# Упражнения: Качество и преработка на кода

## Добър код

Разгледайте внимателно единия от примерите програмен код по-долу (или и двата). **Не е нужно да изтегляте нищо**, просто разгледайте кода:

* Microsoft .NET Framework Reference Source:

<http://referencesource.microsoft.com/#mscorlib,namespaces>

Можете да проучите някои от познатите ви класове, например:

* + **System.Collections.Generic.List<T>**: <http://referencesource.microsoft.com/#mscorlib/system/collections/generic/list.cs,cf7f4095e4de7646>
  + **System.Text.StringBuilder**: <http://referencesource.microsoft.com/#mscorlib/system/text/stringbuilder.cs,adf60ee46ebd299f>
  + **System.DateTime**:  
    <http://referencesource.microsoft.com/#mscorlib/system/datetime.cs,df6b1eba7461813b>
  + **System.Random**:  
    <http://referencesource.microsoft.com/#mscorlib/system/random.cs,bb77e610694e64ca>

Може да разгледате и друг клас (или класове).

* OpenPOS: Open Source Point-Of-Sale System:

<http://openpos.codeplex.com/SourceControl/latest>

Можете да разгледате който и да е от класовете (или няколко класа). Някои добри примери, които си заслужава да погледнете, са:

* + Пространство от имена **Trunk.OpenPOS.OpenPOSData.Models** (всички класове)
  + Пространство от имена **Trunk.OpenPOS.OpenPOSInfrastructure.Intefaces** (всички интерфейси)
  + Пространство от имена **Trunk.OpenPOS.OpenPOSModules.Sales.Services**: CashupService.cs, PaymentService.cs

**Не е необходимо да разбирате структурата на системата.** Просто разгледайте класовете.

Опитайте да разберете каква е целта им, какви добри практики са използвани и т.н.

Документирайте всичко, което намерите, в таблица. Следвайте зададения формат:

|  |  |  |
| --- | --- | --- |
| **Свойство** | **Коментар** | **Пример (по желание)** |
| Коментари в кода | Добре написани коментари, не твърде много; обясняват объркващи части от кода | // A StringBuilder is internally represented as a linked list of blocks each of which holds  // a chunk of the string. It turns out string as a whole can also be represented as just a chunk,  // so that is what we do. |
| Форматиране на кода | Добре форматиран код, вмъкванията помагат да се обясни целта на кода  Кода е разделен на „региони“: най-отгоре променливите, после конструкторите, после методите | - |
| … | … | … |
| **Цялостен коментар** | Как ме кара да се чувствам този код? Какво го прави добре написан? Мога ли да разбера какво става? Мога ли лесно да променя нещо в него? | |

## Лош код

Разгледайте следните примери за лош (или, ако щете, не чак толкова добър ☺) код:

* <https://github.com/crazycat2701/Nhom19-TrangSuc/>

Разгледайте няколко класа (например **Trangsuc.Models**, **Trangsuc.Controllers**) и попълнете таблица, подобна на тази в предишната задача. Този път обърнете внимание на всички проблеми / лоши практики, които намерите. Добавете под таблицата коментар, обясняващ цялостното ви мнение за кода: лесен ли е за **четене**, **разбиране**, **разширение** и **управление**?

**Съвет 1:** Ползвайте конспекта на този курс за идеи (например форматиране на код, именуване…)

**Съвет 2:** Ползвайте OOP принципите за още идеи (проектът следва ли ясна структура, има ли капсулиране и т.н.)

**Не е необходимо да разбирате структурата на системата.** Просто разгледайте класовете.

Ако искате, можете да използвате и други хранилища.

## Подобряване на качеството на кода

Сега, след като сте идентифицирали някои от проблемите на кода, опитайте да предложите подобрения базирани на това, което вече знаете. Можете да добавите нова колона в таблицата от предишната задача или да направите нова, ето така:

|  |  |  |
| --- | --- | --- |
| **Свойство** | **Коментар** | **Подобрения** |
| Коментари по кода, написани на виетнамски | Коментарите не биха били разбираеми за много програмисти | Да се пренапишат коментарите на английски |
| Безполезни / безсмислени имена на променливи | Някои променливи са именувани неподходящо – tt, vmp и т.н. | Да се променят променливите:  tt -> numberOfClients  … |
| Много дълги методи | … | … |
| … | … | … |