



C# Coding Stanard

[기본 원칙](#)

[참조 문서](#)

[코딩 표준 요약 테이블](#)

[Visual Studio 사용할 때 미세 팁](#)

기본 원칙

1. 가독성을 최우선으로 한다.
2. 정말 합당한 이유가 있지 않는 한, 통합개발환경 *IDE*의 자동 서식을 따른다. (Visual Studio의 "Ctrl + K + D" 기능)

참조 문서

- 여기서 만든 코딩 규칙은 포큐 C# 코딩 표준을 기준으로 만들었음

POCU 아카데미용 C# 코딩 표준

이 코딩 표준은 아래의 코딩 표준들에서 영감을 얻었음. public 메서드가 아닌 경우 카멜 표기법을 따른다. Visual Studio를 사용 시에는 별도의 스타일 규칙을 추가해야 할 수도 있음. (자세한 설
<https://docs.popekim.com/ko/coding-standards/pocu-csharp>



- 그 외에 Effective C++, More Effective C++, ... 등의 서적 혹은 문서를 참고하여 Coding Standard를 완성해 나갈 예정임

참고한 서적 목록	제목	출판사
1	Effective C#	프로텍미디어

코딩 표준 요약 테이블

Coding Standard Table

▼ Type	# Number	Aa Coding Standard	≡ Tag	▼ Reference
--------	----------	--------------------	-------	-------------

🔍 Type	# Number	Aa Coding Standard	☰ Tag	🔍 Reference
Main	1	<u>클래스와 구조체의 이름은 파스칼 표기법을 따른다</u>	#네이밍	COMP1500
Main	2	<u>지역 변수 그리고 함수의 매개 변수의 이름은 카멜 표기법을 따른다</u>	#네이밍	COMP1500
Main	3	<u>메서드 이름은 기본적으로 동사(명령형) + 명사(목적어)의 형태로 짓는다</u>	#네이밍	COMP1500
Main	4	<u>단, 단순히 부울(boolean) 상태를 반환하는 메서드의 동사 부분은 최대한 Is, Can, Has, Should를 사용하되 그러는 것이 부자연스러울 경우에는 상태를 나타내는 다른 3인칭 단수형 동사를 사용한다.</u>	#네이밍	COMP1500
Main	5	<u>아래에 제시된 예를 제외하곤 모든 메서드의 이름은 파스칼 표기법을 따른다.</u>	#네이밍	COMP1500
Main	6	<u>public 메서드가 아닌 경우 카멜 표기법을 따른다. Visual Studio를 사용 시에는 별도의 스타일 규칙을 추가해야 할 수도 있음.</u>	#네이밍	COMP1500
Main	7	<u>상수의 이름은 모두 대문자로 하되 밑줄로 각 단어를 분리한다.</u>	#네이밍	COMP1500
Main	8	<u>상수로 사용하는 개체형 변수에는 static readonly를 사용한다.</u>	#네이밍	COMP1500
Main	9	<u>static readonly 변수는 모두 대문자로 하되 밑줄로 각 단어를 분리한다.</u>	#네이밍	COMP1500
Main	10	<u>초기화 후 값이 변하지 않는 변수는 readonly로 선언한다.</u>	#네이밍	COMP1500
Main	11	<u>네임스페이스의 이름은 파스칼 표기법을 따른다.</u>	#네이밍	COMP1500
Main	12	<u>부울(boolean) 변수는 앞에 b를 붙인다.</u>	#네이밍	COMP1500
Main	13	<u>부울 프로퍼티는 앞에 Is, Has, Can, Should 중에 하나를 붙인다.</u>	#네이밍	COMP1500
Main	14	<u>인터페이스를 선언할 때는 앞에 I를 붙인다.</u>	#네이밍	COMP1500

