Основы программирования

Основы ООП

Члены класса

Поля и константы

Данные: переменные и константы

– переменные;
– константы.

Члены класса

Константа – это член класса, представляющий постоянное значение: значение, которое может быть вычислено во время компиляции.

(Спецификация языка C#)

Поле – это член, представляющий переменную, связанную с объектом или классом

(Спецификация языка C#)

Синтаксис описания констант класса

атрибуты модификаторы `const` *тип имя = значение;*

Например:

```
public const double pi=3.141, e=2.81;
```

Модификаторы доступа констант

- `public` – доступ к полю не ограничен;
- `private` – доступ только из данного класса (по умолчанию).
- `protected` – доступ только из самого класса и из производных классов (потомков);
- `internal` – доступ только в пределах текущей сборки (программы).

...

Константы могут зависеть от других констант внутри одной программы, если только зависимости не являются циклическими.

Компилятор автоматически располагает вычисления объявлений констант в соответствующем порядке.

Например:

```
class A   {
public const int X = B.Z + 1;
public const int Y = 10;      }

class B   {
public const int Z = A.Y + 1; }
```

Особенности констант

- Значение константы неизменно, следовательно должно быть проинициализировано при описании.
- Значение константы вычисляется во время компиляции.
- Константа существует в единственном экземпляре на все объекты класса.
- Обращение к константе производится по имени класса, а не по имени объекта.

Поля класса

- Поле описывает одну из характеристик состояния объекта.
- Все поля представляют собой полный набор характеристик экземпляров класса и предназначены для хранения значений этих характеристик.

Синтаксис описания поля класса

атрибуты модификаторы тип
имя = начальное значение;

Пример:

```
public static int myX=1, myY;
```

эквивалентно

```
public static int myX = 1;
public static int myY;
```

Тип, имя, начальное значение

- Тип поля – это тип данных, которому принадлежит значение, хранимое полем.
- Имя поля составляется, как правило, по нотации `Camel` (со строчной (=маленькой) буквы).
- Начальное значение – это то значение, которое будет принимать поле в момент создания любого экземпляра класса. Задавать начальное значение при описании поля не обязательно.

Модификаторы доступа полей

- `public` – доступ к полю не ограничен;
- `private` – доступ только из данного класса (по умолчанию).
- `protected` – доступ только из самого класса и из производных классов (потомков);
- `internal` – доступ только в пределах текущей сборки (программы).

Модификаторы

- `new` – новое поле, скрывающее поле, унаследованное от предка ;
- `readonly` – поле доступно только для чтения;
- `static` – одно поле для всех экземпляров данного класса;
- `volatile` – поле предназначено для совместного использования несколькими потоками инструкций.

Доступ к константам и полям класса

Для доступа используется операция доступа (разыменования) «.» (точка):

```
имя_класса.имя_константы
```

(Константы используются в контексте класса)

```
имя_экземпляра.имя_поля
```

(Поля используются в контексте объекта)

Если поле закрыто, то такое обращение невозможно.

Статические поля

Поле объявленное с модификатором `static` является статическим.

Оно сохраняет состояние класса в целом, а не отдельного объекта и существуют в единственном экземпляре.

