

# C++

## BattleShip 1

국민대학교  
박민근

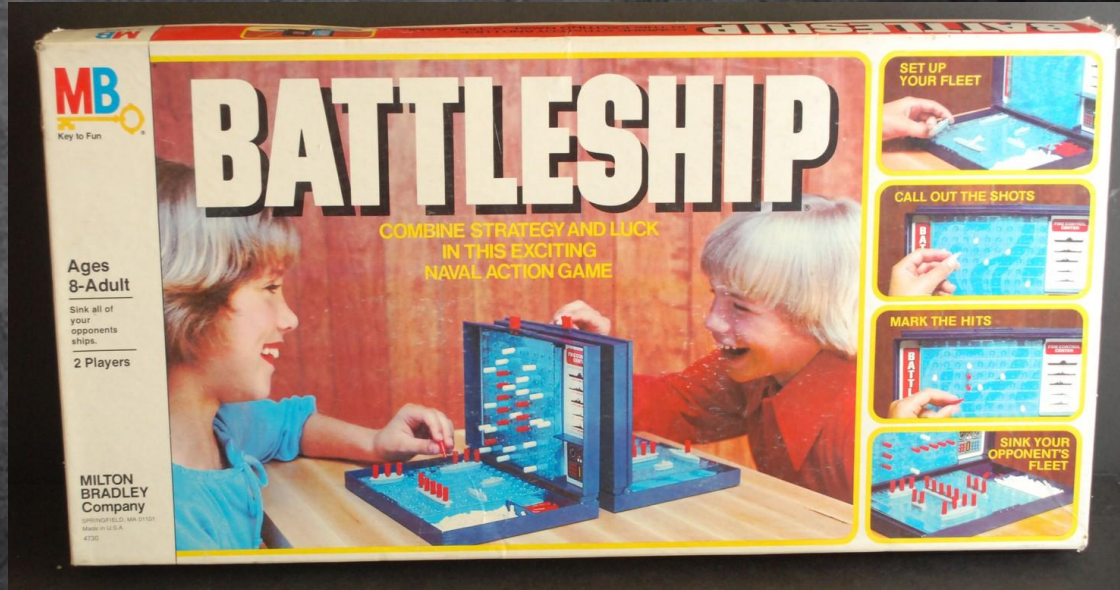


## 1. 개요

같은 이름의 보드게임을 원작으로 만들어진 실사영화. 이 실사영화를 **역**으로 **보드게임화**한 상품도 당연히 나왔다.

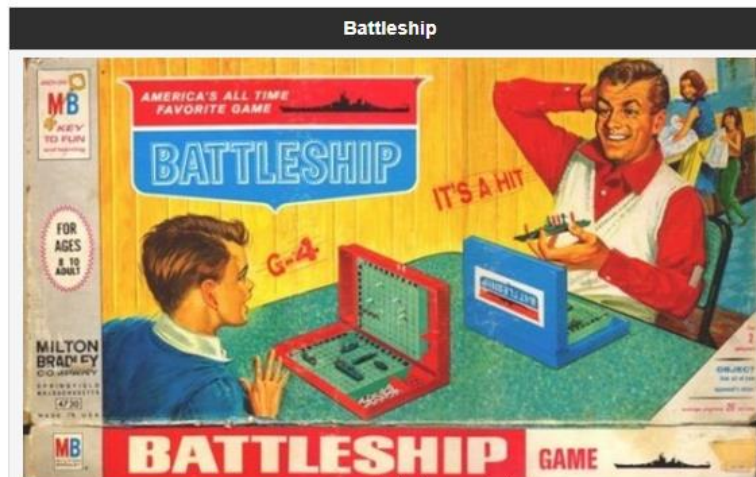
# BattleShip Game





# 1. 보드게임

- 상위 문서: [보드게임](#)



밀튼 브래들리 버전 1967년판

디자이너	Clifford Von Wickler
발매사	<a href="#">Milton Bradley</a> <sup>[1]</sup>
발매년	1931
인원	2명
플레이 시간	30분
연령	8세 이상 (보드게임식 : 6세 이상)
장르	어린이 게임
테마	어린이 게임 / 추리 / 해양 / <a href="#">워게임</a>
시스템	유닛 정보 숨김
홈페이지	<a href="#">유희각 페이지</a>

