sdl-example-dodger

Generated by Doxygen 1.8.13

Contents

Chapter 1

Dodger!

2021학년도 프로그래밍방법론및실습 프로젝트 예제인 총알 피하기 게임이다. SDL2 라이브러리를 이용하여 제작하였으며 SDL2의 기본적인 사용법과 코드 구조를 설명하기 위하여 간단히 제작된 게임이다.

1.1 메인루프 구조

본 프로그램은 GUI 프로그램으로 `main()` 함수는 무한 루프 안에서 다음 서브루틴을 반복한다. 프로그램은 사용자가 창에 있는 "창닫기 버튼"을 눌러야만 종료된다.

1. 배경을 모두 지운다 (clear)
2. 외부 입력(키보드)를 받는다 (input)
3. 객체들의 다음 상태를 계산한다 (action)
4. 바뀐 객체들을 화면에 렌더링한다. (draw)
5. 렌더링한 결과를 화면에 띄운다 (show)
6. 정해진 시간 동안 대기한다 (delay)

1.2 모듈 설명

본 매뉴얼에는 메인 함수 무한루프에 들어가는 각 서브루틴별로 사용되는 함수를 분리하여 Modules 절에서 설명한다.

- Init : 무한루프 진입 전 초기화
- Input : 키보드 입력에 대한 응답
- Action : 주인공과 총알, 그리고 점수판의 변화를 계산하고 변경
- Draw : 실제 오브젝트들을 화면에 보여주기 위한 기능
- Utils : 많이 사용하는 기능을 함수로 구현

1.3 전역변수 사용

본 프로그램은 간단한 프로그램의 구현을 위해 키 입력과 SDL 창 관련 객체 (app) 주인공 (player), 총알 (bullet), 점수판 (score_board), 점수판에 표시할 텍스트 (score_text), 점수 (score)를 전역변수로 지정하여 사용한다. 각 전역변수 내용에 대한 설명은 GlobalVariables 에서 볼 수 있다.

Chapter 2

Module Index

2.1 Modules

Here is a list of all modules:

Chapter 3

Data Structure Index

3.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 4

File Index

4.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 5

Module Documentation

5.1 GlobalVariables

Variables

- App app
- Entity player
- Entity bullet [NUM_BULLETS]
- Entity game_over
- Text score_board
- char score_text [BUFSIZE]
- int score

5.1.1 Detailed Description

프로그램에서 사용하는 전역변수에 대해 설명한다.

5.1.2 Variable Documentation

5.1.2.1 app

```
App app
```

프로그램 전체 단위로 관리하는 전역 객체 모음

5.1.2.2 bullet

```
Entity bullet[NUM_BULLETS]
```

총알의 속성을 설명하는 Entity형 구조체

5.1.2.3 game_over

```
Entity game_over
```

게임오버 화면의 속성을 설명하는 Entity형 구조체

5.1.2.4 player

```
Entity player
```

주인공의 속성을 설명하는 Entity형 구조체

5.1.2.5 score

```
int score
```

게임 스코어

5.1.2.6 score_board

```
Text score_board
```

우상단 스코어보드 문자열을 설명하는 Text형 구조체

5.1.2.7 score_text

```
char score_text[BUFSIZE]
```

스코어보드에 출력할 문자열

5.2 Init

Functions

- void InitSDL (void)
  프로그램 수행에 필요한 초기화 과정 수행
- void InitTTF (void)
  TTF폰트 사용을 위한 초기화 과정 수행
- void QuitSDL (int flag)
  프로그램 종료에 필요한 루틴을 수행하고 프로그램 종료
- void QuitTTF (void)
  연 폰트 파일을 닫고 TTF 엔진 종료
- void InitMemorySet (void)
  전역 변수 메모리 내용 초기화
- void InitScoreBoard (void)
  스코어보드 초기화
- void InitPlayer (void)
  플레이어 초기화
- void InitBullet (void)
  총알 초기화
- void InitGameOver (void)
  게임오버 화면 초기화 (텍스쳐, 위치)

5.2.1 Detailed Description

Init 모듈은 main()에서 무한 루프가 실행되기 전에 전역 변수들과 SDL 요소들의 초기화를 수행하는 함수들이 정의되어 있다.

5.2.2 Function Documentation

5.2.2.1 InitBullet()

```
void InitBullet (
            void  )
```

총알 초기화

bullet 의 텍스쳐를 불러오고 랜덤한 장소로 위치를 설정하고 player 와의 각도를 계산한다. 텍스쳐는 gfx/Bullet.↩ png를 사용한다. 각 객체의 너비와 높이는 텍스쳐 정보를 바탕으로 Draw 모듈의 함수에서 일괄 계산하여 반영하므로 별도로 설정하지 않는다.

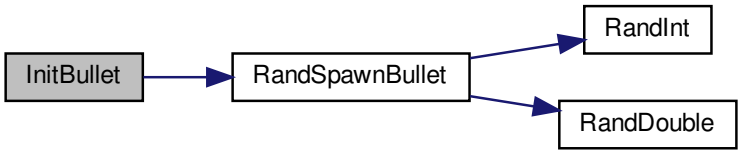
Remarks

전역변수 bullet , app 에 접근한다.

Returns

리턴 값 없음

Here is the call graph for this function:



5.2.2.2 InitGameOver()

```
void InitGameOver (
            void  )
```

게임오버 화면 초기화 (텍스쳐, 위치)

game_over 의 텍스쳐를 불러온다. game_over 의 텍스쳐는 화면 전체 크기와 동일하므로 위치는 (0, 0)으로 설정한다. 각 객체의 너비와 높이는 텍스쳐 정보를 바탕으로 Draw 모듈의 함수에서 일괄 계산하여 반영하므로 별도로 설정하지 않는다.

Remarks

전역변수 game_over , app 에 접근한다.

Returns

리턴 값 없음

5.2.2.3 InitMemorySet()

```
void InitMemorySet (
            void  )
```

전역 변수 메모리 내용 초기화

모든 전역 변수의 메모리 내용을 0으로 초기화한다.

Remarks

초기화 대상: app , game_over player , bullet , score_board .

Returns

리턴 값 없음

5.2.2.4 InitPlayer()

```
void InitPlayer (
            void  )
```

플레이어 초기화

player 의 텍스쳐를 불러오고 위치를 초기화하고 체력을 1로 초기화한다. 텍스쳐는 gfx/Player.png를 사용하고 위치는 화면의 중앙점으로 한다. 각 객체의 너비와 높이는 텍스쳐 정보를 바탕으로 Draw 모듈의 함수에서 일괄 계산하여 반영하므로 별도로 설정하지 않는다.

Remarks

전역변수 app 과 player 에 접근한다.

Returns

리턴 값 없음

5.2.2.5 InitScoreBoard()

```
void InitScoreBoard (
            void  )
```

스코어보드 초기화

score_board 의 글씨 색상과 위치를 초기화한다. 글씨 색상은 검정색, 위치는 (420, 40) 으로 설정한다. 각 객체의 너비와 높이는 텍스쳐 정보를 바탕으로 Draw 모듈의 함수에서 일괄 계산하여 반영하므로 별도로 설정하지 않는다.

Remarks

전역변수 score_board 에 접근한다.

Returns

리턴 값 없음

5.2.2.6 InitSDL()

```
void InitSDL (
            void  )
```

프로그램 수행에 필요한 초기화 과정 수행

SDL 초기화 과정을 수행한다. 또한 .png와 .jpg 파일을 사용하기 위한 작업도 수행한다.

Exceptions

초기화	실패 시 에러를 콘솔에 출력한다.

Remarks

전역변수 app 에 접근한다.

Returns

리턴 값 없음

5.2.2.7 InitTTF()

```
void InitTTF (
              void  )
```

TTF폰트 사용을 위한 초기화 과정 수행

TTF폰트 초기화 과정을 수행한다.

Exceptions

초기화	실패 시 에러를 콘솔에 출력하고 프로그램을 종료한다.

Remarks

전역변수 app 에 접근한다.

