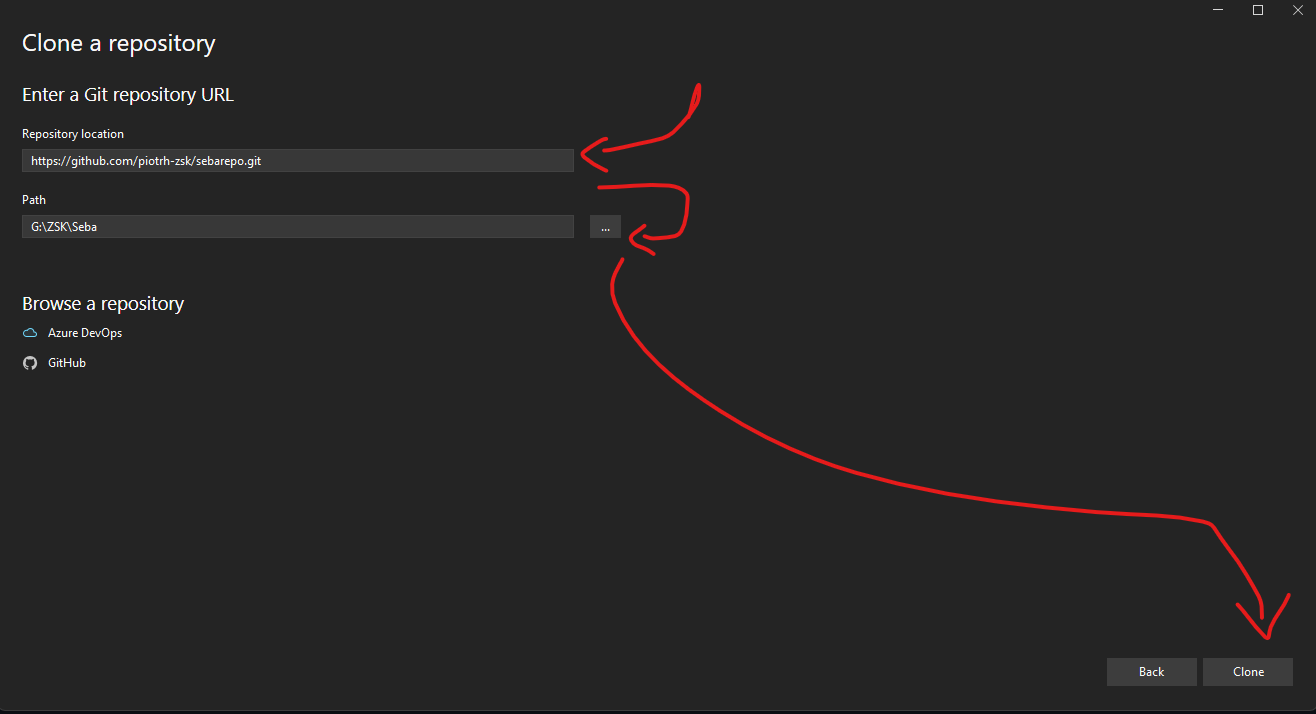
Tworzymy na dysku jakiś folder, w którym docelowo będziemy trzymać wszystkie pliki. Następnie uruchamiamy Visual Studio i wybieramy opcję Clone Repository:



Wracamy na chwilę do Githuba, klikamy na zielony przycisk i kopiujemy adres repozytorium:



W Visual Studio wklejamy adres repozytorium, wskazujemy ścieżkę lokalną i wybieramy opcję Clone:

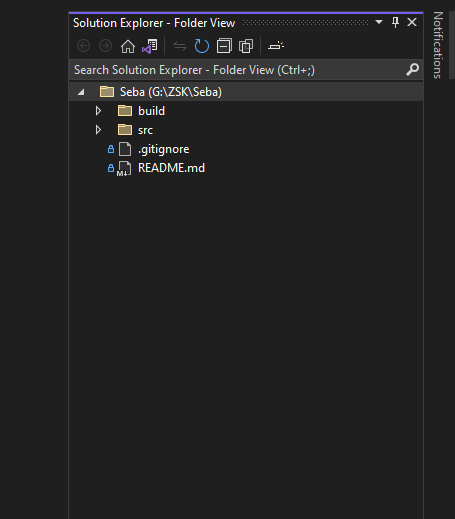


Visual Studio domyślnie po tej operacji otworzy coś tam, ale lepiej kliknąć w Visualu opcję w górnym menu File -> Open folder a następnie wskazać folder, do którego zostało pobrane repozytorium.

