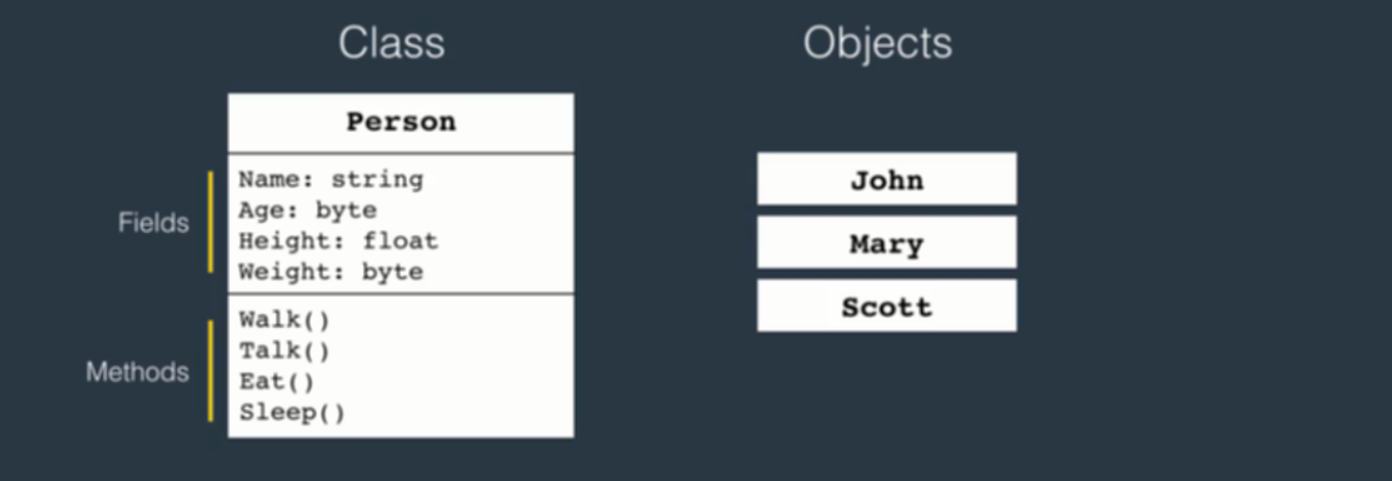
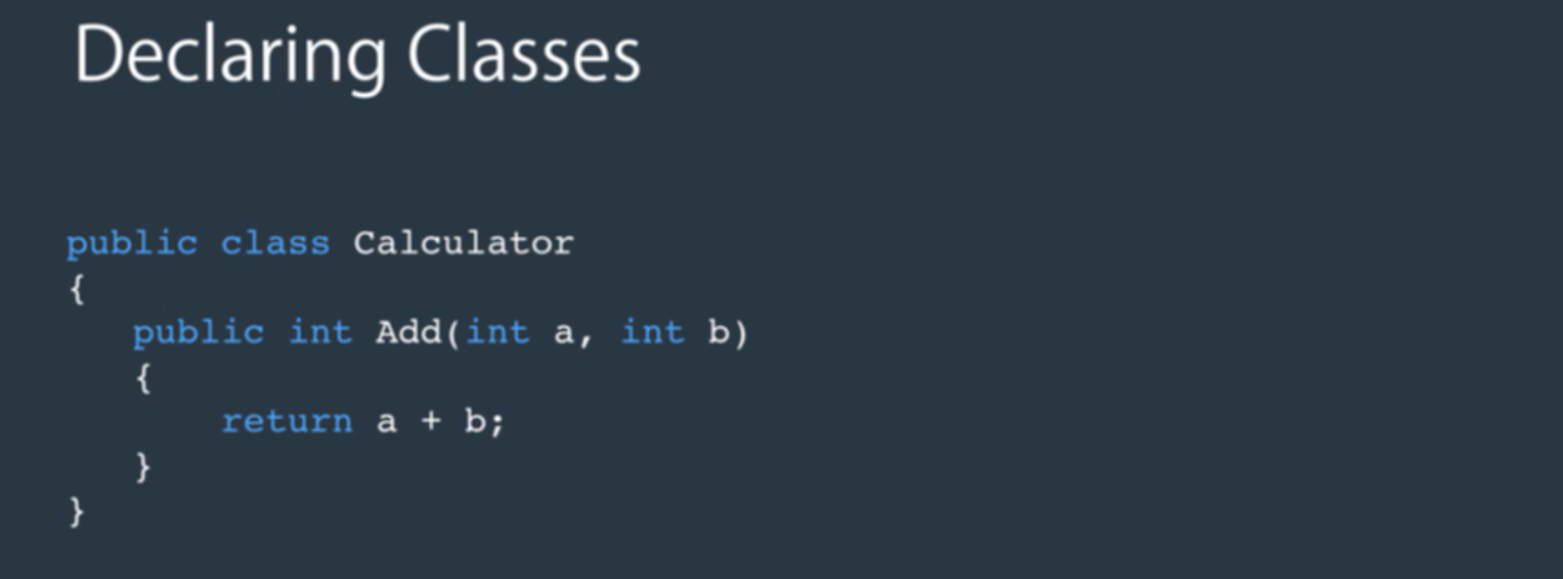
2. Classes



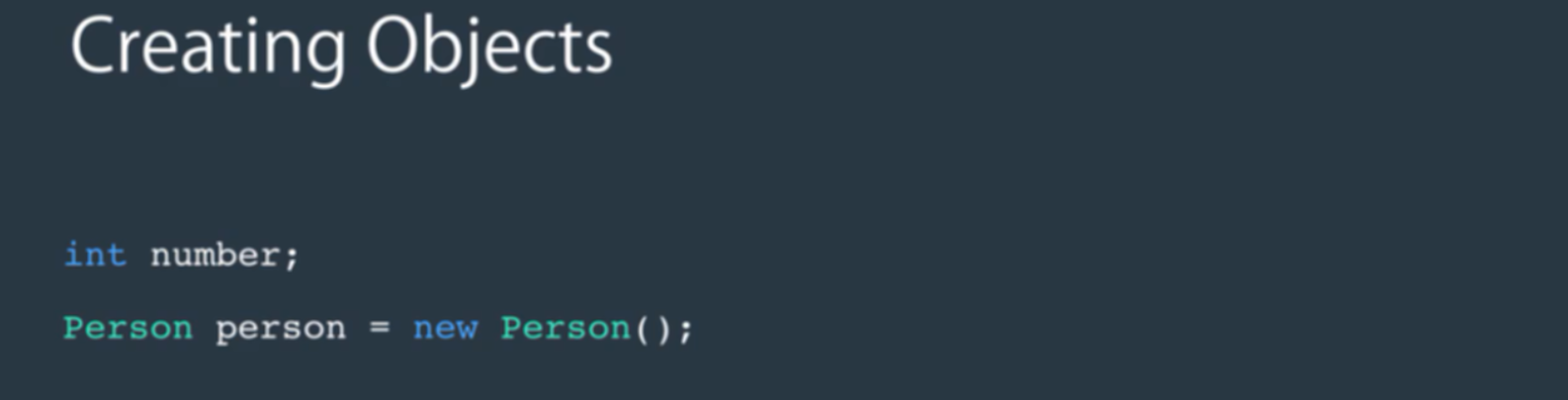
Objects là cái được tạo ra từ class, cần làm rõ xí là trong c# thì object chỉ được tạo ra bằng class, chứ không tự tạo object được nghe man



