Ако търсиш нещо качествено и негодуваш, когато станеш жертва на некачествена работа, значи трябва да се замислиш когато и ти вършиш нещо. Качествената работа предполага отговорно отношения към материала, трябва да вземеш насериозно поетия ангажимент и да го докараш до край според изискванията.

Защо трябва да вършим нещо качествено ? Да си представим, че купуваме нещо – софтуерен продукт или не - даваме му сумати пари и се надяваме да работи коректно. Пускаме го, минава време и крашва. Рестратираме ... Да, но сега пък някой бъг. Още един бъг ... и още един и така докато разберем, че сме сгрешили с купуването на продукта ... Какво бихме направили ? Една дълга майна и следващия път няма да купим от този производител! Ама ако този прозиводител сме ние ? Това ще означава, че ще имам един клиент по-малко ... А ако десетки клиенти се откажат да купуват ? Може да кажем чао на пазара ☺

Когато предлагаме софтуерен продукт и искаме да се докажем, трябва да разчитаме на доверието сред клиентите, а доверието е пряко свързано с качествения продукт. Продукът е качествен, ако кодът е качествен. Тогава как може да разграничим качествения от некачествения код ? Какви са особеностите на един качествен код ?

Поведението на софтуерния продукт трябва да е според изискванията – да няма бъгове, крашове, да е стабиУен. При некоректна работа, да има подходяща реакция поради причините, които описах в началото на есето.

Не трябва да забравяме, че освен нас и някой друг ще мине през нашия код и ще го редактира, ако е нужно. Имайки впредвид тази ситуация, нашият код трябва да бъде четим, лесно разбираем и лесен за поддръжка. Кодът е лесно четим и лесно разбираем, ако сме обърнали внимание на именуването така, че кодът да е самодокументиращ се. За целта имена на порменливите, методите, класовете, интерфейсите трябва да бъдат подбрани правилно.

Особено важно е да се следва някаква конвенция – на фирмата и/или на общността на програмистите, които работят с въпросния език.

Променливите трябва да имат говорящо име, кракът лайфтайм и спам.

Класовете/интерефейсите да са със силна кохезия и в слаба връзка с останалите класове/интерефейси, високо ниво на абстракция, добра енкапсулация.

Методите да са с подходящо име и параметри. Да върши само една работа като името му да е говорящо и еднозначно да дефинира това, което прави – като черна кутия.

Кодът да подходящо форматиран, да следва общоприетите правила, а не да е разхвърлен и следващият, който се заема да разширява нашия код, да си хване главата.

Документацията е написана по подходящ начин. Кодът колкото повече е самодокументиращ се, толкова по-малко коментари са нужни.

Добре написани юнит тесове, които покриват колкото се може повече от кода. Тестовете да обхващат повече сценарии и да гарантират, че прдикут ще работи според изиксвания. Не е възможно да се махнат абсолютно всички бъгове, но идеята е дори да се появят такива или трябва да бъдат хванати, или да са такива, че да не засегнат коректната работа на продукта.

Ако се обърне внимание на тези изисквания, то качеството на кода ще е на доста добро ниво – продуктът също и ние ще се радваме на повече клиенти ☺