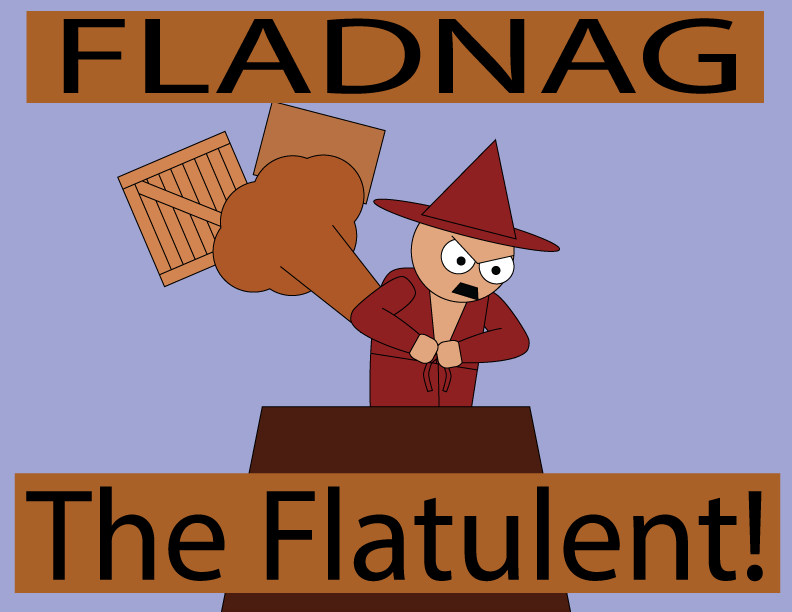
Fladnag the Flatulent



(Concept Image #1)

BEHD Team

Revision: 0.0.1

GDD Template Written by: Benjamin “HeadClot” Stanley

Altered by Brayden Mitchell

Special thanks to Alec Markarian

Otherwise this would not have happened

License

If you use this in any of your games. Give credit in the GDD (this document) to Alec Markarian and Benjamin Stanley. We did work so you don’t have to.

Feel free to Modify, redistribute but **not sell** this document.

TL;DR - Keep the credits section of this document intact and we are good and do not sell it.

[Overview](#_yj5nhqp5cf0j)

[Theme / Setting / Genre](#_5s48wntac2es)

[Core Gameplay Mechanics Brief](#_uzq23hfhdv6e)

[Targeted platforms](#_kvz0cxkhwt0s)

[Monetization model (Brief/Document)](#_421ijgnpyvmc)

[Project Scope](#_rdb2xo3rjh0s)

[Influences (Brief)](#_155cm8v36jpc)

[- <Influence #1>](#_c6nxu1rzd2cc)

[- <Influence #2>](#_ssiemceczw16)

[- <Influence #3>](#_31bxzkfeuvl6)

[- <Influence #4>](#_o4f1wa5aq6q3)

[The elevator Pitch](#_337xnergkz1b)

[Project Description (Brief):](#_z7oe7x50rpf3)

[Project Description (Detailed)](#_exbmsy55zuvb)

[What sets this project apart?](#_s4h84uy3suza)

[Core Gameplay Mechanics (Detailed)](#_a8x4s87df6uk)

[- <Core Gameplay Mechanic #1>](#_jyik8zbcjcio)

[- <Core Gameplay Mechanic #2>](#_y46mn9zee60t)

[- <Core Gameplay Mechanic #3>](#_lmzwvmw5e0hr)

[- <Core Gameplay Mechanic #4>](#_kct9c2l3dr9p)

[Story and Gameplay](#_6pmf08ssy6y0)

[Story (Brief)](#_ctv1wxi9dpll)

[Story (Detailed)](#_kqt2h5q76zyt)

[Gameplay (Brief)](#_ejtq4v6r30ui)

[Gameplay (Detailed)](#_cl69l94amjmx)

[Assets Needed](#_6m1256af7s3j)

[- 2D](#_1wb69txjqarm)

[- 3D](#_xdk2cy4n4ovn)

[- Sound](#_f8xx8iwg5gs9)

[- Code](#_ky1qxs88utre)

[- Animation](#_isk96p5euy3r)

[Schedule](#_kmt9zaowjejr)

[- <Object #1>](#_r3fjjzh8krjg)

[- <Object #2>](#_j584764hn4bz)

[- <Object #3>](#_lbj31oz0xb3v)

[- <Object #4>](#_p0jgh8xq0o3r)

# Overview

**Executive Summary**  
 FtF is a small game about a disgruntled wizard, Fladnag, who wakes up one day and decides to throw boxes at people to ease his suffering. He sits on top of his roof, in his bath robe, and conjures up boxes that drop on passerbys’ heads.