Void trong tên method đó là kiểu trả về của method

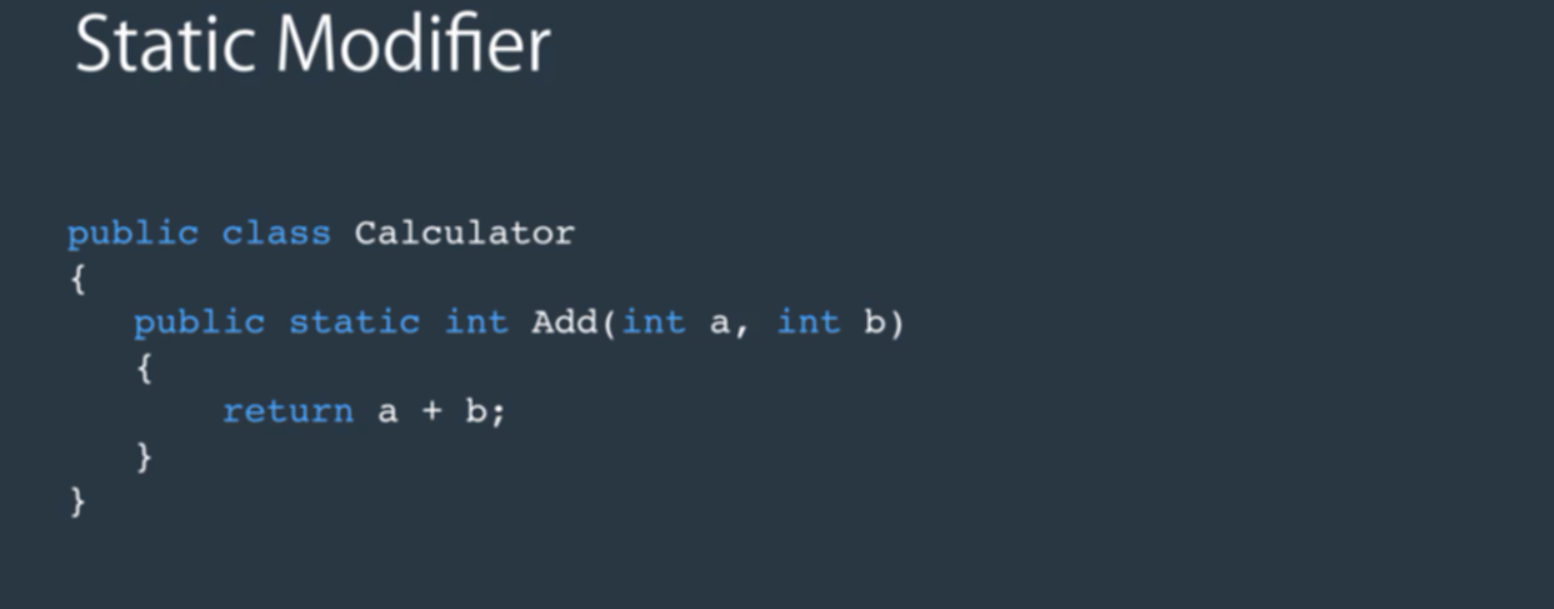


Tương tự kiểu trả về ở đây là int

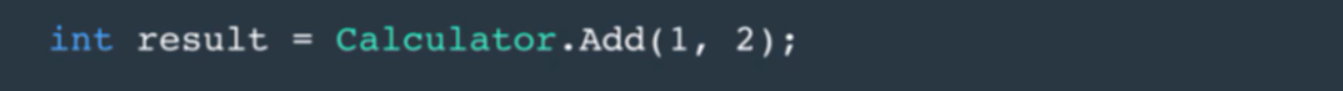




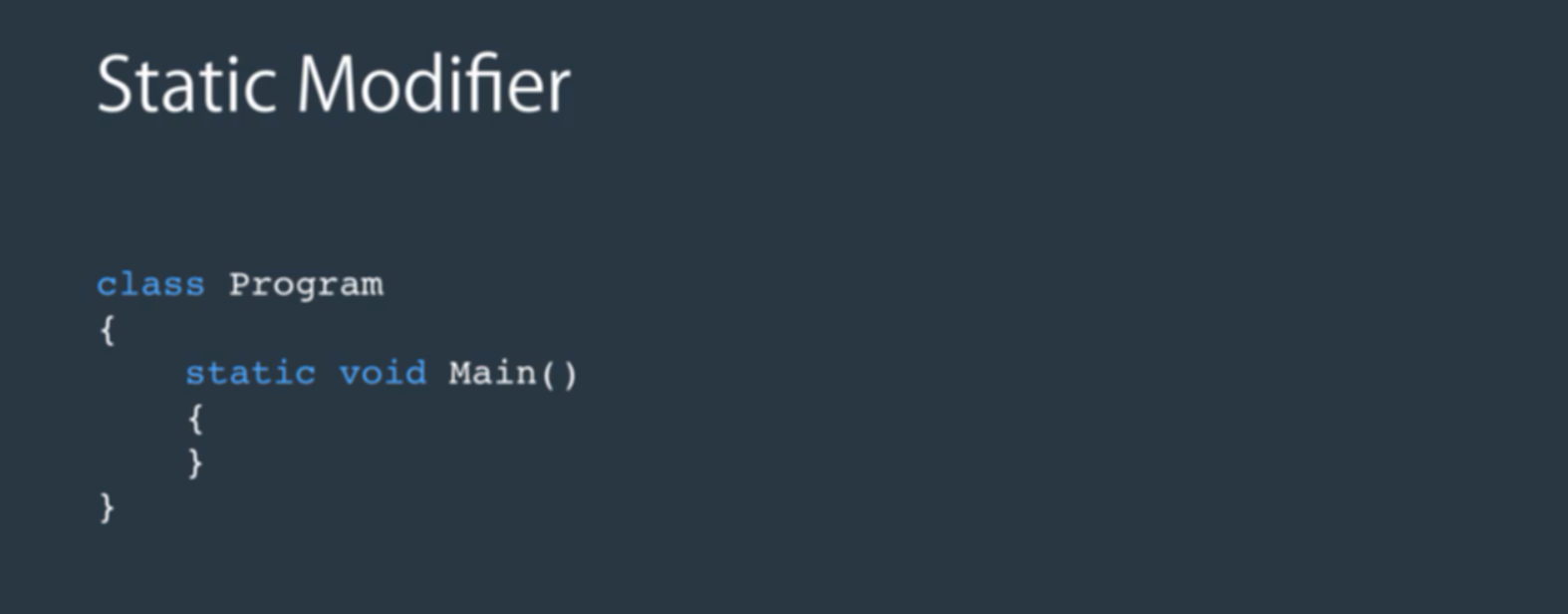
Hên quá dùng dấu chấm, php dùng -> như shit



