

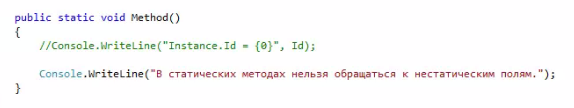
Статические переменные располагаются не в области памяти экземпляра, а в области памяти объекта, а значит поле будет разделяемым для всех экземпляров этого объекта

Из нестатических членах (методов) мы можем обращаться к статическим полям

На классе-объекте NonStaticClass обращаемся к полю field и прсваиваем 1



В статических методах нельзя обращаться к нестатическим полям, в связи с тем что неизвестно к какому полю класса мы обращаемся



Компилятор налагает на статический класс ряд ограничений.

 Класс должен быть прямым потомком System.Object — наследование любому

другому базовому классу лишено смысла, поскольку наследование применимо

только к объектам, а создать экземпляр статического класса невозможно.

 Класс не должен реализовывать никаких интерфейсов, поскольку методы ин-

терфейса можно вызывать только через экземпляры класса.

 В классе можно определять только статические члены (поля, методы, свойства

и события). Любые экземплярные члены вызовут ошибку компиляции.

 Класс нельзя использовать в качестве поля, параметра метода или локальной

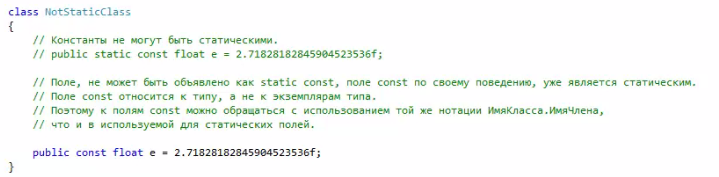
переменной, поскольку это подразумевает существование переменной, ссы-

лающейся на экземпляр, что запрещено. Обнаружив подобное обращение со

статическим классом, компилятор вернет сообщение об ошибке.

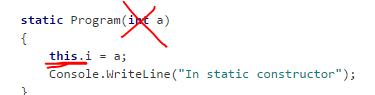
Константные поля

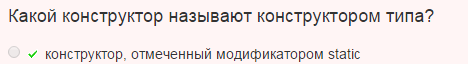
Не могут быть статическими

вызываем 

Обратится к статическому полю или константному полю можно только посредством объекта-класса



*Ключевое слово 'this' нельзя использовать в статическом свойстве, методе или инициализаторе статического поля.* 



Статические конструкторы

// Статические конструкторы обладают следующими свойствами:

// 1. Статический конструктор не имеет модификаторов доступа и не принимает параметров.

// 2. Статический конструктор вызывается автоматически для инициализации класса перед созданием первого экземпляра

// или ссылкой на какие-либо статические члены. (при обращении к статическому члену)

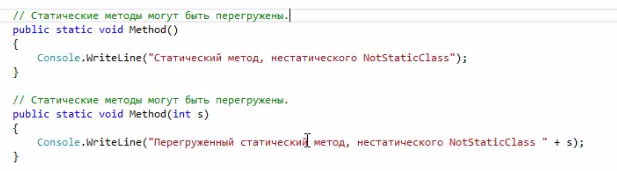
// 3. Статический конструктор нельзя вызывать напрямую.

// 4. Пользователь не управляет тем, когда статический конструктор выполняется в программе.

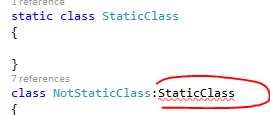
// 5. Типичным использованием статических конструкторов является случай, когда класс использует файл журнала и

// конструктор применяется для добавления записей в этот файл.

