Terminal Adventure

Generated by Doxygen 1.8.13

Contents

Chapter 1

Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Class Documentation

3.1 LevelInformation Struct Reference

LevelInformation Structure.

```
#include <level.h>
```

Public Attributes

- int level
- int hp
- int mp
- int atk
- int def
- int maxexp
- int count
- SkillInfo skills [20]

3.1.1 Detailed Description

LevelInformation Structure.

Structure to store Player Level's Information

3.1.2 Member Data Documentation

3.1.2.1 atk

```
int LevelInformation::atk
```

Attack

3.1.2.2 count

```
int LevelInformation::count
```

Skill count

3.1.2.3 def

```
int LevelInformation::def
```

Defend

3.1.2.4 hp

```
int LevelInformation::hp
```

HP

3.1.2.5 level

```
int LevelInformation::level
```

Level

3.1.2.6 maxexp

```
int LevelInformation::maxexp
```

Max Exp

3.1.2.7 mp

```
int LevelInformation::mp
```

MP

3.1.2.8 skills

```
SkillInfo LevelInformation::skills[20]
```

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

- struct/level.h

3.2 MonsterInformation Struct Reference

MonsterInformation structure.

```
#include <monster.h>
```

Public Attributes

- char name [STRING_LEN]
- int hp
- int mp
- int atk
- int def
- int exp
- Type type
- SkillInfo skills [4]

3.2.1 Detailed Description

MonsterInformation structure.

Structure to store monster's information

3.2.2 Member Data Documentation

3.2.2.1 atk

```
int MonsterInformation::atk
```

Monster attack

3.2.2.2 def

```
int MonsterInformation::def
```

Monster defend

3.2.2.3 exp

```
int MonsterInformation::exp
```

EXP ammount earned on Monster defeat

3.2.2.4 hp

```
int MonsterInformation::hp
```

Monster hp

3.2.2.5 mp

```
int MonsterInformation::mp
```

Monster mp

3.2.2.6 name

```
char MonsterInformation::name[STRING_LEN]
```

Monster name

3.2.2.7 skills

```
SkillInfo MonsterInformation::skills[4]
```

Monster skill

3.2.2.8 type

```
Type MonsterInformation::type
```

Monster Type

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

- struct/monster.h

3.3 SkillInformation Struct Reference

SkillInformation Structure.

```
#include <skill.h>
```

Public Attributes

- Skill skilltype
- char name [STRING_LEN]
- int dmg
- int mpcost
- Type type

3.3.1 Detailed Description

SkillInformation Structure.

Structure to store Skill's Information

3.3.2 Member Data Documentation

3.3.2.1 dmg

```
int SkillInformation::dmg
```

Skill damage

3.3.2.2 mpcost

```
int SkillInformation::mpcost
```

Skill MP cost

3.3.2.3 name

```
char SkillInformation::name[STRING_LEN]
```

Skill name

3.3.2.4 skilltype

```
Skill SkillInformation::skilltype
```

Skill name type

3.3.2.5 type

```
Type SkillInformation::type
```

Skill action type

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

- struct/skill.h

3.4 StageInformation Struct Reference

StageInformation structure.

```
#include <stage.h>
```

Public Attributes

- int number
- MonsterInfo monsters [10]

3.4.1 Detailed Description

StageInformation structure.

Structure to store stage's information

3.4.2 Member Data Documentation

3.4.2.1 monsters

```
MonsterInfo StageInformation::monsters[10]
```

Stage's monsters

3.4.2.2 number

```
int StageInformation::number
```

Stage number

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

- struct/stage.h

3.5 UserInfoNode Struct Reference

UserInfoNode structure.

```
#include <user.h>
```

Public Attributes

- UserInfo user
- int loggedin
- struct UserInfoNode ∗ next

3.5.1 Detailed Description

UserInfoNode structure.

Structure to store user's information as linked list

3.5.2 Member Data Documentation

3.5.2.1 loggedin

```
int UserInfoNode::loggedin
```

Login status

3.5.2.2 next

```
struct UserInfoNode* UserInfoNode::next
```

Next user

3.5.2.3 user

```
UserInfo UserInfoNode::user
```

UserInfo

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

- struct/user.h

3.6 UserInformation Struct Reference

UserInformation structure.

```
#include <user.h>
```

Public Attributes

- char username [STRING_LEN]
- char password [STRING_LEN]
- int level
- int curExp
- int curHP
- int curMP
- int stage

3.6.1 Detailed Description

UserInformation structure.

Structure to store user's information

3.6.2 Member Data Documentation

3.6.2.1 curExp

```
int UserInformation::curExp
```

Current EXP

3.6.2.2 curHP

```
int UserInformation::curHP
```

Current HP

3.6.2.3 curMP

```
int UserInformation::curMP
```

Current MP

3.6.2.4 level

```
int UserInformation::level
```

Level

3.6.2.5 password

```
char UserInformation::password[STRING_LEN]
```

Password

3.6.2.6 stage

```
int UserInformation::stage
```

Current stage

3.6.2.7 username

```
char UserInformation::username[STRING_LEN]
```

Username

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

- struct/user.h

Chapter 4

File Documentation

4.1 client.c File Reference

```
#include <sys/types.h>
#include <ctype.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/socket.h>
#include <netdb.h>
#include <arpa/inet.h>
#include <netinet/in.h>
#include "struct/skill.h"
#include "struct/monster.h"
#include "struct/level.h"
#include "struct/stage.h"
#include "struct/user.h"
#include "game/game.h"
#include "helper/mysocket.h"
#include "interface/clientfunc.h"
```