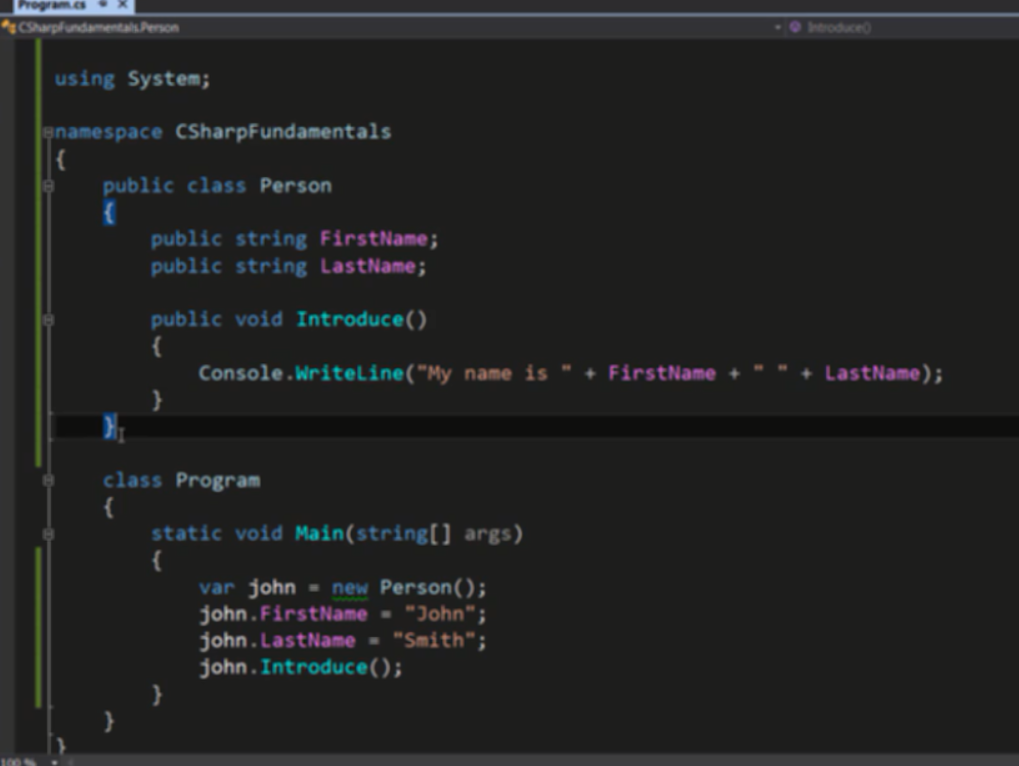
Dùng thêm static cho method



Khi đó method đó có thể dùng class để gọi, bình thường phải gọi từ object



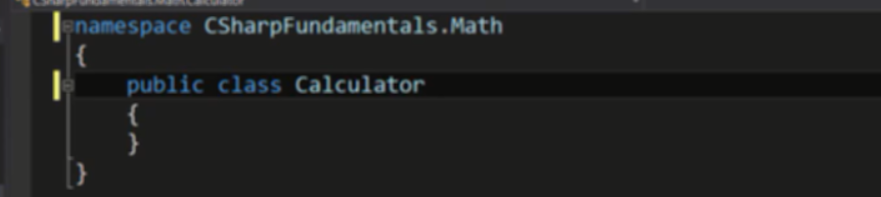
Static ntn object ko gọi được vì không có public



Định nghĩa class program có object john dùng class person để tạo ra

Chọn vào tên class person và chọn tạo file mới để đưa person ra file riêng

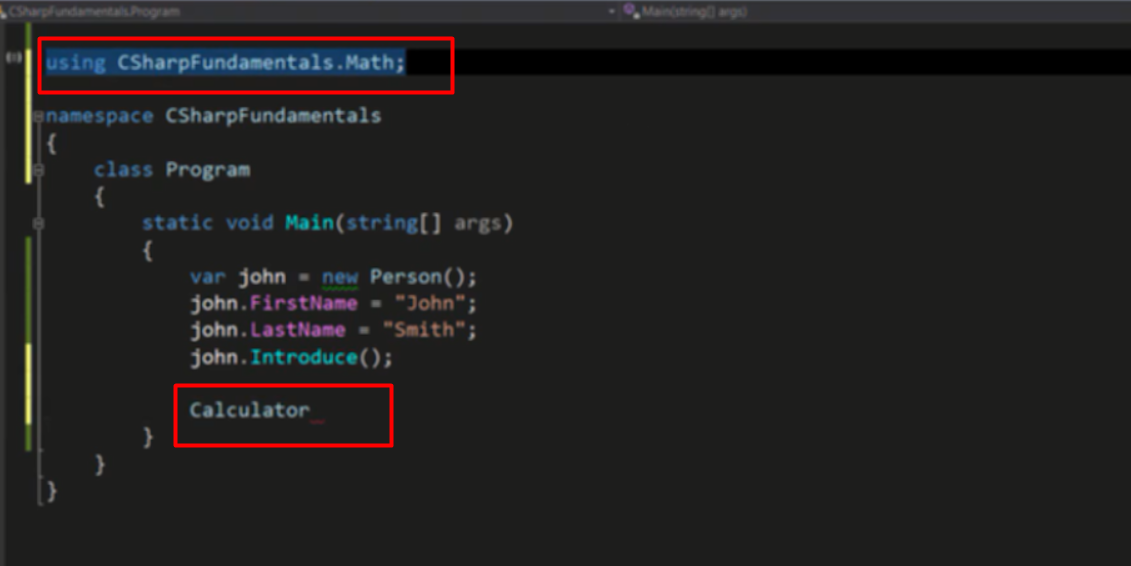
Có thể đưa person vào trong folder, nhưng namespace ko thay đổi thì class của nó vẫn nằm trong namespace đó, nhấn vào tên namespace, ide sẽ gợi ý đổi namespace phù hợp với folder



Thêm từ khóa public trước class

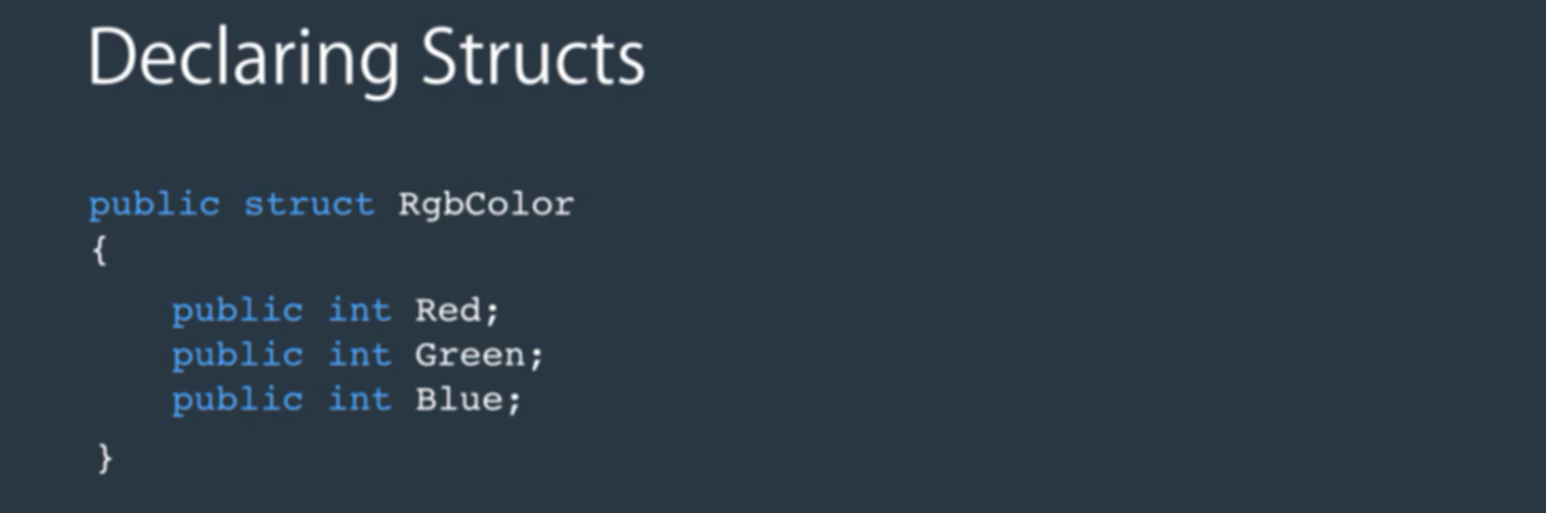
Không có public: chỉ sử dụng được trong class đó, object cũng ko thê gọi tới cái ko có public