Returns

리턴 값 없음

5.2.2.8 QuitSDL()

```
void QuitSDL (
              int flag )
```

프로그램 종료에 필요한 루틴을 수행하고 프로그램 종료

프로그램 종료 전 전역변수 app 의 App::renderer 와 App::window 를 닫고 프로그램을 종료한다.

Parameters

in	*flag*	메인 함수에 넘겨줄 프로그램 리턴값

Remarks

전역변수 app 에 접근한다.

Returns

리턴 값 없음

Here is the call graph for this function:

5.2.2.9 QuitTTF()

```
void QuitTTF (
              void  )
```

연 폰트 파일을 닫고 TTF 엔진 종료

app 의 App::font 를 닫고 TTF 엔진을 종료한다.

Remarks

전역변수 app 에 접근한다.

Returns

리턴 값 없음

5.3 Input

Functions

- void GetInput (void)
  외부 입력을 받아 적절한 동작을 취하도록 한다.
- void ResponseKeyUp (SDL_KeyboardEvent *event)
  키보드를 뗐을 때 상태를 기록한다.
- void ResponseKeyDown (SDL_KeyboardEvent *event)
  키보드를 눌렀을 때 상태를 기록한다.

5.3.1 Detailed Description

Input 모듈은 키보드 입력에 반응하여 app 에 저장된 입력 상태를 저장하는 변수를 바꾸는 역할을 수행한다.

5.3.2 Function Documentation

5.3.2.1 GetInput()

```
void GetInput (
            void  )
```

외부 입력을 받아 적절한 동작을 취하도록 한다.

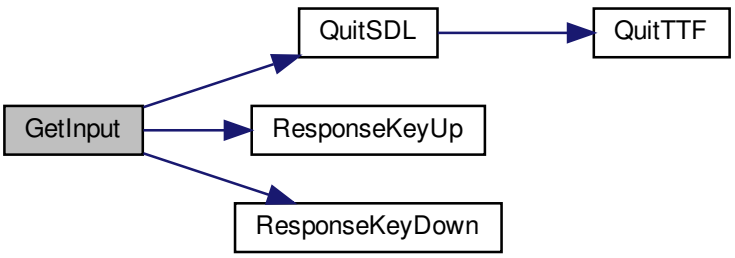
입력 처리에 대한 모든 내용을 담고 있는 함수이다. 다음과 같은 행동을 수행한다.

1. 창닫기 버튼을 누르면 프로그램 종료
2. 키보드를 누르면 ResponseKeyDown 호출
3. 키보드를 떼면 ResponseKeyUp 호출

Returns

리턴 값 없음

Here is the call graph for this function:



5.3.2.2 ResponseKeyDown()

```
void ResponseKeyDown (
            SDL_KeyboardEvent * event )
```

키보드를 눌렀을 때 상태를 기록한다.

app 의 멤버 App::key_up , App::key_down , App::key_left , App::key_right 는 키보드가 눌린 상태를 기록하는 변수이다. 이벤트에 따라 키보드가 눌린 키에 대한 변수를 1으로 설정한다.

Parameters

in	*event*	키보드 이벤트 내용을 담고 있는 구조체

Remarks

전역변수 App app 구조체에 접근한다.

Returns

리턴 값 없음

5.3.2.3 ResponseKeyUp()

```
void ResponseKeyUp (
            SDL_KeyboardEvent * event )
```

키보드를 뗐을 때 상태를 기록한다.

app 의 멤버 App::key_up , App::key_down , App::key_left , App::key_right 는 키보드가 눌린/떼진 상태를 기록하는 변수이다. 이벤트에 따라 키보드가 떼진 키에 대한 변수를 0으로 설정한다.

Parameters

in	*event*	키보드 이벤트 내용을 담고 있는 구조체

Remarks

전역변수 app 구조체에 접근한다.

Returns

리턴 값 없음

5.4 Action

Functions

- void LogicGame (void)
  게임 진행 중일 시 필요한 액션을 순차적으로 수행
- void LogicGameOver (void)
  게임오버된 상태에 필요한 액션을 순차적으로 수행
- void ActPlayer (void)
  키보드 입력 상태에 따라 주인공의 액션을 수행. 누른 화살표 버튼에 따라 움직이고, 화면 밖으로 나갈 수 없음.
- void ActBullet (void)
  총알의 액션을 수행. 주인공이 있는 방향으로 랜덤하게 날아가고, 화면 밖으로 나가면 리스폰됨.
- void ActScoreBoard (void)
  스코어보드 문자열 생성 (전역변수 score에 따라 현재 점수 표시)
- void ActFinalScoreBoard (void)
  게임 오버 시 최종 스코어 문자열 생성 (전역변수 score에 따라 최종 점수 표시)
- void ActCheckDeath (void)
  총알과 주인공의 충돌 여부를 계산하고 충돌 시 게임 오버시킴
- void ActGameOver (void)
  게임 오버 화면의 액션 수행. 아무 것도 수행하지 않다가, R키를 누르면 모든 구조체 초기화

5.4.1 Detailed Description

Action 모듈은 키보드 입력 상태, player 와 bullet 의 현재 위치, 그리고 현재 점수에 따라 Entity 구조체들의 상태를 변화시킨다.

5.4.2 Function Documentation

5.4.2.1 ActBullet()

```
void ActBullet (
            void  )
```

총알의 액션을 수행. 주인공이 있는 방향으로 랜덤하게 날아가고, 화면 밖으로 나가면 리스폰됨.

총알은 정해진 방향으로 BULLET_SPEED 만큼 이동한다. 총알의 이동 방향은 RandSpawnBullet 함수에서 주인공과 총알의 방향 $\pm 18^\circ$ 사이로 랜덤하게 결정되어 Entity::theta 에 저장되어 있다.

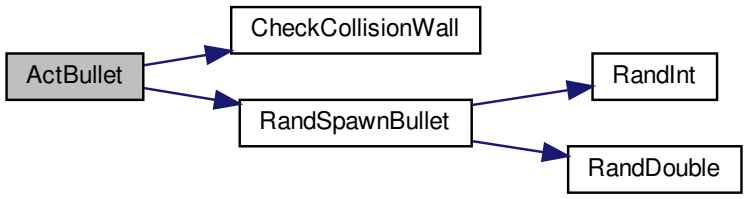
Remarks

전역 변수 bullet 에 접근한다.

Returns

리턴 값 없음

Here is the call graph for this function:



5.4.2.2 ActCheckDeath()

```
void ActCheckDeath (
            void  )
```

총알과 주인공의 충돌 여부를 계산하고 충돌 시 게임 오버시킴

모든 총알과 주인공이 충돌했는지 여부를 계산하여 충돌이 발생 시 player의 health를 0으로 만들어 게임 오버 조건을 만족시킨다. 충돌 여부는 CheckCollisionObjects 함수에서 계산한다.

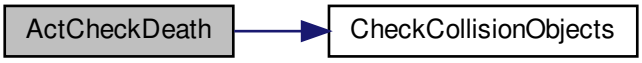
Remarks

전역 변수 bullet , player 에 접근한다.

Returns

리턴 값 없음

Here is the call graph for this function:



5.4.2.3 ActFinalScoreBoard()

```
void ActFinalScoreBoard (
            void  )
```

게임 오버 시 최종 스코어 문자열 생성 (전역변수 score에 따라 최종 점수 표시)

게임 오버 시 결과창에 띄울 문자열 "Your Final Score: [최종 점수]"을 만들고 score_board 의 texture에 저장한다.

Remarks

전역 변수 score_board, app, score_text, score 에 접근한다.

Returns

리턴 값 없음

5.4.2.4 ActGameOver()

```
void ActGameOver (
            void  )
```

게임 오버 화면의 액션 수행. 아무 것도 수행하지 않다가, R키를 누르면 모든 구조체 초기화

R키가 눌렸을 시 주인공, 총알, 스코어보드, 게임오버 Entity 를 초기화시킨다. 이 과정에서 player 의 health가 1로 복구되어 게임이 재시작된다.

