

**CS 244B**

**Review Session I**

**Mazewar 4/1/2011**

# Agenda

- Admin / Overview
- Mazewar
  - Demo
  - Rules
  - Hints
- Form groups and start early!

# Administration

- Two (possibly three) programming assignments (~40%)
  - Mazewar
  - Replicated File System
- Develop and test on myth computers (myth.stanford.edu)
- Midterm (~15%)
- Final Exam (~45%)

# Discussion Forum: Piazza

- This is where assignment questions will be answered
- URL: <http://www.piazza.com>
- Most of you should have been enrolled already
- Otherwise, sign up today!
- For private questions, contact us directly

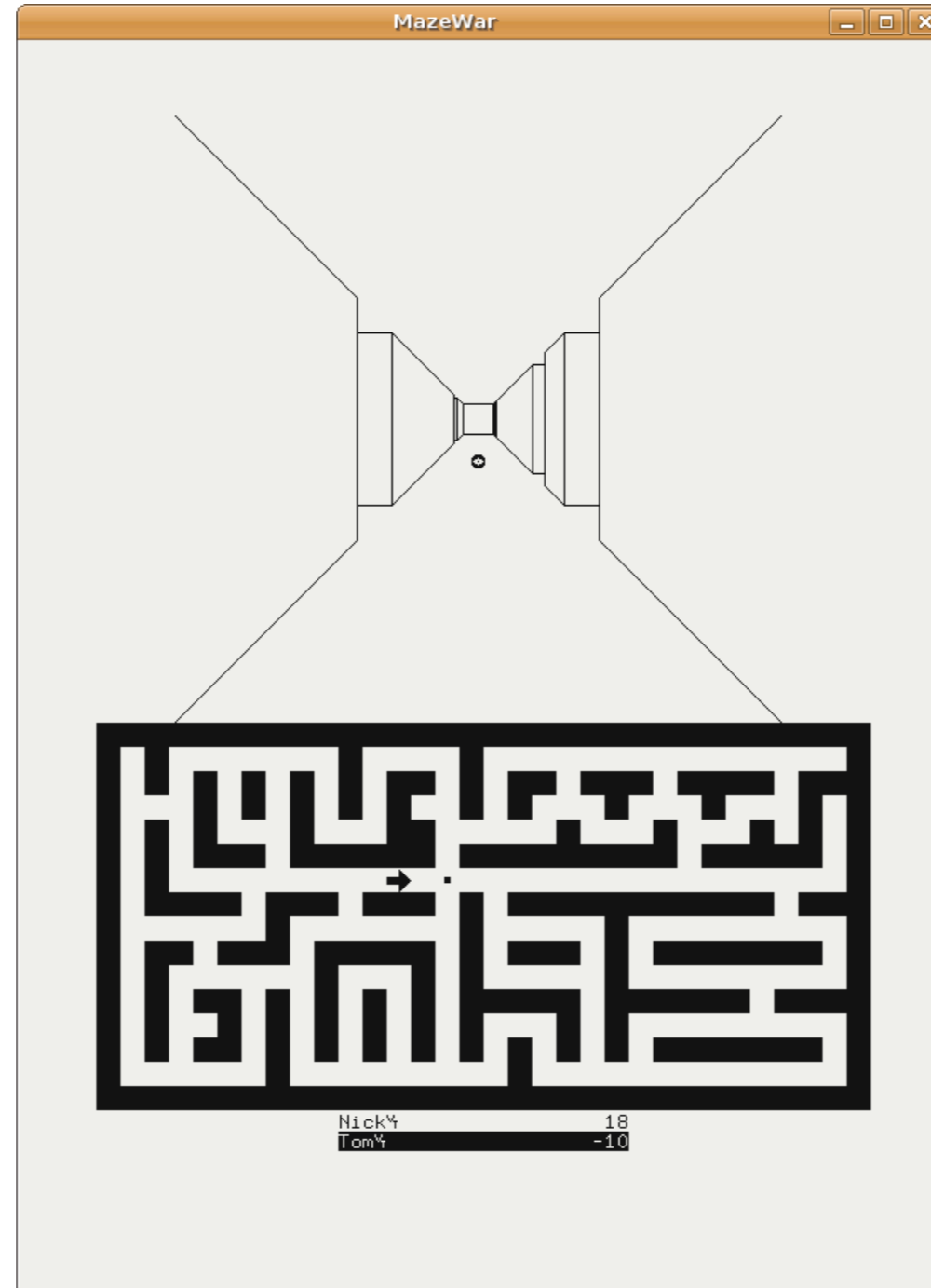
# Mazewar Deadlines

- Tue 4/5: Teams due
  - 2-3 members per team
  - Send us a “team name” and list of members with emails
- Thu 4/14: Protocol specification due
- Thu 4/28: Project due. Demos will be around this time (tentatively May 2nd)

# How to Submit

- Teams and Protocol Specification
  - Email to both “tazim at cs” and “rumble at cs”
  - SCPD students should also cc SCPD ([scpd-distribution@lists.stanford.edu](mailto:scpd-distribution@lists.stanford.edu))
- Project
  - Submit using the script:  
`/afs/ir/class/cs244b/bin/submit`

# Mazewar Demo



# Game Overview

- You control one rat: fwd, back, left, right
- Scoring:
  - Shoot projectile: -1
  - Hit opponent: +11
  - Opponent hits you: -5
- Player sees perspective view plus location in the maze
- Consult the assignment page for specifics:
  - <http://www.stanford.edu/class/cs244b/mazewar.html>

