

# Asteroids Protocol Design

## Frames

Each frame starts with the \* character, followed by an integer indicating its type, followed by the payload.

## Frame Types

Note: S to C means the server will send the frame to *all* clients

### Login (C to S, S to C)

Type = 1

Payload = String with the username

When a client logs on, they send the server their player name. The server then sends all clients the name of the current players

### Countdown (S to C)

Type = 2

Payload = int with number of seconds until the game starts

### Start (S to C)

Type = 3

Payload = 2 ints width and height of the game in pixels

### AsteroidLife (S to C)

Type = 4

Payload = int asteroid ID, int with pixel diameter, followed by two ints  $x$  and  $y$  representing the starting coordinates of the asteroid, followed by two ints  $\Delta x$  and  $\Delta y$  representing the speed of the asteroid in pixels/second

### AsteroidDeath (S to C)

Type = 5

Payload = int with asteroid ID that died, String of player who killed it, int of pts awarded

### ShipDestroyed (S to C)

Type = 6

Payload = String with the player name whose ship got destroyed

### PlayerUpdate (S to C, C to S)

Type = 7

Payload = String with player name, followed by two ints  $x$  and  $y$  representing the position of the player, followed by two ints  $\Delta x$  and  $\Delta y$  representing the speed of the player in pixels/second, followed by an int for the angle  $\theta$  of the ship

**Shoot** (C to S)

Type = 8

Payload = String with player name

**Bullet** (S to C)

Type = 9

Payload = coordinates of bullet, String with player name who shot it

**Quit** (C to S, S to C)

Type = 10

Payload = String with player name

**Chat** (C to S, S to C)

Type = 11

Payload = String with player name, followed by a String with the message

**Update** (S to C)

Type = 12

Payload = double with number of milliseconds since the game began

**GameFinished** (S to C)

Type = 13

Payload = String with name of winner

We will be using `DataInputStreams` and `DataOutputStreams`

Server starts and waits for incoming connections

When a client logs on sends a **Login** frame. Server sends back a **Login** frame with all logged in players names.

After first client logs in the server will begin a countdown to the start of the game and send a **Countdown** frame to the client. All other clients who join will also get **Countdown** frames at that time.

When the countdown reaches 0, all clients will receive a **Start** frame and several **AsteroidLife** frames to populate the game.

The server will send an initial **PlayerUpdate** frame for each logged in player to provide the client with an initial position for each player, and then server and the client will send **PlayerUpdate** frames back and forth for the duration of each player's life, so that both are aware of all players positions.

After the game started the server will also be constantly sending **PlayerUpdate** frames to all clients to assure they are all operating properly.

If a ship ends up hitting an asteroid, the server sends a **ShipDestroyed** frame to all clients.

If a player shoots then a **Shoot** frame is sent from the client to the server. The bullet's direction is determined by the orientation of the ship and its speed will be preset, so only the name of the player who shot the bullet is necessary.

The bullet then repeatedly reports its destination to every client with a **Bullet** frame.

When a bullet hits an asteroid an **AsteroidDeath** frame is sent to all clients.

If a player quits, he informs the server with a **Quit** frame and then the server informs all remaining clients with **Quit** frames.

If a player sends a chat message, he informs the server with a **Chat** frame and then the server informs all clients with **Chat** frames.

When all asteroids or every player is destroyed the server sends out a **GameFinished** frame.