Статические методы допустимо перегружать, статические методы наследуются производным классом

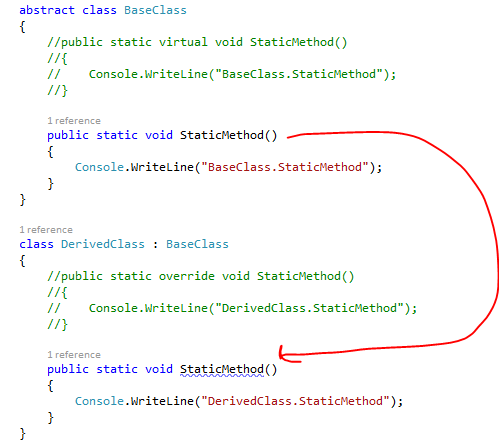


От статических классов наследование запрещено



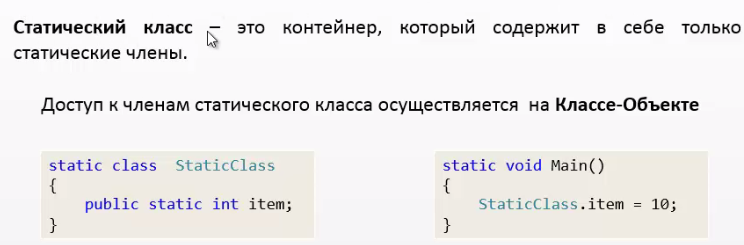
Статические члены не могут быть виртуальными (в связи с этим не поддерживают переопределения), абстрактными

НО ПОДДЕРЖИВАЮТ ЗАМЕЩЕНИЕ можно добавить ключевое слово new



Статические классы

По умолчанию имеет модификатор доступа SEALED но его явно использовать нельзя, т.к. уже по умолчанию он (это как static на const)



Может в себе содержать не статический класс

// Статический класс в C#, выражает идею паттерна проектирования - Singleton.

// Правила:

// 1. Экземпляр статического класса нельзя создать.

// 2. Static class всегда наследуется от Оbject (Попытка наследования от чего либо другого приводит к ошибке компиляции).

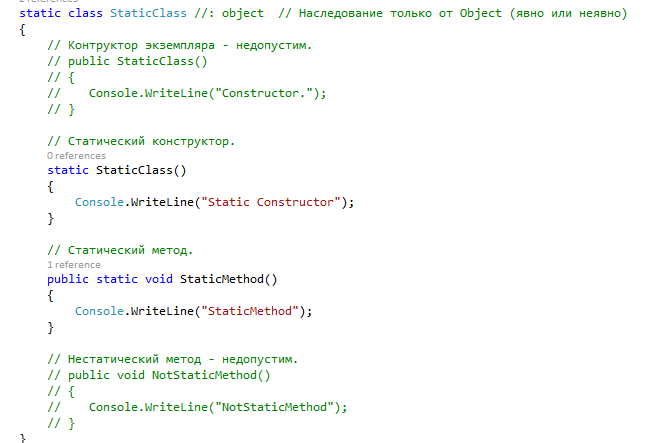
// 3. Static class не реализует интерфейсы. Попытка наследования от интерфейса приводит к ошибке уровня компиляции.

// 4. Содержит только статические члены (наличие в нем нестатического члена приведет к ошибке компиляции).

// 5. Статический класс не может содержать конструкторов экземпляров.

// 6. Статический класс закрыт для наследования от него. Попытка наследования от статического класса приводит

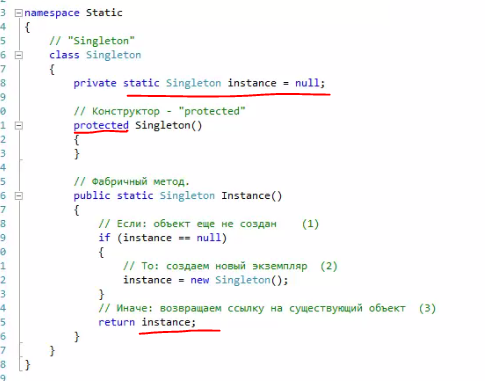
//к ошибке уровня компиляции.