# More Rules

- Each instantiation of the game is a separate player
- After firing, the rat needs time to recover before firing again
- Rat is tagged instantly and reappears at a random spot
- 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
  - Functions in `display.cpp` are useful
- Event loop (`void play()`)
  - Receives Packets (`EVENT_NETWORK`)
  - Keyboard Events
  - Timer Events (`EVENT_TIMEOUT`)

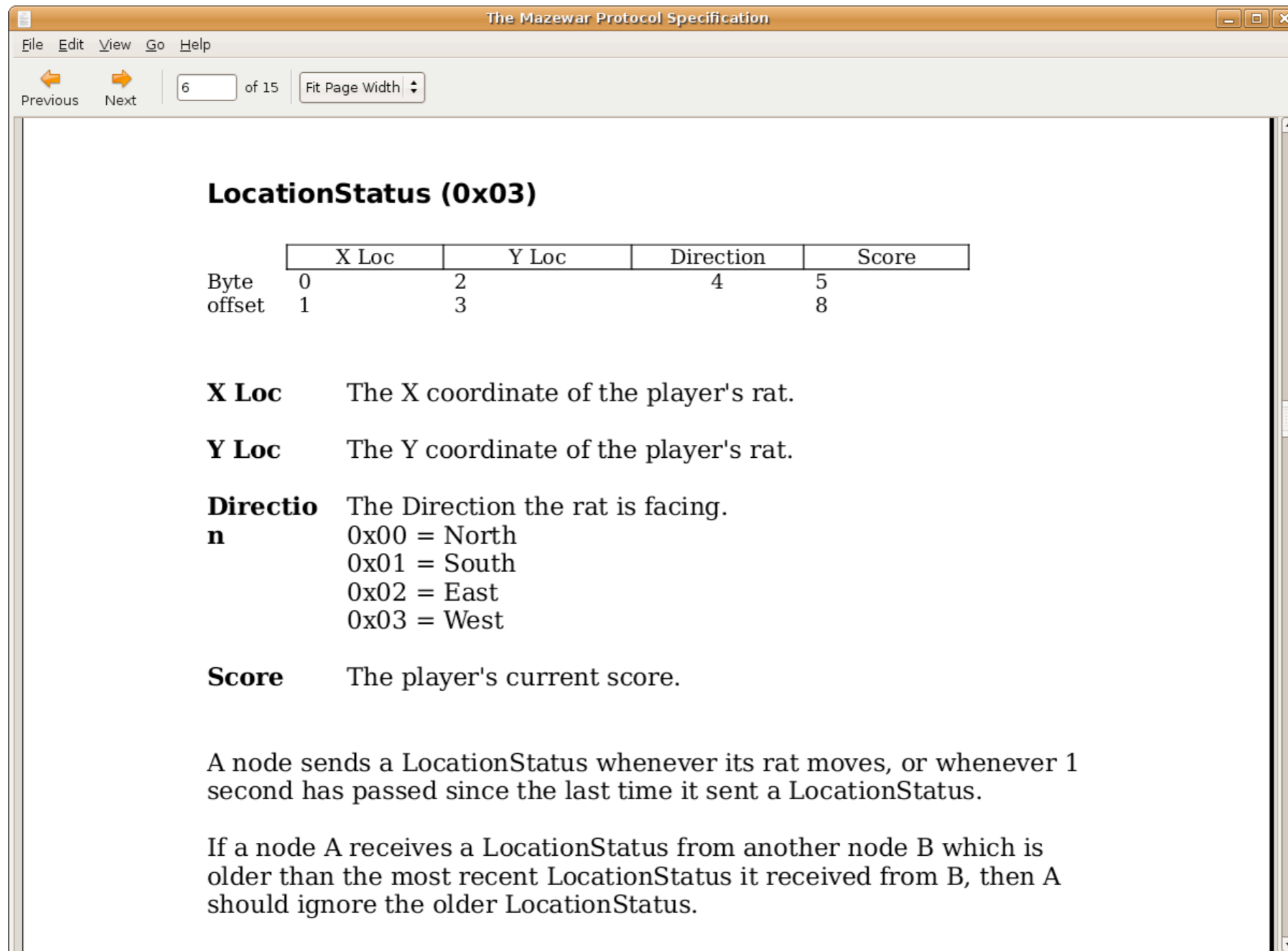
# What You Need to Edit

- `netInit()`
- Most of your work will be in editing `play()`
  - Main event loop
  - Receive packets and update local state
  - Change local state and send packets
- Stub networking code in place

# Protocol Design

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

# Protocol Example



The screenshot shows a document viewer window titled "The Mazewar Protocol Specification". The window has a menu bar with "File", "Edit", "View", "Go", and "Help". Below the menu bar are navigation buttons for "Previous" and "Next", a page number "6 of 15", and a "Fit Page Width" button. The main content area displays the following information:

### 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.

# Protocol Assumptions

- Players know the UDP multicast group for the game
  - Multicast group identified by MAZEPORT in mazewar.h
- No cheating
- One standard maze (M->maze\_)
- One game per UDP multicast group

# Protocol Hints

- Many cases to handle, below are some cases to think about
- Sometimes need both request and response packet types
- Need a way to receive player names and scores when joining a game

# Protocol Hints (contd.)

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

# Useful Functions

- `struct timeval`
  - `gettimeofday(&timeval, NULL);`
- Network byte order:
  - `htonl, htons, ntohs` etc.
- In the framework:
  - `clearSquare, ClearRatPosition, showMissile, sendPacketToPlayer` etc.

# Questions