게임 규칙

* 본 게임은 턴 방식입니다.
* 플레이어는 귀퉁이 중 임의의 한 곳에서부터 출발하게 됩니다.
* 게임을 시작하게 되면 플레이어는 경로 상의 장애물을 제거하면서 자원을 획득하여 점수를 획득합니다.
* 자원은 다이아몬드, 금, 은, 동이 있으며 각각 5점, 3점, 2점, 1점의 점수를 가지고 있습니다.
* 장애물을 제거하면 특수 아이템을 획득할 수도 있습니다.
* 플레이어가 맵 내의 모든 자원을 획득하게 되면 게임이 종료됩니다.
* 게임이 종료되는 시점에서 가장 점수를 많이 획득한 플레이어가 승리하게 됩니다.

장애물

* 장애물은 1부터 6까지의 내구성을 가지고 있습니다.
* 장애물의 내구성이 0 이하가 되면 제거됩니다.

플레이어

* 플레이어는 자신의 턴 차례에서 상하좌우 1칸 이동 또는 장애물 내구성을 자신이 보유하고 있는 파워만큼 제거할 수 있습니다.
* 플레이어의 기본 파워는 1입니다.
* 특수 아이템을 통해 파워를 증가시킬 수 있습니다.

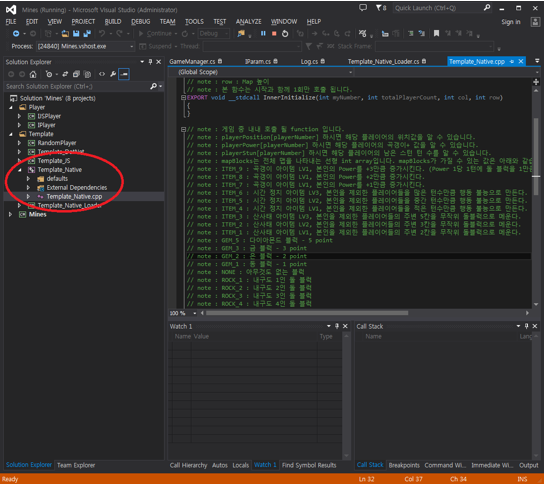
특수 아이템

* 각 특수 아이템은 3단계로 나누어집니다. (회색 : 1단계, 녹색 : 2단계, 청색 : 3단계)
* 곡괭이
  + 플레이어의 파워를 증가시킵니다. 증가된 파워는 게임 종료 시까지 유지됩니다.
  + 1단계 : 파워 1증가
  + 2단계 : 파워 2증가
  + 3단계 :파워 3증가
* 시간정지
  + 다른 플레이어를 일정 턴 동안 행동불능 상태로 만듭니다.
  + 1단계 : 맵 대각선 블럭 수 \* 1/3 만큼 행동 정지(예 : 20x16 크기의 맵인 경우 대각선 블럭 수는 약 25.6이므로 8턴 정지)
  + 2단계 : 맵 대각선 블럭 수 \* 2/3 만큼 행동 정지 (예 : 20x16 크기의 맵인 경우 17턴 정지)
  + 3단계 : 맵 대각선 블럭 수만큼 행동 정지 (예 : 20x16 크기의 맵인 경우 25턴 정지)
* 산사태
  + 다른 플레이어의 위치 주변에 장애물이 생깁니다.
  + 1단계 : 다른 플레이어 위치를 기준으로 2칸 반경에 장애물이 생깁니다.
  + 2단계 : 다른 플레이어 위치를 기준으로 3칸 반경에 장애물이 생깁니다.
  + 3단계 : 다른 플레이어 위치를 기준으로 5칸 반경에 장애물이 생깁니다.