# 1.5. 바리에이션

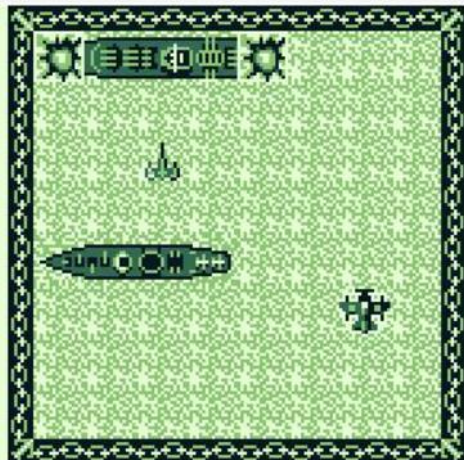
상당히 인기가 있었기 때문에 여러가지 버전으로도 포팅되었다.



1991년도에 나온 윈도우용 게임 Battle Sat.



1991년도에 나온 도스용 게임 Battle Fleet.



게임보이용 네이비블루.



# BattleShip Game

## 숫자야구의 2차원 버전

# Battleships!

## My Ships

A								
B								
C								
D								
E								
F								
G								
H								
	1	2	3	4	5	6	7	8



Aircraft Carrier 

A	A	A	A	A
---	---	---	---	---

Battleship 

B	B	B	B
---	---	---	---



Cruiser 

C	C	C
---	---	---

Destroyers 

D	D
---	---

D	D
---	---

Submarines 

S
---

S
---

## Their Ships

A								
B								
C								
D								
E								
F								
G								
H								
	1	2	3	4	5	6	7	8



Aircraft Carrier 

A	A	A	A	A
---	---	---	---	---

Battleship 

B	B	B	B
---	---	---	---



Cruiser 

C	C	C
---	---	---

Destroyers 

D	D
---	---

D	D
---	---

Submarines 

S
---

S
---

# BattleShip Game Rule

- 자신의 배를 임의의 위치에 배치
- 턴 순서를 결정
- 공격 턴에 상대방의 좌표를 지정
- 수비측은 결과를 통보
  - 빗나감, 히트(종류는 알려주지 않는다)
  - 격추 - 배의 영역이 모두 히트 하였을 때, 종류도 알려준다.
- 턴을 교대하면서 상대방의 모든 배를 격추 시키면 승리!



**Are you Ready?**

# BattleShip Project 목표

C++ 문법 및 프로그래밍 이해



```
graph TD; A[C++ 문법 및 프로그래밍 이해] --> B[C++ 객체 지향 설계]; B --> C[C++ 자료구조 및 알고리즘]; C --> D[게임 로직 프로그래밍 학습]; D --> E[AI 기반 게임 프로젝트 프로그래밍];
```