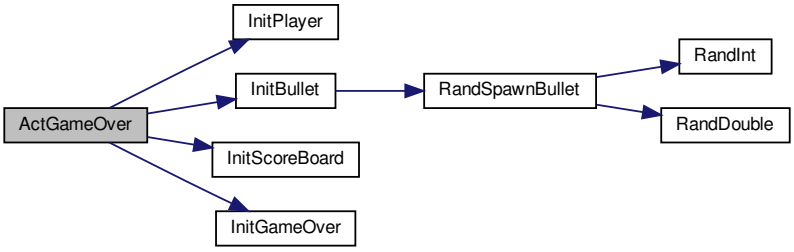
Remarks

전역 변수 app 에 접근한다.

Returns

리턴 값 없음

Here is the call graph for this function:



5.4.2.5 ActPlayer()

```
void ActPlayer (
            void  )
```

키보드 입력 상태에 따라 주인공의 액션을 수행. 누른 화살표 버튼에 따라 움직이고, 화면 밖으로 나갈 수 없음.

- 위 방향키가 눌린 상태인 경우 위 방향으로 PLAYER_SPEED 만큼 이동한다.
- 아래 방향키가 눌린 상태인 경우 아래 방향으로 PLAYER_SPEED 만큼 이동한다.
- 왼쪽 방향키가 눌린 상태인 경우 왼쪽 방향으로 PLAYER_SPEED 만큼 이동한다.
- 오른쪽 방향키가 눌린 상태인 경우 오른쪽 방향으로 PLAYER_SPEED 만큼 이동한다.

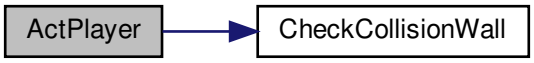
Remarks

전역 변수 app 과 player 에 접근한다.

Returns

리턴 값 없음

Here is the call graph for this function:



5.4.2.6 ActScoreBoard()

```
void ActScoreBoard (
            void  )
```

스코어보드 문자열 생성 (전역변수 score에 따라 현재 점수 표시)

"Your Score: [현재 점수]"의 문자열을 만들고 score_board 의 texture에 저장한다.

Remarks

전역 변수 score_board, app, score_text, score 에 접근한다.

Returns

리턴 값 없음

5.4.2.7 LogicGame()

```
void LogicGame (
            void  )
```

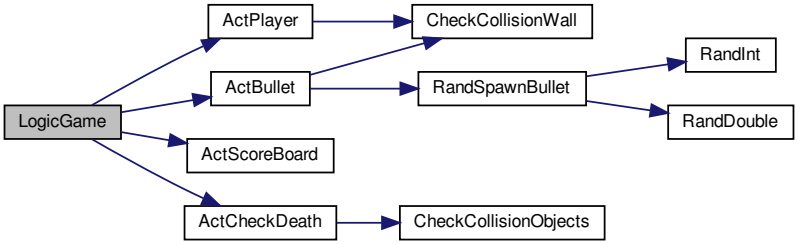
게임 진행 중일 시 필요한 액션을 순차적으로 수행

게임이 진행 중인 경우에 수행하는 액션을 모두 모아 순차적으로 실행하는 함수이다. main()함수에서 호출한다.

Returns

리턴 값 없음

Here is the call graph for this function:



5.4.2.8 LogicGameOver()

```
void LogicGameOver (
            void  )
```

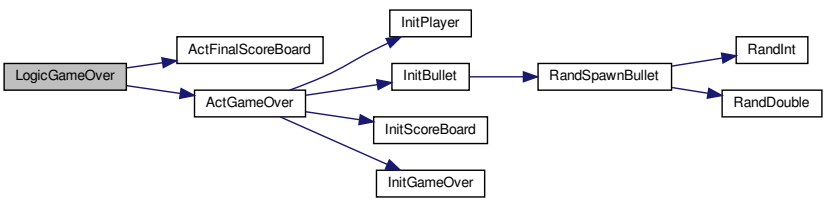
게임오버된 상태에 필요한 액션을 순차적으로 수행

게임오버된 경우에 수행하는 액션을 모두 모아 순차적으로 실행하는 함수이다. main()함수에서 호출한다.

Returns

리턴 값 없음

Here is the call graph for this function:



5.5 Draw

Functions

- void ClearWindow (void)
  배경을 흰색으로 지정하고 화면에 렌더링된 모든 요소 제거
- void ShowWindow (void)
  화면에 렌더링 결과 표시(보여주기)
- void DrawGame (void)
  게임 진행 상태에서 주인공, 총알, 스코어보드 렌더링
- void DrawGameOver (void)
  게임 오버 상태에서 게임 오버 화면, 최종 스코어 렌더링
- void RenderEntity (Entity *object)
  Entity 구조체 렌더링
- void RenderScoreBoard (Text *object)
  Text 구조체 렌더링 (스코어보드)

5.5.1 Detailed Description

Draw 모듈은 Action 이 수행된 이후 상태가 변경된 player , bullet , score_board 를 화면에 렌더링시켜주는 역할을 수행하는 함수들이 정의되어 있다.

그리고 ClearWindow 와 ShowWindow 는 각각 main() 내 무한 루프 처음과 끝에서 화면을 초기화하고 표시해주는 역할을 수행한다.

5.5.2 Function Documentation

5.5.2.1 ClearWindow()

```
void ClearWindow (
            void  )
```

배경을 흰색으로 지정하고 화면에 렌더링된 모든 요소 제거

배경 내용을 모두 지우고 흰색으로 칠한다. main() 내 무한 루프에서 가장 먼저 호출한다.

Remarks

전역변수 app 에 접근한다.

Returns

리턴 값 없음

5.5.2.2 DrawGame()

```
void DrawGame (
            void  )
```

게임 진행 상태에서 주인공, 총알, 스코어보드 렌더링

게임 진행 상태시 main()에서 호출하는 함수이다. 게임 진행 상태에서 주인공(player), 총알 (bullet), 스코어보드 (score_board)를 렌더링한다.

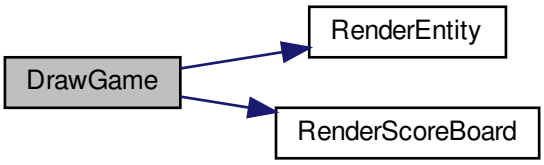
Remarks

전역변수 player , bullet , score_board 에 접근한다.

Returns

리턴 값 없음

Here is the call graph for this function:



5.5.2.3 DrawGameOver()

```
void DrawGameOver (
            void  )
```

게임 오버 상태에서 게임 오버 화면, 최종 스코어 렌더링

게임 오버 상태시 main()에서 호출하는 함수이다. 게임 오버 상태에서 게임 오버 화면 (game_over), 스코어보드 (score_board)를 렌더링한다.

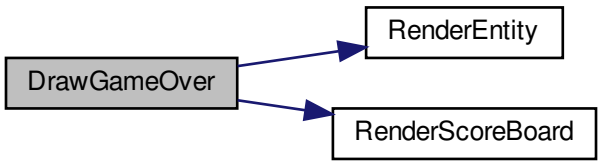
Remarks

전역변수 game_over , score_board 에 접근한다.

Returns

리턴 값 없음

Here is the call graph for this function:



5.5.2.4 RenderEntity()

```
void RenderEntity (
            Entity * object )
```

Entity 구조체 렌더링

Entity 를 렌더링한다. 대상 Entity 는 적절하게 초기화되어 Entity::texture 와 Entity::pos 의 위치 (x, y)가 설정되어 있어야 한다. SDL_QueryTexture 함수에서는 texture의 정보를 바탕으로 너비와 높이를 자동으로 계산하여 App.pos 에 반영한다.

Parameters

in	*object*	렌더링 대상 Entity 구조체

Remarks

전역변수 app 에 접근한다.

Returns

리턴 값 없음

5.5.2.5 RenderScoreBoard()

```
void RenderScoreBoard (
            Text * object )
```

Text 구조체 렌더링 (스코어보드)

Text 를 렌더링한다. 대상 Text 는 적절하게 초기화되어 Text::texture 와 Text::pos 의 위치 (x, y)가 설정되어 있어야 한다. SDL_QueryTexture 함수에서는 texture의 정보를 바탕으로 너비와 높이를 자동으로 계산하여 App.pos 에 반영한다.