Macros

- #define BUFFER_LEN 1024

Functions

- int main (int argc, char const ∗argv[ ])

Variables

- char ch
- char temp
- UserInfo user
- struct sockaddr_in servaddr
- int sockfd

4.1.1 Macro Definition Documentation

4.1.1.1 BUFFER_LEN

```
#define BUFFER_LEN 1024
```

4.1.2 Function Documentation

4.1.2.1 main()

```
int main (
             int argc,
             char const * argv[] )
```

4.1.3 Variable Documentation

4.1.3.1 ch

```
char ch
```

4.1.3.2 servaddr

```
struct sockaddr_in servaddr
```

4.1.3.3 sockfd

```
int sockfd
```

4.1.3.4 temp

```
char temp
```

4.1.3.5 user

```
UserInfo user
```

4.2 game/game.c File Reference

```
#include <sys/types.h>
#include <ctype.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <sys/socket.h>
#include <netdb.h>
#include <arpa/inet.h>
#include <netinet/in.h>
#include "../struct/skill.h"
#include "../struct/monster.h"
#include "../struct/level.h"
#include "../struct/stage.h"
#include "../struct/user.h"
#include "game.h"
#include "../interface/clientfunc.h"
```

Functions

- void campaign ()

  PVE mode.
- int damageCalculationPlayer (SkillInfo skill, LevelInfo player, MonsterInfo monster)

  Player's damage calculation.
- int damageCalculationMonster (SkillInfo skill, LevelInfo player, MonsterInfo monster)

  Monster's damage calculation.
- SkillInfo monsterAI (MonsterInfo monster, int curHP)

  Monster's behavior.
- char gameoverChoice ()

  Game over menu.
- char stageoverChoice ()

  Game over menu.
- void printUserLog (char user[ ], char skill[ ], int dmg, Type type)

  Print User log.
- void printMonsterLog (char user[ ], char skill[ ], int dmg, Type type)

  Print Monster log.

Variables

- SkillInfo skills [ ]
- LevelInfo levels [ ]
- MonsterInfo monsters [ ]
- StageInfo stages [ ]
- int userCurHP
- int userCurMP
- int userCurLevel
- int userCurExp
- int userCurStage
- int monsterCurHP
- int monsterCurMP
- int curDmg
- int userStageHP
- int userStageMP
- SkillInfo monsterSkill
- UserInfo user
- int sockfd

4.2.1 Function Documentation

4.2.1.1 campaign()

```
void campaign ( )
```

PVE mode.

4.2.1.2 damageCalculationMonster()

```
int damageCalculationMonster (
            SkillInfo skill,
            LevelInfo player,
            MonsterInfo monster )
```

Monster's damage calculation.

Parameters

SkillInfo	Monster skill
LevelInfo	Player's level info
MonsterInfo	Monster information

4.2.1.3 damageCalculationPlayer()

```
int damageCalculationPlayer (
            SkillInfo skill,
            LevelInfo player,
            MonsterInfo monster )
```

Player's damage calculation.

Parameters

SkillInfo	Player skill
LevelInfo	Player's level info
MonsterInfo	Monster information

4.2.1.4 gameoverChoice()

```
char gameoverChoice ( )
```

Game over menu.

Returns

Player choice

4.2.1.5 monsterAI()

```
SkillInfo monsterAI (
            MonsterInfo monster,
            int curHP )
```

Monster's behavior.

Parameters

MonsterInfo	Monster information
int	Monster current HP

Returns

Monster Skill

4.2.1.6 printMonsterLog()

```
void printMonsterLog (
             char monster[ ],
             char skill[ ],
             int dmg,
             Type type )
```

Print Monster log.

Parameters

String	Monster name
String	Skill name
int	Damage
Type	Skill type

4.2.1.7 printUserLog()

```
void printUserLog (
             char user[ ],
             char skill[ ],
             int dmg,
             Type type )
```

Print User log.

Parameters

String	Username
String	Skill name
int	Damage
Type	Skill type

4.2.1.8 stageoverChoice()

```
char stageoverChoice ( )
```

Game over menu.

Returns

Player choice

4.2.2 Variable Documentation

4.2.2.1 curDmg

```
int curDmg
```

4.2.2.2 levels

```
LevelInfo levels[]
```

4.2.2.3 monsterCurHP

```
int monsterCurHP
```

4.2.2.4 monsterCurMP

```
int monsterCurMP
```

4.2.2.5 monsters

```
MonsterInfo monsters[]
```

4.2.2.6 monsterSkill

```
SkillInfo monsterSkill
```

4.2.2.7 skills

```
SkillInfo skills[]
```

4.2.2.8 sockfd

```
int sockfd
```

4.2.2.9 stages

```
StageInfo stages[]
```

4.2.2.10 user

```
UserInfo user
```

4.2.2.11 userCurExp

```
int userCurExp
```

4.2.2.12 userCurHP

```
int userCurHP
```

4.2.2.13 userCurLevel

```
int userCurLevel
```

4.2.2.14 userCurMP

```
int userCurMP
```

4.2.2.15 userCurStage

```
int userCurStage
```

4.2.2.16 userStageHP

```
int userStageHP
```

4.2.2.17 userStageMP

```
int userStageMP
```

4.3 game/game.h File Reference

```
#include "../struct/monster.h"
#include "../struct/skill.h"
#include "../struct/level.h"
```