## Core Gameplay Mechanics Brief

- Box Conjuring- when the player clicks left or right of Fladnag within a certain area, he conjures a box that drops.

- Gravity/Drop Physics - the box falls according to a set gravity and collides with anything in its path. Once it hits something, it bounces once and then falls off of the screen.

- Points system - depending on who the player hits with their boxes, they will get a certain number of points.

- Timer - Fladnag will get tired after a while and go to bed, so it’s up to the player to rack up as many points as possible.

**Core Gameplay Features**

## Box-dropping on mouse click

1. Background + ‘playground’
2. Physics & Collision with Boxes
3. Walkers to drop boxes on + collision + animations
4. Points system + Timer (Gamification of app)
5. Wizard Animations + Particle/Effect animations
6. Aesthetic additions/Polish (Sun moving across background for time, more detailed animations for getting hit by boxes, more box types)

## 

## Project Scope

- Development Time: April 18th, 2017 - May 11th, 2017

- Cost: Nothing whatsoever, because we’re broke.

- Time Scale:

- Approx. 3 weeks of development time

- 23 Days

- 6-8 Team Meetings

- Dev Team & Skills Needed

- Coders:

- Brayden Mitchell

- p5.js (Known)

- Basic game collision/structural logic (Known)

- p5.play (Unknown)

- Physics Library/Functions (Unknown)

- On-click functionality without relying on event listeners (Unknown)

- Henry Dustrude

- P5.js (Known)

- P5.sound (Known)

- Physics Engine (Unknown)

- Animation States + Gameplay (Unknown)

- Basic AI Coding (Unknown)

- Artists:

- Ethan Suhr

- Paint (Known)

- Aesprite (Known)

- Photoshop (Unknown)

- Animation States (Unknown)

- Environment Animation (Unknown)

- Devin O’Connor

- Photoshop (Known)

- Illustrator (Known)

- After Effects (Known)

- Frame-by-frame effects (Unknown)

- Responsive Character Design (Unknown)

- Possible Working LIbraries:

- matter.js (gravity + rigidbodies)

- p5.play (collision detection, animation support)

- anime.js (possible animation support (additional))

## The elevator Pitch

A loser wizard throws boxes at people from the top of his house to make himself feel better.

## Project Description:

The player operates from an elevated static position while dropping objects on passing npcs. Most of the objects will be simple boxes that just hit the npcs although we might include an object with a special effect. The goal of the game will be to hit as many npcs as possible until the timer runs out. The way this will work will be that the character the player controls, Fladnag, will keep summoning boxes and dropping them on NPCs until he gets tired and heads back inside. Boxes will be summoned in some sort of special effect/particle effect, nothing has been concretely decided as of yet.

## 

## 

## Core Gameplay Mechanics (Detailed)

### - Gravity/Box Dropping

- Gravity will apply to certain objects at a set rate of force, using matter.js functionality to support core functionality.

When a box is created by the player, gravity will apply after a certain rate of time. Certain objects will have a ‘gravity apply’ boolean that determines whether or not gravity will apply to them (in theory).

### - Player Control/Box Conjuring

- When the player clicks within a predefined region of the world, left or right of the main character, they will conjure a box at either a predetermined position or the mouse position.

Either through event listeners or a constantly running function in the background, player mouseX and mouseY will be detected and assigned to a variable. On mouseClick/mouseDown, if mouse position is within Region Left, conjure a box at Region Left, and vice versa for Region Right. Call animations based on which direction/position.

NOTE: Might be worth considering a ‘mana’ bar that regens over time, to prevent the player from spamming boxes everywhere.

### - Points System/Collision Detection

- Depending on progress of project, player will glean points based on which type of NPC they hit, added to a total displayed somewhere on the screen.

On box collision w/ Walker, detect which kind of Walker was hit, and add their points variable to the total (once). Delete Walker immediately afterwards, to avoid multiplicity of point addition/weird bugs. In other words, on collision detection (box+Walker), execute a function that retrieves the walker’s point value, deletes them from the game, and allows the box to continue falling past the screen.

### - Game Timer

- Eventually, Fladnag will get tired of throwing boxes at people and go into his house for bed. As such, we will need a timer to reflect this action, represented by either a straight up countdown, or a sun moving across the sky as everything gets darker.

On game start, set the timer to dayLength, and as soon as the player presses ‘start’, begin to count down to 0 at a set rate. If sun is included, have progress along the sky be a percentage of currentTime/dayLength. When currentTime = 0, the game is over; display final score, disable player control, and have Fladnag ‘go to bed’. Allow player to restart the game.

# 

# 

# 

# Story and Gameplay

## Story (Brief)

The disgruntled wizard, Fladnag the Flatulent, seeks out innocent passersby to crush by conjuring wooden boxes that fall on his victims.

