*Сборка* является неотъемлемой частью программирования на С#. Она представляет

собой один или несколько файлов, содержащих все необходимые сведения о развертывании

программы и ее версии. Сборки составляют основу среды .NET. Они предоставляют

механизмы для надежного взаимодействия компонентов, межъязыковой возможности

взаимодействия и управления версиями. Кроме того, сборки определяют

область действия программного кода.

Сборка состоит из четырех разделов. Первый раздел представляет собой *декларацию*

сборки. Декларация содержит сведения о самой сборке. К этой информации относится,

в частности, имя сборки, номер ее версии, сведения о соответствии типов

и параметры культурной среды (язык и региональные стандарты). Второй раздел

сборки содержит метаданные типов, т.е. сведения о типах данных, используемых в

программе. Среди прочих преимуществ *метаданные типов* способствуют межъязыковой

возможности взаимодействия. Третий раздел сборки содержит *программный*

*код* в формате MSIL (Microsoft Intermediate Language — промежуточный язык корпорации

Microsoft). И четвертый раздел сборки содержит ресурсы, используемые программой.

Правда, при программировании на C# сборки получаются автоматически, требуя

от программирующего лишь минимальных усилий. Дело в том, что исполняемый

файл, создаваемый во время компиляции программы на С#, на самом деле представляет

собой сборку, содержащую исполняемый код этой программы, а также другие

виды информации. Таким образом, когда компилируется программа на С#, сборка

получается автоматически.

У сборок имеется много других особенностей, и с ними связано немало актуальных

вопросов программирования, но, к сожалению, их обсуждение выходит за рамки этой

книги. Ведь сборки являются неотъемлемой частью процесса разработки программного

обеспечения в среде .NET, но формально они не относятся к средствам языка С#.

Тем не менее в C# имеется одно средство, непосредственно связанное со сборкой. Это

модификатор доступа internal, рассматриваемый в следующем разделе.

**Модификатор доступа internal**

Помимо модификаторов доступа public, private и protected, использовавшихся

в представленных ранее примерах программ, в C# предусмотрен также модификатор

доступа internal. Этот модификатор определяет доступность члена во всех

файлах сборки и его недоступность за пределами сборки. Проще говоря, о члене, обозначенном

как internal, известно только в самой программе, но не за ее пределами.

Модификатор доступа internal особенно полезен для создания программных компонентов.

Модификатор доступа internal можно применять к классам и их членам, а также

к структурам и членам структур. Кроме того, модификатор internal разрешается

использовать в объявлениях интерфейсов и перечислений.

Из модификаторов protected и internal можно составить спаренный модификатор

доступа protected internal. Уровень доступа protected internal может

быть задан только для членов класса. Член, объявленный как protected internal,

доступен лишь в пределах собственной сборки или для производных типов.

Ниже приведен пример применения модификатора доступа internal.

(***glava16\_12***)

class InternalTest

{

internal int x;

}

class InternalDemo

{

static void Main()

{

InternalTest ob = new InternalTest();

ob.x = 10; //доступно, потому что находится в том же файле

Console.WriteLine("Value ob.x: " + ob.x);

}

}

В классе InternalTest поле х объявляется как internal. Это означает, что поле х

доступно в самой программе, но, как показывает код класса InternalDemo, оно недоступно

за пределами программы.