Có public có thể sử dụng ngoài class



Import và sử dụng calculator, import= using namespace

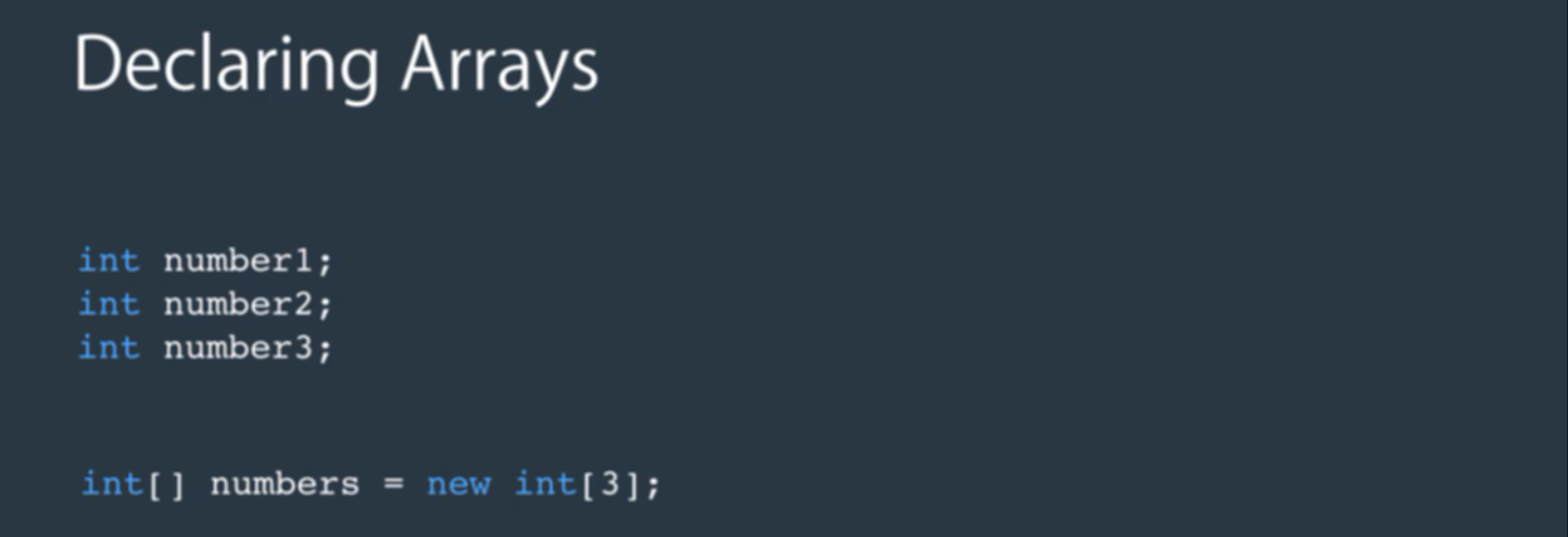
4. Structs



Systax giống class, thay chữ class thành struct

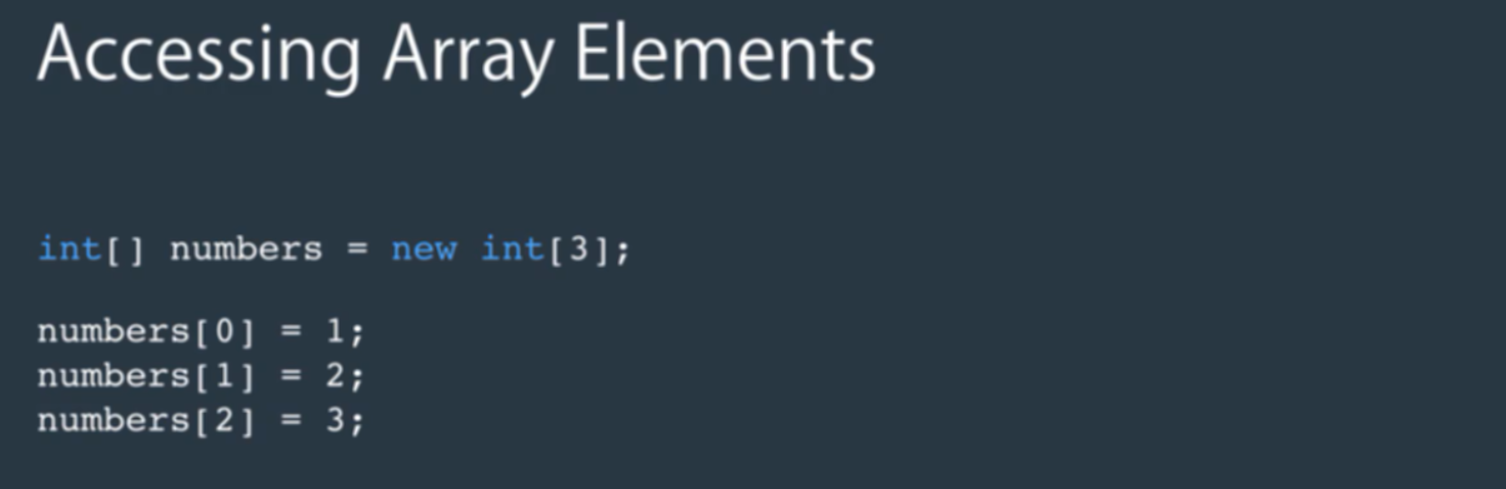
Dùng class, khỏi dùng struct ( struct gần tương tự class)

5. Arrays



Array thực ra lớp dưới là object

Với c# phải chỉ ra kích thước array (dm ghét họ nhà C vler)







Hoặc dùng var cũng đk

Khi tạo ra array, giá trị mặc định của các phần tử là 0, sau đó ta có thể gán giá trị của từng phần tử cho 1 giá trị khác

