3강. 데이터형

프로그래밍 강사 :이태성

목차

1. 데이터형
2. 변환
3. 표준 입력
4. 사용자 지정형
5. 값 형식과 참조 형식
6. 정리

01. 데이터형

기본 데이터형

- C#의 데이터형 object로부터 파생된 객체

  System.Object == object

- 데이터형은 CTS에서 정의된 객체

기본 데이터형

- 정수형

  bool: System.Boolean, 1 byte
  char: System.Char , 2 byte
  byte: System.Byte , 1 byte
  sbyte: System.SByte , 1 byte
  short: System.Int16 , 2 byte
  ushort: System.UInt16 , 2 byte
  int : System.Int32 , 4 byte
  uint : Syste.UInt32 , 4 byte
  long: System.Int64 , 8 byte
  ulong: System.UInt64 , 8 byte

기본 데이터형

- 실수형
  float : System.Single , 4 byte
  double : System.Double , 8 byte
  decimal : System.Decimal, 16 byte

- 문자열형
  string : System.String

bool 형

- true, false
  true와 false 대신 0과 그 외의 값은
  사용 금지

- 클래스의 정적 변수 bool형의
  기본 값과 지역 bool 변수 값을
  출력해 보자.

코드 보기

char 형

- 유니코드
  2 byte
- char 형의 암시적 값 변환
  ushort, int, uint, long, ulong,
  float,
  double, decimal
  int Numer = ‘7’;
- 문자 상수 7을 int형으로 변환하고
  1를 더한 결과를 출력해 보자.

코드 보기

byte, sbyte 형

- byte 유효 범위
  부호 없는 0 ~ 255

- sbyte 유효 범위
  -128 ~ 127

- byte형 두 값을 더하고 그 값을 int형 변수에 대입하여 출력해 보자

코드 보기

short, ushort 형

- short 유효 범위
  -32768 ~ 32767
- ushort 유효 범위
  0 ~ 65535
- short 형의 유효 범위 값을 출력해 보자.

```
public      const      short
MinValue
public      const      short
```

코드 보기

int, uint 형, float 형

- int 유효값 범위
  -2,147,483,648 ~ 2,147,483,647
- uint 유효값 범위
  0 ~ 4,294,967,295
- float 형
  소수점형 뒤에 f, F 접미사 명시
  없으면 double 형

string 형

- **C/C++ 문자열과 차이점**
  문자열 끝에 0, '₩0'

- **'+' : 문자열 연결**
  == : 문자열 비교
  [인덱스] : 문자

string 형

- 문자열에 '₩' 를 포함하는 경우

  @"C:₩temp₩test.txt"
  "C:₩₩temp₩₩test.txt"

string 형

- 두 문자열을 합친 후에 합친 문자열과
  문자를 출력해 보자.

코드 보기

- **문자열의 문자 개수를 Length로**
  확인해 보자.

코드 보기

string 형

- 백스페이스가 있는 문자열을 출력해 보자.

코드 보기

암시적 데이터형 var

- 대입되는 데이터에 따라 데이터형 결정

- var를 사용할 수 없는 예

  (1) null 값 초기화, 매개변수로는 사용 못함
  (2) var는 지역변수로만 사용
  　　클래스 멤버로는 사용 못함
  (3) 연속적으로 초기화 하는 경우
  　　var m = 10, n = 20;

암시적 데이터형 var

- var형 변수에 데이터가 분명한 값으로 선언과 초기화를 한 후에 두 값을 더한 결과를 출력해 보자

코드 보기

nullable 형

- null을 허용하지 않는 데이터형이 null값을 허용
- 형식
  데이터형? 변수명;

  int? Var1;
  bool? Var2 = null; // true, false, null

nullable 형

- **속성**
  .HasValue // true, false
  .Value // 읽기 전용

- nullable 형을 선언하고 판독해 보자

코드 보기

02. 변환

데이터 변환

- ToString()
- 기본 데이터형.Parse()
- Convert.ToInt32()
  Convert.ToSingle()
  Convert.ToXXXXX()

코드 보기

박싱과 언박싱

- 박싱(boxing)
  데이터 형을 최상위 object 형으로 변환하여 힙(heap 메모리)에 저장
  int m = 123;
  object obj = m;

- 언박싱(unboxing)
  힙에 저장된 형식을 다시 원래의 형식으로 변환
  int n = (int)obj;

박싱과 언박싱

- int형 값을 박싱한 후에 다시 언박싱하여 출력해 보자

  코드 보기

- 박싱과 언박싱 과정에서 메모리 공유가 발생하는지, 또는 복사가 발생하는지 확인해 보자.

  코드 보기

03. 표준 입력

표준 입력

- Console.ReadKey()
  사용자가 눌린 키 한 문자 정보를 리턴하는 메서드
- 함수 원형
  public static ConsoleKeyInfo ReadKey()
  public static ConsoleKeyInfo ReadKey
  (bool intercept)
  true : 화면 출력 안 함 false : 화면 출력
- ConsoleKeyInfo
  키의 문자와 Shift, Alt, Ctrl 보조키 상태 포함

표준 입력

- ConsoleKeyInfo 속성

  - ConsoleKeyInfo.Key
    ConsoleKey 열거형 값
    ConsoleKey.A, ConsoleKey.Escape 등….
    MSDN 에서 찾아 볼 것 코드 보기

  - ConsoleKeyInfo.KeyChar
    눌린 키의 유니코드를 얻는 속성으로
    대소문자 등을 모두 구분할 수 있다.
    코드 보기

