A C# app is composed of Classes and Interfaces.

> 一个 C# 应用由类和接口组成。

Class is like a blueprint of specific object. A Class wraps fields/properties and methods, can generate an event (defined by delegates) and throw an exception.

> 类就像特定对象的蓝图。类封装字段/属性和方法，可以生成事件（由委托定义）并抛出异常。

A C# class is composed of:

> 一个 C# 类由以下部分组成：

constructor

> 构造函数

Fields

> 字段

Properties

> 属性

methods

> 方法

Events

> 事件

Delegates

> 委托

The members of a class could be public, protected, private, internal, protected internal and private protected.

> 类的成员可以是 public、protected、private、internal、protected internal 和 private protected。

The C# access modifiers are:

> 这些 C# 访问修饰符为：

public

private
The type or member can be accessed only by code in the same class or struct.

> 该类型或成员只能被同一类或结构中的代码访问。

protected
The type or member can be accessed only by code in the same class, or in a class that is derived from that class.

> 该类型或成员只能被同一类中的代码或从该类派生的类中的代码访问。

internal
The type or member can be accessed by any code in the same assembly, but not from another assembly.

> 该类型或成员可以被同一程序集中任何代码访问，但不能被其他程序集访问。

protected internal The type or member can be accessed by any code in the assembly in which it is declared, or from within a derived class in another assembly.

> protected internal 该类型或成员可以被其声明所在程序集中的任何代码访问，或被另一个程序集中派生类的代码访问。

private protected The type or member can be accessed only within its declaring assembly, by code in the same class or in a type that is derived from that class.

> private protected 该类型或成员仅能在其声明的程序集内由同一类中的代码或从该类派生的类型中的代码访问。

A C# class can be:

> A C# 类可以是：

public

private

> 私有

internal

> 内部

abstract

sealed (java final)

> 密封的（java final）

nested (java inner)

> 嵌套的（java inner）

partial

A nested class can be private (as now it is simply a member of the outer class). A class could be partial.

> 嵌套类可以是私有的（因为它现在仅仅是外部类的一个成员）。 一个类可以是部分类。

Classes could be grouped/scoped as follows:

> 类可以按以下方式分组/ 作用域化：

Namespace (same as java package}

> 命名空间（与 Java 包相同}

Assemblies/dlls (contain one or more namesspaces, somewhat parallel to jar files but not quite)

> 程序集/dll（ 包含一个或多个命名空间， 在某种程度上类似于 jar 文件，但又不完全相同）

A method could be abstract or virtual, or can have override, new or sealed (final) keywords.

> 方法可以是抽象的或虚拟的，或可以带有 override、new 或 sealed（最终） 关键字。

new

hides an inherited (i.e. non virtual) member from a base class member

> 从基类成员中隐藏一个继承的 （即非虚拟 ） 成员

override

extends the base class method

> 扩展 基类方法

virtual

virtual methods have an implementation and provide the derived classes with the option of overriding it.

> 虚方法有实现，并为派生类提供覆盖它的选项。

abstract

abstract methods do not provide an implementation and force the derived classes to override the method. So, abstract methods have no actual code in them, and subclasses HAVE TO override the method.

> abstract 方法不提供实现，并强制派生类重写该方法。因此，abstract 方法内部没有实际代码，子类必须重写该方法。

Note: The virtual modifier cannot be used with static, abstract and override modifiers.

> 注意：virtual 修饰符不能与 static、abstract 和 override 修饰符一起使用。

Everything inherits from Object. C# supports single inheritance only. The following are not inherited:

> 所有类型都继承自 Object。C# 仅支持单继承。以下内容不被继承：

Static constructors, which initialize the static data of a class.

> 静态构造函数，用于初始化类的静态数据。

Instance constructors, which you call to create a new instance of the class. Each class must define its own constructors.

> 实例构造函数，用于创建类的新实例。每个类必须定义它自己的构造函数。

Finalizers, which are called by the runtime's garbage collector to destroy instances of a class.

> 终结器，由运行时的垃圾回收器调用以销毁类的实例。

Base class constructor, for example, could be called as follows:

> Base 类构造函数，例如，可以按如下方式调用：

```
public subclass() : base() {
```

// do subclass constructor stuff here

> // 在此处执行子类构造函数的相关操作

}

Try-catch-finally block is as follows:

> try-catch-finally 块如下：

try – A try block is used to encapsulate a region of code. ...

> try – try 块用于封装一段代码区域。...

catch – When an exception occurs, the Catch block of code is executed. ...

> catch – 当发生异常时，将执行 catch 代码块。...

finally – The finally block allows you to execute certain code if an exception is thrown or not.

> finally – finally 块允许你在是否抛出异常的情况下执行某些代码。

A c# generic is class encapsulates operations that are not specific to a particular data type.

> A c# 泛型是封装不特定于某一数据类型的操作的类。

A c# interfaces can be declared as public or internal; its members are always public. Interfaces can contain events, indexers, methods, and properties but no implementation of methods.

> A c# 接口 可以 声明为 public 或 internal；其 成员始终为 public。 接口 可以包含事件、索引器、方法和属性，但 不包含方法的实现。

Self study other c# keywords e.g. volatile, const, readonly and concept of an indexer

> 自学其他 c# 关键字，例如 volatile、const、readonly 以及索引器的概念.