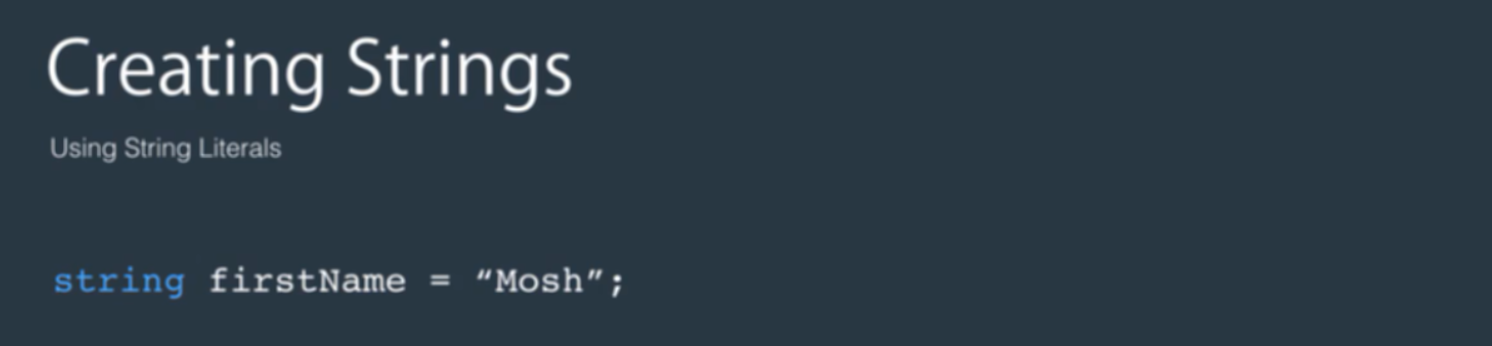


Với bool thì giá trị mặc định là false

Với string

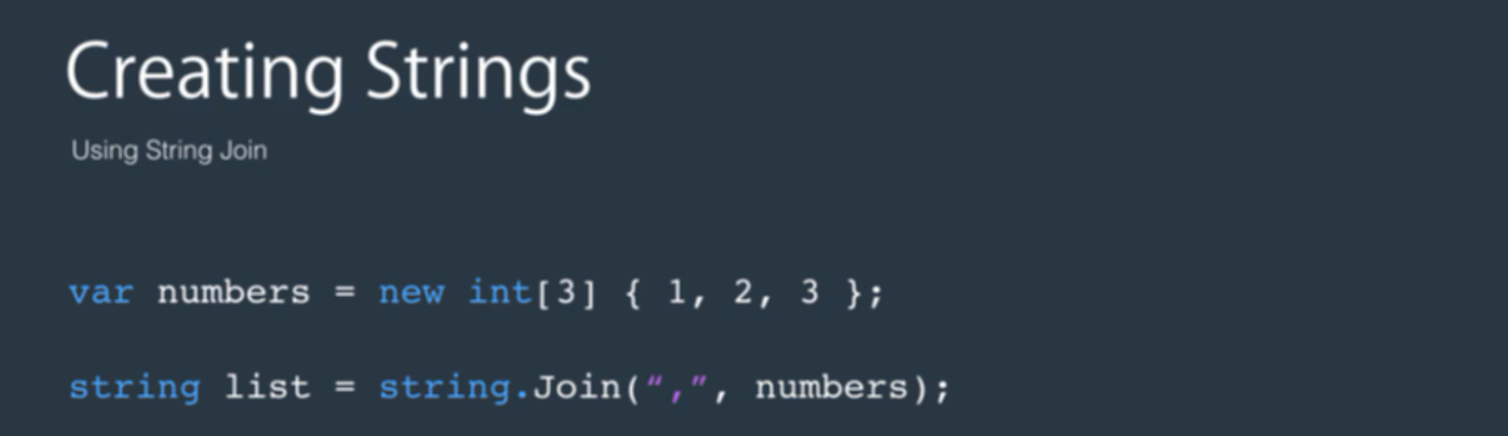


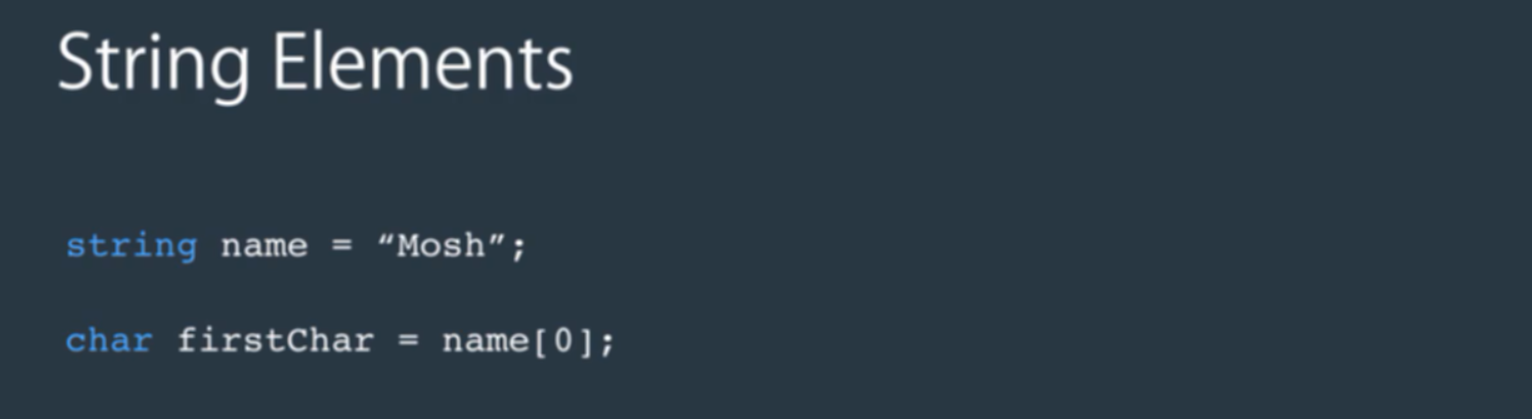
7. Strings



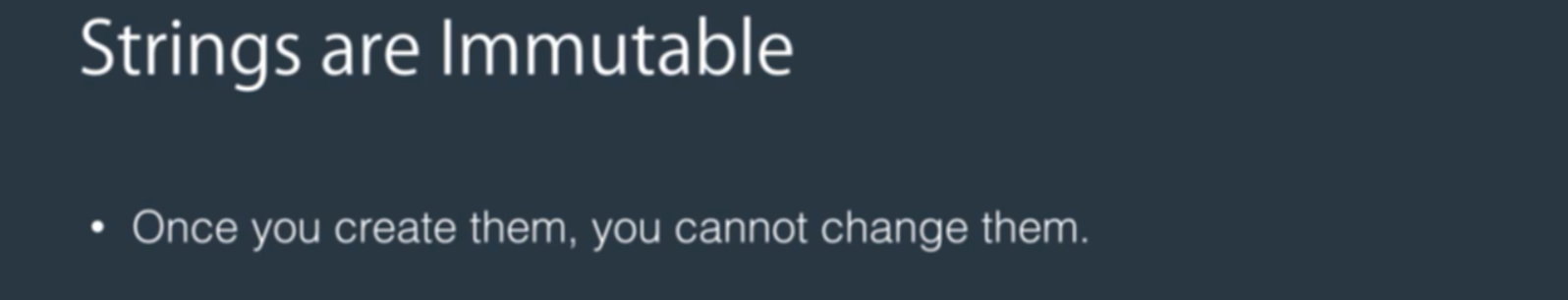








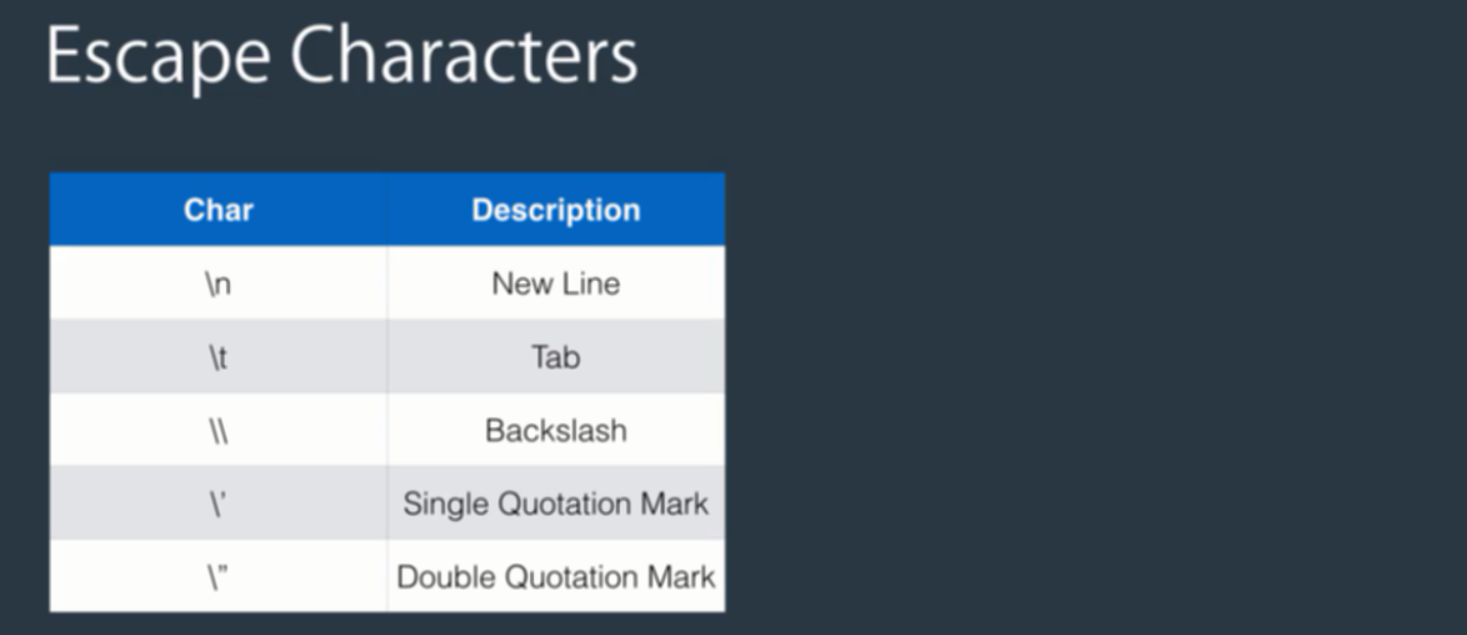
Access từng ký tự





Không dùng ntn được

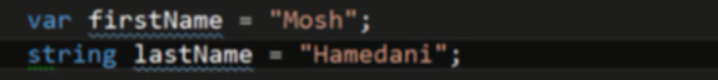
Chỉ access để xem thôi, không thay đổi được





Thay vì link 1 thì link 2 với @

8. Demo Strings



Có thẻ declare dùng string hoặc var



Ngoài ra có thẻ dùng class tương ứng của string hoặc class của int là Int32 đê khai báo