Parameters

in	*object*	렌더링 대상 Text 구조체

Remarks

전역변수 app 에 접근한다.

Returns

리턴 값 없음

5.5.2.6 ShowWindow()

```
void ShowWindow (
            void  )
```

화면에 렌더링 결과 표시(보여주기)

Action 과 Draw 모듈에서 수행한 렌더링 결과를 화면에 표시한다.

Remarks

전역변수 app 에 접근한다.

Returns

리턴 값 없음

5.6 Utils

Functions

- int CheckCollisionWall (Entity *object)
  주인공 혹은 총알이 벽 밖으로 넘어갔는지 확인
- int CheckCollisionObjects (Entity *object_a, Entity *object_b)
  두 Entity 간 충돌 여부를 판단
- void RandSpawnBullet (Entity *object)
  좌상, 좌하, 우상, 우하단 중 랜덤한 위치에 총알 소환 후 주인공과의 각도 계산
- int RandInt (int min_val, int max_val)
  [min_val, max_val) 사이의 무작위 정수를 리턴
- double RandDouble (double min_val, double max_val)
  [min_val, max_val) 사이의 무작위 실수를 리턴

5.6.1 Detailed Description

Utils 모듈은 자주 사용하는 계산을 함수 형태로 구현한 모듈이다.

5.6.2 Function Documentation

5.6.2.1 CheckCollisionObjects()

```
int CheckCollisionObjects (
            Entity * object_a,
            Entity * object_b )
```

두 Entity 간 충돌 여부를 판단

두 Entity 간 충돌 여부를 판단한다. Entity::pos 는 SDL2 라이브러리에서 정의한 SDL_Rect 구조체이다. 이 구조체는 멤버 변수로 (x, y)좌표와 (w, h) 너비/높이 쌍을 가진다. (x, y)는 초기화 과정 또는 Action 모듈에서 루프마다 계산되며 (w, h)는 Draw 모듈에서 텍스쳐 정보를 바탕으로 계산된다. SDL2에서 제공하는 SDL_HasIntersection 함수를 이용하여 두 개의 SDL_Rect 구조체에 교집합이 있는지 판별하는 방법으로 충돌을 계산한다.

Parameters

in	*object_a*	충돌 여부를 판단할 첫 번째 Entity 구조체
in	*object_b*	충돌 여부를 판단할 두 번째 Entity 구조체

Returns

충돌했으면 1, 충돌하지 않았으면 0

5.6.2.2 CheckCollisionWall()

```
int CheckCollisionWall (
            Entity * object )
```

주인공 혹은 총알이 벽 밖으로 넘어갔는지 확인

Entity 객체가 벽 밖으로 넘어갔는지 판정한다.

Parameters

in	*object*	탐지 대상 Entity형 구조체

Returns

벽 밖으로 넘어가면 1, 벽 안에 있으면 0

5.6.2.3 RandDouble()

```
double RandDouble (
            double min_val,
            double max_val )
```

[min_val, max_val) 사이의 무작위 실수를 리턴

Parameters

in	*min_val*	최솟값
in	*max_val*	최댓값

Returns

생성된 무작위 실수

5.6.2.4 RandInt()

```
int RandInt (
            int min_val,
            int max_val )
```

[min_val, max_val) 사이의 무작위 정수를 리턴

Parameters

in	*min_val*	최솟값
in	*max_val*	최댓값

Returns

생성된 무작위 정수

5.6.2.5 RandSpawnBullet()

```
void RandSpawnBullet (
            Entity * object )
```

좌상, 좌하, 우상, 우하단 중 랜덤한 위치에 총알 소환 후 주인공과의 각도 계산

랜덤한 위치에 총알을 소환한다. 화면 너비를 W, 화면 높이를 H 라고 했을 때, 총알은 지점 1: $(0 \leq x \leq \frac{1}{8}W, 0 \leq y \leq \frac{1}{8}H)$, 지점 2: $(\frac{7}{8}W \leq x \leq W, 0 \leq y \leq \frac{1}{8}H)$, 지점 3: $(0 \leq x \leq \frac{1}{8}W, \frac{7}{8}H \leq y \leq H)$, 지점 4: $(\frac{7}{8}W \leq x \leq W, \frac{7}{8}H \leq y \leq H)$ 중 한 군데에 랜덤 소환된다.

총알은 소환된 이후 주인공과의 각도를 계산한다. 그리고 Entity::theta 에 주인공과 총알 사이의 각도 $\pm 18°$ 을 저장한다. Action 모듈에서는 그 방향으로 계속 총알을 움직인다.

Parameters

in	*object*	랜덤 소환할 총알 Entity

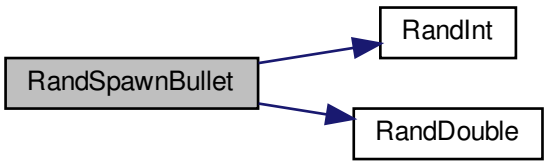
Remarks

전역변수 player 에 접근한다.

Returns

리턴 값 없음

Here is the call graph for this function:



Chapter 6

Data Structure Documentation

6.1 App Struct Reference

App: 프로그램 전체적으로 관리해야 하는 요소를 모아 놓은 구조체

`#include <defs.h>`

Data Fields

- SDL_Renderer * renderer
- SDL_Window * window
- TTF_Font * font
- int key_up
- int key_down
- int key_left
- int key_right
- int key_r

6.1.1 Detailed Description

App: 프로그램 전체적으로 관리해야 하는 요소를 모아 놓은 구조체

6.1.2 Field Documentation

6.1.2.1 font

`TTF_Font* font`

폰트 관리를 위한 구조체

6.1.2.2 key_down

```
int key_down
```

아래 방향키가 눌린 상태를 저장하는 변수

6.1.2.3 key_left

```
int key_left
```

왼쪽 방향키가 눌린 상태를 저장하는 변수

6.1.2.4 key_r

```
int key_r
```

R키가 눌린 상태를 저장하는 변수

6.1.2.5 key_right

```
int key_right
```

오른쪽 방향키가 눌린 상태를 저장하는 변수

6.1.2.6 key_up

```
int key_up
```

위 방향키가 눌린 상태를 저장하는 변수

6.1.2.7 renderer

```
SDL_Renderer* renderer
```

렌더링 관리를 위한 구조체

6.1.2.8 window

```
SDL_Window* window
```

창 관리를 위한 구조체

The documentation for this struct was generated from the following file:

- defs.h

6.2 Entity Struct Reference

Entity: 게임 내에서 움직이는 물체를 구현하기 위한 구조체(주인공, 총알)

`#include <defs.h>`

Data Fields

- SDL_Rect pos
- double theta
- int health
- SDL_Texture * texture

6.2.1 Detailed Description

Entity: 게임 내에서 움직이는 물체를 구현하기 위한 구조체(주인공, 총알)

6.2.2 Field Documentation

6.2.2.1 health

`int health`

주인공의 체력 상태를 나타내는 변수 (생존 1, 사망 0)

6.2.2.2 pos

`SDL_Rect pos`

직사각형 객체의 상태를 나타내기 위한 구조체 여기에 객체의 좌표, 위치 저장

6.2.2.3 texture

`SDL_Texture* texture`

텍스쳐를 담고 있는 구조체 (그림파일을 열어 텍스쳐에 저장)

6.2.2.4 theta

`double theta`

총알-주인공 간 각도를 저장하는 변수

The documentation for this struct was generated from the following file:

- defs.h

6.3 Text Struct Reference

Text: 게임 내에 문자열을 표시할 경우 문자열을 나타내는 구조체(스코어보드)

`#include <defs.h>`

Data Fields

- SDL_Rect pos
- SDL_Color color
- SDL_Surface * surface
- SDL_Texture * texture

6.3.1 Detailed Description

Text: 게임 내에 문자열을 표시할 경우 문자열을 나타내는 구조체(스코어보드)