표준 입력

- Console.ReadLine()
  엔터키가 눌려질 때까지 입력 받은 문자열을 리턴하는 메서드

- 활용
  입력 받은 문자열을 숫자로 사용할 때는 Convert.Toint32() 등의 메서드를 사용

  코드 보기

04. 사용자 지정형
struct, enum, class, interface

구조체

- 형식

```
public struct  구조체명
{
    // 멤버, 속성, 메서드
 }
```

구조체

- 제한 사항

  - 구조체에 선언된 const, static 변수만 초기화 가능

  코드 보기

  - **구조체 안에 선언할 수 있는 생성자는 매개변수가 반드시 있어야 함**

  코드 보기

구조체

- 제한 사항

  - 구조체를 같은 구조체에 대입하게 되면 값이 복사

    코드 보기

  - **구조체는 값 형식이고 클래스는 참조 형식임**

    코드 보기

구조체

- 제한 사항
  구조체는 값 형식이므로 선언만으로도 사용 가능

  new를 사용했을 때만
  ➔ 생성자가 호출
  ➔ 기본값으로 초기화

  코드 보기

구조체

- 제한 사항
  - 구조체는 구조체 또는 클래스에 상속할 수 없음

  - 구조체는 인터페이스를 상속하여 메서드를 구현할 수 있음

구조체

- 국어, 영어, 수학 점수를 구조체 멤버 변수로 입력하고 Compute() 메서드를 호출하면 평균과 총점이 계산되도록 하여 프로그래밍해 보면
다음과 같다.

코드 보기

열거형

- 상수를 문자열로 대치하여 선언
  상수에 의미 부여

- 형식
  enum 열거형 명칭 { 문자열1, 문자열2 };
  enum 열거형 명칭 { 문자열1 = 상수,
  문자열2 = 상수};
  enum 열거형 명칭 { 문자열1 = 상수, 문자열2 };

열거형

- 기본은 int형이지만 char 형을 제외한 형식 지정할 수 있음

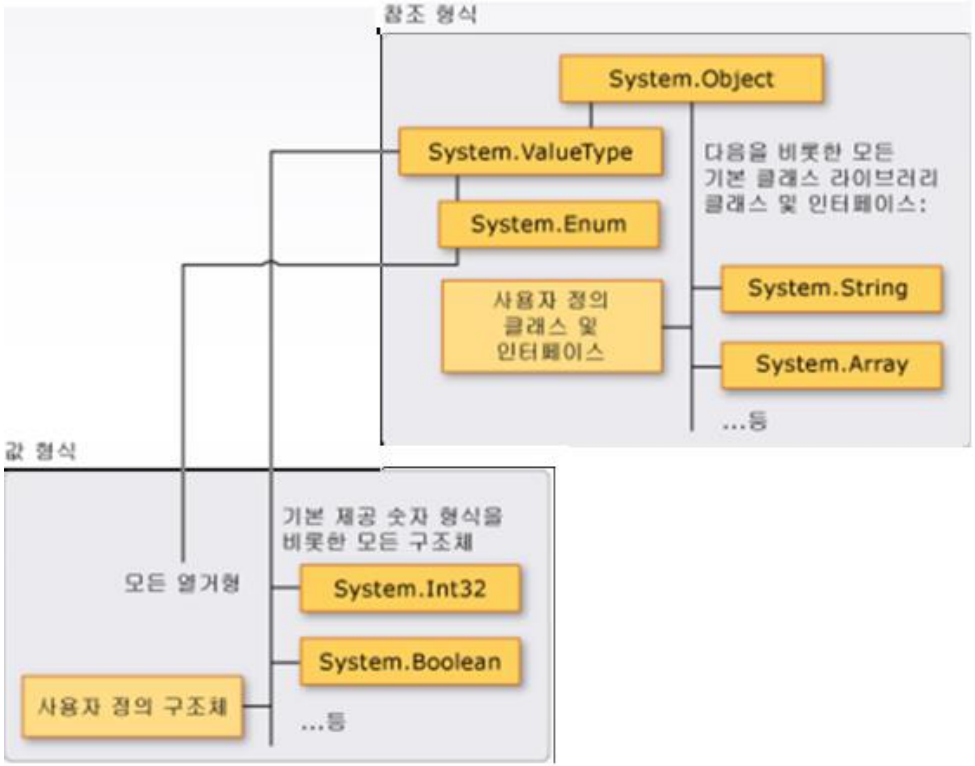
```
enum Days : byte { Sun = 0, Mon,
                   Tue, Wed, Thu };
```

- 열거형 변수가 아닌 변수에 열거형 값을 대입할 때는 데이터형을 명시할 것

코드 보기

05. 값 형식과 참조 형식

형식 상속 관계


참조 형식
System.Object
System.ValueType
System.Enum
사용자 정의
클래스 및
인터페이스
다음을 비롯한 모든
기본 클래스 라이브러리
클래스 및 인터페이스:
System.String
System.Array
...등
값 형식
모든 열거형
사용자 정의 구조체
기본 제공 숫자 형식을
비롯한 모든 구조체
System.Int32
System.Boolean
...등

값 형식

- System.Object

  + System.ValueType 에서 파생

- 변수가 직접 값을 저장하는 형
- 기본 데이터형, 구조체, 열거형
- 선언 vs 생성(new)

코드 보기

참조 형식

- 참조 형식
  한 객체를 참조형 변수가 사용할 때 참조형에 의해 내용이 바뀌면 객체에 영향을 줌
- class, interface, delegate, 배열, string
- 객체와 참조형 사이의 대입은 객체의 메모리 주소가 복사됨

코드 보기

06. 정리

정리

- 기본 데이터형과 CTS 형식을 익혀둔다.
- 데이터형에 관한 검증 코드를 작성해 보자.
- 값 형식과 참조 형식의 차이점을 이해한

구분	설명
값 형식	기본 데이터형, struct, enum
참조 형식	class, interface, delegate, 배열