Описание методов в Си#. Синтаксис

В языке C# нет специальных ключевых слов - **procedure** и **function**, но присутствуют сами эти понятия. Синтаксис объявления метода позволяет однозначно определить, чем является метод - *процедурой* или *функцией*.

Синтаксис описания метода:

[**Модификаторы**] **Тип Имя** **([список формальных параметров])**
{**Тело метода**}

Модификаторы могут отсутствовать и определяют область видимости метода. Имеют четыре возможных значения:

Модификатор	Назначение
public	метод открыт и доступен для вызова клиентами и потомками класса
private	метод предназначен для внутреннего использования в классе и доступен для вызова только в теле методов самого класса
protected	метод будет доступен как из класса, в котором он определен, так и из любого производного класса. Для остальных вызовов из вне этот метод будет недоступен.
internal	метод будет доступен из всех классов внутри сборки, в которой он определен. Из-за пределов этой сборки обратиться к нему будет нельзя.

Если модификатор доступа опущен, то по умолчанию предполагается, что он имеет значение private и метод является ***закрытым*** для клиентов и потомков *класса*.

Обязательным при ***описании заголовка*** является **указание типа результата**, **имени метода** и **круглых скобок**, наличие которых необходимо и в том случае, если сам список формальных аргументов отсутствует.

Тело метода заключается в фигурные скобки.

Формально тип результата метода указывается всегда, но значение void однозначно определяет, что метод реализуется *процедурой*.

Тип результата, отличный от void, указывает на *функцию*.

В теле функции обязательно указывается оператор **return (возврат),** который присваивает функции нужное значение.

Если метод представляет собой процедуру, то в нем нельзя указывать оператор **return.**

Примеры описания методов:

```
void A() {...};
int B(){. . .};
public void C(){...};
```

Методы A и B являются *закрытыми*, а метод C - *открыт*. Методы A и C реализованы *процедурами*, а метод B - *функцией*, возвращающей целое значение.

Методы могут быть объявлены как статические — с использованием ключевого слова **static**.

Это значит, что статический метод (static method) может быть вызван напрямую через уровень класса, без необходимости создавать хотя бы один экземпляр объекта данного класса.

По этой причине метод Main() всегда объявляется как static — чтобы этот метод мог начать выполняться еще до создания первого экземпляра класса, в котором он определен.

Список формальных параметров

Список формальных параметров метода может быть пустым или содержать фиксированное число параметров, разделяемых символом запятой.

Синтаксис объявления формального параметра:

[ref|out|params] тип_параметра имя_параметра

Обязательным является указание типа и имени параметра, причем никаких ограничений на тип не накладывается. Он может быть любым скалярным типом, массивом, классом, структурой, интерфейсом, перечислением, функциональным типом.

Несмотря на фиксированное число формальных аргументов, есть возможность при вызове метода передавать ему **произвольное число фактических аргументов**. Для реализации этой возможности в списке формальных аргументов необходимо задать ключевое слово **params**.

Оно задается один раз и указывается только для последнего параметра списка, объявляемого как массив произвольного типа. При вызове метода этому формальному параметру соответствует произвольное число фактических аргументов.

Выделяют три группы аргументов метода:

- **входные in** (задаются без ключевого слова, передают информацию методу, их значения в теле метода только читаются)
- **выходные out** (результаты метода, они получают значения в ходе работы метода, следовательно, их не надо инициализировать)
- **обновляемые ref** (выполняют обе функции [in, out], их значения используются в ходе вычислений и обновляются в результате работы метода).

Если параметр объявлен как выходной с ключевым словом out, то в теле метода обязательно должен **присутствовать оператор присваивания, задающий значение этому аргументу.** В противном случае возникает ошибка еще на этапе компиляции.

Выделяют **два способа передачи параметров**:

- **по значению** (работа с копией переменной, которая уничтожается по окончанию работы метода);
- **по ссылке** (работа с самой переменной по ссылке).