6.3.2 Field Documentation

6.3.2.1 color

`SDL_Color color`

글씨 색깔을 저장하는 구조체

6.3.2.2 pos

`SDL_Rect pos`

직사각형 객체의 상태를 나타내기 위한 구조체 여기에 객체의 좌표, 위치 저장

6.3.2.3 surface

`SDL_Surface* surface`

폰트 렌더링을 위해 필요한 구조체

6.3.2.4 texture

`SDL_Texture* texture`

텍스쳐를 담고 있는 구조체 (문자열을 surface로 만들고, 그 후 texture에 저장)

The documentation for this struct was generated from the following file:

- defs.h

Chapter 7

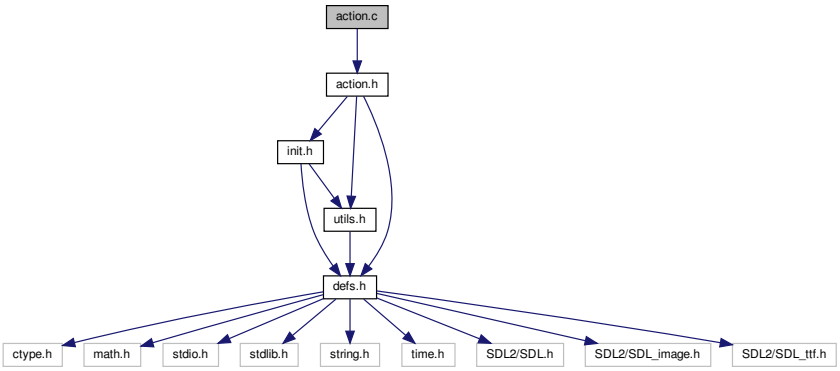
File Documentation

7.1 action.c File Reference

키보드 입력, 현재 주인공 및 총알의 상태를 바탕으로 액션을 수행 (상태를 변경)하는 함수 정의

```
#include "action.h"
```

Include dependency graph for action.c:



Functions

- void LogicGame (void)
  게임 진행 중일 시 필요한 액션을 순차적으로 수행
- void LogicGameOver (void)
  게임오버된 상태에 필요한 액션을 순차적으로 수행
- void ActPlayer (void)
  키보드 입력 상태에 따라 주인공의 액션을 수행. 누른 화살표 버튼에 따라 움직이고, 화면 밖으로 나갈 수 없음.
- void ActBullet (void)
  총알의 액션을 수행. 주인공이 있는 방향으로 랜덤하게 날아가고, 화면 밖으로 나가면 리스폰됨.
- void ActScoreBoard (void)
  스코어보드 문자열 생성 (전역변수 score에 따라 현재 점수 표시)
- void ActFinalScoreBoard (void)

게임 오버 시 최종 스코어 문자열 생성 (전역변수 score에 따라 최종 점수 표시)

- void ActCheckDeath (void)

  총알과 주인공의 충돌 여부를 계산하고 충돌 시 게임 오버시킴

- void ActGameOver (void)

  게임 오버 화면의 액션 수행. 아무 것도 수행하지 않다가, R키를 누르면 모든 구조체 초기화

7.1.1 Detailed Description

키보드 입력, 현재 주인공 및 총알의 상태를 바탕으로 액션을 수행 (상태를 변경)하는 함수 정의

Author

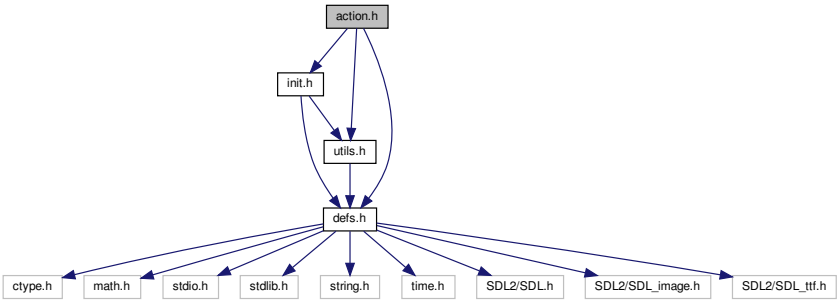
이성재 (seongjae.lee.1118@gmail.com)

7.2 action.h File Reference

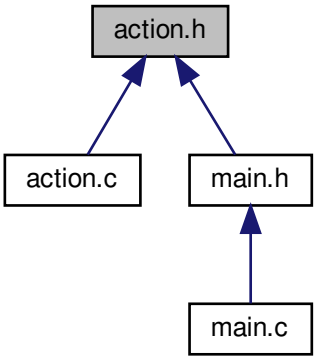
키보드 입력, 현재 주인공 및 총알의 상태를 바탕으로 액션을 수행 (상태를 변경)하는 함수 선언

```
#include "init.h"
#include "defs.h"
#include "utils.h"
```

Include dependency graph for action.h:



This graph shows which files directly or indirectly include this file:



Functions

- void LogicGame (void)
  게임 진행 중일 시 필요한 액션을 순차적으로 수행
- void LogicGameOver (void)
  게임오버된 상태에 필요한 액션을 순차적으로 수행
- void ActPlayer (void)
  키보드 입력 상태에 따라 주인공의 액션을 수행. 누른 화살표 버튼에 따라 움직이고, 화면 밖으로 나갈 수 없음.
- void ActBullet (void)
  총알의 액션을 수행. 주인공이 있는 방향으로 랜덤하게 날아가고, 화면 밖으로 나가면 리스폰됨.
- void ActScoreBoard (void)
  스코어보드 문자열 생성 (전역변수 score에 따라 현재 점수 표시)
- void ActFinalScoreBoard (void)
  게임 오버 시 최종 스코어 문자열 생성 (전역변수 score에 따라 최종 점수 표시)
- void ActCheckDeath (void)
  총알과 주인공의 충돌 여부를 계산하고 충돌 시 게임 오버시킴
- void ActGameOver (void)
  게임 오버 화면의 액션 수행. 아무 것도 수행하지 않다가, R키를 누르면 모든 구조체 초기화

Variables

- App app
- Entity player
- Entity bullet [NUM_BULLETS]
- Entity game_over
- Text score_board
- char score_text [BUFSIZE]
- int score

7.2.1 Detailed Description

키보드 입력, 현재 주인공 및 총알의 상태를 바탕으로 액션을 수행 (상태를 변경)하는 함수 선언

Author

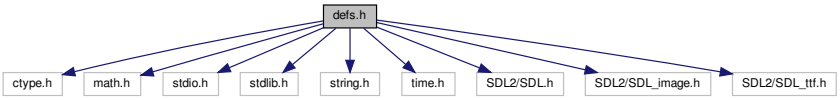
이성재 (seongjae.lee.1118@gmail.com)

7.3 defs.h File Reference

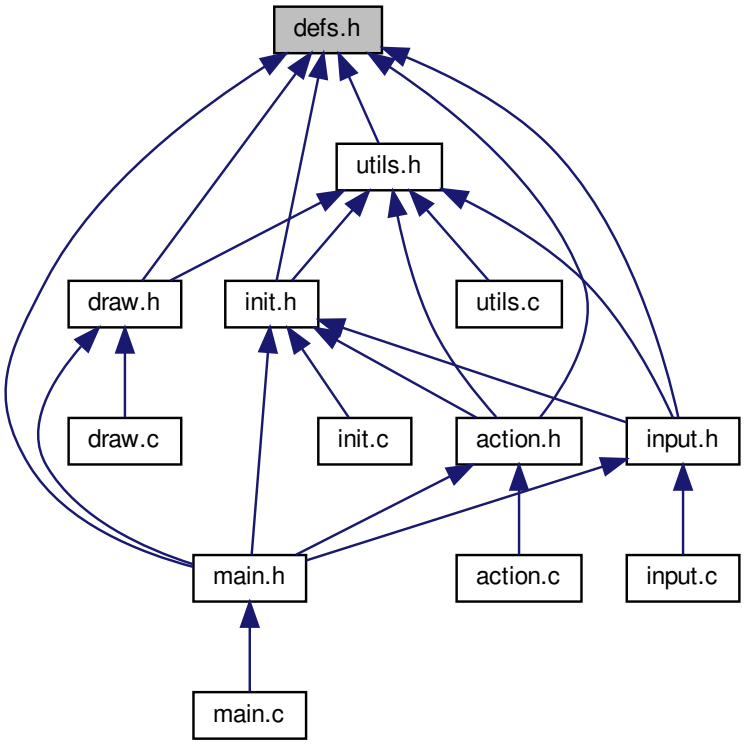
데이터타입 및 상수 정의

```
#include "ctype.h"
#include "math.h"
#include "stdio.h"
#include "stdlib.h"
#include "string.h"
#include "time.h"
#include "SDL2/SDL.h"
#include "SDL2/SDL_image.h"
#include "SDL2/SDL_ttf.h"
```