Macros

- #define BUF_SIZE 100
- #define RED "\x1B[31m"
- #define GRN "\x1B[32m"
- #define YEL "\x1B[33m"
- #define BLU "\x1B[34m"
- #define MAG "\x1B[35m"
- #define CYN "\x1B[36m"
- #define WHT "\x1B[37m"
- #define RESET "\x1B[0m"

Functions

- int damageCalculationPlayer (SkillInfo skill, LevelInfo player, MonsterInfo monster)

  Player's damage calculation.
- int damageCalculationMonster (SkillInfo skill, LevelInfo player, MonsterInfo monster)

  Monster's damage calculation.
- SkillInfo monsterAI (MonsterInfo monster, int curHP)

  Monster's behavior.
- void campaign ()

  PVE mode.
- char gameoverChoice ()

  Game over menu.
- char stageoverChoice ()

  Game over menu.
- void printUserLog (char user[ ], char skill[ ], int dmg, Type type)

  Print User log.
- void printMonsterLog (char monster[ ], char skill[ ], int dmg, Type type)

  Print Monster log.

4.3.1 Macro Definition Documentation

4.3.1.1 BLU

```
#define BLU "\x1B[34m"
```

Print in Blue color

4.3.1.2 BUF_SIZE

```
#define BUF_SIZE 100
```

Maximum buffer size

4.3.1.3 CYN

```
#define CYN "\x1B[36m"
```

Print in Cyn color

4.3.1.4 GRN

```
#define GRN "\x1B[32m"
```

Print in Greed color

4.3.1.5 MAG

```
#define MAG "\x1B[35m"
```

Print in Mag color

4.3.1.6 RED

```
#define RED "\x1B[31m"
```

Print in Red color

4.3.1.7 RESET

```
#define RESET "\x1B[0m"
```

Print Reset

4.3.1.8 WHT

```
#define WHT "\x1B[37m"
```

Print in White color

4.3.1.9 YEL

```
#define YEL "\x1B[33m"
```

Print in Yellow color

4.3.2 Function Documentation

4.3.2.1 campaign()

```
void campaign ( )
```

PVE mode.

4.3.2.2 damageCalculationMonster()

```
int damageCalculationMonster (
             SkillInfo skill,
             LevelInfo player,
             MonsterInfo monster )
```

Monster's damage calculation.

Parameters

SkillInfo	Monster skill
LevelInfo	Player's level info
MonsterInfo	Monster information

4.3.2.3 damageCalculationPlayer()

```
int damageCalculationPlayer (
             SkillInfo skill,
             LevelInfo player,
             MonsterInfo monster )
```

Player's damage calculation.

Parameters

SkillInfo	Player skill
LevelInfo	Player's level info
MonsterInfo	Monster information

4.3.2.4 gameoverChoice()

```
char gameoverChoice ( )
```

Game over menu.

Returns

Player choice

4.3.2.5 monsterAI()

```
SkillInfo monsterAI (
            MonsterInfo monster,
            int curHP )
```

Monster's behavior.

Parameters

MonsterInfo	Monster information
int	Monster current HP

Returns

Monster Skill

4.3.2.6 printMonsterLog()

```
void printMonsterLog (
            char monster[],
            char skill[],
            int dmg,
            Type type )
```

Print Monster log.

Parameters

String	Monster name
String	Skill name
int	Damage
Type	Skill type

4.3.2.7 printUserLog()

```
void printUserLog (
            char user[],
            char skill[],
            int dmg,
            Type type )
```

Print User log.

Parameters

String	Username
String	Skill name
int	Damage
Type	Skill type

4.3.2.8 stageoverChoice()

```
char stageoverChoice ( )
```

Game over menu.

Returns

Player choice

4.4 game/gamemaster.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include "gamemaster.h"
```

Functions

- UserNode ∗ loadUserInfo ()
  Load UserInfo from file.

4.4.1 Function Documentation

4.4.1.1 loadUserInfo()

```
UserNode* loadUserInfo ( )
```

Load UserInfo from file.

Returns

A list of UserNode

4.5 game/gamemaster.h File Reference

```
#include "../struct/user.h"
```

Functions

- UserNode ∗ loadUserInfo ()
  Load UserInfo from file.

Variables

- UserNode ∗ root

4.5.1 Function Documentation

4.5.1.1 loadUserInfo()

```
UserNode* loadUserInfo ( )
```

Load UserInfo from file.

Returns

A list of UserNode

4.5.2 Variable Documentation

4.5.2.1 root

```
UserNode* root
```

4.6 helper/helper.c File Reference

```
#include "helper.h"
```

Functions

- char * userHighScoreFormat (char *username, int stage)

  Format highscore message.

4.6.1 Function Documentation

4.6.1.1 userHighScoreFormat()

```
char* userHighScoreFormat (
             char * username,
             int stage )
```

Format highscore message.

Parameters

String	username
int	user current stage

Returns

String with Format: "username : stage"

4.7 helper/helper.h File Reference

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
```

Macros

- #define MAX_STAGE_IN_STRING 4

Functions

- char ∗ userHighScoreFormat (char ∗username, int stage)
  Format highscore message.