C++ 객체 지향 설계

C++ 자료구조 및 알고리즘

게임 로직 프로그래밍 학습

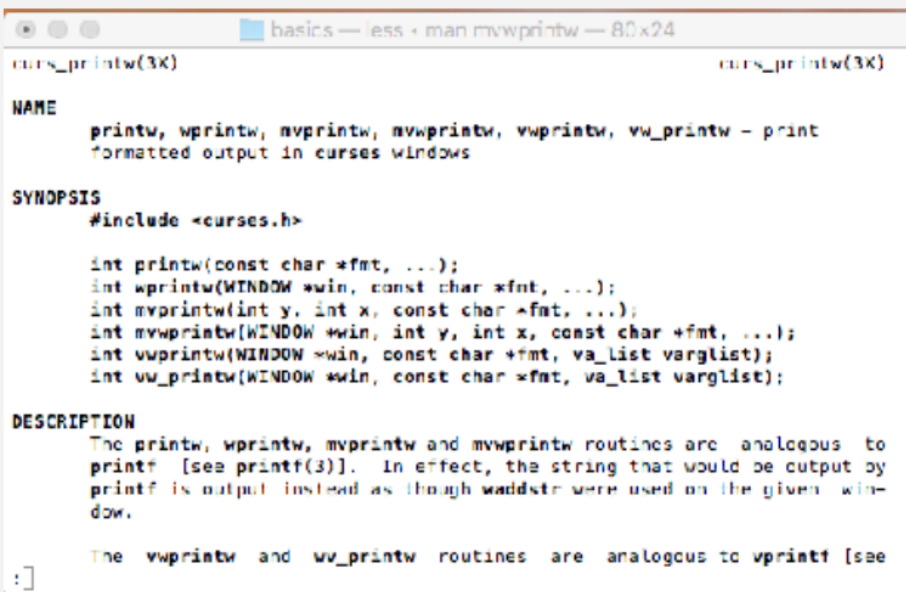
AI 기반 게임 프로젝트 프로그래밍



**NCURSES**

# ncurses

- text-based user-interface library
  - GPU ncurses <http://www.gnu.org/software/ncurses/>
  - NCURSES Programming HOWTO <http://tldp.org/HOWTO/NCURSES-Programming-HOWTO/> : with helpful examples
  - 한국어 번역 <https://wiki.kldp.org/wiki.php/NCURSES-Programming-HOWTO>
  - \$man ncurses on linux command prompt shows manual page
  - \$man mvvline etc.



```
basics -- less - man mvwprintw -- 80x24
curs_printw(3X)                                curs_printw(3X)

NAME
    printw, wprintw, mvprintw, mvwprintw, vwprintw, vw_printw - print
    formatted output in curses windows

SYNOPSIS
    #include <curses.h>

    int printw(const char *fmt, ...);
    int wprintw(WINDOW *win, const char *fmt, ...);
    int mvprintw(int y, int x, const char *fmt, ...);
    int mvwprintw(WINDOW *win, int y, int x, const char *fmt, ...);
    int vwprintw(WINDOW *win, const char *fmt, va_list varglist);
    int vw_printw(WINDOW *win, const char *fmt, va_list varglist);

DESCRIPTION
    The printw, wprintw, mvprintw and mvwprintw routines are analogous to
    printf [see printf(3)]. In effect, the string that would be output by
    printf is output instead as though waddstr were used on the given win-
    dow.

    The vwprintw and vw_printw routines are analogous to vprintf [see
    :]
```



# Hello\_world.cpp for ncurses

- compile command
  - g++ -o hello hello.cpp -lncurses

```
#include <ncurses.h>
int main()
{
    initscr();          /* Start curses mode      */
    printw("Hello World !!!"); /* Print Hello World */
    refresh();          /* Print it on to the real screen */
    getch();            /* Wait for user input */
    endwin();           /* End curses mode    */

    return 0;
}
```

# creating windows using ncurses library

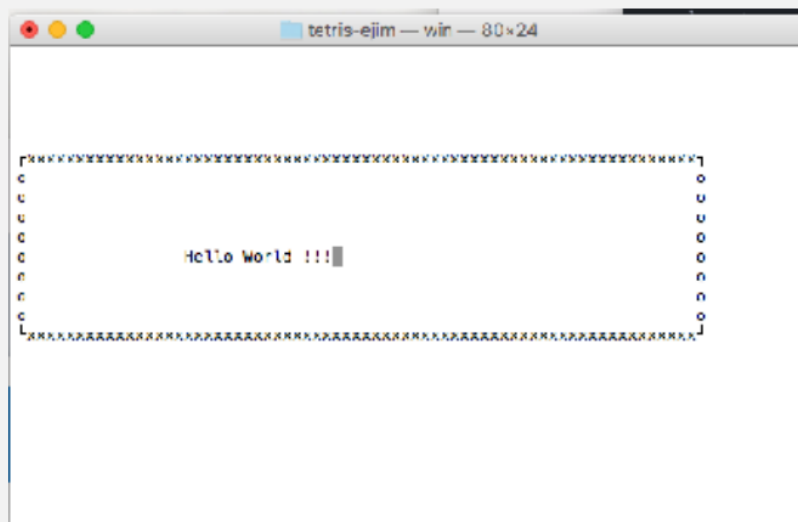
- window is a non-overlapping part of text screen.
- when curses is initialized, a default window named stdscr which represents the whole size of window in which your terminal (xterm) is running.

```
#include <ncurses.h>

int main()
{
    initscr();
    refresh(); // 꼭 필요함!!
    int height = 10, width = 70, x = 0, y = 5;
    WINDOW *w = newwin(height, width, y, x);
    box(w, 'o', 'x');
    mvwprintw(w, height/2, width/4, "Hello World !!!");
    wrefresh(w);

    getch(); /* Wait for user input */
    delwin(w);
    endwin(); /* End curses mode

    return 0;
}
```





