

cs244b Review Session 1

Mazewar

4/3/2009

Agenda

- Administration / Overview
- Mazewar Outline and Hints
- Finish a bit early to allow time to form groups

Administration

- Class meets Tuesday and Thursday from 12:50-2:05 in Gates B1
- Review sessions as needed in Gates B01 on Fridays from 3:15 - 4:05 pm
- Prerequisites:
 - Knowledge of C, and C++, and basic programming methodology as developed in cs106b or cs106x, and networking and sockets knowledge at the level of CS144 or CS244A and ideally, CS249.

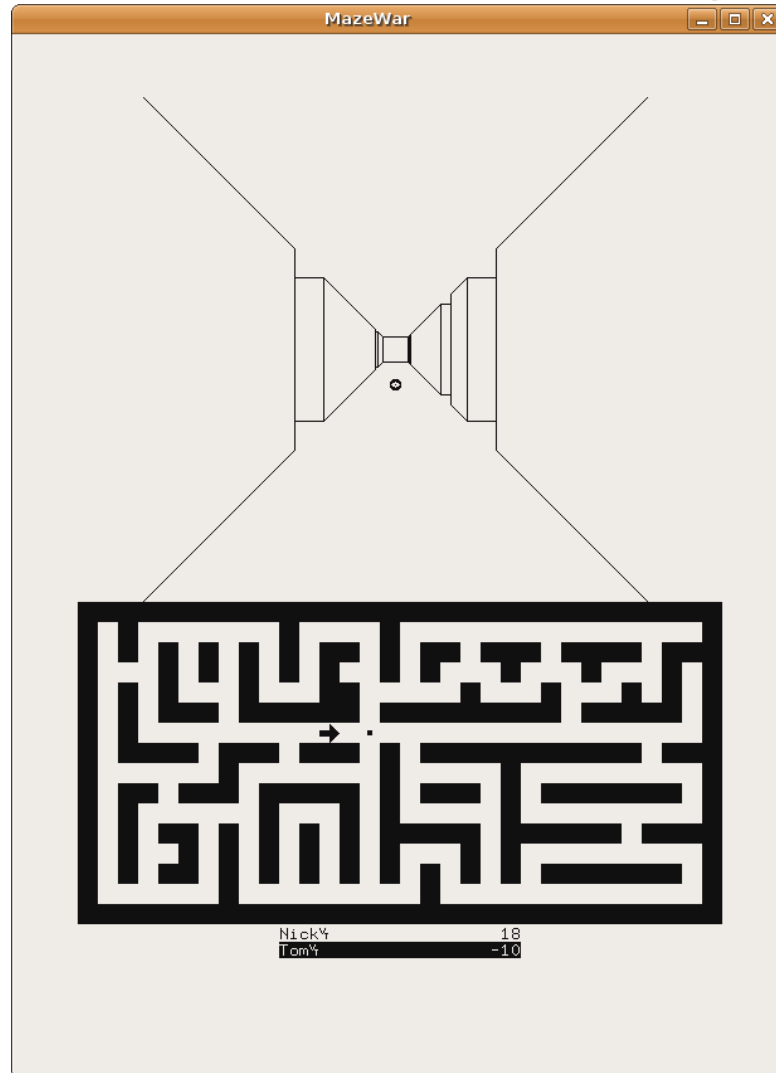
Administration

- Use newsgroup at su.class.cs244b for questions about the homework
 - Connecting directions found at:
 - http://www.stanford.edu/services/ess/pc/docs/oe/oe_news.html
- Correspondence with staff: cs244b-spr0809-staff@lists.stanford.edu

Administration

- Two (Possibly three) programming assignments (~40%)
 - Mazewar
 - Distributed File System
 - Develop and test on 32 bit pod computers (pod.stanford.edu).
- Midterm (~15%)
- Final Exam (~45%)

Mazewar Image



Mazewar Overview

- Mazewar is a distributed multiplayer game.
- Form Groups (By 4/7)
- Develop protocol
- Implement protocol (Due: By 4/16)
- Demo results (Code Due: 4/30)
- SCPD students will demo remotely.

Game Overview

- You control a rat
- Gain points by tagging other rats
- Lose points when tagged
- Lose a point when you shoot

More Specific Rules

- Each instantiation of the game is a separate player
- Each player controls one rat
- Player sees only what his rat sees and his location in the maze
- Players can move and rotate about the maze

More Rules

- After firing the rat needs time to recover before firing again
- Rat is tagged instantly and reappears at a random spot
- Rat loses 5 points for being tagged, gains 11 for tagging another rat, and loses 1 for shooting a projectile.

More Rules

- Only one rat can occupy a square at once
- Multiple projectiles can occupy the same square

What You are Given

- Implementation of the graphics for the game.
- Event loop (void play())
 - Receives Packets
 - Keyboard events

What You Need to Edit

- `netInit()`
 - May need to add some more code (possibly not)
- Most of your work will be in editing `play()`
 - Main event loop
 - Receive packets and update local state
 - Change local state and send packets

Protocol Design

- Team creates protocol, members implement **individually**
 - At most three team members
- Must be distributed
 - Can't have central point of failure
- Over UDP
- Must deal with lost packets in a reasonable way

Protocol Assumptions

- Players know the UDP multicast group for the game
- No cheating
- One standard maze
 - Maze name field most likely unused
- One game per UDP multicast group

