

RFC R-TYPE

Status of this Memo

This memo provide information for EPITECH APE about R-TYPE Protocol between Client and R-TYPE Server. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (2018). All Rights Reserved.

Table of Contents

1.	Introduction.	2
2.	Server.	2
2.1	UDP.	2
2.2.	Packets	2
3.	Client.	3
4.	Message Details.	3
5.	Command client-server connection.	3
5.1	Establishing a client-server connection.	4
5.2.	Disconnect from the server.	4
6.	Command start and end game.	4
6.1.	Launch the game.	4
6.2.	Start the game.	5
6.3.	End the game.	5
7.	Command game information.	5
7.1	Send the players or enemies position.	5
7.2	Send the projectiles position.	5
7.3	Send the collide with player or enemies.	6
7.4	Send the events of the player.	6
8.	Author's Address.	6
9.	Full Copyright Statement.	7

1. Introduction

This document is intended for people working on implementing an Server Client Communication for R-TYPE Project.

2. Server

Server MUST send all the information about the game to client by different packets with in each packet an unsigned short "cmd".

2.1 UDP

The data transfers between client and server MUST be done by UDP (User Datagram Protocol).

Packets can be lost. So, if a command between server and client failed server don't send any error message.

The client MUST send multiple commands (For example: CONNECT) till he get the SUCCESS message.

2.2. Packets

ObjectPacket {

For each item:

unsigned int	id
float	X
float	Y
bool	hit
bool	animated
enum	ALIVE, DEAD
unsigned char	animation_id

}

ScorePacket {

For each player:

enum State	INGAME, WON, LOST
struct	player_score[4]
{	
char	playerId
unsigned int	score
}	

}

```
LobbyPacket {  
  
    bool                gameStarted  
    unsigned short      seed  
    unsigned char       numberOfPlayers  
}  
  
OnInput {  
  
    short              X_velocity  
    short              Y_velocity  
    bool               Release_shot  
    bool               Charge_shot  
}
```

3. Clients

For each client, all servers MUST have the following information first: a seed to get the lobby you want by using the CONNECT command. The client MUST use the CONNECT command first.

4. Message Details

The server to which a client is connected is required to parse the complete message. A fatal error may follow from incorrect command, a destination which is otherwise unknown to the server, not enough parameters or incorrect privileges.

If a full set of parameters is presented, then each MUST be checked for validity and appropriate responses sent back to the client.

Example:

COMMAND parameter

5. Command client-server connection

All the command server and client MUST use to establish a success connection.

5.1. Establishing a client-server connection.

The client MUST use the CONNECT command to establish a client-server connection:

Command: CONNECT

Parameters: <packet> (LobbyPacket)

Default IP address is set to the local IP address (127.0.0.1)

When the CONNECT command is send by the client, the server responds CONNECTED if the connection success.

The server MUST NOT use this command.

5.2. Disconnect from the server

The client MUST use the DISCONNECT command to exit the connection between himself and the server:

Command: DISCONNECT

Parameters: NONE

When the DISCONNECT command is send by the client, the server respond DISCONNECTED if the disconnection success.

6. Command start and end game

All the command the client and server MUST use to start or end the game properly.

6.1. Launch the game

To launch the game the client MUST use the READY command:

Command: READY

Parameter: NONE

If the command success the server respond by the STARTGAME Command else the server return E_READY.

6.2. Start the game

To start the game the server MUST use the STARTGAME command with all the sprite in parameter:

Command: STARTGAME

Parameter: NONE

The sprite will be send to the client.

6.3. End the game

To end the game the server MUST use the ENDGAME command:

Command: ENDGAME

Parameters: NONE

7. Command Game Information

All the command the server and the client MUST use to send the informations about the game like players or enemies position, events etc...

7.1. Send the players or enemies position

To send the players position the server MUST use the PLAYERPOSITION command:

Command: PLAYERPOSITION

Parameter: <params> (ObjectPacket)

Like the players position to send the enemies position the server MUST send the same parameter. Only the command name change, the server MUST use ENEMIESPOSITION command.

7.2 Send the projectiles position

To send the projectiles position the server MUST use the PROJECTILESPOSITION command:

Command: PROJECTILESPOSITION

parameter: <params> (ObjectPacket)

7.3 Send the collide with players or enemies

To send the collide position of the player the server MUST use the HITPLAYER command:

Command: HITPLAYER

Parameter: <params> (ObjectPacket)

Like the player collision to send the enemies position the server MUST send the same parameter. Only the command name change, the server MUST use HITENEMIES command.

7.4 Send the events of the player

To send the events of the player the client MUST use the EVENTS command:

Command: EVENTS

Parameter: <events> (OnInput)

8. Author's Address

EPITECH R-TYPE team
85 rue du Jardin Public
33000 BORDEAUX
FRANCE

EMail: clement.peau@epitech.eu
paul.wery@epitech.eu
alexandre2.nguyen@epitech.eu
luka.villeuneuve@epitech.eu
pierre.buis@epitech.eu
romain.samuel@epitech.eu

9. Full Copyright Statement

Copyright (C) The Internet Society (2018). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.