Classes

Classes are building blocks of software which encapsulate data and behavior.

Anatomy: Name, Fields, Methods

Declaration: </> `Access Mod ifier + class + ClassName`

Constructors: (code snippet: **ctor**)

- Constructors are methods that are called when an instance of a class is instantiated.
- Constructors have no return type.
- Constructors can be overloaded with different signatures.

</> `public class ClassName() // default constr uctor { }`

</> `public class ClassName(int parameter) // overloa d : this() //calls the default constr uctor {}`

Functions/Methods

- Methods are the functions of classes.
- Methods can be overloaded with different signatures.

Declaration: </> `Access Parameter + Return Type + (Argument Type argum ent, ...) {}`

Varying number of parameters:

</> `public void MethodName(params int[] arguments) {}`

Calling a function/method:

`return Value = Functi onName(argum ents)`

`return Value = ObjectName.MethodName(argum ents)`

Conventions

PascalCase	classes, methods, properties
camelCase	method parameters
_camelCase	private fields

Access Modifiers

`public`	accessible everywhere
`private`	accessible only from inside the class
`protected`	
`internal`	
`protected internal`	

Usable for classes, methods and variables.

Objects

`var objectName = new ClassName();`	Create an object
`objectName.Method Name()`	Access public class methods
`var x = Object Nam e.P rop ert yN ame()`	*Gets* an object property.
`Object Nam e.P rop ert yName = x`	*Sets* an object property.

Sponsored by **Readable.com**
Measure your website readability!
https://readable.com