CLIMBING THE CITADEL
Many try. Few succeed. Begin the ascent on December 16th, 2007!

Game Specification

Written by Chuck Bolin
November 2007
Game Challenge 4 Submission
www.gameinstitute.com

Revised November 21, 2007

Table of Contents

Purpose and Game Overview

This document specifies requirements for a video game to be submitted for Game Challenge 4. The contest start is November 16, 2007 and the deadline for submissions is December 16, 2007.

Contest Theme: Winter

Game Title: Climbing the Citadel

Game Genre: 2D Side scroller

Game Concept: The Citadel (fictional) is the most deadliest and one of the tallest mountains in the Swiss Alps. The objective is for the player to climb to the top of the Citadel and hoist a flag.

Game Levels: ~~There are 5 game levels. The first level begins at 1,000 feet and ascends to 5,000 feet. Successful completion of each level advances the player's elevation by 4,000 feet. Therefore the peak is located at 21,000 feet. (1000 + (5 x 4,000)~~. ***(See Update 2)***

NOTE: The game should come across and very serious. The game level 1 will start off in a serious manner. However pure 'silliness' breaks out and the game because fun with less emphasis on realism and seriousness.

Technology

Graphics: 2D sprites using DirectX 9. All art is contained in 512 x 512 pixel graphic files. File format is .PNG using alpha channel for transparency. (***See Update 1 for exceptions***)

Sound: WAV or MP3 files using FMOD as the sound library.

Software: Visual Studio.net 2003, C++

Tools: Written using Visual Basic 6.0

Game Data: CSV data for game level and configuration information.

Game Play Phases

Phase I: Introductory screen. Artwork is displayed conveying cold and treacherous mountain terrain. Music conveys feeling of magnitude and expresses the near impossibility of reaching the Citadel. This phase is 'eye candy' and excites the player to start climbing. The music should loop no less than 30 seconds.

Phase II: Main menu. Player can configure the mouse and keyboard for his or her preferences. The player may review previous accomplishments in highest elevation hall of fame. They may exit the program or continue to play. The music should connect the introduction to game play. The music should loop no less than 30 seconds.

Phase III: Game play. The player begins at level 1 or 1000 feet. This level is relatively easy. The player is allowed 120 seconds to complete the level. A 5 to 10 second victory celebration is permitted at the completion of levels 1, 2, 3 and 4. A musical interlude should be celebratory. Completing level 5 wins the player the game. The player may stay there as long as necessary. The victory music should loop no less than 30 seconds.

Each successive level lasts 120 seconds and allows the player to climb to 21,000 feet.

Phase IV: Closing credits. The music should be grand and with the effects of drawing the player back to the game. The music should loop no less than 30 seconds.

Saving Games

A criticism of past games is that the player is not allowed to resume a game at a later time at the last level completed. Consequently players would not finish the game. The player's name and last level successfully completed is saved to a file “players.dat”. The game data is saved at the completion of a level.

Graphic Files

All art is saved as PNG files with the alpha channel for transparency. Data pertaining to each file is saved in “***graphicfiles.dat***”. The following data is stored.

File ID - Unsigned integer beginning with '0'
Filename with extension.

Sprites

All images used in the game are called sprites. All sprites are identified by a unique number in the “***sprites.dat***” file. The following data is stored.

Sprite ID – Unsigned integer beginning with '0'
File ID – corresponds to number in “graphicfiles.dat”
x,y – top-left corner of sprite location in the .png file
width, height – width and height of the sprite within the graphics file
rotX, rotY – this is the center of rotation of the sprite in the event it can be rotated
scale – a float representing the sprite scale. '1.0' is the standard scale size.
alpha – the degree of transparency is controlled by this value (0 – 255)
angle – the intial angle of rotation about rotX and rotY
name – sprite's name is for reference only
layer – a number corresponding to the layer or depth (see Sprite Layers below)

Sprite Layers (See Update 1 for Changes)

~~To facilitate depth and perspective sprites are drawn on layers. Layer 1 is farthest away and will be a distant moutain range. Its' vertical position drops with each level...creating the illusion the player is climbing higher in altitude. The color of the mountain blends into the cold blue sky.~~

~~Layer 2 is closer and used to display mountain pines or clouds.~~

~~Layers 3, 4 and 5 are used for game play. The player begins on layer 4. The player may move among the three layers as necessary to avoid obstacles and attacks.~~

~~Layers 6 is for foreground objects such as small snow drifts. The player cannot play on layer 6.~~

~~Layer 7 is for the UI (user inteface).~~

Game Level Data (See Update 1)

~~All data is stored in a single file “gamelevel.dat”. The file format is TBD (to be determined). This data will identify the sprites that should be displayed and their respective level. The level also indentifies those things that work to prevent the player form succeeding.~~

Players Animated Sprite

The player will be of stick man form. Key frames will be used to indicate major body changes for particular behaviors. The program will interpolate the frames in between the key frames.