Include dependency graph for defs.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct App

  App: 프로그램 전체적으로 관리해야 하는 요소를 모아 놓은 구조체
- struct Entity

  Entity: 게임 내에서 움직이는 물체를 구현하기 위한 구조체(주인공, 총알)
- struct Text

  Text: 게임 내에 문자열을 표시할 경우 문자열을 나타내는 구조체(스코어보드)

Macros

- #define FPS 60
- #define BUFSIZE 1024
- #define SCREEN_WIDTH 640
- #define SCREEN_HEIGHT 480
- #define PLAYER_WIDTH 24
- #define PLAYER_HEIGHT 24
- #define PLAYER_SPEED 4
- #define BULLET_WIDTH 8
- #define BULLET_HEIGHT 8
- #define BULLET_SPEED 6
- #define NUM_BULLETS 16
- #define FONTSIZE 20

7.3.1 Detailed Description

데이터타입 및 상수 정의

Author

이성재 (seongjae.lee.1118@gmail.com)

7.3.2 Macro Definition Documentation

7.3.2.1 BUFSIZE

```
#define BUFSIZE 1024
```

문자열 버퍼 크기

7.3.2.2 BULLET_HEIGHT

```
#define BULLET_HEIGHT 8
```

총알 객체 높이(픽셀)

7.3.2.3 BULLET_SPEED

```
#define BULLET_SPEED 6
```

총알 객체 속도(단위시간당 이동량)

7.3.2.4 BULLET_WIDTH

```
#define BULLET_WIDTH 8
```

총알 객체 너비(픽셀)

7.3.2.5 FONTSIZE

```
#define FONTSIZE 20
```

출력할 문자열 폰트 크기

7.3.2.6 FPS

```
#define FPS 60
```

게임 FPS

7.3.2.7 NUM_BULLETS

```
#define NUM_BULLETS 16
```

총알 전체 갯수

7.3.2.8 PLAYER_HEIGHT

```
#define PLAYER_HEIGHT 24
```

플레이어 객체 높이(픽셀)

7.3.2.9 PLAYER_SPEED

```
#define PLAYER_SPEED 4
```

플레이어 객체 속도(단위시간당 이동량)

7.3.2.10 PLAYER_WIDTH

```
#define PLAYER_WIDTH 24
```

플레이어 객체 너비(픽셀)

7.3.2.11 SCREEN_HEIGHT

```
#define SCREEN_HEIGHT 480
```

화면 높이(픽셀)

7.3.2.12 SCREEN_WIDTH

```
#define SCREEN_WIDTH 640
```

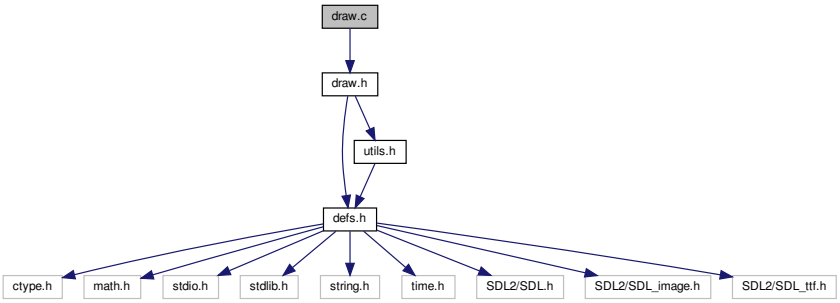
화면 너비(픽셀)

7.4 draw.c File Reference

텍스쳐 렌더링을 수행하는 함수 정의

```
#include "draw.h"
```

Include dependency graph for draw.c:



Functions

- void ClearWindow (void)

  배경을 흰색으로 지정하고 화면에 렌더링된 모든 요소 제거
- void ShowWindow (void)

  화면에 렌더링 결과 표시(보여주기)
- void DrawGame (void)

  게임 진행 상태에서 주인공, 총알, 스코어보드 렌더링
- void DrawGameOver (void)

  게임 오버 상태에서 게임 오버 화면, 최종 스코어 렌더링
- void RenderEntity (Entity *object)

  Entity 구조체 렌더링
- void RenderScoreBoard (Text *object)

  Text 구조체 렌더링 (스코어보드)

7.4.1 Detailed Description

텍스쳐 렌더링을 수행하는 함수 정의

Author

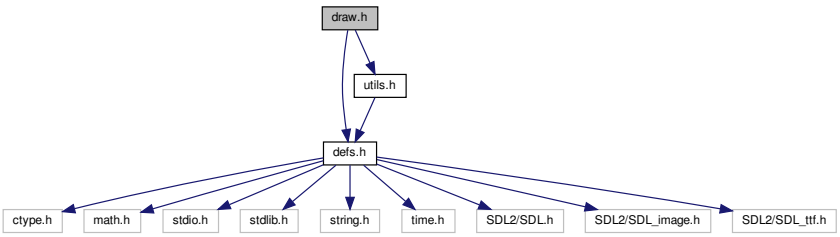
이성재 (seongjae.lee.1118@gmail.com)

7.5 draw.h File Reference

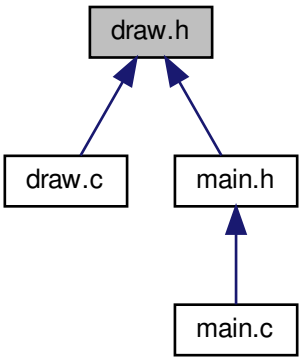
텍스쳐 렌더링을 수행하는 함수 선언

```
#include "defs.h"
#include "utils.h"
```

Include dependency graph for draw.h:



This graph shows which files directly or indirectly include this file:



Functions

- void ClearWindow (void)
  배경을 흰색으로 지정하고 화면에 렌더링된 모든 요소 제거
- void ShowWindow (void)
  화면에 렌더링 결과 표시(보여주기)
- void DrawGame (void)
  게임 진행 상태에서 주인공, 총알, 스코어보드 렌더링
- void DrawGameOver (void)
  게임 오버 상태에서 게임 오버 화면, 최종 스코어 렌더링
- void RenderEntity (Entity *object)
  Entity 구조체 렌더링
- void RenderScoreBoard (Text *object)
  Text 구조체 렌더링 (스코어보드)

Variables

- App app
- Entity player
- Entity bullet [NUM_BULLETS]
- Entity game_over
- Text score_board
- char score_text [BUFSIZE]
- int score

7.5.1 Detailed Description

텍스쳐 렌더링을 수행하는 함수 선언

Author

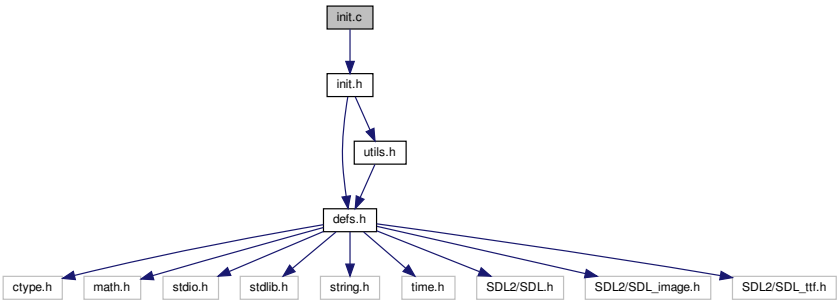
이성재 (seongjae.lee.1118@gmail.com)