## Story (Detailed)

Fladnag the Flatulent, an elderly, aggrieved wizard, awakes one morning with a resounding feeling of resentment, due to (ONE of the following:

•The unjust divorce of his wife, Leirdalag.

•The despair of realizing McDonalds will never bring back the Mulan Szechuan Sauce.

•Disgruntled wizard things.) As a result he decides his feelings of depravity must be extinguished by transferring his pain to others in the form of crushing boxes.

## Gameplay (Brief)

Player will conjure boxes as Fladnag to boop passerbys on the head, to make Fladnag happy.

## Gameplay (Detailed)

As the day starts, Fladnag will come up onto his roof and begin to keep watch for Walkers along the sidewalk. The player will then click where they want each box to drop, predicting the fall of the box to properly hit as many people as possible. Meanwhile, the game will count via points how many people they’ve bonked, which kinds, and how long the game has gone on for. At a certain time, the game will halt, Fladnag will go back inside, and they will be presented with a score. Different Walker types give different amounts of points.

# Assets Needed

## - 2D

- Textures

- Environment Textures

- Heightmap data (If applicable)

- List required data required - Example: DEM data of the entire UK.

- Etc.

- Characters

- Fladnag (A drunk wizard in a bathrobe who summons boxes)

- Passerby one (A random person with random clothes)

- Passerby two (Old lady with a walker?)

- Passerby three (Some kid, smaller than the other passerbys)

- Passerby four (Apple employee? Blue shirt with kakis?)

## - Sound

- Sound List (Ambient)

- Outside

- Birds chirping/some other simple ambiance.

- Sound List (Player)

- Character Hit / Collision Sound list

- “OOF!” or “BONK” on Walker+Box collision

- Character on Conjured box

- Voice Line, randomized (2-5 different character VO that’s cycled through at occasional intervals)

- Conjure sound (“POOF” or fart sound)

## - Code

- Character Scripts/Functions (Player Pawn/Player Controller)

- Box Dropping - On mouse click, if player is within region L or R, drop a box at set location, allow gravity to come into effect after a short delay (for particle effects).

- Various UI Functions (Reset button, pause button, etc.)

- Ambient Scripts (Runs in the background)

- Gravity script - apply constant gravity to certain objects (boxes), allow tweaking of force.

- Collision Detection - check + possibly draw collision boxes around characters, execute events on collision (points go up, person gets deleted, etc.)

- Walker Generation - generate NPCs at set/random intervals that follow the WalkAI code.

- NPC Scripts

- Walker AI (Constructor) - For each generated Walker, have them walk left/right (Depending on region) towards the tower, until either A: they get hit by a box or B: they reach the other side of the screen, and get deleted. Walk based on a set rate of movement, which may/may not fluctuate. Consider different ‘classes’ of Walker, with different speeds/point values assigned.

## - Animation

- Environment Animations

- Sun flicker/movement from one side of the screen to the other.

- Possible clouds drifting across the screen.

- Character Animations

- Player

- Conjure animation - plays on box conjure

- Idle animations - Shaking of fist at passerbys, etc.

- NPC

- Walking animation - walkers in a simple walk cycle (may differ per character)

# 

# 

# Schedule

### - Basic Mechanics/Framework (Code Outline, Basic Environment, Main Character model in-game)

- Final (Ideal) Deadline: April 25th (By end of first weekend)

- Begin work on April 20th

- Create a basic framework for the code to function within.

- Create the main character model. (Fladnag)

- Create the play environment/playspace (Background, tower, etc.)

### - Core Gameplay (Gravity scripts, point functions, rudimentary animations)

- Final (Ideal) Deadline: April 27th

- Get functioning gravity + collision detection for boxes/walkers

- Detect if box + walker registers, make sure the boxes fall after being created and collide with walkers.

- Create regions and scripts for player input, create boxes on click

- Create box designs + Walker Models/Designs

- Finish main character animations

### - Movement (Walker AI, Collision Functions, Walker Animations, Conjure/Environment Animation)

- Final (Ideal) Deadline: May 2nd

- At this point, meet and decide which features we want to focus on/what’s feasible.

- Code AI for Walkers, make each function as a separate object

- Create proper collision functions for box+walker on-hit.

- Make sure collisions properly destroy/de-register the walker, to avoid stacking events.

- Create animations for conjuring boxes + other environmental effects (Sun moving+flickering in the sky, clouds moving back and forth, etc.)

- Create animations for each Walker type.

### - Final Stretch

- Deadline: May 11th

- Finish any and all features leftover for polish/aesthetic purposes. Make sure group is on point.

- Debug.