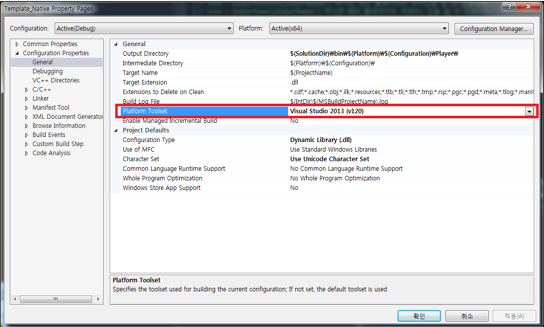
개발 가이드

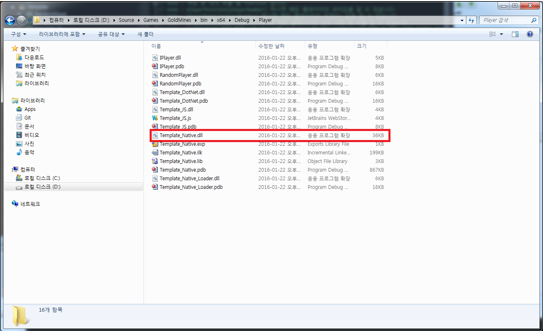
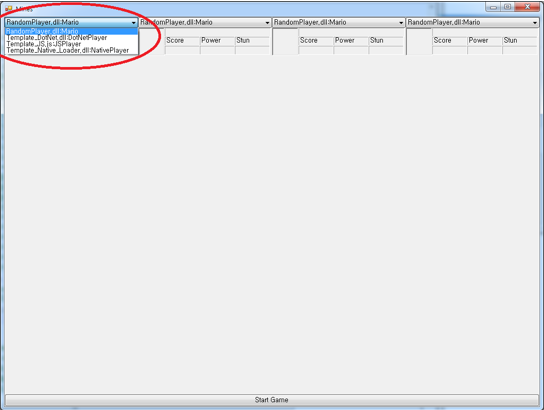
**C++**

* Visual Studio를 기준으로 설명하였습니다.
* 다운로드 : <https://github.com/elrha/GoldMine>

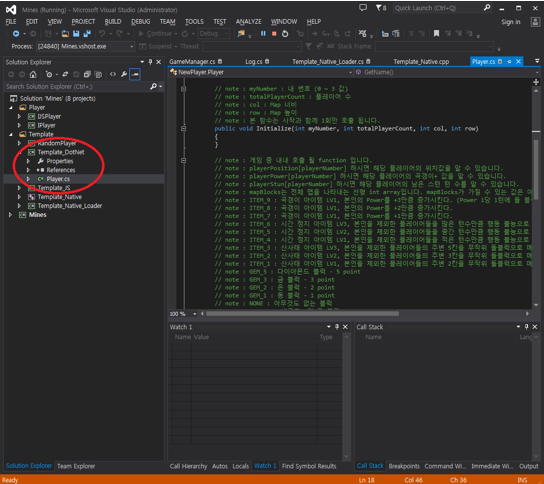
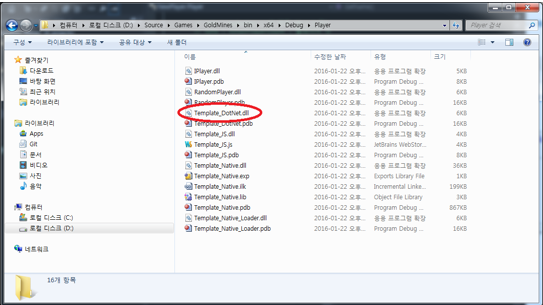
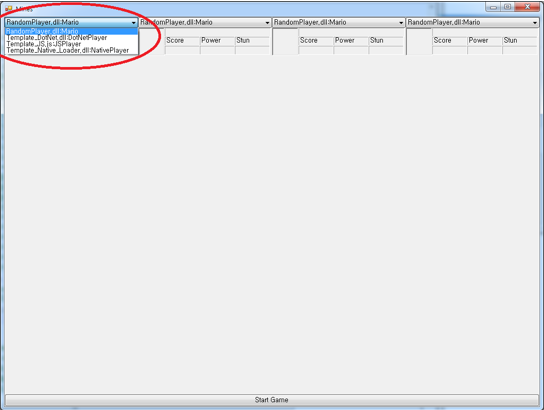


* 개발 프로젝트 : Template\_Native (위 그림 참조)
* 빌드가 안될 경우 (아래 그림 참조)
  + Solution Explorer에서 프로젝트 선택 > 마우스 우측버튼 클릭 > Properties(속성) 클릭합니다.
  + Configuration Properties > General > Platform Tooset에서 현재 개발 환경에서 지원하는 Platform 선택합니다.



