# Bad tempered Birds

## Design Doc

### Status:

The game is ready for a beta or an early release. The game has a 10 levels for the players to start with, along with two different birds for them to use. The entire gameplay loop has been implemented, and the game can be played from the start to the end without issues. Some things the game needs to add as development continues are as follows: The game needs additional birds to keep the different levels interesting and varied. The game will also benefit from different “pieces” with which to build the level. This could be a new material type to build the pig’s house with, or even a TNT block to explode when hit. Both of these give the game more variety, as the new material will force the player to approach the level in a different way, and the TNT block will give the player another target in the level, as opposed to just the pigs. The game also needs some quality of life changes. Some of these include, a settings menu to change things like sound volume or controls. The game has a basic UI when playing, with a score indicator and bomb counter on screen in the top left and right respectively (see below). This UI also has an onscreen button to mute the music. The UI will need to be expanded to include a settings menu and level selector. The level selector will let the player replay specific levels, which enhances replayabillity. The game also needs sound effects to better convey to the player what is occurring. This makes the game more readable and will make it easier to play.

This screenshot from the game shows the the UI that the player sees during the levels. Top left is the score counter, top right is the (remaining) bomb counter, and bottom right is the mute button.

A screenshot of a video game

Description automatically generated

### Elevator Statement:

The game is a casual angry birds style artillery game, but instead of using a catapult, the player has to launch the birds using bombs they can place anywhere, giving unique versatility for an angry birds style game. Each level has lots of ways to complete it, which gives each level replayability. In addition to this, there is also a high score system for completionists, which gived each level even more replayability.

### Concept:

The game is the very similar to Angry Birds, however, instead of launching the birds from a fixed catapult the player uses bombs to launch the birds. This gives the player more freedom in how they approach the level, as the bombs can be placed anywhere in the level, however it also makes it more difficult. The game is more difficult because it is harder to predict how the bombs will launch the bird compared to the catapult from the original games. The catapult also used a trail in the sky to show where the birds would go, but Bad tempered Birds players will need to use their intuitive understanding of physics to predict the flight of the birds.

### Genre:

The game is a casual mobile game, with taking from genres such as the catapult/artillery game, and puzzle games. The artillery game inspiration is obvious as the game primarily consists of launching an object (bird) at a target structure. The puzzle aspect comes from how to actually hit the pig. The pig is protected by various structures and materials each with different properties. How to get past all of this using the fewest bombs possible is the puzzle. Both of these are also in the angry birds game, however the artillery aspect of the game is different because the same bird is always used, instead of dying on impact. Furthermore, the player also must use bombs to launch the bird instead of the catapult or trebuchet, which is typically seen in artillery games.

### Target Audience:

The game has a wide target audience. The mobile games industry is massive, taking up 49% of the games industry revenue in 2023 at $90 billion. This means there is a large potential audience. This is capitalized on by making a casual game, that can appeal to lots of people. The game is a casual artillery game, with a child friendly art style and content. The target audience is largely the same as angry birds, however it does deviate slightly as Bad tempered Birds is more difficult than angry birds. This is because it uses bombs instead of the typical catapult to launch the birds, which makes it harder to accurately predict where the birds go. It is worth mentioning, although it is harder to predict the birds flight path, it is consistent, and therefore skill based.

A target ESRB rating would be as low as possible, preferably “E for everyone”, as the aim of this game is to appeal to as wide an audience as possible. The best age rating available to achieve this is clearly the least restrictive “E for everyone” rating. The equivalent rating for PEGI would be the 3 years old rating, or the easier to achieve 7 years old rating.

The 7 year old rating for PEGI is much more likely for this game, as it does contain very mild violence against the pigs. It seems that any violence in the game will get it the 7 year old rating, which basically forces the PEGI age rating target to be 7 years old.

### Key Moments:

The gameplay loop has a few key moments for every level. The first is when the level first starts, and the player is seeing the level’s challenge for the first time. Here the player has to figure out and plan ahead what they will do in this level. This part should be kept as visually clear as possible so as not to distract or confuse the player. The next key moment is when the player starts to destroy the pig’s building. This should also be kept relatively clear visually, so as not to overwhelm the player, or prevent them from seeing what is actually happening. This second key moment should have at least some visual shine to make it enjoyable to destroy the building and progress with the level. This is achieved through particle effects on the bomb and pieces, and, later in development, camera shake when the bomb explodes. The final key moment is when the level is completed, and the end level menu appears. This menu shows the number of stars the player got and is the primary reward. This should be full of visual charm, such as bright particles or fireworks etc. It also should have a custom victory or loss sound later in development.

### Player Objectives:

The primary objective for the player is to kill all the pigs in the level. However, there is a secondary objective which is to finish the level with the highest score possible. This is achieved by destroying as much as possible, while using as few bombs as possible. This objective is represented by the stars system, whereupon completing each level, the player will be assigned zero to three stars depending on their success with the level.