4.7.1 Macro Definition Documentation

4.7.1.1 MAX_STAGE_IN_STRING

```
#define MAX_STAGE_IN_STRING 4
```

4.7.2 Function Documentation

4.7.2.1 userHighScoreFormat()

```
char* userHighScoreFormat (
             char * username,
             int stage )
```

Format highscore message.

Parameters

String	username
int	user current stage

Returns

String with Format: "username : stage"

4.8 helper/mysocket.c File Reference

```
#include "mysocket.h"
```

Functions

- int getPort (const char ∗port_argument)

  Get port number from argument.
- void die (char ∗msg, int type)

  Exit program when error encounter or get exit message.
- struct sockaddr_in configAddress (const char ∗ip, int port)

  Config an address with ip and port.

4.8.1 Function Documentation

4.8.1.1 configAddress()

```
struct sockaddr_in configAddress (
              const char * ip,
              int port )
```

Config an address with ip and port.

Parameters

String	ip address
int	port number

Returns

sockaddr_in

4.8.1.2 die()

```
void die (
              char * msg,
              int type )
```

Exit program when error encounter or get exit message.

Parameters

String	message
int	type

4.8.1.3 getPort()

```
int getPort (
          const char * port_argument )
```

Get port number from argument.

Parameters

String	port number

Returns

-1 on invalid
port number on valid

4.9 helper/mysocket.h File Reference

```
#include <sys/types.h>
#include <sys/socket.h>
#include <sys/wait.h>
#include <netdb.h>
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <arpa/inet.h>
#include <netinet/in.h>
#include <string.h>
#include <ctype.h>
#include <sys/select.h>
#include <sys/time.h>
#include <errno.h>
```

Functions

- int getPort (const char ∗port_argument)
  Get port number from argument.
- void die (char ∗msg, int type)
  Exit program when error encounter or get exit message.
- struct sockaddr_in configAddress (const char ∗ip, int port)
  Config an address with ip and port.

4.9.1 Function Documentation

4.9.1.1 configAddress()

```
struct sockaddr_in configAddress (
          const char * ip,
          int port )
```

Config an address with ip and port.

Parameters

String	ip address
int	port number

Returns

sockaddr_in

4.9.1.2 die()

```
void die (
            char * msg,
            int type )
```

Exit program when error encounter or get exit message.

Parameters

String	message
int	type

4.9.1.3 getPort()

```
int getPort (
            const char * port_argument )
```

Get port number from argument.

Parameters

String	port number

Returns

-1 on invalid
port number on valid

4.10 interface/clientfunc.c File Reference

```
#include "clientfunc.h"
#include "../game/game.h"
```

```
#include <string.h>
#include <stdlib.h>
```

Functions

- void login (int connfd)

    Fill login form and send to server.
- void registerFunc (int connfd)

    Fill register form and send to server.
- void fetchPlayerData (int connfd)

    Get player data from server.
- void fetchHighScore (int connfd)

    Get highscore list from server.
- void updateUserInfo (int connfd, UserInfo user)

    Request for update user current status.
- void playMenu (int connfd)

    Play menu.

Variables

- UserInfo user
- struct sockaddr_in servaddr

4.10.1 Function Documentation

4.10.1.1 fetchHighScore()

```
void fetchHighScore (
             int connfd )
```

Get highscore list from server.

Parameters

int	connection file descriptor

4.10.1.2 fetchPlayerData()

```
void fetchPlayerData (
             int connfd )
```

Get player data from server.

Parameters

int	connection file desciptor

4.10.1.3 login()

```
void login (
             int connfd )
```

Fill login form and send to server.

Parameters

int	connection file desciptor

4.10.1.4 playMenu()

```
void playMenu (
             int connfd )
```

Play menu.

Parameters

int	connection file desciptor

4.10.1.5 registerFunc()

```
void registerFunc (
             int connfd )
```

Fill register form and send to server.

Parameters

int	connection file desciptor

4.10.1.6 updateUserInfo()

```
void updateUserInfo (
          int connfd,
          UserInfo user )
```

Request for update user current status.

Parameters

int	connection file desciptor

4.10.2 Variable Documentation

4.10.2.1 servaddr

```
struct sockaddr_in servaddr
```

4.10.2.2 user

```
UserInfo user
```

4.11 interface/clientfunc.h File Reference

```
#include <sys/types.h>
#include <sys/socket.h>
#include <sys/wait.h>
#include <netdb.h>
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <arpa/inet.h>
#include <netinet/in.h>
#include <string.h>
#include <ctype.h>
#include <sys/select.h>
#include <sys/time.h>
#include <errno.h>
#include "../helper/mysocket.h"
#include "../struct/user.h"
```

Macros

- #define BUFFER_LEN 1024

Functions

- void login (int connfd)

    Fill login form and send to server.
- void registerFunc (int connfd)

    Fill register form and send to server.
- void fetchPlayerData (int connfd)

    Get player data from server.
- void playMenu (int connfd)

    Play menu.
- void updateUserInfo (int connfd, UserInfo user)

    Request for update user current status.
- void fetchHighScore (int connfd)

    Get highscore list from server.

4.11.1 Macro Definition Documentation

4.11.1.1 BUFFER_LEN

```
#define BUFFER_LEN 1024
```

Maximum of buffer lenght

4.11.2 Function Documentation

4.11.2.1 fetchHighScore()

```
void fetchHighScore (
             int connfd )
```

Get highscore list from server.

Parameters

int	connection file descriptor

4.11.2.2 fetchPlayerData()

```
void fetchPlayerData (
             int connfd )
```

Get player data from server.