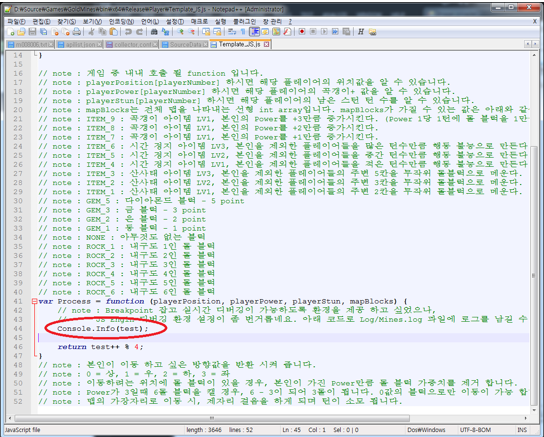
* Map Field 정의
  + ITEM\_9 = -1009 // 특수아이템(곡괭이 3단계)
  + ITEM\_8 = -1008 // 특수아이템(곡괭이 2단계)
  + ITEM\_7 = -1007 // 특수아이템(곡괭이 1단계)
  + ITEM\_6 = -1006 // 특수아이템(시간정지 3단계)
  + ITEM\_5 = -1005 // 특수아이템(시간정지 2단계)
  + ITEM\_4 = -1004 // 특수아이템(시간정지 1단계)
  + ITEM\_3 = -1003 // 특수아이템(산사태 3단계)
  + ITEM\_2 = -1002 // 특수아이템(산사태 2단계)
  + ITEM\_1 = -1001 // 특수아이템(산사태 1단계)
  + GEM\_5 = -5 // 자원(다이아몬드)
  + GEM\_3 = -3 // 자원(금)
  + GEM\_2 = -2 // 자원(은)
  + GEM\_1 = -1 // 자원(동)
  + NONE = 0 // 빈칸
  + ROCK\_1 = 1 // 장애물(내구성 1)
  + ROCK\_2 = 2 // 장애물(내구성 2)
  + ROCK\_3 = 3 // 장애물(내구성 3)
  + ROCK\_4 = 4 // 장애물(내구성 4)
  + ROCK\_5 = 5 // 장애물(내구성 5)
  + ROCK\_6 = 6 // 장애물(내구성 6)
* Return Value : Process Function이 Return해야 하는 값(플레이어의 이동 방향)
  + 0 : 상
  + 1 : 우
  + 2 : 하
  + 3 : 좌
  + 이동 방향에 장애물이 있는 경우, 플레이어의 파워만큼 장애물의 내구성을 감소시킵니다.
* 디버깅
  + Solution Explorer에서 프로젝트 선택 > 마우스 우측버튼 클릭 > Set as StartUp Project 선택 후 디버깅 메뉴 또는 F5 단축키를 이용하여 디버깅 합니다.
* 제출 파일
  + 빌드 후 Player 폴더의 Template\_Native.dll 파일을 이메일로 보내주시면 되겠습니다.
  + 빌드 적용했던 Platform Toolset Version도 같이 알려주세요.
* 결과물 확인
  + 초기 화면에서 작성한 DLL을 선택하고 하단의 [Start Game] 버튼을 클릭합니다.

