

C# Overview

Part I

Tomáš Matoušek
&
Ladislav Prošek

tmd.havit.cz/teaching/csharp.htm

C# Program Structure

- types
 - code and data descriptors
 - visibility: public, internal (default)
- code defined in methods
 - no global functions (C# limitation, not the CLR one)
 - visibility: public, private (default), protected, internal, internal protected
- data defined by fields
 - called “attributes” in Java
 - no global variables (C# limitation)
 - visibility: public, private (default), protected, internal, internal protected
- metadata
 - generated by the C# compiler
 - custom metadata (attributes) can be defined and used in C#

Preprocessor Directives

- `#define`, `#undef`
 - define/remove a symbol
 - no value possible (no macros)
 - at the beginning of a source file
 - compiler option `/define`
- `#if`, `#else`, `#elif`, `#endif`
 - conditional compilation depending on defined symbols
- `#region`, `#endregion`
 - ignored by preprocessor, useful for tools like VS.NET

Namespaces

- types declared in namespaces
 - dot notation, e.g. System.Collections.Hashtable
 - subnamespaces
 - not related to directory structure nor source files (unlike Java packages)

```
namespace A.B.C
{
    namespace D
    {
        class MyClass1 { ... } // class's namespace is A.B.C.D
        class MyClass2 { ... }
    }
    class MyClass3 { ... } // class's namespace is A.B.C
}
```

- using keyword
 - allows using short names of types declared in the specified namespace
 - allows create type name aliases and namespace aliases

```
using MyCls = A.B.C.D.MyClass1;
using MyNamespace = A.B;
using A.B.C;
```

BCL Namespaces

- Basic Class Library namespaces (implemented in mscorlib)
 - System
 - System.Text
 - System.Reflection
 - System.Threading
 - System.IO
 - System.Collections
 - System.Globalization
 - System.Diagnostics
 - System.Security
 - ...