9. Enums

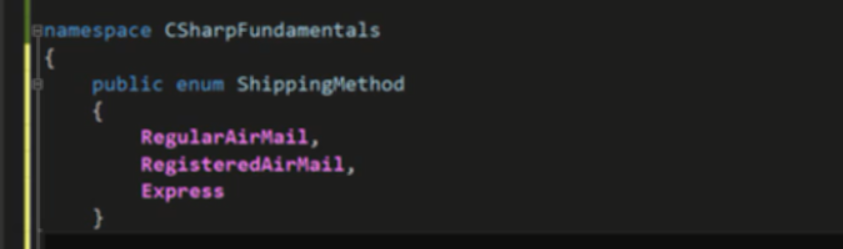


Nghe giống object vcl

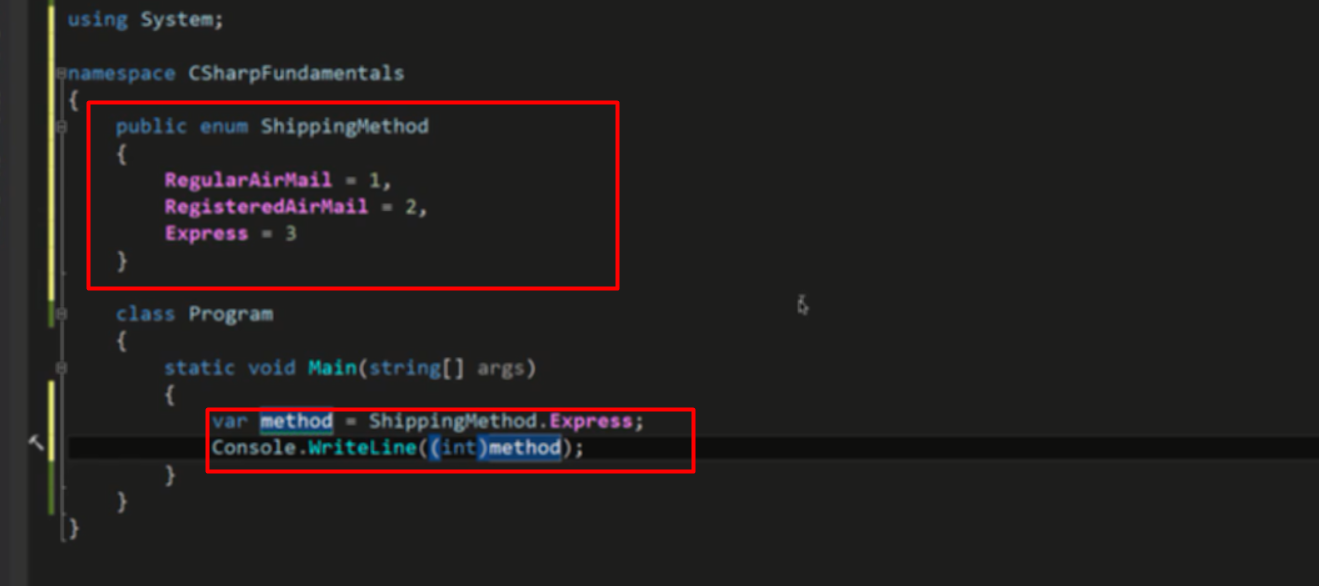


Mặc định kiểu dữ liệu trong enum là int

10. Demo Enums

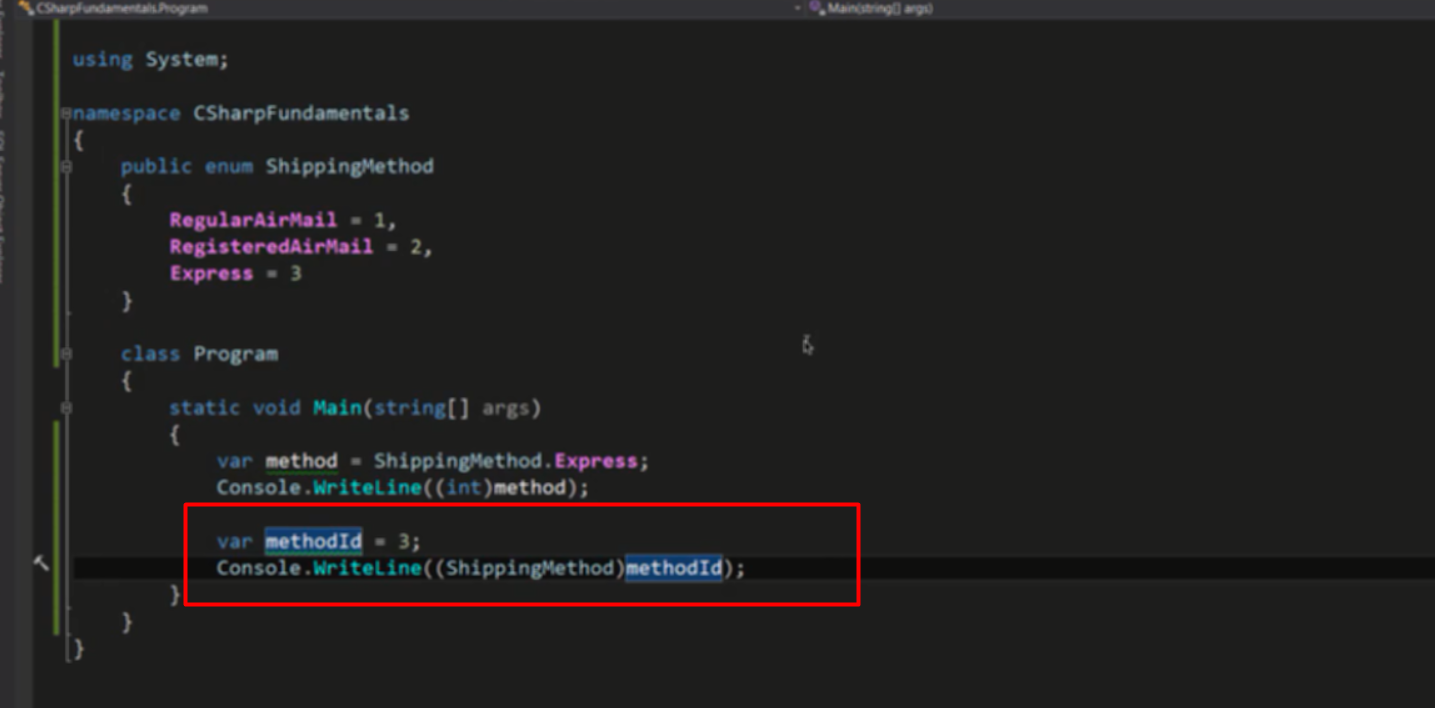


Nếu dùng enum ntn chưa set giá trị thì mặc định giá trị của nó là 0



Phải ép kiểu mới gọi ra giá trị, ko ép kiểu gọi ra key

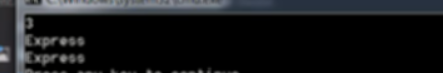
