**Null Commandos**

A 3-D side scrolling shooter

Created by:

Michael Rojas

Geoff Smith

Ronnie Hout

Dylan Miller

Ross Wang

Table of Contents

1. **Game Overview**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3
	1. ***Introduction\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***3
	2. ***Features\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***3
2. **Game Play**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3
	1. ***Mechanics\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***3
	2. ***Inventory\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***3
	3. ***Characters\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***4
3. **Player Controls** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_4
	1. ***Player 1\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***4
	2. ***Player 2\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***5
4. **Arhcitecture**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_5
	1. ***Input Framework\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***5
	2. ***State Framework\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***6
5. **Level Framework**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_8
	1. ***World Elements\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***8
	2. ***Spatial Elements \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***8
	3. ***Dynamic Elements \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***8
6. **Credits**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_8
	1. ***Creators \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***8
	2. ***Sounds\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***8
	3. ***Music\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***8
	4. ***Models\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***9
	5. ***3rd Party Tools \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***9

***1.0) Overview***

# 1.1) Introduction: What is Null Commandos?

Null Commandos will be a 3-D side scrolling shooter game created in XNA by this group. The design of the game is modeled after classic side-scrolling shooters such as Contra or Metal Slug, with some added improvements.

# 1.2) Features

Our game will take the basic idea of side-scrolling shooters of old and add to these ideas some new approaches:

1. A 3-Dimensional environment allowing for a richer game experience
2. An inventory system to allow different weapons to be used at different times (no more being forced to use a weapon when picked up initially)
3. A Level Editor tool to allow creation of new and unique stages
4. A Developer’s Console for stage and game manipulation, mainly for debugging and testing, but sometimes just for fun

# *2.0) Game Play*

**2.1) Mechanics:**

The gameplay will be simple: get to the end of the level and defeat boss without dying a set number of times, and destroying all enemies in your way to the end of the level. The player will be able to run, shoot, and jump, along with being able to choose between an inventory of weapons, allowing for a unique strategy to the game. The game is set up in a ‘wave’ format, which means each level is broken down into a number of stages (or waves), in which the player must defeat a set number of enemies before moving on the next wave. The player is given 30 lives and 2 extra credits (each worth 10 lives) at the outset of the game, and each time the player is hit by a bullet or enemy, the player loses a life.

**2.2) Inventory:**

There are 4 available guns for use by the player: Hand Gun, Machine Gun, Shotgun, And the Sniper Rifle. Each gun is available from the start, but only the Hand Gun ill have any ammo; the other 3 guns will all have empty ammo stashes. Ammo for the other guns can be picked up by killing certain enemies, which will add the ammo to the proper gun’s inventory, and thus allow the player to now use the gun. The Machine Gun, Shotgun, and Sniper Rifle all have an ammo capacity of 99 bullets, while the Hand Gun has an infinite amount of ammo. Guns can be switched by using the X button on the Xbox controller, or the 1 through 4 keys on the keyboard. The weapon inventory is as such:

Hand Gun (default gun): single shot, low damage, low fire rate
Machine Gun: 3 bullet burst shot, low/mid damage, high fire rate
Shotgun: 2 bullets per shot parallel to each other, mid damage, mid fire
rate
Sniper Rifle: single shot, high damage, low fire rate, piercing
effect

**2.3) Characters:**

Main Character (Player 1): The Nameless Hero

Main Character (Player 2): The Other Nameless Hero

Enemies:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Enemy Name | Regular Grunt | Grunt Commander | Enemy vehicles | Final Boss |
| Actions | Fast enemies that will swarm and run straight at you, fearlessly | Armed soldiers who will keep their distance while shooting at you | Armored Mech walkers that will assault you from long range | A heavily armored enemy Tank accompanied by mechs and foot soldiers |
| Health | 2 | 3 | 5 | 9 |

# *3.0) Player Control*

**3.1) Player 1**

Keyboard:

1. Run – Left/ Right arrow Keys
2. Jump – Up arrow, L/R Ctrl
3. Shoot – Space
4. Switch Weapons – 1,2,3,4 keys
5. Pause – Escape, Delete

Xbox:

1. Run – Left Thumbstick/D-Pad
2. Jump – A
3. Shoot – Right Trigger
4. Switch Weapons – X
5. Pause – Start Button

**3.2) Player 2 (If necessary)**

Keyboard:

1. Run – A,S
2. Jump – W
3. Shoot – Left Shift
4. Switch Weapons – Q
5. Pause – Escape, Delete

Xbox:

1. Run – Left Thumbstick/D-Pad
2. Jump – A
3. Shoot – Right Trigger
4. Switch Weapons – X
5. Pause – Start Button

# *4.0) Architecture*

## 4.1) Input Framework

The input system is built around the idea of “input paradigms”—that is, it is built around the idea that, at any point in the operation of the game (usually corresponding on a per-state basis), there is a specific set of control inputs, tied intimately with the function of the state. That set of control inputs and functions, the input paradigm, is used to design input adapter classes that decode the appropriate control inputs into the appropriate functionality.

Example input paradigms are listed below:

* Press Any – a “press any key/button” continuation
* Menu – navigation of form-like GUI systems
* Terminal – navigation of console-like interface systems
* Game Play – controls during basic game play

There are two distinct forms of input paradigm: event-based and state-based. The XNA framework by default uses a state-based control framework. Examples of event-based paradigms are Menu and Terminal paradigms. An example of a state-based paradigm is Game Play. Some paradigms, such as the Press Any paradigm, could arguably be event-based but are implemented as state-based due to their extreme simplicity.

Event-based adapters tend to require instantiation and contain internal state information, while state-based adapters tend to be utility classes with static functions that return state structures.

## 4.2) State Framework

The game state framework is a variant of a state machine that places states on a state stack and is capable of running more than one state at once. States are placed on the stack from the bottom up (although ordering may be explicitly specified for some special effects). States are both processed and later rendered from the bottom up. Only the topmost state receives input (as dictated by the input controller adapters).

### State Index

The states of the game are given below:

* Loading – load all anticipated resources, based on an XML script
* Main Menu – display the entry-point menu for top-level game options
* Options – display game configuration menu
* Game Play – execute the game play
* Fade Transition – graphically fade between two game states
* Pause – allow for pausing a game in progress

Transition Diagram: Arrows indicate fade transitions between states

## 5.0) Level Framework

The level system is a paged world model. Data are deserialized from an XML script into an intermediate data object. The level system then uses the data to page in world segments as needed by the game.

### 5.1) World Elements

World elements are objects that interact with and are visible in the world. Basic world elements have the attribute of position and can be drawn. Specific world element subclasses add functionality to world elements.

### 5.2) Spatial Elements

Spatial Elements are objects that have physics-based properties, and are listed in Octrees, which keep track of location and collisions (ICollideable objects). Each spatial object has a bounding box and bounding sphere for collision detection and resolving.

### 5.3) Dynamic Elements

Dynamic elements are moving, active objects in the game, like Characters and bullets. Dynamic objects are controlled by physics and collision rules, which are defined by Spatial element rules. A dynamic object has specifications for movement, location, acceleration, and any forces acting on it.

## 6.0) Credits

### 6.1) Creators

Michael Rojas – Guns, Console, Documentation

Geoff Smith – Level Design, AI, Webmaster, Menus

Ronnie Hout – Sounds

Dylan Miller – Models, Animations, Textures

Ross Wang – Game State Control, Sounds, Installer

### 6.2) Sounds

 Sound Effects found at [http://www.a1freesoundeffects.com/weapons.html](https://outlook.com/owa/redir.aspx?C=32ee9bfa434f4d9c8f4403095911d75a&URL=http%3a%2f%2fwww.a1freesoundeffects.com%2fweapons.html)

### 6.3) Music

 “Attack From Outer Space” – 2DPolygon, Newgrounds.com

 “LCD” (Loading Music) – Created by Ross Wang

### 6.4) Models

 Models were obtained from Turbosquid.com

* Soldiers
* Mechs
* Tank
* Guns

### 6.5) 3rd Party Tools

 Audacity

 XACT

 3DS Max

 XNA Console + IronPython

 Gimp

 Paint

Sibelius 5