C# Statements

Statement — **Example**

Statement lists and block statements

```
static void Main() {
    F();
    G(); {
         H();
         I();
    }
}
```

Labeled statements and goto statements

```
static void Main(string[] args) {
    if (args.Length == 0)
         goto done;
    Console.WriteLine(args.Length);
done:
    Console.WriteLine("Done");
}
```

Local constant declarations

```
static void Main() {
    const float pi = 3.14f;
    const int r = 123;
    Console.WriteLine(pi * r * r);
}
```

Local variable declarations

```
static void Main() {
    int a;
    int b = 2, c = 3;
    a = 1;
    Console.WriteLine(a + b + c);
}
```

Expression statements

```
static int F(int a, int b) {
    return a + b;
}
static void Main() {
    F(1, 2); // Expression statement
}
```

if statements

```
static void Main(string[] args) {
    if (args.Length == 0)
         Console.WriteLine("No args");
    else
         Console.WriteLine("Args");
}
```

switch statements

```
static void Main(string[] args) {
    switch (args.Length) {
case 0:
    Console.WriteLine("No args");
    break;
case 1:
    Console.WriteLine("One arg ");
    break;
default:
    int n = args.Length;
    Console.WriteLine("{0} args", n);
    break;
    }
}
```

while statements

```
static void Main(string[] args) {
    int i = 0;
    while (i < args.Length) {
         Console.WriteLine(args[i]);
         i++;
    }
}
```

do statements

```
static void Main() {
    string s;
    do {
         s = Console.ReadLine();
    } while (s != "Exit");
}
```

SOURCE: ECMA Standard ECMA-334, December 2001

Statement

Example

for statements

```
static void Main(string[] args) {
    for (int i = 0; i < args.Length; i++)
         Console.WriteLine(args[i]);
}
```

foreach statements

```
static void Main(string[] args) {
    foreach (string s in args)
         Console.WriteLine(s);
}
```

break statements

```
static void Main(string[] args) {
    int i = 0;
    while (true) {
         if (i == args.Length)
              break;
         Console.WriteLine(args[i++]);
    }
}
```

continue statements

```
static void Main(string[] args) {
    int i = 0;
    while (true) {
         Console.WriteLine(args[i++]);
         if (i < args.Length)
              continue;
         break;
    }
}
```

return statements

```
static int F(int a, int b) {
    return a + b;
}
static void Main() {
    Console.WriteLine(F(1, 2));
    return;
}
```

throw statements and try statements

```
static int F(int a, int b) {
    if (b == 0)
         throw new Exception("Divide by zero");
    return a / b;
}
static void Main() {
    try {
         Console.WriteLine(F(5, 0));
    }
    catch(Exception e) {
         Console.WriteLine("Error");
    }
}
```

checked and unchecked statements

```
static void Main() {
    int x = Int32.MaxValue;
    Console.WriteLine(x + 1); // Overflow
    checked {
         Console.WriteLine(x + 1); // Exception
    }
    unchecked {
         Console.WriteLine(x + 1); // Overflow
    }
}
```

lock statements

```
static void Main() {
    A a = foo;
    lock(a) {
         a.P = a.P + 1;
    }
}
```

using statements

```
static void Main() {
    using (Resource r = new Resource()) {
         r.F();
    }
}
```