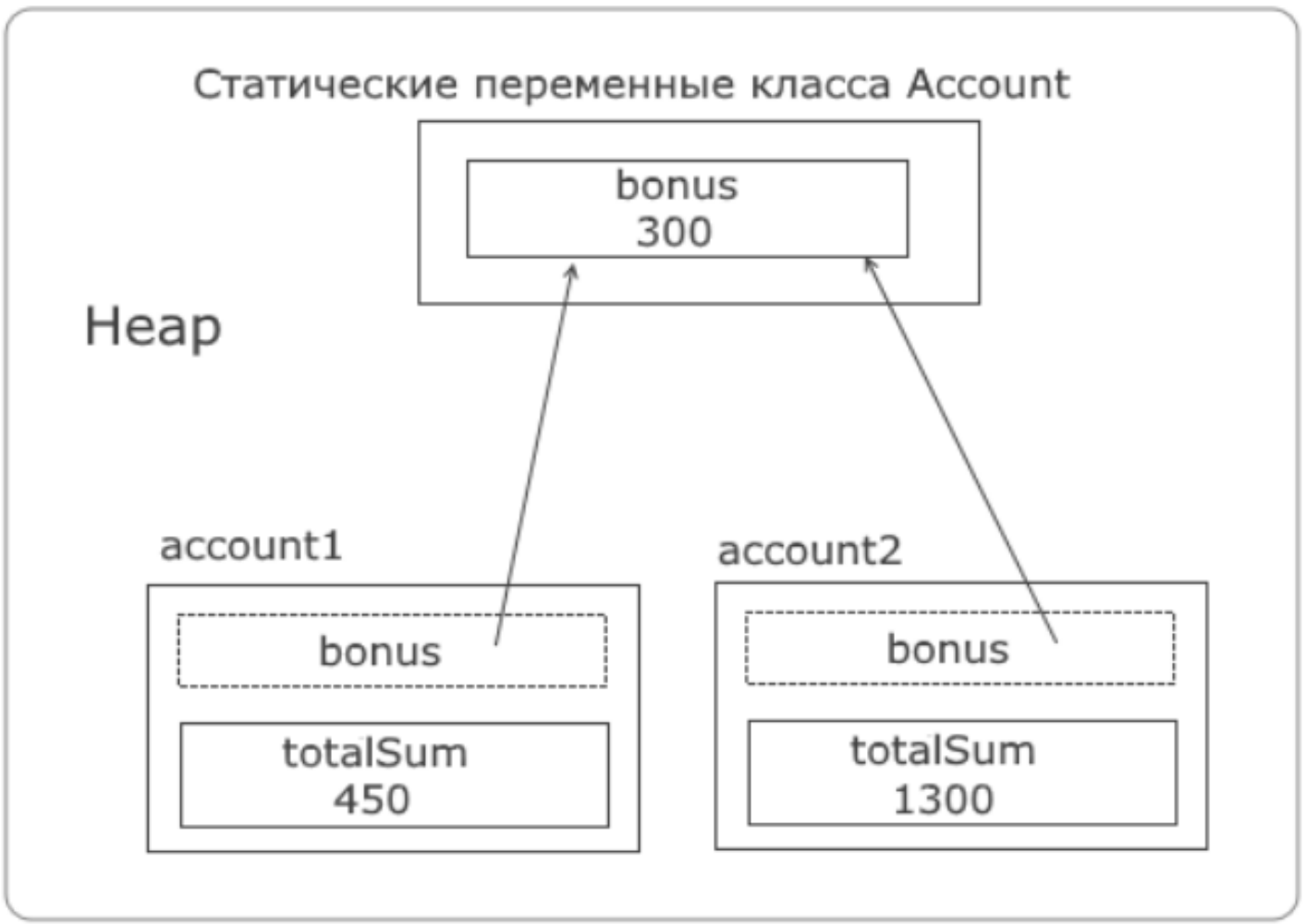
Должны быть сразу проинициализированы.

Их значения могут быть изменены.

Обращаться к этому полю необходимо по имени класса:

```
Account.bonus += 200;
```

Статические поля



Статические поля и поля экземпляров

Если объявление поля включает модификатор `static`, поля, являются *статическими полями*.

Если нет, то введенные объявлением поля являются *полями экземпляров*.

Их иногда называют *статическими переменными* и *переменными экземпляров*, соответственно.

Пример

```
class MyClass {
  public const int publicConst = 2018;
                    //открытая константа
  public double publicField = 28;
                    // открытое поле (не рекомендуется!)
  public static string staticField =
           ”6100”;
                    //статическое поле
  float privateField = 1.2;
                    //непубличное (private) поле
}
```

Пример. Продолжение

```
class Program {
  public static void Main()    {
  MyClass c = new MyClass();
            //создание объекта (экземпляра)
  double x =
     c.publicField+MyClass.publicConst;
            //обращение к открытому полю и константе
  Console.WriteLine(MyClass.staticField);
            //обращение к статическому полю
  Console.WriteLine(c.privateField);
            //НЕДОПУСТИМО!
  }
}
```

Модификатор readonly

Если объявление поля включает модификатор `readonly`, поля, введенные этим объявлением, являются *полями только для чтения*.

Поля для чтения можно инициализировать при их объявлении либо на уровне класса, либо инициализировать и изменять в конструкторе.

В других местах инициализировать или изменять их значение нельзя, можно только считывать их значение.

Сравнение констант и полей readonly

- Константы должны быть определены во время компиляции, а поля для чтения могут быть определены во время выполнения программы. Соответственно инициализировать константу можно только при ее определении.
- Поле для чтения можно инициализировать либо при его определении, либо в конструкторе класса.
- Константы не могут быть статическими. Поля для чтения могут быть статическими.

Использование статических полей `readonly` вместо констант

- Если значение не может быть вычислено во время компиляции, значит их нельзя объявить как `const`. В этом случае полезно поле `static readonly`.
- Это объявление `static readonly` имеет почти такой же результат.

Инициализация поля

- Начальным значением поля, как статического, так и поля экземпляра, является значение по умолчанию типа поля.
- Таким образом поле никогда не бывает «неинициализированным». Иначе он было бы просто не видно.