Physics Engine

Physics will apply gravity, coefficient of friction for various surfaces to allow user to slip, slide and fall. Collision detection and reaction will be determined by the physics engine. The physics engine will be created by the team.

Game Engine

The frame work created by the author of the specification and used in previous games will serve as the base engine. The engine has been used in “Skunked”, “Rommel's Fantasy Revenge” and “Jed, Ted and Fred”.

Sound Effects

Some effects will transition on and off suddenly such as explosions. Other sounds will vary volume in relation to the distance between the player and the object generating the sound.

Game Details

Each level requires fun and unusual things to happen. The player can power up using a variety of possible items such as snow shoes, skiis, jet pack, pole used for pole vaulting, poggo stick, etc. Weapons consists of throwing snow balls, exploding snow over bad guys, and other such items.

A more comprehensive list will be determined later.

Supplemental Updates

Update 1 (November 18, 2007)

Graphics – All files sizes are 512 x 512 with the following exceptions: Splash screen and sky backgrounds for the game play.

Tile and Sprite Sizes – It has been determined that numerous small sprites consume more framerates than a few larger frame rates. Every effort should be made to avoid numerous small sprites.

Sprite Layers

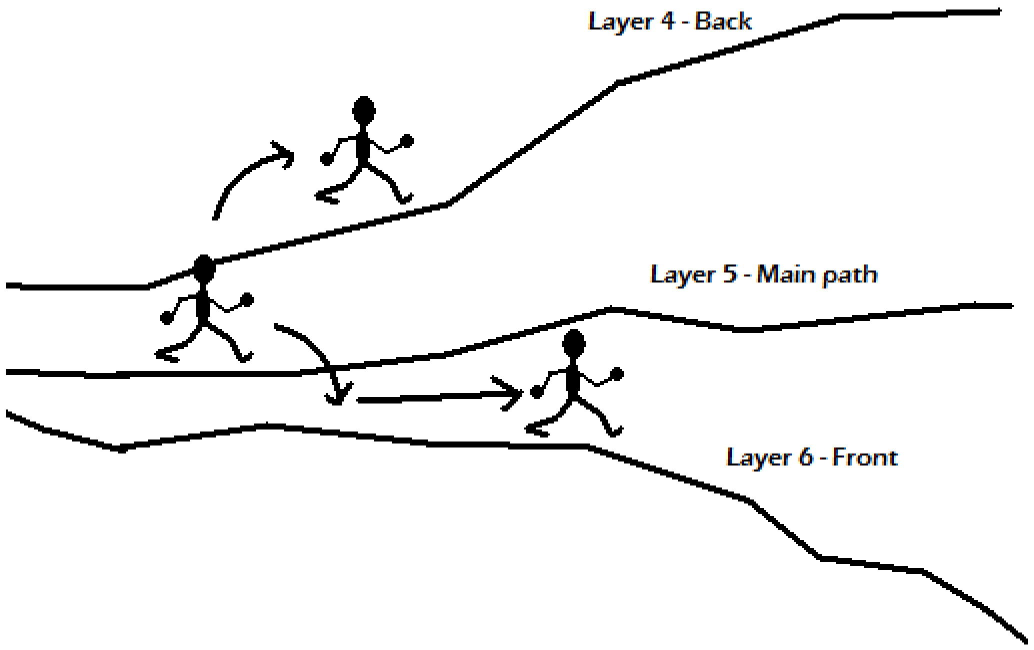
Layers 0 and 1 are fixed to the screen and do not move. This includes items such as sky images, clouds and distant moutains.

Layers 2, 3 are closer but moves slower than closer layers.

Layers 4, 5 and 6 are player layers. The player sprite can jump and play on these three levels.

Layers 7 are foreground items. The player moves behind these. This may be trees, snow, etc.

Layer 8 is the user interface. It is fixed to the window.



Player Sprite Layers

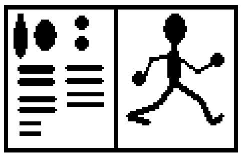
The above diagram indicates three levels used by the player. The player moves his sprite left and right and is able to transition to the back layer (4) to the front layer (6).

The scale of the player will change accordingly.

Level	**Scale**
Layer 4	0.9
Layer 5	1.0
Layer 6	1.1

The scale will be interpolated between levels to show a smooth scale transition.

Player Sprite



The player sprite will be constructed from several body parts such as the head, trunk, thigh, leg, arm, lower arm, hands and feet. A class will be created to manage the animation.

