Static Initializers

When a class has instance variables that need initialization that involves more than simple first values, the initialization can be done in a constructor that is executed when objects of the class are instantiated.

Class variables, however, belong to the class, not to the objects.

- Class variables need to be initialized when the class is first loaded.

  static double interest = 6.75;

- If this initialization involves

  iteration

  decision making

  multiple commands

  it can be perform in a special anonymous block, called a *static initializer*, that resides in the class.

```
static
{
      // commands that initialize
      // class variables
}
```

When a class is loaded, initialization of all class (static) variables is performed and all static initializers are executed in the order they occur in the class definition.


```
class StrangeExample
{
    static int num, size;                          // first
    static double total = 0.0;                     // second
    static double [] list;                         // third
    static
    {   size = readInt();                          // fourth
        if (size<0) size = 10;                     // fifth
        list = new double [size];                  // sixth
    }
    static String message = "Strange";             // seventh
    static int readInt()
    {
        try
        {
            BufferedReader br = new BufferedReader(
                        new InputStreamReader(System.in));
            System.out.print("Enter size: ");
            return Integer.parseInt(br.readLine());
        }
        catch (Exception e)     // IOException or
        {    return 0;      }    // NumberFormatException
    }

    public static void main(String [] a)
    {
        System.out.println(message);
        System.out.println(list.length);
    }
}
```

Good style suggests that a class have only one static initializer, but no rule limits their number.

Note: Static initializers may access only class variables and class methods (directly), not instance variables and methods.

Example

Suppose a particular class has a class variable that is intended to refer to an array containing the 55 dominoes in a set (MAXSPOTS = 9).

Although this class variable can be initialized using an array literal,

{ **new** Domino(0,0,**false**), **new** Domino(0,1,**false**), ...,

new Domino(8,9,**false**), **new** Domino(9,9,**false**) }

a static initializer will do the job with much less programmer effort.

Using a Static Initializer

This static initializer builds a set of dominoes with any value of MAXSPOTS≥0.

The number of dominoes in such a set is

(MAXSPOTS+1)*(MAXSPOTS+2)/2.

For MAXSPOTS = 9,

number of dominoes = 10*11/2 = 55.


```
class UseDominoes
{
   static int size = (Domino.MAXSPOTS+1)*
                                 (Domino.MAXSPOTS+2)/2;

   static Domino [ ] dominoSet = new Domino [size];

   static
   {
        int index = 0;
        for (int m=0; m<=Domino.MAXSPOTS; m++)
            for (int n=m; n<=Domino.MAXSPOTS; n++)
            {
                dominoSet[index] = new Domino(m,n,false);
                index++;
            }
   }              // end of static initializer

   :              // rest of class
}
```