Parameters

int	connection file desciptor

4.11.2.3 login()

```
void login (
             int connfd )
```

Fill login form and send to server.

Parameters

int	connection file desciptor

4.11.2.4 playMenu()

```
void playMenu (
             int connfd )
```

Play menu.

Parameters

int	connection file desciptor

4.11.2.5 registerFunc()

```
void registerFunc (
             int connfd )
```

Fill register form and send to server.

Parameters

int	connection file desciptor

4.11.2.6 updateUserInfo()

```
void updateUserInfo (
            int connfd,
            UserInfo user )
```

Request for update user current status.

Parameters

int	connection file desciptor

4.12 interface/serverfunc.c File Reference

```
#include "serverfunc.h"
#include "../helper/helper.h"
```

Functions

- int handleRequest (int connfd, UserNode ∗∗root)

  Redirect client request to right function.
- int login (int connfd, UserNode ∗root)

  User login handle.
- int sendPlayerInfo (int connfd, UserNode ∗root)

  Send player info to client.
- int sendHighScore (int connfd, UserNode ∗root)

  Send users high score to client.
- int updatePlayerInfo (int connfd, UserNode ∗∗root)

  Update user stat.
- int addUser (int connfd, UserNode ∗∗root)

  Add new user when client register.

4.12.1 Function Documentation

4.12.1.1 addUser()

```
int addUser (
            int connfd,
            UserNode ** root )
```

Add new user when client register.

Parameters

int	connection file descriptor
UserNode	pointer to user list

Returns

1 on success
0 on error

4.12.1.2 handleRequest()

```
int handleRequest (
            int connfd,
            UserNode ** root )
```

Redirect client request to right function.

Parameters

4.12.1.3 login()

```
int login (
            int connfd,
            UserNode * root )
```

User login handle.

Parameters

int	connection file description
UserNode	user list root

Returns

1 on success
0 on error

4.12.1.4 sendHighScore()

```
int sendHighScore (
             int connfd,
             UserNode * root )
```

Send users high score to client.

Parameters

int	connection file description
UserNode	user list root

Returns

1 on success
0 on error

4.12.1.5 sendPlayerInfo()

```
int sendPlayerInfo (
             int connfd,
             UserNode * root )
```

Send player info to client.

Parameters

int	connection file description
UserNode	user list root

Returns

1 on success
0 on error

4.12.1.6 updatePlayerInfo()

```
int updatePlayerInfo (
             int connfd,
             UserNode ** root )
```

Update user stat.

Parameters

int	connection file descriptor
UserNode∗	pointer to user list

Returns

1 on success
0 on error

4.13 interface/serverfunc.h File Reference

```
#include <sys/types.h>
#include <sys/socket.h>
#include <sys/wait.h>
#include <netdb.h>
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <arpa/inet.h>
#include <netinet/in.h>
#include <string.h>
#include <ctype.h>
#include <sys/select.h>
#include <sys/time.h>
#include <errno.h>
#include "../helper/mysocket.h"
#include "../struct/user.h"
```

Macros

- #define BUFFER_LEN 1024

Functions

- int handleRequest (int connfd, UserNode ∗∗root)

  Redirect client request to right function.
- int login (int connfd, UserNode ∗root)

  User login handle.
- int addUser (int connfd, UserNode ∗∗root)

  Add new user when client register.
- int sendPlayerInfo (int connfd, UserNode ∗root)

  Send player info to client.
- int updatePlayerInfo (int connfd, UserNode ∗∗root)

  Update user stat.
- int sendHighScore (int connfd, UserNode ∗root)

  Send users high score to client.

4.13.1 Macro Definition Documentation

4.13.1.1 BUFFER_LEN

```
#define BUFFER_LEN 1024
```

Maximum of buffer lenght

4.13.2 Function Documentation

4.13.2.1 addUser()

```
int addUser (
             int connfd,
             UserNode ** root )
```

Add new user when client register.

Parameters

int	connection file descriptor
UserNode	pointer to user list

Returns

1 on success
0 on error

4.13.2.2 handleRequest()

```
int handleRequest (
             int connfd,
             UserNode ** root )
```

Redirect client request to right function.

Parameters

4.13.2.3 login()

```
int login (
            int connfd,
            UserNode * root )
```

User login handle.

Parameters

int	connection file description
UserNode	user list root

Returns

1 on success
0 on error

4.13.2.4 sendHighScore()

```
int sendHighScore (
            int connfd,
            UserNode * root )
```

Send users high score to client.

Parameters

int	connection file description
UserNode	user list root

Returns

1 on success
0 on error

4.13.2.5 sendPlayerInfo()

```
int sendPlayerInfo (
            int connfd,
            UserNode * root )
```

Send player info to client.

Parameters

int	connection file description
UserNode	user list root

Returns

1 on success
0 on error

4.13.2.6 updatePlayerInfo()

```
int updatePlayerInfo (
            int connfd,
            UserNode ** root )
```

Update user stat.