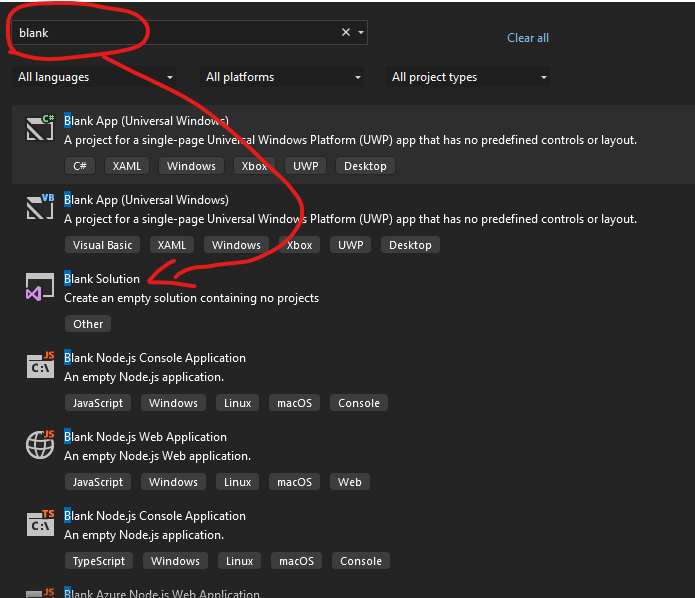
Następnie w panelu po prawej stronie powinniśmy mieć okno zatytułowane „Solution Explorer”. Jeżeli okienko nie jest obecne możemy je włączyć korzystając z opcji VIew -> Solution Explorer. Będziemy z tego okienka wielokrotnie korzystać, więc lepiej go nie wyłączać.

W okienku Solution Explorer klikamy PPM na najwyższy element drzewa, czyli po prostu nazwę Twojego folderu i wybieramy opcję Add -> Folder.

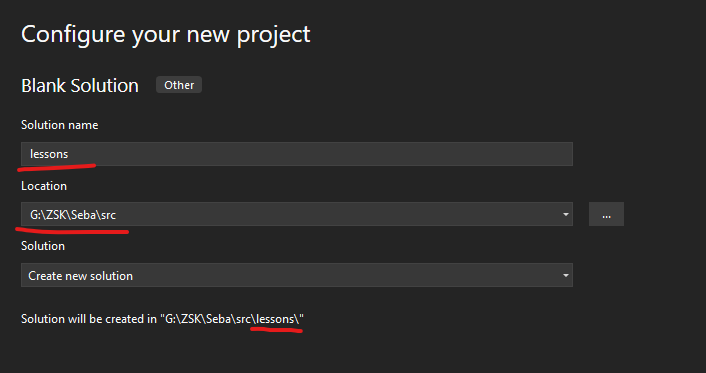
To co teraz wykonamy nie jest oczywiście konieczne, ale przyjęte standardy są takie, że solucja zawierająca cały projekt zawiera najczęściej minimum dwa foldery, czyli build oraz src. W folderze build kiedyś w odległej przyszłości znajdować się będą skrypty do automatycznego budowania projektu. Tak czy inaczej solucja powinna wyglądać tak:



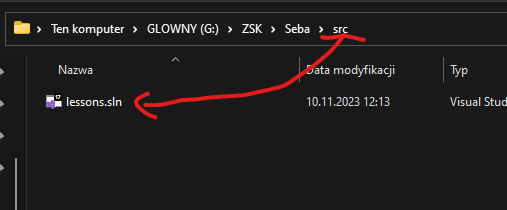
Następnie klikamy w górnym menu opcje File -> New -> Project. W okienku wyboru szablonów projektu w szukajkę wpisujemy „blank” i wybieramy opcję Blank solution.



W kreatorze nowej solucji wypełniamy nazwę, wskazujemy folder „src”, który wcześniej utworzyliśmy. Głupie zachowanie Visual Studio jest takie, że solucja zostanie utworzona w podfolderze o nazwie naszej solucji:

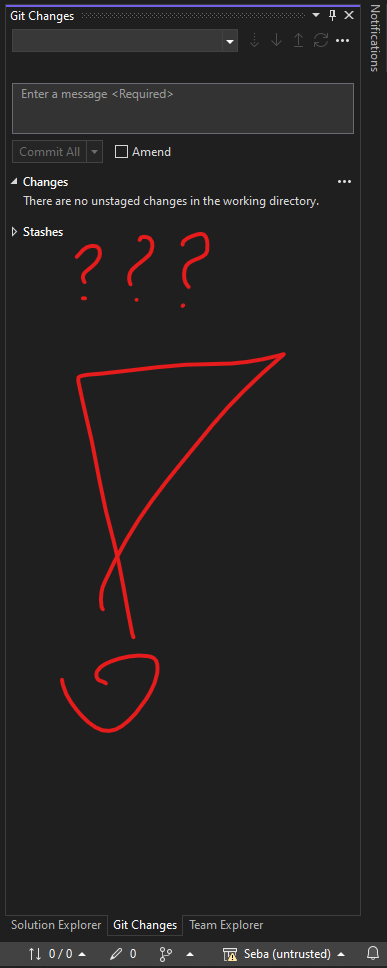


Jest to zbędny folder w strukturze plików. Dlatego po utworzeniu solucji wyłączamy Visual Studio i ręcznie „naprawiamy” tę przypadłość poprzez wykopiowanie pliku „sln” wprost do folderu „src” i usunięcia zbędnych rzeczy:



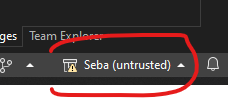
Włączamy następnie Visual Studio poprzez dwukrotne kliknięcie w nasz plik sln.