**C#**

* 다운로드 : <https://github.com/elrha/GoldMine>
* 개발 대상 프로젝트 : Template\_DotNet (위 그림 참조)
* Map Field 정의
  + - * + ITEM\_9 = -1009 // 특수아이템(곡괭이 3단계)
        + ITEM\_8 = -1008 // 특수아이템(곡괭이 2단계)
        + ITEM\_7 = -1007 // 특수아이템(곡괭이 1단계)
        + ITEM\_6 = -1006 // 특수아이템(시간정지 3단계)
        + ITEM\_5 = -1005 // 특수아이템(시간정지 2단계)
        + ITEM\_4 = -1004 // 특수아이템(시간정지 1단계)
        + ITEM\_3 = -1003 // 특수아이템(산사태 3단계)
        + ITEM\_2 = -1002 // 특수아이템(산사태 2단계)
        + ITEM\_1 = -1001 // 특수아이템(산사태 1단계)
        + GEM\_5 = -5 // 자원(다이아몬드)
        + GEM\_3 = -3 // 자원(금)
        + GEM\_2 = -2 // 자원(은)
        + GEM\_1 = -1 // 자원(동)
        + NONE = 0 // 빈칸
        + ROCK\_1 = 1 // 장애물(내구성 1)
        + ROCK\_2 = 2 // 장애물(내구성 2)
        + ROCK\_3 = 3 // 장애물(내구성 3)
        + ROCK\_4 = 4 // 장애물(내구성 4)
        + ROCK\_5 = 5 // 장애물(내구성 5)
        + ROCK\_6 = 6 // 장애물(내구성 6)
* Return Value : Process Function이 Return해야 하는 값(플레이어의 이동 방향)
  + 0 : 상
  + 1 : 우
  + 2 : 하
  + 3 : 좌
  + 이동 방향에 장애물이 있는 경우, 플레이어의 파워만큼 장애물의 내구성을 감소시킵니다.
* 제출 파일
  + 빌드 후 Player 폴더의 Template\_ DotNet.dll 파일을 이메일로 보내주시면 되겠습니다.
* 주의사항
  + Template\_DotNet 프로젝트 외의 코드를 변경할 경우(특히 IPlayer 프로젝트) 접수 확인시 DLL 구동 불가로 참가가 불가할 수 있습니다.
* 결과물 확인
  + 초기 화면에서 작성한 DLL을 선택하고 하단의 [Start Game] 버튼을 클릭합니다.

**JavaScript**

* 다운로드 : <https://github.com/elrha/GoldMine>
* /bin/GoldMines.zip 압축 해제
* Player 폴더 내의 Template\_JS.js 파일을 수정하여 개발
* Log를 통한 Debugging 환경 제공
  + Console.Info(stringstring) 명령으로 Log폴더 내에 Log 기록을 남길 수 있습니다. (아래 화면 참조)



* Map Field 정의
  + - * + ITEM\_9 = -1009 // 특수아이템(곡괭이 3단계)
        + ITEM\_8 = -1008 // 특수아이템(곡괭이 2단계)
        + ITEM\_7 = -1007 // 특수아이템(곡괭이 1단계)
        + ITEM\_6 = -1006 // 특수아이템(시간정지 3단계)
        + ITEM\_5 = -1005 // 특수아이템(시간정지 2단계)
        + ITEM\_4 = -1004 // 특수아이템(시간정지 1단계)
        + ITEM\_3 = -1003 // 특수아이템(산사태 3단계)
        + ITEM\_2 = -1002 // 특수아이템(산사태 2단계)
        + ITEM\_1 = -1001 // 특수아이템(산사태 1단계)
        + GEM\_5 = -5 // 자원(다이아몬드)
        + GEM\_3 = -3 // 자원(금)
        + GEM\_2 = -2 // 자원(은)
        + GEM\_1 = -1 // 자원(동)
        + NONE = 0 // 빈칸
        + ROCK\_1 = 1 // 장애물(내구성 1)
        + ROCK\_2 = 2 // 장애물(내구성 2)
        + ROCK\_3 = 3 // 장애물(내구성 3)
        + ROCK\_4 = 4 // 장애물(내구성 4)
        + ROCK\_5 = 5 // 장애물(내구성 5)
        + ROCK\_6 = 6 // 장애물(내구성 6)
* Return Value : Process Function이 Return해야 하는 값(플레이어의 이동 방향)
  + 0 : 상
  + 1 : 우
  + 2 : 하
  + 3 : 좌
  + 이동 방향에 장애물이 있는 경우, 플레이어의 파워만큼 장애물의 내구성을 감소시킵니다
* 제출 파일
  + 작성하신 Template\_JS.js파일을 이메일로 보내주시면 되겠습니다.
* 결과물 확인
  + 초기 화면에서 작성한 DLL을 선택하고 하단의 [Start Game] 버튼을 클릭합니다.
  + 파일 수정 / 저장 후 [Start Game]을 다시 누르면 새로 저장된 파일을 재 로드하여 실행 하도록 되어 있으니, 굳이 프로그램을 재시작 하지 않아도 됩니다