# 1. 기본 UI 구성

<< Battle Ship Game >>

< MAP >	
A	00000000
B	00000000
C	00000000
D	00000000
E	00000000
F	00000000
G	00000000
H	00000000

12345678

< STATUS >

AIRCRAFT : AAAAA  
 BATTLESHIP : BBBB  
 CRUISER : CCC  
 DESTROYER: DD DD

< INPUT >

Input position...(ex A 3)  
 Input :

# Main.cpp

```
1  // BattleShip.cpp: 콘솔 응용 프로그램의 진입점을 정의합니다.  
2  //  
3  
4  #include "stdafx.h"  
5  #include "CBattleShipApp.h"  
6  
7  int main()  
8  {  
9      CBattleShipApp battleShip;  
10     battleShip.Play();  
11     return 0;  
12 }  
13
```



# BattleShipApp.h

```
class CBattleShipApp
{
public:
    CBattleShipApp();
    ~CBattleShipApp();

    void Play();

protected:
    void Init();
    void Render();
    void Destroy();

protected:
    CBattleShipMap* m_pMap;
    StatPane* m_pStatPane;
    InputPane* m_pInputPane;
};
```

# BattleShipApp.cpp

```
void CBattleShipApp::Init()
{
    initscr();
    start_color();
    cbreak();
    refresh();

    // 컬러 세팅
    init_pair(1, COLOR_GREEN, COLOR_BLACK);
    init_pair(2, COLOR_CYAN, COLOR_BLACK);
    init_pair(3, COLOR_YELLOW, COLOR_BLACK);

    m_pMap = new CBattleShipMap();
    m_pStatPane = new StatPane(30, 3, 30, 6);
    m_pInputPane = new InputPane(30, 15, 30, 4);
}
```

```
void CBattleShipApp::Play()
{
    Init();
    Render();
    Destroy();
}
```

```
void CBattleShipApp::Render()
{
    mvprintw(1, 1, "<< Battle Ship Game >>");

    m_pMap->Draw();
    m_pStatPane->Draw();
    m_pInputPane->Draw();

    refresh();
}
```

```
void CBattleShipApp::Destroy()
{
    getch();
    endwin();
    delete m_pMap;
}
```

# Pane.h

```
// 화면을 구성하는 Pane의 부모 클래스
class Pane
{
public:
    Pane(int x, int y, int width, int height);
    virtual ~Pane();

    virtual void Draw();

protected:
    int m_width, m_height;
    int m_x, m_y;
    WINDOW* m_pWindow;
};
```

```
Pane::Pane(int x, int y, int width, int height)
    :m_x(x), m_y(y), m_width(width), m_height(height)
{
    m_pWindow = newwin(height, width, y, x);
    box(m_pWindow, 0, 0);
    wrefresh(m_pWindow);
}

Pane::~Pane()
{
    delwin(m_pWindow);
}

void Pane::Draw()
{
    box(m_pWindow, 0, 0);
    wrefresh(m_pWindow);
}
```



# Pane.cpp

```
Pane::Pane(int x, int y, int width, int height)
: m_x(x), m_y(y), m_width(width), m_height(height)
{
    m_pWindow = newwin(height, width, y, x);
    box(m_pWindow, 0, 0);
    wrefresh(m_pWindow);
}

Pane::~Pane()
{
    delwin(m_pWindow);
}

void Pane::Draw()
{
    box(m_pWindow, 0, 0);
    wrefresh(m_pWindow);
}
```

# BattleShipMap.h

```
#include "Pane.h"

#define MAP_SIZE 8

// 게임 화면 맵을 표시하는 클래스
class CBattleShipMap : Pane
{
public:
    CBattleShipMap();
    ~CBattleShipMap();

    void Draw();

protected:
    char m_mapData[MAP_SIZE][MAP_SIZE];
};
```

