EE433 Final Project

In an attempt to further our education and expand into non-trivial software projects, we are developing a multiplayer command-economy and military low-fantasy Real-Time Strategy-esque videogame.

Description

The project will be a real-time strategy game that features a platform agnostic view for portanility. The game is set in a low-fantasy medieval realm with different feature maps to have different gameplay mechanics. Our focus will be a competitive arena like gameplay to see which player can win utilizing a combination of both economy and military. Each player may choose different gameplay routes via a different strategy or reacting to an enemy player's attack or expansion. Unlike most real time strategy games, this game will feature each unit being trainable to multiple tasks within a village. A villager can either be trained for mining, woodcutting, farming, a soldier, defender, or other such task -- each unit has their own skill tree and can take on multiple roles over the course of a play session.

Like individual units, players can use resources to build different buildings to improve the economy or train soldiers. Each building will also have its own health associated so that attacking players can destroy key buildings to stunt growth. A player can win a game by destroying all of the enemy players buildings or completeing some other objective. The game will have a client-server model where players can theoretically play worldwide. Each player will have their own account and have a rank associated. The ranking system will make sure that new players are not queued with veteran players given options. Veteran players will have a challenge to reach the top of the ranking system making this game have potential replay value.

Purpose

Entertainment and Education (for us). While other projects such as scenario-designer frameworks with unit control via coding are enticing, as junior developer yuppies we need a more linear game design model first to learn the game designing and implementation procedure. As such, the game is nigh solely a vain entertainment, although as with other games, the author expression and decision-making that players use to excel at the game lends itself to practice, multi-variable analysis, and dedicated problem-solving.

Major Features

Gameplay Features

- Control and manage multiple units to do various tasks.
- Gather resources from the world around you.
- Process materials into other materials.
- Craft various tools and weapons to enhance your units.
- Construct buildings with which units can interact.
- Food to upkeep units, and implicitly the large space needed to make food, as pseudo-limit to unit growth.

Technical Features

- Connect to one or more other players based on usernames.

    - Re-connect in lost sessions should both player attempt to reset; periodic save states.
- Save states that can be passed and loaded by another copy of the game.
- Updator that allows easy updating to the next version without reinstalling.
- Menu and GUI to access and navigate customization options of the game.
    - Options Menu
    - Find Game
    - Exit Game
    - Load Game
    - Profile
- Profile creation and management.
- System-agnostic Controller and Model (distinguished View) -- relatively trivial porting.
- Client-Server Connection Model
    - When lacking a server, the player with the stronger specs (as judged by network connectivity and multi-processing capability) hosts both a client and server -- the server is automatically managed and relatively inaccessible by the user except in turning off and on unless a dedicated customizable playsession is activated.
    - One computer can potentially host a server as well as multiple clients, assumming the necessary resources and input devices.
    - Multiple clients can connect to a single server, thus expanding player-count is relatively trivial.
    - Latency-guarding mechanics to ease higher latency delays and connection loss.

Stakeholders

- EE433 Instructor (Myers)
    - List of software requirements
    - Seeking to test knowleedge of non-trivial software design and interaction of several components.
    - Desires a well-planned project and project review and presentation (via at least a show-board).
- EE433 Students/Developers (Reece, Alex, Elijah)
    - Seeking to learn game designing concepts.
    - Learn advanced software interfaces and facets.
    - Create a well-organized and enjoyable project and product.
- Gamers-RTS Fans
    - Desire multiple gameplay options and choices
    - Player authorship and expression through playstyle
    - Game knowledge and interaction dependent skill.

Languages

- C# - Major game component interactions, object-oriented.
- C++ (maybe)
- C (maybe) - Low-level processes that must be done fast and efficiently.
- Batch Scripting - Opening, saving, loading game state files.

Frameworks

- Unity 3D
- Logging System
- DB-OO connection Module (maybe)

- .NET
- Windows Development Kit