🔍 Type	# Number	Aa Coding Standard	☰ Tag	🔍 Reference
Main	15	<u>열거형을 선언할 때는 앞에 E를 붙인다.</u>	#네이밍	COMP1500
Main	16	<u>구조체를 선언할 때는 앞에 S를 붙인다. 단, readonly struct일 때는 그렇지 아니한다.</u>	#네이밍	COMP1500
Main	17	<u>private 멤버 변수명은 앞에 m을 붙이고 파스칼 표기법을 따른다.</u>	#네이밍	COMP1500
Main	18	<u>값을 반환하는 함수의 이름은 무엇을 반환하는지 알 수 있게 짓는다.</u>	#네이밍	COMP1500
Main	19	<u>단순히 반복문에 사용하는 변수가 아닌 경우에는 i, e 같은 변수명 대신 index, employee 처럼 변수에 저장되는 데이터를 한 눈에 알아볼 수 있는 변수명을 사용한다.</u>	#네이밍	COMP1500
Main	20	<u>뒤에 추가적인 단어가 오지 않는 경우 줄임말은 모두 대문자로 표기한다.</u>	#네이밍	COMP1500
Main	21	<u>getter와 setter 대신 프로퍼티를 사용한다.</u>	#코드 작성	COMP1500
Main	22	<u>지역 변수를 선언할 때는 그 지역 변수를 사용하는 코드와 동일한 줄에 선언하는 것을 원칙으로 한다.</u>	#코드 작성	COMP1500
Main	23	<u>double이 반드시 필요한 경우가 아닌 이상 부동 소수점 값에 f를 붙여 준다.</u>	#코드 작성	COMP1500
Main	24	<u>switch 문에 언제나 default: 케이스를 넣는다.</u>	#코드 작성	COMP1500
Main	25	<u>switch 문에서 default: 케이스가 절대 실행될 일이 없는 경우, default: 안에 Debug.Fail() 또는 Debug.Assert(false) 란 코드를 추가한다.</u>	#코드 작성	COMP1500
Main	26	<u>재귀 함수는 이름 뒤에 Recursive를 붙인다.</u>	#네이밍	COMP1500
Main	27	<u>클래스 안에서 멤버 변수와 메서드의 등장 순서는 다음을 따른다. (페이지 참고).</u>	#코드 작성	COMP1500

🔍 Type	# Number	Aa Coding Standard	≡ Tag	🔍 Reference
Main	28	<u>매개변수 자료형이 범용적인 경우, 함수 오버로딩을 피한다</u>	#코드 작성	COMP1500
Main	29	<u>클래스는 각각 독립된 소스 파일에 있어야 한다. 단, 작은 클래스 몇 개를 한 파일 안에 같이 넣어두는 것이 상식적일 경우 예외를 허용한다.</u>	#코드 작성	COMP1500
Main	30	<u>파일 이름은 대소문자까지 포함해서 반드시 클래스 이름과 일치해야 한다</u>	#파일 작성	COMP1500
Main	31	<u>여러 파일이 하나의 클래스를 이룰 때(즉, partial 클래스), 파일 이름은 클래스 이름으로 시작하고, 그 뒤에 마침표와 세부 항목 이름을 붙인다.</u>	#파일 작성	COMP1500
Main	32	<u>특정 조건이 반드시 충족되어야 한다고 가정(assertion)하고 짤 코드는 모든 곳에 assert를 사용한다. assert는 복구 불가능한 조건이다. (예: 대부분의 함수는 다음과 같은 assert를 가질 수도... Debug.Assert(매개변수의 null 값 검사)).</u>	#코드 작성	COMP1500
Main	33	<u>비트 플래그 열거형은 이름 뒤에 Flags를 붙인다.</u>	#네이밍	COMP1500
Main	34	<u>디폴트 매개 변수 대신 함수 오버로딩을 선언한다.</u>	#코드 작성	COMP1500
Main	35	<u>디폴트 매개 변수를 사용하는 경우, null이나 false, 0 같이 비트 패턴이 0인 값을 사용한다.</u>	#코드 작성	COMP1500
Main	36	<u>변수 가리기(variable shadowing)은 허용되지 않는다. 외부 변수가 동일한 이름을 사용중이라면 내부 변수에는 다른 이름을 사용한다.</u>	#코드 작성	COMP1500
Main	37	<u>언제나 System.Collections에 들어있는 컨테이너 대신에 System.Collections.Generic에 들어있는 컨테이너를 사용한다. 순수 배열을 사용하는 것도 괜찮다.</u>	#코드 작성	COMP1500

