KeyDown, KeyUp은 이름대로 키가 눌러질 때와 떨어질 때 발생하며 KeyPress 이벤트는 문자가 입력될 때 발생한다. 세 이벤트 중에 상대적으로 간단한 KeyPress 이벤트에 대해 먼저 알아보자. KeyDown은 모든 키에 대해 발생하는데 비해 KeyPress는 문자 키에 대해서만 발생한다는 점이 다르다. 예를 들어 A, B 같은 키를 누르면 KeyDown과 KeyPress 이벤트가 동시에 발생하지만 커서 이동키나 Home, Ins처럼 문자가 아닌 기능 키에 대해서는 KeyDown 이벤트만 발생하며 KeyPress 이벤트는 발생하지 않는다.

KeyPress 이벤트의 인수는 KeyPressEventArgs인데 이 객체에는 KeyChar와 Handled 두 개의 프로퍼티가 포함되어 있다. KeyChar 프로퍼티는 누른 키의 유니코드 문자 코드인데 이 프로퍼티를 읽으면 어떤 문자가 입력되었는지를 알 수 있다. 단순히 키에 할당된 문자를 전달하기만 하는 것이 아니라 Caps Lock, Shift 키와 키보드의 언어 설정 등을 종합적으로 계산하여 문자 코드를 계산해 주므로 KeyChar만 읽으면 최종적으로 입력된 문자를 알 수 있다.

Handled 프로퍼티는 이벤트의 처리 여부를 설정하는데 이 프로퍼티에 true를 대입하면 컨트롤이 키 입력을 완전히 처리했다는 뜻이며 운영체제에게 더 이상 처리하지 말라는 뜻을 전달한다. 디폴트는 false인데 이 값을 그대로 두면 컨트롤이 키 입력을 처리한 후 운영체제가 이 키를 디폴트 처리한다. 대화상자에서 Enter 키나 Esc처럼 특별한 의미가 있는 키 입력은 운영체제가 처리하도록 내버려 두는 것이 좋으므로 이 프로퍼티는 보통 건드리지 않는다. 특수한 목적으로, 예를 들어 Esc를 눌러도 대화상자를 닫지 않도록 하고 싶을 때 Handled 프로퍼티를 활용한다.

예제 KeyPress

KeyPress 예제는 키보드로부터 문자들을 하나씩 입력받아 문자열을 조립한다. 콘솔에서처럼 문자열을 한 번에 입력받는 메서드가 없으므로 이벤트가 전달될 때마다 눌러진 문자들을 누적시켜 문자열을 입력받아야 한다. KeyPress라는 이름으로 프로젝트를 생성하고 Form1.cs 파일에 다음 코드를 입력한다. string 타입의 필드 str을 선언했으며 생성자에서 빈 문자열로 초기화했다.

```
namespace KeyPress
{
    public partial class Form1 : Form
    {
        private string str;

        public Form1()
        {
            InitializeComponent();
            str = "";
        }
    }
}
```