属性:

> 可以看成是在直接访问字段的过程中间增加对字段的访问控制,本质就是一个get方法 和一个set方法

语法:

```
访问修饰符 类型  属性名
{
      get{ } //需要返回值
      set{ } //value关键字 表示调用时 =号 右边的内容
}
```

```
//自动属性
public int Age{get;set;}// 自动属性在编译时自动生成字段,不需要手动写字段
```

结构体

结构和类的语法几乎一样,结构体是值类型的

为什么要使用结构体?

答:为了让一些高频率小体积的数据能够快速的访问,且不产生内存垃圾,在Unity 中Vector3被定义成结构体就是这个道理

定义结构体的语法

```
//语法格式:
struct 结构名
{
}
//例如定义一个结构体表示位置
struct Vector3
{
    public int x;
    public int y;
```

```
    public int z;
    //和类一样也可以包含方法,这里就不写了
}


```

创建结构对象

1.使用new关键字创建结构对象(和创建类对象相同)
2.结构体可以不用new

结构体的特点

结构可带有方法、字段、索引、属性、运算符方法和事件。
结构体无法申明默认构造函数(公共无参构造函数)
 结构体不可以继承,但可以实现接口
 构造函数中必须对所有字段赋值
结构可以不使用 New 操作符即可被实例化。
如果不使用 New 操作符，只有在字段被初始化(被赋值)以后字段才可以使用;

结构体和类的区别

```
1.类是引用类型，结构是值类型。
2.结构不支持继承。
3.结构不能声明默认的构造函数。
```

常量:

const　　编译时常量 定义时候必须赋值不能使用静态修饰符
readonly　运行时常量 只能在定义时或者构造函数中赋值 其他任何地方不能修改或赋值

枚举:

枚举是一组命名整型常量。
使用枚举将有限的选项列举出来,类似于将填空题改为选择题,方便调用者

定义枚举语法:

```
//语法格式:
enum 枚举名
{
    项名1=值1,
    项名2=值2,
}
//例如表示性别的枚举
enum Sex
{
    boy=1,
    girl=2
}
```

使用枚举中的常量:

枚举名.枚举项

标志枚举:[Flags]

标志枚举用于多个枚举值组合的情况,例如文件属性中的可读,可写,隐藏
枚举中各标志的值应该是以 2 的幂来赋值，即：1、2、4、8、16、32……
使用按位或运算符"|" 表示多个状态

原理:
2的n次方数的二进制只有1位是1, Unity中层也是使用这个原理

枚举转换

```
    class Program
    {
        static void Main(string[] args)
        {
```

```
            Sex sex = Sex.girl;

            //枚举项转整数
            int i = (int)sex;
            Console.WriteLine(i);
            //整数转枚举项
            Sex sex1 = (Sex)2;
            Console.WriteLine(sex1);
            //枚举项转字符串
            string str = sex1.ToString();
            Console.WriteLine(str);
            //字符串转枚举项
            string str1 = "girl";
            Sex sex2= (Sex) Enum.Parse(typeof(Sex), str1) ;
            Console.WriteLine(sex2);
        }
    }
    enum Sex
    {
        boy=1,
        girl=2
    }
```

命名空间

定义命名空间的语法

```
    namespace 名称
     {
        //可以在命名空间中定义类,结构体,枚举 等
     }
```

引用命名空间

```
using 命名空间名称;
```

作业

画C#基本语法的思维导图