# BattleShipMap.cpp #1

```
CBattleShipMap::CBattleShipMap()
:Pane(4, 4, MAP_SIZE + 3, MAP_SIZE + 2)
{
    for (int i = 0; i < MAP_SIZE; i++)
    {
        for (int j = 0; j < MAP_SIZE; ++j)
        {
            // 맵 데이터 초기화
            m_mapData[i][j] = '0';
        }
    }

    // 칸 구별 이름
    for (int i = 0; i < MAP_SIZE; ++i)
    {
        mvprintw(i + 1 + m_y, m_x - 1, "%c", 'A' + i);
        mvprintw(m_y + m_height, m_x + 2 + i, "%d", 1 + i);
    }

    // 타이틀
    mvprintw(m_pWindow, 0, 3, "< MAP >");
}
```

	< MAP >							
A	0	0	0	0	0	0	0	0
B	0	0	0	0	0	0	0	0
C	0	0	0	0	0	0	0	0
D	0	0	0	0	0	0	0	0
E	0	0	0	0	0	0	0	0
F	0	0	0	0	0	0	0	0
G	0	0	0	0	0	0	0	0
H	0	0	0	0	0	0	0	0
12345678								



# BattleShipMap.cpp #2

```
CBattleShipMap::~CBattleShipMap()
{
}

void CBattleShipMap::Draw()
{
    watttron(m_pWindow, COLOR_PAIR(1));
    for (int i = 0; i < MAP_SIZE; ++i)
    {
        for (int j = 0; j < MAP_SIZE; ++j)
        {
            mvwprintw(m_pWindow, i + 1, j + 2, "%c", m_mapData[i][j]);
        }
    }
    wattroff(m_pWindow, COLOR_PAIR(1));

    wrefresh(m_pWindow);
}
```

← MAP →

A	00000000
B	00000000
C	00000000
D	00000000
E	00000000
F	00000000
G	00000000
H	00000000

12345678

# StatPane.h

```
#include "Pane.h"
```

```
// 스태터스를 표시하는 윈도우
```

```
class StatPane : Pane
```

```
{
```

```
public:
```

```
    StatPane(int x, int y, int width, int height);
```

```
    ~StatPane();
```

```
    virtual void Draw();
```

```
};
```

```
StatPane::StatPane(int x, int y, int width, int height)  
    :Pane(x, y, width, height)
```

```
{
```

```
    // 타이틀
```

```
    mvprintw(m_pWindow, 0, 3, "< STATUS >");
```

```
}
```

# StatPane.cpp

```
void StatPane::Draw()  
{  
    wattron(m_pWindow, COLOR_PAIR(2));  
    mvwprintw(m_pWindow, 1, 2, "AIRCRAFT : AAAAA");  
    mvwprintw(m_pWindow, 2, 2, "BATTLESHIP : BBBB");  
    mvwprintw(m_pWindow, 3, 2, "CRUISER : CCC");  
    mvwprintw(m_pWindow, 4, 2, "DESTROYER: DD DD");  
    wattroff(m_pWindow, COLOR_PAIR(2));  
  
    wrefresh(m_pWindow);  
}
```

```
← STATUS →  
AIRCRAFT : AAAAA  
BATTLESHIP : BBBB  
CRUISER : CCC  
DESTROYER: DD DD
```



# StatPane.h

```
#include "Pane.h"
class InputPane :
{
    public Pane
{
public:
    InputPane(int x, int y, int width, int height);
    ~InputPane();

    virtual void Draw();
};
```

← INPUT →  
Input position...(ex A 3)  
Input :

# StatPane.cpp

```
InputPane::InputPane(int x, int y, int width, int height)
:Pane(x, y, width, height)
{
    // 타이틀
    mvwprintw(m_pWindow, 0, 3, "< INPUT >");
}

InputPane::~InputPane()
{
}

void InputPane::Draw()
{
    wattron(m_pWindow, COLOR_PAIR(3));
    mvwprintw(m_pWindow, 1, 2, "Input position...(ex A 3)");
    mvwprintw(m_pWindow, 2, 2, "Input : ");
    wattroff(m_pWindow, COLOR_PAIR(3));

    wrefresh(m_pWindow);
}
```



```
< INPUT >
Input position...(ex A 3)
Input :
```