# 4) 고급 자료형 및 기능

**매 강의 강의자료 시작에 PDF파일을 올려두었어요!**

▼ PDF 파일

**[수업 목표]**

- C#에서 Nullable 형과 문자열 빌더에 대한 이해와 사용 방법을 익힙니다.
- Nullable 형과 문자열 빌더를 활용하여 프로그램을 작성해봅니다.

**[목차]**



모든 토글을 열고 닫는 단축키
Windows : `Ctrl` + `alt` + `t`
Mac : `⌘` + `⌥` + `t`

---

## 01. Nullable 형

▼ **1) null**

- null은 "아무것도 없음"을 의미한다.
- 참조형 변수가 어떠한 객체를 참조하지 않을 때 사용됩니다.

▼ **2) Nullable 란?**

- Nullable은 C#에서 null 값을 가질 수 있는 값형에 대한 특별한 형식입니다.
- 기본적으로 값형은 null을 허용하지 않습니다.
- **값형 변수에 null 값을 지정할 수 있는 방법을 제공**하여 값형이나 구조체를 사용하는 프로그램에서 null 상태를 나타낼 수 있습니다. 주로 값형 변수가 null인지 아닌지를 확인하고 처리해야 할 때 유용합니다.
- 형식은 `?` 연산자를 사용하여 선언됩니다. 예를 들어, `int?`는 int 형식에 null을 할당할 수 있는 Nullable<int> 형식을 나타냅니다.

### ▼ 3) 사용 예제

```
// Nullable 형식 변수 선언
int? nullableInt = null;
double? nullableDouble = 3.14;
bool? nullableBool = true;

// 값 할당 및 접근
nullableInt = 10;
int intValue = nullableInt.Value;

// null 값 검사
if (nullableDouble.HasValue)
{
    Console.WriteLine("nullableDouble 값: " + nullableDouble.Value);
}
else
{
    Console.WriteLine("nullableDouble은 null입니다.");
}

// null 병합 연산자 사용
// nullableInt ?? 0과 같이 사용되며, nullableInt가 null이면 0을 반환합니다.
int nonNullableInt = nullableInt ?? 0;
Console.WriteLine("nonNullableInt 값: " + nonNullableInt);
```

## 02. 문자열 빌더 (StringBuilder)

### ▼ 1) StringBuilder 란?

1. **문자열 조작**
   StringBuilder는 Append(), Insert(), Replace(), Remove() 등 다양한 메서드를 제공하여 문자열에 대한 추가, 삽입, 치환, 삭제 작업을 수행할 수 있습니다.

2. **가변성**
   StringBuilder는 **내부 버퍼를 사용하여 문자열 조작**을 수행하므로 크기를 동적으로 조정할 수 있습니다. 따라서 문자열의 크기가 늘어나거나 줄어들어도 추가적인 메모리 할당이 발생하지 않습니다.
3. **효율적인 메모리 관리**
   문자열 조작 시 StringBuilder는 내부 버퍼를 사용하여 문자열을 조작하므로, 반복적인 문자열 조작 작업이 발생해도 **메모리 할당 및 해제 오버헤드가 크게 감소**합니다.

### ▼ 2) 주요 메서드

- Append: 문자열을 뒤에 추가합니다.
- Insert: 문자열을 지정한 위치에 삽입합니다.
- Remove: 지정한 위치에서 문자열을 제거합니다.
- Replace: 문자열의 일부를 다른 문자열로 대체합니다.
- Clear: StringBuilder의 내용을 모두 지웁니다.

### ▼ 3) 사용예제

```
StringBuilder sb = new StringBuilder();

// 문자열 추가
sb.Append("Hello");
sb.Append(" ");
sb.Append("World");

// 문자열 삽입
sb.Insert(5, ", ");

// 문자열 치환
sb.Replace("World", "C#");

// 문자열 삭제
sb.Remove(5, 2);

// 완성된 문자열 출력
string result = sb.ToString();
Console.WriteLine(result);
```

이전 강의

📕 3) 델리게이트, 람다 및 LINQ

다음 강의

📕 1) 알고리즘 기초