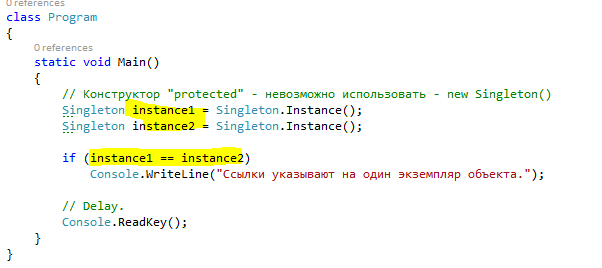


Шаблон проектирования

Шаблон проектирования Singelton – когда необходимо иметь только один экземпляр класса

Конструктор protected на случай если захочется расширить функциональность класса и мы смогли бы унаследоваться от этого класса (при модификаторе private сделать так бы не удалось)

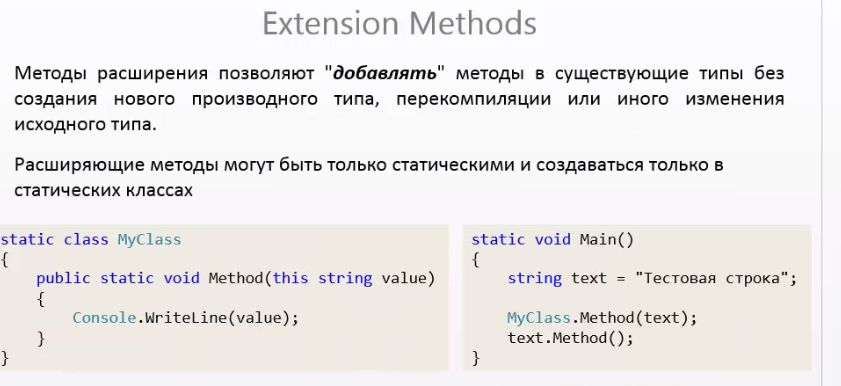
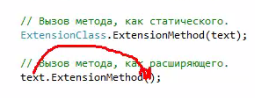




Расширяющие методы

this – аргумент расширения, только один и только первым аргументом

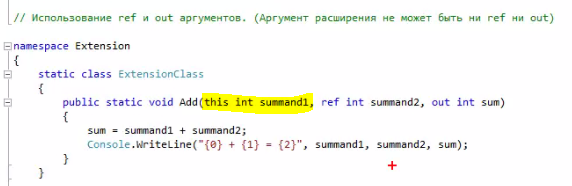
значок - 

, 

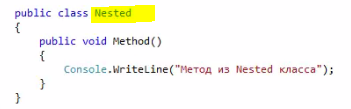
На строковом литерале вызываем метод расширения

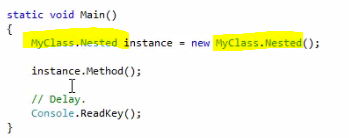


Сам аргумент расширения (this) не может быть ref out

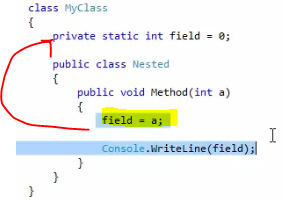


Вложенные классы

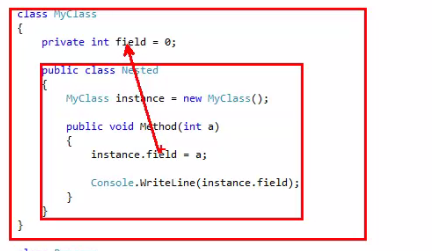




Можно обращаться к закрытым полям внешнего класса



и в данном примере инкапсуляция не работает

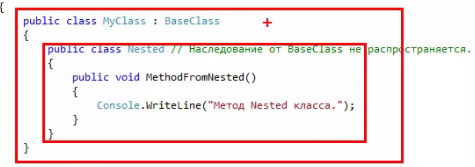


Наследование во вложенных классах допустима, но не распространяется на внешний класс (MyClass)

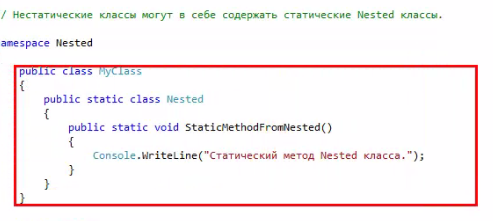


При наследовании в основном классе, наследование на вложенный класс не распространяется

Члены BaseClass в Nested доступны не будут



Нестатические классы могут содержать в теле статические вложенные классы



Может быть вложен интерфейс