7.6 init.c File Reference

무한 루프 진입 전 객체 및 SDL 요소 초기화를 위한 함수 정의

```
#include "init.h"
```

Include dependency graph for init.c:



Functions

- void InitSDL (void)
  프로그램 수행에 필요한 초기화 과정 수행
- void InitTTF (void)
  TTF폰트 사용을 위한 초기화 과정 수행
- void QuitSDL (int flag)
  프로그램 종료에 필요한 루틴을 수행하고 프로그램 종료
- void QuitTTF (void)
  연 폰트 파일을 닫고 TTF 엔진 종료
- void InitMemorySet (void)
  전역 변수 메모리 내용 초기화
- void InitScoreBoard (void)
  스코어보드 초기화
- void InitPlayer (void)
  플레이어 초기화
- void InitBullet (void)
  총알 초기화
- void InitGameOver (void)
  게임오버 화면 초기화 (텍스쳐, 위치)

7.6.1 Detailed Description

무한 루프 진입 전 객체 및 SDL 요소 초기화를 위한 함수 정의

Author

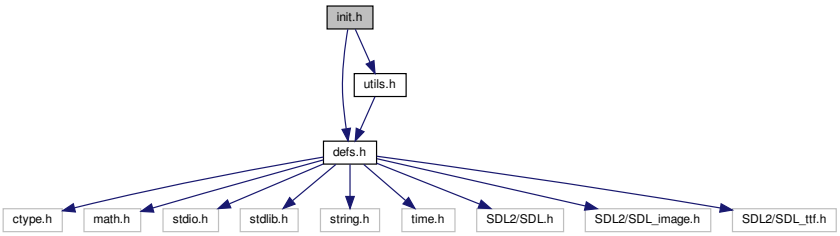
이성재 (seongjae.lee.1118@gmail.com)

7.7 init.h File Reference

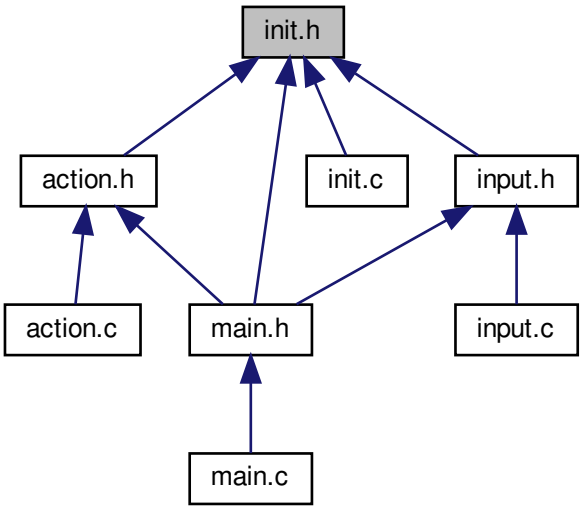
무한 루프 진입 전 객체 및 SDL 요소 초기화를 위한 함수 선언

```
#include "defs.h"
#include "utils.h"
```

Include dependency graph for init.h:



This graph shows which files directly or indirectly include this file:



Functions

- void InitSDL (void)
  프로그램 수행에 필요한 초기화 과정 수행
- void InitTTF (void)
  TTF폰트 사용을 위한 초기화 과정 수행
- void QuitSDL (int flag)
  프로그램 종료에 필요한 루틴을 수행하고 프로그램 종료

- void QuitTTF (void)
  연 폰트 파일을 닫고 TTF 엔진 종료
- void InitMemorySet (void)
  전역 변수 메모리 내용 초기화
- void InitScoreBoard (void)
  스코어보드 초기화
- void InitPlayer (void)
  플레이어 초기화
- void InitBullet (void)
  총알 초기화
- void InitGameOver (void)
  게임오버 화면 초기화 (텍스쳐, 위치)

Variables

- App app
- Entity player
- Entity bullet [NUM_BULLETS]
- Entity game_over
- Text score_board
- char score_text [BUFSIZE]
- int score

7.7.1 Detailed Description

무한 루프 진입 전 객체 및 SDL 요소 초기화를 위한 함수 선언

Author

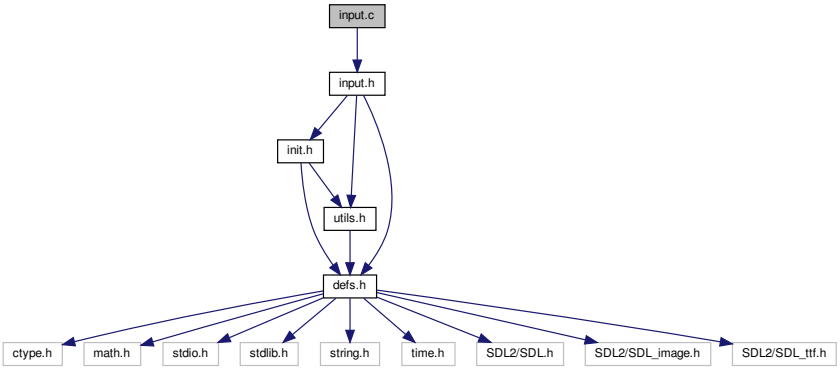
이성재 (seongjae.lee.1118@gmail.com)

7.8 input.c File Reference

키보드 입력 발생 시 처리하는 함수 정의

```
#include "input.h"
```

Include dependency graph for input.c:



Functions

- void GetInput (void)

  외부 입력을 받아 적절한 동작을 취하도록 한다.
- void ResponseKeyUp (SDL_KeyboardEvent *event)

  키보드를 뗐을 때 상태를 기록한다.
- void ResponseKeyDown (SDL_KeyboardEvent *event)

  키보드를 눌렀을 때 상태를 기록한다.

7.8.1 Detailed Description

키보드 입력 발생 시 처리하는 함수 정의

Author

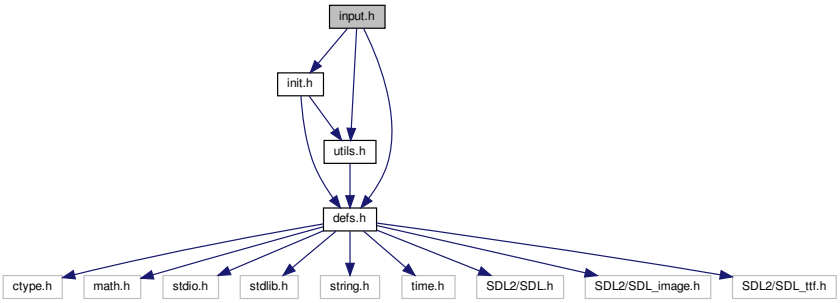
이성재 (seongjae.lee.1118@gmail.com)

7.9 input.h File Reference

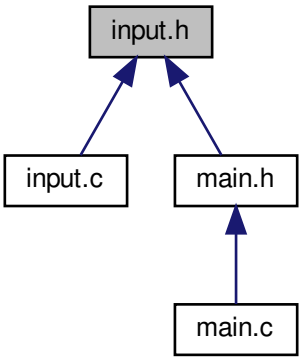
키보드 입력 발생 시 처리하는 함수 선언

```
#include "init.h"
#include "defs.h"
#include "utils.h"
```

Include dependency graph for input.h:



This graph shows which files directly or indirectly include this file:



Functions

- void GetInput (void)

  외부 입력을 받아 적절한 동작을 취하도록 한다.
- void ResponseKeyUp (SDL_KeyboardEvent *event)

  키보드를 뗐을 때 상태를 기록한다.
- void ResponseKeyDown (SDL_KeyboardEvent *event)

  키보드를 눌렀을 때 상태를 기록한다.