Gọi enum ra sử dụng bằng dấu .

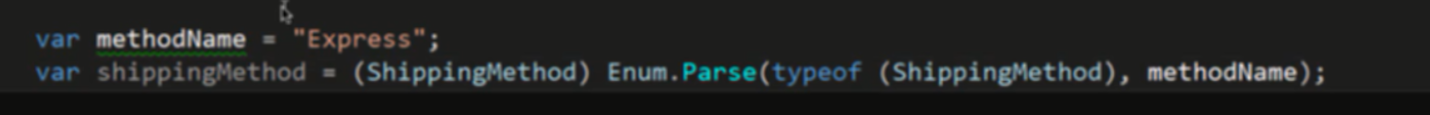


Sẽ in ra Express, ép kiểu bằng tên của enum sẽ cho ra key của enum đó



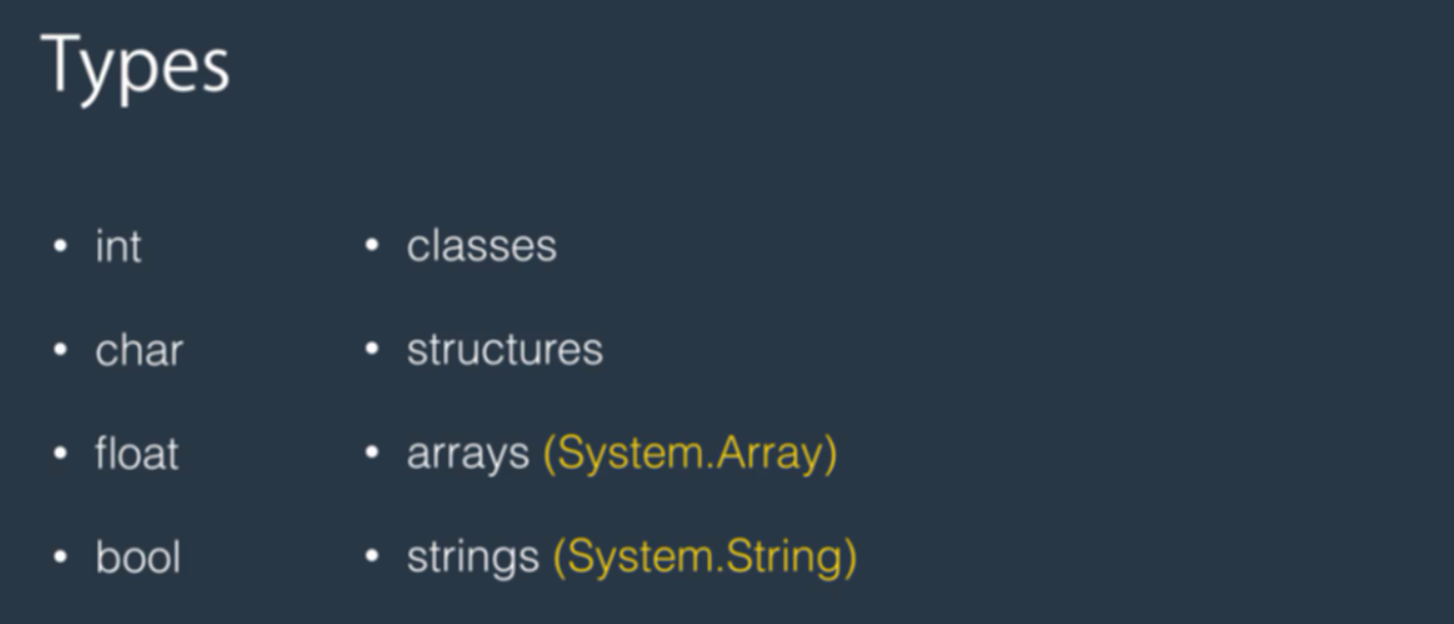
Convert method to string, convert to string sẽ trả về key của enum

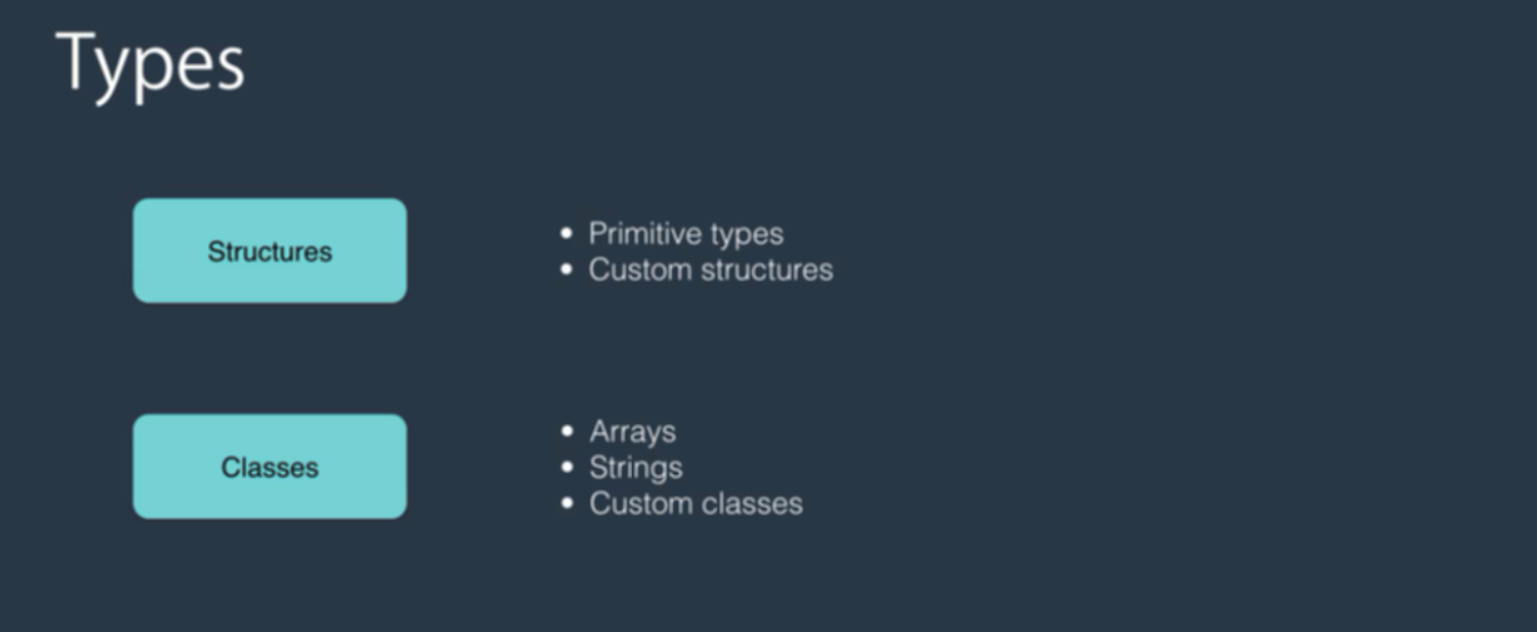


Convert string thành enum method, in ra chữ express ( key của enum)