🔍 Type	# Number	Aa Coding Standard	≡ Tag	🔍 Reference
Main	38	<u>var 키워드를 사용하지 않으려 노력한다. 단, 대입문의 우항에서 데이터 형이 명확하게 드러나는 경우, 또는 데이터형이 중요하지 않은 경우에는 예외를 허용한다. IEnumerable에 var를 사용하거나 우항의 new 키워드를 통해 어떤 개체가 생성되는지 알 수 있는 등의 허용되는 경우의 좋은 예이다.</u>	#코드 작성	COMP1500
Main	39	<u>싱글턴 패턴 대신에 정적 클래스를 사용한다.</u>	#코드 작성	COMP1500
Main	40	<u>async void 대신에 async Task를 사용한다. async void가 허용되는 유일한 곳은 이벤트 핸들러이다.</u>	#코드 작성	COMP1500
Main	41	<u>외부로부터 들어오는 데이터의 유효성은 외부/내부 경계가 바뀌는 곳에서 검증(validate)하고 문제가 있을 경우 내부 함수로 전달하기 전에 반환해 버린다. 이는 경계를 넘어 내부로 들어온 모든 데이터는 유효하다고 가정한다는 뜻이다.</u>	#코드 작성	COMP1500
Main	42	<u>따라서 내부 함수에서 예외(익셉션)를 던지지 않으려 노력한다. 예외는 경계에서만 처리하는 것을 원칙으로 한다.</u>	#코드 작성	COMP1500
Main	43	<u>위 규칙의 예외: enum 형을 switch 문에서 처리할 때 실수로 처리 안 한 enum 값을 찾기 위해 default: 케이스에서 예외를 던지는 것은 허용.</u>	#코드 작성	COMP1500
Main	44	<u>함수의 매개변수로 null을 허용하지 않는 것을 추구한다. 특히 public 함수일 경우 더욱 그러하다.</u>	#코드 작성	COMP1500
Main	45	<u>null 매개변수를 사용할 경우 변수명 뒤에 OrNull를 붙인다.</u>	#코드 작성	COMP1500
Main	46	<u>함수에서 null을 반환하지 않는 것을 추구한다. 특히 public 함수일 경우 더욱 그러하다. 그러나 때로는 예외를 던지는 것을 방지하기 위해 그래야 할 경우도 있다.</u>	#코드 작성	COMP1500


🔍 Type	# Number	Aa Coding Standard	☰ Tag	🔍 Reference
Main	47	<u>함수에서 null을 반환할 때는 함수의 이름 뒤에 OrNull을 붙인다.</u>	#코드 작성	COMP1500
Main	48	<u>개체 초기자(object initializer)를 사용하지 않으려고 노력한다. 그 대신 생성자와 이름으로 지정한 매개변수(named parameter)를 사용하는 게 더 좋은 방법이다. 이 원칙에 대한 두 가지 예는 다음과 같다.</u>	#코드 작성	COMP1500
Main	49	<u>함수에 전달하는 out 매개변수는 별도의 라인에 선언한다. 즉, 인자 목록 안에서 선언하지 않는다.</u>	#코드 작성	COMP1500
Main	50	<u>using 선언(C# 8.0)의 사용을 금지한다. 대신 using 문을 사용한다.</u>	#코드 작성	COMP1500
Main	51	<u>new 키워드 뒤에 반드시 명시적으로 자료형을 적어준다. (즉, C# 9.0의 new() 사용 금지).</u>	#코드 작성	COMP1500
Main	52	<u>프로퍼티에 private init(C# 9.0)을 최대한 사용한다.</u>	#코드 작성	COMP1500
Formatting	1	<u>탭은 비주얼 스튜디오 기본 값을 사용하며, 비주얼 스튜디오를 사용하지 않을 씨 띄어쓰기 4칸을 탭으로 사용한다.</u>	#코드 작성	COMP1500
Formatting	2	<u>중괄호({})를 열 때는 언제나 새로운 줄에 연다.</u>	#코드 작성	COMP1500
Formatting	3	<u>중괄호 안({})에 코드가 한 줄만 있더라도 반드시 중괄호를 사용한다.</u>	#코드 작성	COMP1500
Formatting	4	<u>한 줄에 변수 하나만 선언한다.</u>	#코드 작성	COMP1500

Visual Studio 사용할 때 미세 팁

1. 스크롤 막대에 지도 모드 사용, `Ctrl + M + O` 누르면 선언만 간추려서 볼 수 있음, 코드 블록 지정하기

비주얼스튜디오 핵꿀팁 총정리

알아두면 개발 시간이 효과적으로 단축되고, 모르면 손해인 비주얼스튜디오 꿀팁을 정리해봤습니다. 본인이 알고있는 또다른 팁이 있고 공유하고 싶으시다면 댓글로 부탁드립니다 다음번 꿀팁정리 영상에 포

 <https://www.youtube.com/watch?v=24edLCBwoWc>