Variables

- App app
- Entity player
- Entity bullet [NUM_BULLETS]
- Entity game_over
- Text score_board
- char score_text [BUFSIZE]
- int score

7.9.1 Detailed Description

키보드 입력 발생 시 처리하는 함수 선언

Author

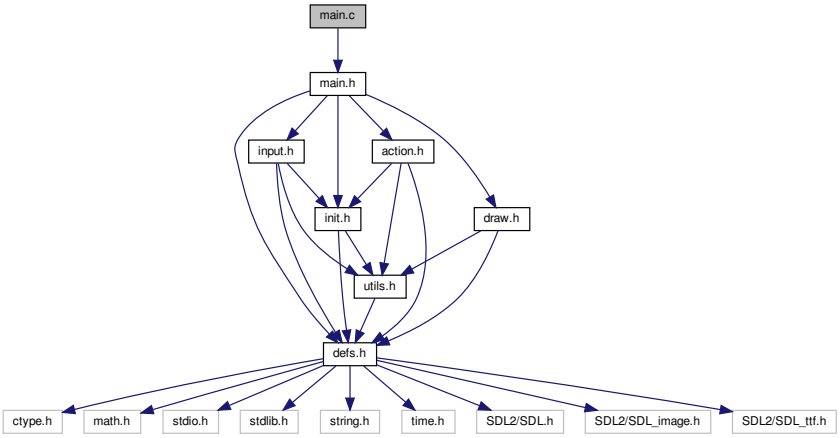
이성재 (seongjae.lee.1118@gmail.com)

7.10 main.c File Reference

dodger 게임 main 함수를 정의한 소스 파일

```
#include "main.h"
```

Include dependency graph for main.c:



Functions

- int **main** (void)

7.10.1 Detailed Description

dodger 게임 main 함수를 정의한 소스 파일

Author

이성재 (seongjae.lee.1118@gmail.com)

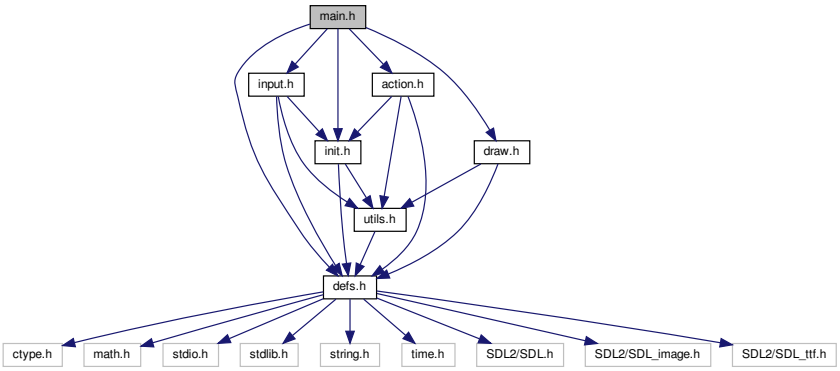
7.11 main.h File Reference

각 모듈 헤더 파일 include 및 전역 변수 선언

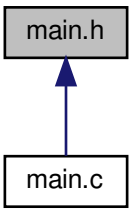
```
#include "defs.h"
#include "init.h"
#include "input.h"
#include "action.h"
```

```
#include "draw.h"
```

Include dependency graph for main.h:



This graph shows which files directly or indirectly include this file:



Variables

- App app
- Entity player
- Entity bullet [NUM_BULLETS]
- Entity game_over
- Text score_board
- char score_text [BUFSIZE]
- int score

7.11.1 Detailed Description

각 모듈 헤더 파일 include 및 전역 변수 선언

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA.

The author modified 2D Shoot'Em Up Tutorial by Parallel Realities (URL: https://www.parallelrealities.↩ co.uk/tutorials/) [21 Oct 2021]

Author

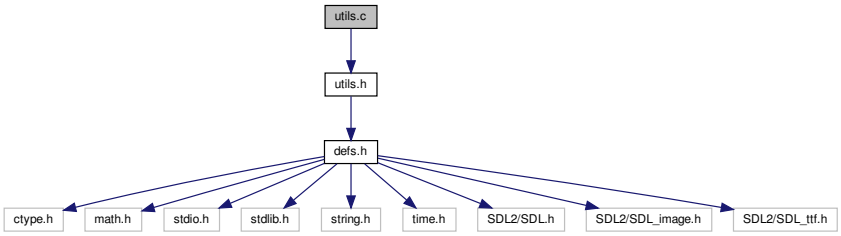
이성재 (seongjae.lee.1118@gmail.com)

7.12 utils.c File Reference

액션 수행에 필요한 부가적 계산을 수행하는 함수 정의

```
#include "utils.h"
```

Include dependency graph for utils.c:



Functions

- int CheckCollisionWall (Entity *object)
  주인공 혹은 총알이 벽 밖으로 넘어갔는지 확인
- int CheckCollisionObjects (Entity *object_a, Entity *object_b)
  두 Entity 간 충돌 여부를 판단
- void RandSpawnBullet (Entity *object)
  좌상, 좌하, 우상, 우하단 중 랜덤한 위치에 총알 소환 후 주인공과의 각도 계산
- int RandInt (int min_val, int max_val)
  [min_val, max_val) 사이의 무작위 정수를 리턴
- double RandDouble (double min_val, double max_val)
  [min_val, max_val) 사이의 무작위 실수를 리턴

7.12.1 Detailed Description

액션 수행에 필요한 부가적 계산을 수행하는 함수 정의

Author

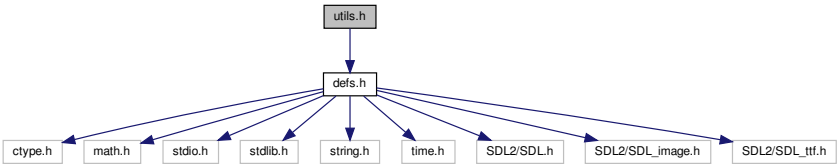
이성재 (seongjae.lee.1118@gmail.com)

7.13 utils.h File Reference

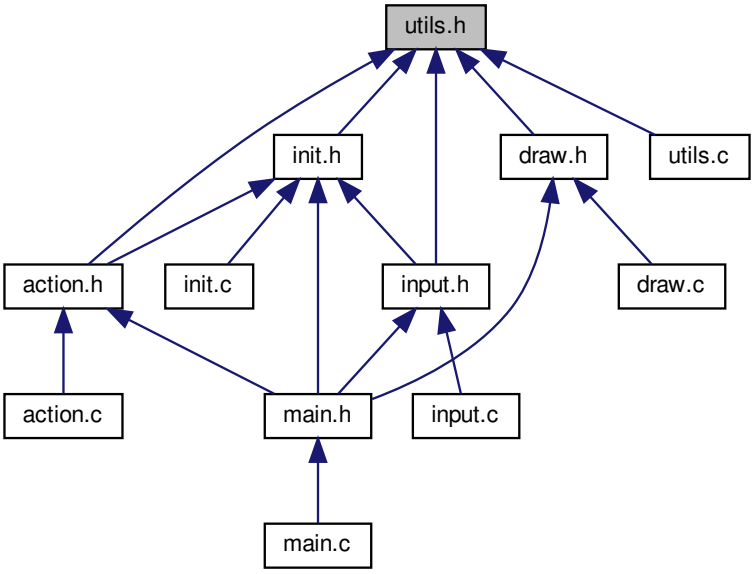
액션 수행에 필요한 부가적 계산을 수행하는 함수 선언

```
#include "defs.h"
```

Include dependency graph for utils.h:



This graph shows which files directly or indirectly include this file:



Functions

- int CheckCollisionWall (Entity *object)
  주인공 혹은 총알이 벽 밖으로 넘어갔는지 확인
- int CheckCollisionObjects (Entity *object_a, Entity *object_b)
  두 Entity 간 충돌 여부를 판단
- void RandSpawnBullet (Entity *object)
  좌상, 좌하, 우상, 우하단 중 랜덤한 위치에 총알 소환 후 주인공과의 각도 계산
- int RandInt (int min_val, int max_val)
  [min_val, max_val) 사이의 무작위 정수를 리턴
- double RandDouble (double min_val, double max_val)
  [min_val, max_val) 사이의 무작위 실수를 리턴

Variables

- App app
- Entity player
- Entity bullet [NUM_BULLETS]
- Entity game_over
- Text score_board
- char score_text [BUFSIZE]
- int score

7.13.1 Detailed Description

액션 수행에 필요한 부가적 계산을 수행하는 함수 선언

Author

이성재 (seongjae.lee.1118@gmail.com)

Index