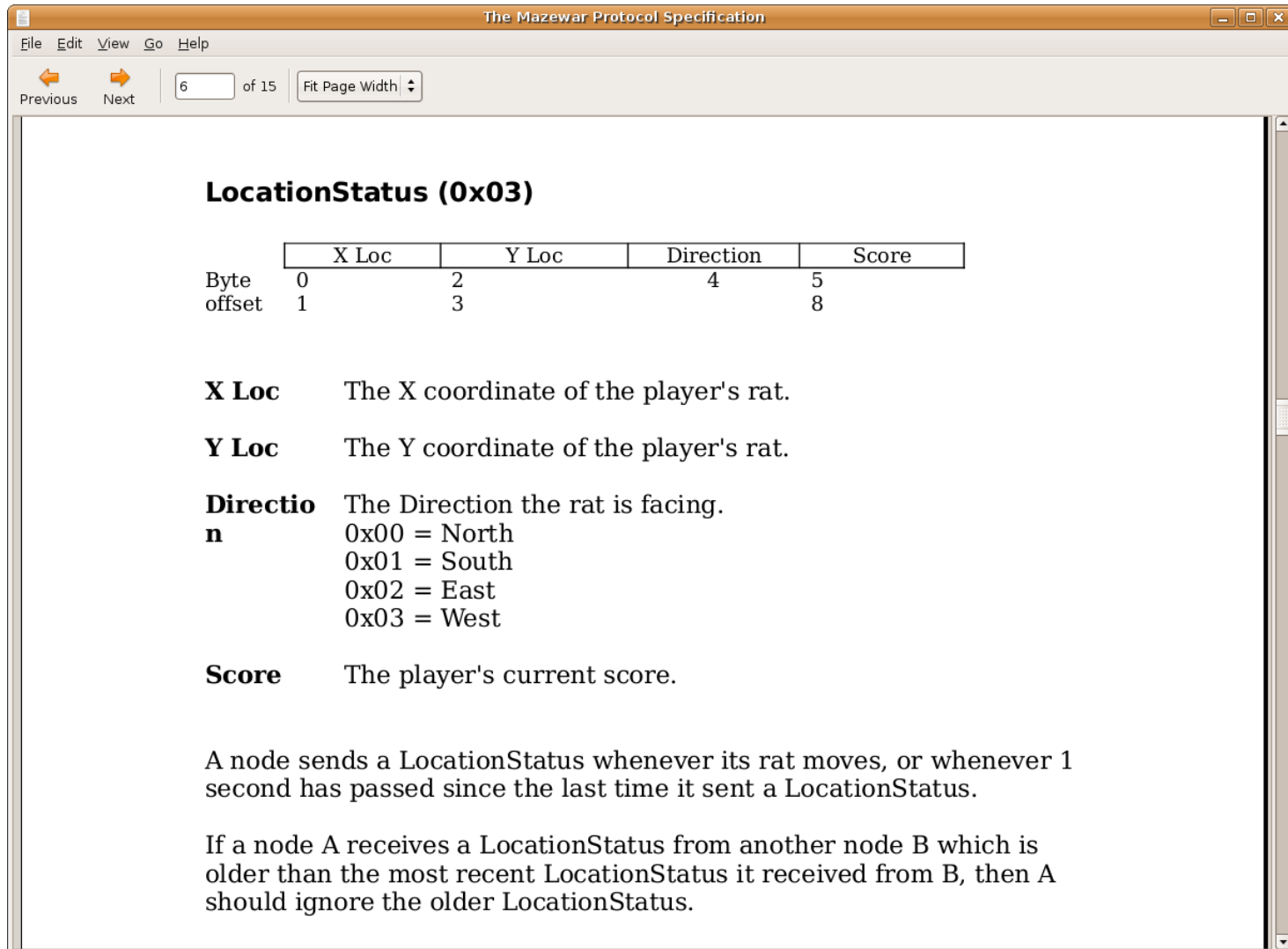
Protocol Hints

- Many cases to handle, below are some cases to think about
 - Common situation: there is the request and then the answer packet type
- Way to receive player names and scores when joining a game

Protocol Hints

- Current or new location
 - Multiple players at same spot
 - Timing functions useful here
- Missile Information
 - Movement
- Hit Information
- Exiting game

Protocol Example



The screenshot shows a web browser window with the title "The Mazewar Protocol Specification". The browser's address bar shows "6 of 15" and "Fit Page Width". The page content includes a table and several definitions.

LocationStatus (0x03)

	X Loc	Y Loc	Direction	Score
Byte	0	2	4	5
offset	1	3		8

X Loc The X coordinate of the player's rat.

Y Loc The Y coordinate of the player's rat.

Direction The Direction the rat is facing.
0x00 = North
0x01 = South
0x02 = East
0x03 = West

Score The player's current score.

A node sends a LocationStatus whenever its rat moves, or whenever 1 second has passed since the last time it sent a LocationStatus.

If a node A receives a LocationStatus from another node B which is older than the most recent LocationStatus it received from B, then A should ignore the older LocationStatus.

cs249 Style Functions

- No requirements to use cs249 functions or style.
- Initial code uses some:
 - Smart Pointers
 - reference counting
 - Nominal / Ordinal types

Useful Functions - Timing

- struct timeval
 - gettimeofday(&timeval, NULL);
- Network byte order:
 - uint32_t htonl(uint32_t *hostlong*);
 - uint16_t htons(uint16_t *hostshort*);
 - uint32_t ntohl(uint32_t *netlong*);
 - uint16_t ntohs(uint16_t *netshort*);

Questions?