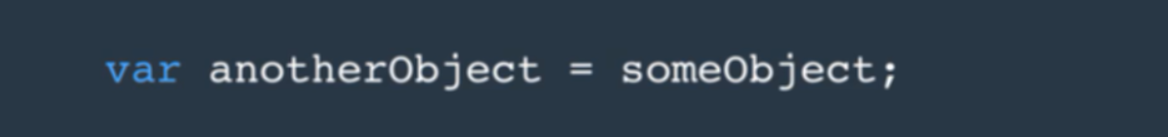
Hàm Enum.Parse sẽ trả về object, do đó cần phải ép kiểu (ShippingMethod)

11. Reference Types and Value Types

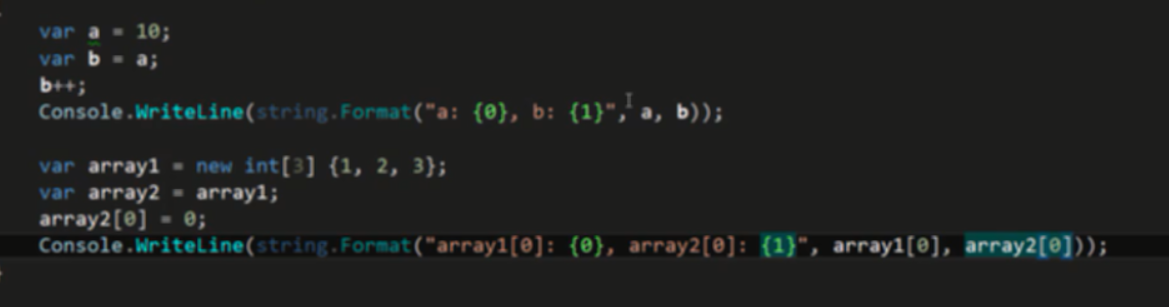






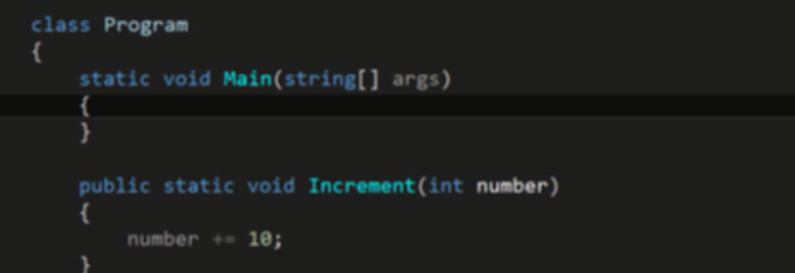


Phải phân biệt sự khác nhau giữa phép gán và phép trỏ

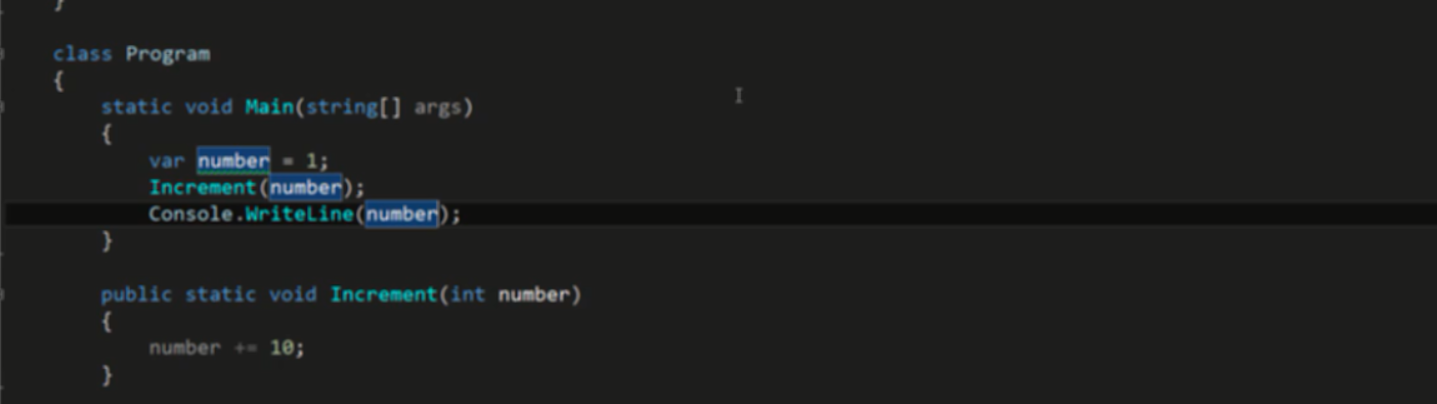


Trường hợp variable thì 2 giá trị khác nhau

Trường hợp object thì 2 giá trị giống nhau



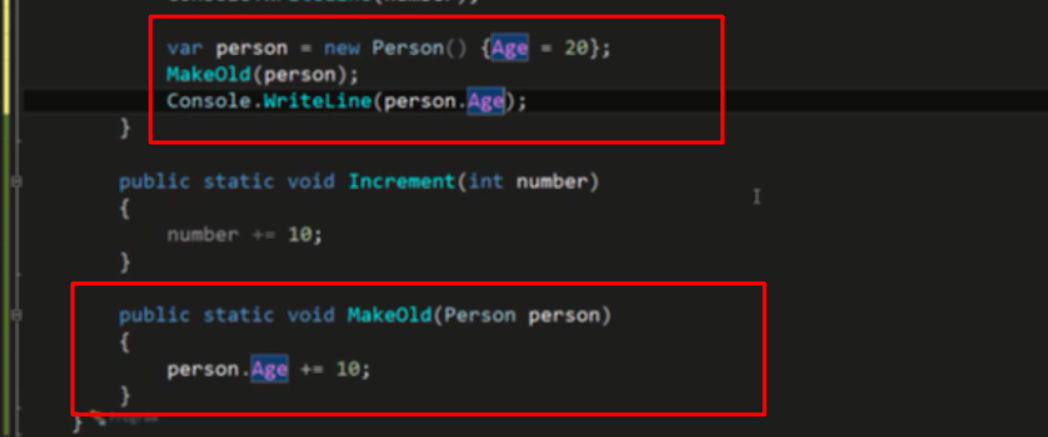
Khi dùng static thì có thể dùng class để gọi method mà không cần phải create object



Kết quả in ra là 1

Nếu in trong hàm increment là 11

Lý do: đầu vào của hàm là biến number=1, sau khi vào hàm biến được đưa lên number=11, nhưng kết thúc hàm number=11 được giải phóng, khi đó number=1 lại như ban đầu ( hồi xưa làm ở C muốn có tác dụng sau khi giải phóng hàm thì khi định nghĩa hàm đối số phải là con trỏ)



Thông số truyền vào hàm là object nên đây là phép trỏ -> kết quả là 30