Ripasso su git

Il branch che crei per primo è il main, stacco il branch che sarà una release. Dopo che faccio un hotfix mergio sia nella release che nella main. La feature è un branch di sviluppo e poi si mergia nel main, la release invece è la versione fixata e testata del main, quella che poi forse andrà al cliente.

Comandi principali

Git clone --> clonare la repository remota in locale

Git checkout --> serve a switchare tra varie branch o a restorare il file ad una certa versione

Git brench-->

Git commit--> aggiorna lo stato del file e della repository

Git push-->

Git fetch -->

Git pull--> aggiorna la storia

Git stash--> salvare il locale i propri fix ma senza committarli, tiene conto solo delle modifiche fatte fino al momento dello stash stesso

Git blame--> per vedere chi ha committato certe versioni

Git cherrypick--> ci permette di portare le modifiche di uno o più commit nella nostra branch

Git log--> Storia delle modifiche

Git diff--> Differenze tra file

Git reset--> Ci resetta le nostre modifiche e torniamo all'ultimo commit

Git revert--> Possiamo scegliere un commit a cui tornare, portando con noi se vogliamo alcune modifiche

Strategie di merge

Fast forward merge--> Mergi in automatico quando non ci sono conflitti

Three way merge--> Merge manuale con locale remoto e codice unito ma potrebbe avere dei conflitti, sta a te andare a mettere quello che ti serve effettivamente

Pull request

Chiedo di inserire le modifiche del mio branch in un altro branch, faccio richiesta di controllo delle mie modifiche per far si che finiscano nella release.La pull request può avere delle regole che determinano chi può accettarla.

Ripasso visual studio

Progetto--> core del nostro programma che contiene un insieme di classi, è quello che stiamo sviluppando, può essere una libreria o un applicazione (application) e vari dll.

Classe--> file effettivo dove andiamo a scrivere il codice

Assembly--> compilato del nostro progetto, quando faccio la build mi darà in ouput se è segnato come application un exe .

Se compilo invece un progetto class library mi darà un dll

Debugging

Breakpoint

Step over/into/out

Watch variables

Call stack inspection--> lista di tutti i metodi chiamati fino al metodo in cui ci siamo fermati noi

Quando scrivo il mio codice e provo a buildarlo, per build si intende che ciò che vengo scritto viene convertito in exe o dll e poi posso usarlo, posso fare anche direttamente lo start , perchè visual studio comunque builda prima.

Nuget package manager

Librerie hostate da vari enti, da cui il nostro progetto avrà delle dipendenze , non le vedrò su visual studio come soluzione, ho il mio progetto localizzato singolo, pusho sul mio nuget , e il mio nuget può essere usato in parallelo , evita le complessità di accedere a repository altrui.

Differenza tra console application e class library

L’exe è runnabile e ha un entry point , il dll è un elenco di metodi e classi del nostro progetto

Differenza tra debug e release mode in visual studio

La debug mode ci permette di debuggare, ci da accesso ai debugging tools di visual studio, ci permette di mettere breakpoint, osservare le variabili ed avere il call stack

La release mode invece non ci permette di debuggarla, anche se metto il breakpoint non si ferma il programma, mi mancano alcuni file che nella release non ci sono, serve praticamente quando viene mandata all’utente finale, è senza debug, c’è questa differenza perchè per progetti grossi i file di debug sono molto grandi e non vanno consegnati all’utente finale

.Net Framework

Un framework in generale è un insieme di classi di librerie predefinita che noi possiamo usare liberamente, ci facilita il lavoro perchè molte cose che dovremo scrivere ci sono già. Il framework ha anche una gestione di memoria.

Nello specifico .Net è di microsoft e si usa in C#, usiamo il framework e non il core perchè è quello più diffuso dagli anni 2000 (usata per tutte le applicazioni windows).

Oltre avere le nostre librerie ha il CLR , ci permette di avere una gestione più facilitata della memoria , il garbage collector.

Microsoft l’ha sviluppato in concomitanza con C# per dare concorrenza a java.

Il core è un evoluzione di dotnet , di fatto comprende anche molte librerie del framework , rendendo le applicazioni crossplatform , runnabile anche su altri os . Dotnet core comprende alcune librerie del framework.

Quando si dice dotnet 5 6 7 si intende il core.

Grazie al CLR si ha quindi il garbage collector, che ci dispone di oggetti che non abbiamo più nel tempo, può essere anche forzato il garbage collector, tutte le gestione delle eccezioni le abbiamo grazie al framework.

.Net in generale ha una serie di class library, per input output , collegamento al db

Introduzione a C#

Linguaggio pensato per la programmazione ad oggetti, ci aiuta a programmare in una maniera unificata, usato con .net framework ma col core può essere utilizzato ovunque.

In c# abbiamo gli operatori e le espressioni, gli operatori possono essere aritimetici, di assegnazione o comparazione.

I++ operatore che ci permette di incrementare in un intero il nostro numero. Invece ++i ci permette di incrementare la variabile e successivamente mostrarla, è un procedimento diverso a livello di memoria.

Il c# ha dei construct , che sono sequenze selection (if, switch) e iterazioni.

In c# posso avere i tipi che possono essere:

Value types: il valore di quella variabile è salvato in memoria

Reference types: Ho l'indirizzo di dove è salvato in memoria

Se io immagino un intero a=5 a=b , sto di fatto avendo due volte in memoria il numero 5

Invece con il reference type mi sto copiando l'indirizzo ,creo l'oggetto una volta.

Int ha numeri interi ed è a 32 bit .

Il double ci permette di avere dei decimali ed è a 64 bit.

Il decimal è utilizzato per essere più preciso del double per la gestione del floating point.

Il float è meno preciso.

Gli array sono una collection dello stesso tipo, in c# ha una size fissa che decido quando lo creo, poi immodificabile--> int[].

Il char invece è un carattere unicode scritto con apici singoli.

La stringa invece è una sequenza di caratteri, ed è già di per se un array di char , ha i doppi apici, quando modifico una stringa in c# è come se facessi un array nuovo.

Loop construct

For , while, do while