Parameters

int	connection file descriptor
UserNode∗	pointer to user list

Returns

1 on success
0 on error

4.14 server.c File Reference

```
#include "helper/mysocket.h"
#include "game/gamemaster.h"
#include "interface/serverfunc.h"
```

Macros

- #define MAX_CLIENT 30

Functions

- int main (int argc, char const ∗argv[ ])

Variables

- UserNode ∗ root = NULL

4.14.1 Macro Definition Documentation

4.14.1.1 MAX_CLIENT

```
#define MAX_CLIENT 30
```

4.14.2 Function Documentation

4.14.2.1 main()

```
int main (
            int argc,
            char const * argv[] )
```

4.14.3 Variable Documentation

4.14.3.1 root

```
UserNode* root = NULL
```

4.15 struct/level.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include "level.h"
#include "skill.h"
```

Functions

- void loadLevelInfo ()

  Load LevelInfo from file.

Variables

- SkillInfo skills [ ]
- LevelInfo levels [MAX_LEVEL]

4.15.1 Function Documentation

4.15.1.1 loadLevelInfo()

```
void loadLevelInfo ( )
```

Load LevelInfo from file.

4.15.2 Variable Documentation

4.15.2.1 levels

```
LevelInfo levels[MAX_LEVEL]
```

4.15.2.2 skills

```
SkillInfo skills[]
```

4.16 struct/level.h File Reference

```
#include "skill.h"
```

Classes

- struct LevelInformation

  LevelInformation Structure.

Macros

- #define MAX_LEVEL 5

Typedefs

- typedef struct LevelInformation LevelInfo

  LevelInformation Structure.

Functions

- void loadLevelInfo ()

  Load LevelInfo from file.

4.16.1 Macro Definition Documentation

4.16.1.1 MAX_LEVEL

```
#define MAX_LEVEL 5
```

4.16.2 Typedef Documentation

4.16.2.1 LevelInfo

```
typedef struct LevelInformation LevelInfo
```

LevelInformation Structure.

Structure to store Player Level's Information

4.16.3 Function Documentation

4.16.3.1 loadLevelInfo()

```
void loadLevelInfo ( )
```

Load LevelInfo from file.

4.17 struct/monster.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include "monster.h"
#include "skill.h"
```

Functions

- void loadMonsterInfo ()

    Load Monster Info from file.

Variables

- SkillInfo skills [ ]
- MonsterInfo monsters [MAX_MONSTER]

4.17.1 Function Documentation

4.17.1.1 loadMonsterInfo()

```
void loadMonsterInfo ( )
```

Load Monster Info from file.

4.17.2 Variable Documentation

4.17.2.1 monsters

```
MonsterInfo monsters[MAX_MONSTER]
```

4.17.2.2 skills

```
SkillInfo skills[]
```

4.18 struct/monster.h File Reference

```
#include "skill.h"
```

Classes

- struct MonsterInformation

  MonsterInformation structure.

Macros

- #define MAX_MONSTER 2

Typedefs

- typedef struct MonsterInformation MonsterInfo

  MonsterInformation structure.

Functions

- void loadMonsterInfo ()

  Load Monster Info from file.

4.18.1 Macro Definition Documentation

4.18.1.1 MAX_MONSTER

```
#define MAX_MONSTER 2
```

Maximum number of Monster

4.18.2 Typedef Documentation

4.18.2.1 MonsterInfo

```
typedef struct MonsterInformation MonsterInfo
```

MonsterInformation structure.

Structure to store monster's information

4.18.3 Function Documentation

4.18.3.1 loadMonsterInfo()

```
void loadMonsterInfo ( )
```

Load Monster Info from file.

4.19 struct/skill.c File Reference

```
#include "skill.h"
```

Variables

- SkillInfo skills [SKILL_COUNT]

4.19.1 Variable Documentation

4.19.1.1 skills

```
SkillInfo skills[SKILL_COUNT]
```

Initial value:

```
= {
    {S_EMBER, "Ember", 20, 5, T_FIRE},
    {S_VINEWHIP, "Vine Whip", 20, 5, T_GRASS},
    {S_WATERGUN, "Water Gun", 20, 5, T_WATER},
    {S_HEADBUTT, "Headbutt", 30, 7, T_NORMAL},
    {S_SLASH, "Slash", 35, 8, T_NORMAL},
    {S_FLAMEWHEEL, "Flame Wheel", 30, 10, T_FIRE},
    {S_MAGICALLEAF, "Magical Leaf", 30, 10, T_GRASS},
    {S_WATERPULSE, "Water Pulse", 30, 10, T_WATER},
    {S_FLAMETHROWER, "FlameThrower", 35, 15, T_FIRE},
    {S_LEAFBLADE, "Leaf Blade", 35, 15, T_GRASS},
    {S_SURF, "Surf", 35, 15, T_WATER},
    {S_MEGAPUNCH, "Mega Punch", 40, 15, T_NORMAL},
    {S_FIREBLAST, "Fire Blast", 40, 20, T_FIRE},
    {S_SOLARBEAM, "Solar Beam", 40, 20, T_GRASS},
    {S_HYDROPUMP, "Hydro Pump", 40, 20, T_WATER},
    {S_MEGAKICK, "Mega Kick", 45, 20, T_NORMAL},
    {S_BLASTBURN, "Blast Burn", 45, 25, T_FIRE},
    {S_FRENZYPLANT, "Frenzy Plant", 45, 25, T_GRASS},
    {S_HYDROCANNON, "Hydro Cannon", 45, 25, T_WATER},
    {S_HYPERBEAM, "Hyper Beam", 45, 25, T_NORMAL},
    {ES_TACKLE, "Tackle", 10, 0, T_NORMAL},
    {ES_ACID, "Acid", 15, 0, T_NORMAL},
    {ES_SLUDGEBOMB, "Sludge Bomb", 20, 0, T_NORMAL},
    {ES_GUNKSHOT, "Gunk Shot", 25, 0, T_NORMAL},
    {ES_SCRATCH, "Scratch", 10, 0, T_NORMAL},
    {ES_THRUST, "Thrust", 15, 0, T_NORMAL},
    {ES_POTION, "Potion", 5, 0, T_HEAL},
    {ES_GOBLINPUNCH, "Goblin Punch", 25, 0, T_NORMAL},
    {S_NORMALATTACK, "Normal Attack", 10, 0, T_NORMAL}}
```

4.20 struct/skill.h File Reference

Classes

- struct SkillInformation

  SkillInformation Structure.

