
CS 456, Spring 2009

Assignment #9

Planet Project

Due 4/22/09, in class

Introduction

Planet Project — A multi-player cooperative planet simulation based on the program `planets` (<http://planets.homedns.org>). The game is built from three parts: a web client, a server, and a status reporting system.

Definitions

- (1) The **goal** of the game is to create a stable orbit for a collection of planets. The more planets the higher the score.
- (2) The **universe** is two dimensional and includes a collection planets all affecting each other through gravitational attraction.
- (3) A **planet** has a location, a speed, a direction, and a mass. The size of the rendered planet reflects its mass.
- (4) Each **player** is expected to control a single planet through a single client.

The Web Client

The client shows two things to the player: the current universe and details of the the player's planet (*e.g.*, its image (color), mass, speed, direction, and location). Players can update these attributes in real time. These updates are sent to the server where they impact the universe.

The Server

The server maintains a collection of current players and their planets. It periodically sends each client details regarding the other planets in the universe.

Status and Scoring

The server maintains the current status of all planets and the current score. It can report this information to interested web surfers as an XML document. This provides a mechanism for the reporting of scores and planets' status to an external reader. A simple approach would be to display the data nicely using an XML style sheet.

Smarts

At one extreme the server does all the computation and simply sends the clients what to display. At the other extreme each client is essentially running the simulation locally. It only communicates with the server to exchange updates regarding planet details (*e.g.*, when one of the players makes a change).

What is the impact of smart's placement on resources (*e.g.*, processor utilization, bandwidth, etc.)? How about the impact on the synchronization between players?

What to hand in (on or before 4/22/09 unless noted below)

- (1) A test plan of not more than two pages (**due on or before 4/15/09**).
- (2) A **well formatted** 2-up printout of your source code (client and server).
- (3) An annotated copy of your test plan including screen shots showing the results of each test.
- (4) A log of working session each include a 1 line description and duration.
- (5) Email me the url for your server and a tar ball named `<your-account-name>-<your-partners-account-name>.tar.gz` that includes your source (all of it including support files), your log and test plan, the annotated test plan, and a sequence of screen shots (names prefixed so that they are listed in the same order as they were generated) showing off how well your program works.

Notes

- (1) Work in different pairs.

Extra credit (admittedly not worth the points, but sometimes life is about more than points)

- (1) [1] Add random events (something like Simcity disasters) (*e.g.*, a super nova?).
- (2) [4] Go 3d. For example, represent the dimension that goes “into” the screen using blue and red shifts.
- (3) [2] Turn on a trail of the each planet’s positions, which fades into the background over time.
- (4) [3] Add a slider to zoom in an out.
- (5) [2] Use cookies to remember and recall the user’s planet between browser sessions.
- (6) [4] Between updates from the server move the planets using a sequence of linear *micro steps* to provide a smother animation.
- (7) [1] Each player gets a different planet color.

To get credit, each of the above much be highlighted in the source and tests and annotated with the numbn-ers found above.