Najprawdopodobniej po tej operacji repozytorium GIT „odepnie” się od naszej solucji. Rozpoznać to będzie można po sprawdzeniu zawartości „Git changes”:

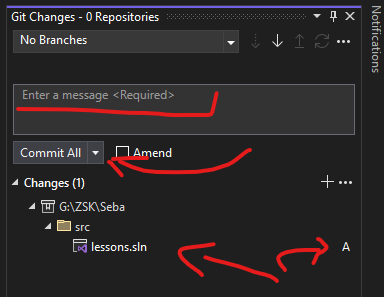


Jeżeli okienko Git Changes nie będzie widoczne w panelu po prawej stronie można to okienko włączyć korzystając z górnej belki z opcji View -> Git changes.

Jak widać na powyższym zrzucie ekranu, można stwierdzić, że okno jest w zasadzie puste. Z kolei w prawym dolnym narożniku Visual Studio najprawdopodobniej widoczne będzie ostrzeżenie, mówiące o tym, że repozytorium nie jest zaufane.



Rozwijamy listę, klikamy PPM na repozytorium i wybieramy opcję Trust Repository. Po tej operacji w okienku Git Changes powinny pojawić się już pierwsze zmiany, które wprowadziliśmy od czasu stworzenia Repozytorium:

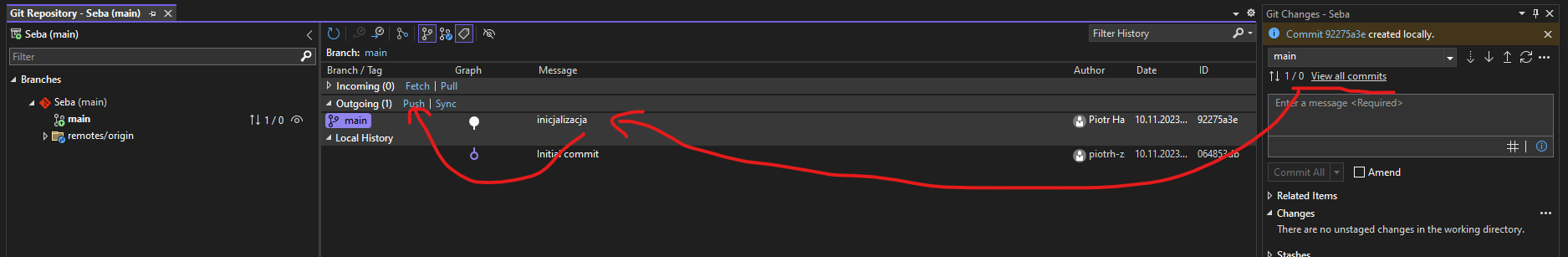


Widać na dole, że dodaliśmy plik \*.sln, oznaczony jest on literą „A” czyli informacja o tym, że jest to nowy dodany plik. W polu „Enter a message” wpisujemy jakiś komentarz, który krótko opisze nam co zostało wykonane, na przykład „Inicjalizacja repozytorium, utworzenie solucji”, następnie klikamy opcję „Commit All”.

Po zacommitowaniu zmian do repozytorium powinna pojawić się opcja View all commits. Po jej wybraniu otwarte zostanie okno prezentujące stan naszego repozytorium.

Należy pamiętać, że w obecnym stanie kod jest wrzucony do repozytorium jedynie LOKALNIE, na Twoim komputerze. Zmiany te nie zostały jeszcze wypchnięte do faktycznego repozytorium w sieci. Jest to bardzo fajne zachowanie GITa, ponieważ potem, pracując w pracy nad zmianą w jakimś dużym projekcie możesz swoją pracę dzielić na etapy, wrzucać do repozytorium lokalnego, analizować co sam w ramach tych zmian zostało pozmieniane, które pliki i jak, i wszystko to nie naruszając repozytorium globalnego.

My jednak docelowo chcemy aby zmiany zostały wypchnięte do globalnego repozytorium. W oknie widać zmiany, „commity”, oczekujące na wypchnięcie. Wybieramy więc opcję „Push”. Jeśli wszystko przebiegnie jak należy, zmiany powinny wrzucić się do Repo w internecie.



Na tym etapie, klikając w swoim repozytorium na stronie GitHub widoczny powinien być plik sln i historię Twoich commitów.