Macros

- #define STRING_LEN 20
- #define SKILL_COUNT 29

Typedefs

- typedef struct SkillInformation SkillInfo

  SkillInformation Structure.

Enumerations

-

enum Skill {
S_EMBER, S_VINEWHIP, S_WATERGUN, S_HEADBUTT,
S_SLASH, S_FLAMEWHEEL, S_MAGICALLEAF, S_WATERPULSE,
S_FLAMETHROWER, S_LEAFBLADE, S_SURF, S_MEGAPUNCH,
S_FIREBLAST, S_SOLARBEAM, S_HYDROPUMP, S_MEGAKICK,
S_BLASTBURN, S_FRENZYPLANT, S_HYDROCANNON, S_HYPERBEAM,
ES_TACKLE, ES_ACID, ES_SLUDGEBOMB, ES_GUNKSHOT,
ES_SCRATCH, ES_THRUST, ES_POTION, ES_GOBLINPUNCH,
S_NORMALATTACK }

Skill enum.

- enum Type {
T_FIRE, T_GRASS, T_WATER, T_NORMAL,
T_HEAL }

  Type enum.

4.20.1 Macro Definition Documentation

4.20.1.1 SKILL_COUNT

```
#define SKILL_COUNT 29
```

Maximum number of skill

4.20.1.2 STRING_LEN

```
#define STRING_LEN 20
```

Maximum number of character

4.20.2 Typedef Documentation

4.20.2.1 SkillInfo

```
typedef struct SkillInformation SkillInfo
```

SkillInformation Structure.

Structure to store Skill's Information

4.20.3 Enumeration Type Documentation

4.20.3.1 Skill

```
enum Skill
```

Skill enum.

List of skill

Enumerator

S_EMBER	
S_VINEWHIP	
S_WATERGUN	
S_HEADBUTT	
S_SLASH	
S_FLAMEWHEEL	
S_MAGICALLEAF	
S_WATERPULSE	
S_FLAMETHROWER	
S_LEAFBLADE	
S_SURF	
S_MEGAPUNCH	
S_FIREBLAST	
S_SOLARBEAM	
S_HYDROPUMP	
S_MEGAKICK	
S_BLASTBURN	
S_FRENZYPLANT	
S_HYDROCANNON	
S_HYPERBEAM	
ES_TACKLE	
ES_ACID	
ES_SLUDGEBOMB	
ES_GUNKSHOT	
ES_SCRATCH	
ES_THRUST	
ES_POTION	
ES_GOBLINPUNCH	
S_NORMALATTACK	

4.20.3.2 Type

```
enum Type
```

Type enum.

List of Type

Enumerator

T_FIRE	
T_GRASS	
T_WATER	
T_NORMAL	
T_HEAL	

4.21 struct/stage.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include "stage.h"
#include "monster.h"
```

Functions

- void loadStageInfo ()

  Load Stage Info from file.

Variables

- MonsterInfo monsters [ ]
- StageInfo stages [MAX_STAGE]

4.21.1 Function Documentation

4.21.1.1 loadStageInfo()

```
void loadStageInfo ( )
```

Load Stage Info from file.

4.21.2 Variable Documentation

4.21.2.1 monsters

```
MonsterInfo monsters[ ]
```

4.21.2.2 stages

```
StageInfo stages[MAX_STAGE]
```

4.22 struct/stage.h File Reference

```
#include "monster.h"
```

Classes

- struct StageInformation

  StageInformation structure.

Macros

- #define MAX_STAGE 2

Typedefs

- typedef struct StageInformation StageInfo

  StageInformation structure.

Functions

- void loadStageInfo ()

  Load Stage Info from file.

4.22.1 Macro Definition Documentation

4.22.1.1 MAX_STAGE

```
#define MAX_STAGE 2
```

Maximum number of stage

4.22.2 Typedef Documentation

4.22.2.1 StageInfo

```
typedef struct StageInformation StageInfo
```

StageInformation structure.

Structure to store stage's information

4.22.3 Function Documentation

4.22.3.1 loadStageInfo()

```
void loadStageInfo ( )
```

Load Stage Info from file.

4.23 struct/user.c File Reference

```
#include <stdlib.h>
#include <string.h>
#include "user.h"
```

Functions

- UserNode ∗ findUser (UserNode ∗root, char ∗username)

  Find an User from UserNode list.
- UserNode ∗ makeNewNode (UserInfo data)

  Create new UserNode with UserInfo.
- UserNode ∗ insertNode (UserNode ∗root, UserNode ∗new)

  Insert a node to UserNode list.
- void freeList (UserNode ∗root)

  Free an UserNode list.
- void printList (UserNode ∗root)

  Print an UserNode list.
- UserNode ∗ sortUserlist (UserNode ∗root)

  Sort a node to UserNode list.
- UserInfo initUserInfo (char ∗username, char ∗password)

  Initial an UserInformation from username and password.
- void writeUserData (UserNode ∗root)

  Write an UserNode list to file.

4.23.1 Function Documentation

4.23.1.1 findUser()

