C# Cheat Sheet

Data Types	
bool	Boolean value
byte	8-bit unsigned integer
char	16-bit Unicode character
decimal	128-bit precise decimal values with digits
double	64-bit double-precision floating poi
float	32-bit single-precision floating poin
int	32-bit signed integer
long	64-bit signed integer
object	Base type for all other types
sbyte	8-bit signed integer
short	16-bit signed integer
string	String value
uint	32-bit unsigned integer
ulong	64-bit unsigned integer
ushort	16-bit unsigned integer

Type Conversion Methods
ToBoolean
ToByte
ToChar
ToDateTime
ToDecimal
ToDouble
ToInt16
ToInt32
ToInt64
ToSbyte
ToSingle
ToString
ToType
ToUInt16
ToUInt32
ToUInt64

Naming Conventions	
Class	MyClass
Method	MyMethod
Local variable	myLocalVariable
Private variable	_myPrivateVariable
Constant	MyConstant

Arrays
int[] array = new int[] {1, 2, 3}
int[] array = {1, 2, 3}
var array = new int[] {1, 2, 3}
int[] array = new int[3]

C# Cheat Sheet

Statements	
if-else	if (true) {...} {{nl}} else {{nl}} else {...}
switch	switch (var) {{{nl}}case default: break; }
for	for (int i =1; i < 5; i++)
foreach-in	foreach (int item in array
while	while (true) {...}
do... while	do {...} {{nl}} while (tru
try-catch-finally	try {...} {{nl}} catch (Ex {{nl}} catch {...} {{nl}}

Classes		
Class	public class Dog {...}	
Inheritance	public class Dog: Pet {...}	
Constructor (no parameters)	public Dog () {...}	Cor co-e
Constructor (one parameter)	public Dog (string var) {...}	Cor co-e
Field	public string name	
Static Class	public static class Dog {...}	Mus mer
Static Member	public static int = 1	
Finalizer (destructor)	~Dog () {...}	Cannot have modifiers or parameters

Access Modifiers	
public	Accessible by any other code in the same assembly or another assembly that references it
private	Only accessible by code in the same class or struct
protected	Only accessible by code in the same class or struct, or in a derived class
internal	Accessible by any code in the same assembly, but not from another assembly
protected internal	Accessible by any code in the same assembly, or by any derived class in another assembly

C# Cheat Sheet

Other Modifiers	
abstract	Indicates that a class is intended c class of other classes
async	Indicates that the modified metho expression, or anonymous method
const	Specifies that the value of the fiel variable cannot be modified
event	Declares an event
extern	Indicates that the method is impl
new	Explicitly hides a member inherite
override	Provides a new implementation o inherited from a base class
partial	Defines partial classes, structs and throughout the same assembly
readonly	Declares a field that can only be a part of the declaration or in a con same class
sealed	Specifies that a class cannot be in
static	Declares a member that belongs t instead of to a specific object
unsafe	Declares an unsafe context

Other Modifiers (cont)	
virtual	Declares a method or an accessor implementation can be changed b member in a derived class
volatile	Indicates that a field can be modi by something such as the operati hardware, or a concurrently execu

Assignment Operators	
=	Simple assignment
+=	Addition assignment
-=	Subtraction assignment
*=	Multiplication assignment
/=	Division assignment
%=	Remainder assignment
&=	AND assignment
|=	OR assignment
^	XOR assignment
<<=	Left-shift assignment
>>=	Right-shift assignment

Comparison Operators	
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to
==	Equal to
!=	Not equal to

Arithmetic Operators	
+	Add numbers
-	Subtract numbers
*	Multiply numbers
/	Divide numbers
%	Compute remainder of division of numbers
++	Increases integer value by 1
--	Decreases integer value by 1

C# Cheat Sheet

Logical and Bitwise Operators	
&&	Logical AND
||	Logical OR
!	Logical NOT
&	Binary AND
|	Binary OR
^	Binary XOR
~	Binary Ones Complement
<<	Binary Left Shift
>>	Binary Right Shift

Other Operators	
sizeof()	Returns the size of a data type
typeof()	Returns the type of a class
&	Returns the address of a variable
*	Pointer to a variable
? :	Conditional expression
is	Determines whether an object is of a specific type
as	Cast without raising an exception if the cast fails