Game Level Data

The data is saved in a CSV file. Each item in the game world contains the following data.

```
Name, x, y, scale, angle, SpriteID
```

The following data represents the example.

```
sky1, 0, 0, 1.5, 0, 13
snow1, 1476, 2500, 1, 0, 9
snow2, 1476, 2618, 1, 0, 10
tree1, 2500, 2530, 0.9, 0, 11
tree2, 1900, 2690, .4, 0, 12
```

Update 2 (November 19, 2007)

Scale of Distance and Pixels: 10 pixels corresponds to 1 foot

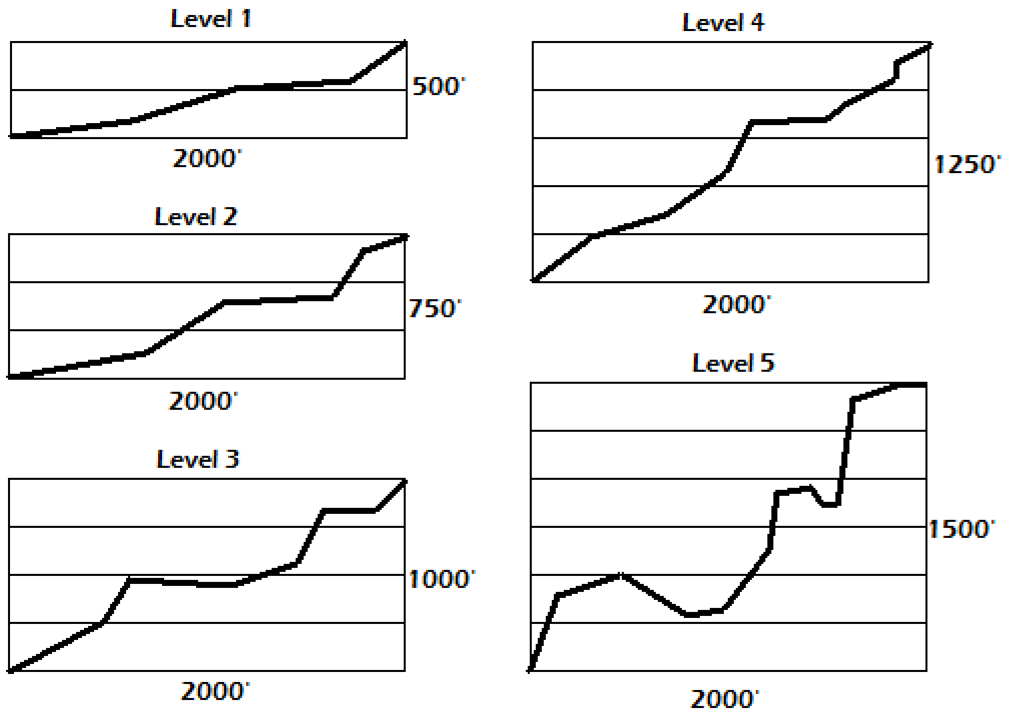
Game Levels: There are 5 game levels. The first level begins at 16,000 feet and ascends to 21,000 feet. Each successive level presents the player to climb higher in elevation.

Level	Elevation Change
1	500'
2	750'
3	1000'
4	1250'
5	1500'

<u>Update 3 (November 19, 2007)</u>

Game Level Profiles

The solid lines are only representative of a suggestive profile. The overall slope is correct.



<u>Update 4 (November 19, 2007)</u>

Beginning and End of Levels: The player steps out of his tent on each level(automatic). Player takes control of sprite movement. At the end of each level is another tent. The level is complete when player enters the final tent. Level 5 is the exception. The player plants or raises a flag.

<u>Update 5 (November 21, 2007)</u>

Sprites

The sprites.dat file will be modified to add an additional parameter to indicate a framecount. This allows sprites.dat to indicate single sprites and animated sprites. A framecount = 0 indicates a single sprite. A value greater than 0 indicates

Graphic File Size

An exception is made to the 512x512 file size. Files containing animated sprite frames will be as wide as necessary to allow for all frames for the animation. An effort should be made to keep the sprite and files sizes as small as possible.

Update 6 (November 26, 2007)

Sprites

Another parameter is appended to the sprites.dat file. The parameter is time interval between animated frames.

Player Sprite

The player sprite will be able to move on 3 layers: 4, 5 and 6. Layer 6 is in the front and 4 is in the back. The player sprite moves on layers. If the current level is behind another level the sprite will disappear from the view.

Game Editor

The editor will read sprites.dat file and existing game level files. It will be used to edit and create game levels.

Slopes and Player Sprite

The player's sprite can display 3 possible profiles: walking, sliding and climbing.

Sliding: The downward slope is greater than 128x80 down.
Climbing: The upward slope is greater than 128x96 up.
Walking: Slope between the two values above.

Game Challenge

As each game level advances, the overall climb increases from 500' over 2000' to 1500' over 2000'. Obstacles will roll, bounce and slide down the hill towards the player. As the leve increases, the downward speed of the obstacles will increase. The slope should be used to affect bouncing of sprites.

Configuration File

A config.ini file is read at program startup. Following is a sample config file.

```
//Screen mode 1 = windowed, 0 = fullscreen
WindowedMode = 1

//-1 is maximum screen width, 1024
ScreenWidth = 1024
```

```
//-1 is maximum screen height, 768
ScreenHeight = 768

//starting game level
GameLevel = 5

//show snow in game 1 = yes, 0 = no
GameSnow = 0

//enable game audio 1 = yes, 0 = no
GameAudio = 0

//display framerate on each screen
FrameRate = 1
```