```
UserNode* findUser (
            UserNode * root,
            char * username )
```

Find an User from UserNode list.

Parameters

UserNode	Root of an UserNode list
String	Username to find

Returns

An UserNode on found
NULL on not found

4.23.1.2 freeList()

```
void freeList (
            UserNode * root )
```

Free an UserNode list.

Parameters

UserNode	Root of an UserNode list

4.23.1.3 initUserInfo()

```
UserInfo initUserInfo (
            char * username,
            char * password )
```

Initial an UserInformation from username and password.

Parameters

String	username
String	password

Returns

new UserInfo

4.23.1.4 insertNode()

```
UserNode* insertNode (
            UserNode * root,
            UserNode * new )
```

Insert a node to UserNode list.

Parameters

UserNode	Root of an UserNode list
UserNode	Node to insert

Returns

Root of an UserNode list with inserted node

4.23.1.5 makeNewNode()

```
UserNode* makeNewNode (
             UserInfo data )
```

Create new UserNode with UserInfo.

Parameters

UserInfo	Data for UserNode

Returns

A node created with provided data

4.23.1.6 printList()

```
void printList (
             UserNode * root )
```

Print an UserNode list.

Parameters

UserNode	Root of an UserNode list

4.23.1.7 sortUserlist()

```
UserNode* sortUserlist (
             UserNode * root )
```

Sort a node to UserNode list.

Parameters

UserNode	Root of an UserNode list

Returns

Root of an UserNode list with sorted node

4.23.1.8 writeUserData()

```
void writeUserData (
             UserNode * root )
```

Write an UserNode list to file.

Parameters

UserNode	Root of an UserNode list

4.24 struct/user.h File Reference

File containing User structure and usage.

```
#include <stdio.h>
```

Classes

- struct UserInformation

    UserInformation structure.
- struct UserInfoNode

    UserInfoNode structure.

Macros

- #define STRING_LEN 20

Typedefs

- typedef struct UserInformation UserInfo

    UserInformation structure.
- typedef struct UserInfoNode UserNode

    UserInfoNode structure.

Functions

- UserNode ∗ makeNewNode (UserInfo data)

  Create new UserNode with UserInfo.
- UserNode ∗ insertNode (UserNode ∗root, UserNode ∗new)

  Insert a node to UserNode list.
- void freeList (UserNode ∗root)

  Free an UserNode list.
- void printList (UserNode ∗root)

  Print an UserNode list.
- UserNode ∗ findUser (UserNode ∗root, char ∗username)

  Find an User from UserNode list.
- UserNode ∗ sortUserlist (UserNode ∗root)

  Sort a node to UserNode list.
- UserInfo initUserInfo (char ∗username, char ∗password)

  Initial an UserInformation from username and password.
- void writeUserData (UserNode ∗root)

  Write an UserNode list to file.

4.24.1 Detailed Description

File containing User structure and usage.

4.24.2 Macro Definition Documentation

4.24.2.1 STRING_LEN

```
#define STRING_LEN 20
```

Maximum number of character

4.24.3 Typedef Documentation

4.24.3.1 UserInfo

```
typedef struct UserInformation UserInfo
```

UserInformation structure.

Structure to store user's information

4.24.3.2 UserNode

```
typedef struct UserInfoNode UserNode
```

UserInfoNode structure.

Structure to store user's information as linked list

4.24.4 Function Documentation

4.24.4.1 findUser()

```
UserNode* findUser (
            UserNode * root,
            char * username )
```

Find an User from UserNode list.

Parameters

UserNode	Root of an UserNode list
String	Username to find

Returns

An UserNode on found
NULL on not found

4.24.4.2 freeList()

```
void freeList (
            UserNode * root )
```

Free an UserNode list.

Parameters

UserNode	Root of an UserNode list

4.24.4.3 initUserInfo()

```
UserInfo initUserInfo (
            char * username,
            char * password )
```

Initial an UserInformation from username and password.

Parameters

String	username
String	password

Returns

new UserInfo

4.24.4.4 insertNode()

```
UserNode* insertNode (
            UserNode * root,
            UserNode * new )
```

Insert a node to UserNode list.

Parameters

UserNode	Root of an UserNode list
UserNode	Node to insert

Returns

Root of an UserNode list with inserted node

4.24.4.5 makeNewNode()

```
UserNode* makeNewNode (
            UserInfo data )
```

Create new UserNode with UserInfo.

Parameters

UserInfo	Data for UserNode

Returns

A node created with provided data

4.24.4.6 printList()

```
void printList (
             UserNode * root )
```

Print an UserNode list.

Parameters

UserNode	Root of an UserNode list

4.24.4.7 sortUserlist()

```
UserNode* sortUserlist (
             UserNode * root )
```

Sort a node to UserNode list.

Parameters

UserNode	Root of an UserNode list

Returns

Root of an UserNode list with sorted node

4.24.4.8 writeUserData()

```
void writeUserData (
             UserNode * root )
```

Write an UserNode list to file.

Parameters

UserNode	Root of an UserNode list

Index