A screenshot of a game

Description automatically generatedA screenshot of a video game

Description automatically generatedThe levels are designed to first introduce the primary objective of killing the pigs. This is achieved by level one where the only thing in front of the bird is a singular pig (see below left). This intuitively guides the player towards killing the pig in this level and consequently all future ones. The secondary objective is only introduced once the player finishes the level with the win/loss menu (see below right). This means the player is typically focused on the primary objective of killing the pigs, with the secondary objective being available for players who want more of a challenge, without distracting everyone else.

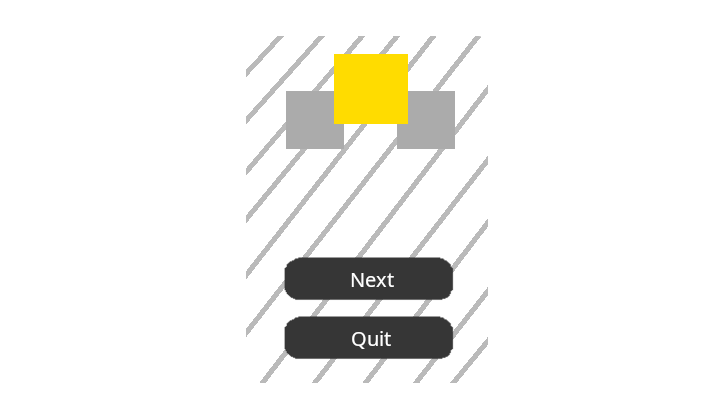
### Progression:

There is only one form of progression in the game: the levels. The player progresses through each level one after the other, as they play. This is typically quite rewarding as the player can look at the level selector and see a real time tracking of their progress through each level, and sometimes how many stars they achieved. The game currently doesn’t have a level selector, but later in development it will be added. This is because without the level selector, the game lacks a lot of the reward from the progression. There is currently no way for the player to see how far they got, or how well they did on that level.

### User Interface:

The UI should use the same assets throughout the game, e.g. the same sprites for the buttons, the same text etc. This will add visual cohesion to the game. It also increases the readability of the game, for example the buttons will always look the same, so the player will always know what a button looks like.

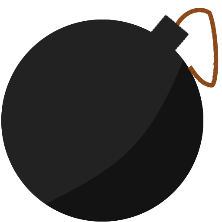
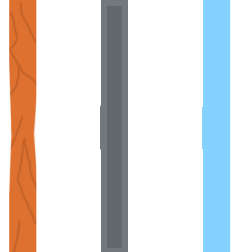
The first thing the player will see is the main menu. This should have two buttons: Start and Quit. These should be under the title, with a generic contrasting background behind them. This background will wiggle side to side to keep the background visually interesting. The buttons should be centred and rectangular. The buttons will have contrasting coloured text on them, explaining their purpose. The top button will play the first level, the bottom button will quit out of the application.



The menu that appears when the player completes a level should be a vertical rectangle, with rounded corners. This rectangle should have a solid fill contrasting background and take up about 80-90% of the screen (vertically). The menu will have two centred buttons, very similar to the main menu. The top button will play the next level (or restart the level if the player lost), while the bottom button will quit to the main menu. At the top of the menu will be three stars that are grey when not achieved, and gold when achieved. This will be controlled by the script and come from the score achieved in the level. There is also a particle system that spawns some particles across the screen when the menu spawns in. These particles are gold, just like the stars. These particles are not included in the wireframe.

The menu that appears when the player loses the level should be identical to the win menu, however, it should have a restart button instead of a next button. This restart button will call on RestartLevel() instead of NextLevel() and should have the text Restart. There will also be no particles spawning from this menu.

### Art Style:

The art style should be cohesive throughout the game. It should be cartoonish, with simple shadows and minimal shading. The pigs and birds should be circular, with a main strong colour. They should have childish, and cartoony faces. The characters can have some extra detail on their sprite in a slightly darker colour. The pieces should all be cartoony as well, but also all the same dimensions in the scene. They can also have some detail on their sprite, but it should be mainly one colour, more muted than the characters. Other assets should roughly follow these rules, but not necessarily the same shape.

### Tools:

The game is made almost entirely within the Unity Engine. The only other tools used are GIMP for creating sprites, Visual Studio for creating scripts, and Git for version control. The Unity Engine handles the physics, rendering, input, scene management, window handling, asset loading, etc. All of these systems are designed to work in harmony with each other. This helps streamline development, as the developers don’t need a lot of interoperability between toolchains. The other tools used use formats that Unity can handle directly, such as .png or .cs. This again simplifies the toolchains and helps interoperability.

To actually create levels, the developer should use the “SampleScene” as a base. It contains all the required elements for a level, and a basic building as an example. When adding pieces, they need to be assigned a reference to the GameManager in the inspector. Once all the pieces are placed, the pigs should be placed next. For each pig placed in the scene, the maxPigs variable on the GameManager should be incremented. The bombCount variable should be adjusted so all the pigs can be killed with one bomb remaining. Finally, maxScore, the score required for 3 stars, should be set on GameManager to complete the functionality. When each scene is added to the build index, the maxSceneIndex constant should be incremented on EndLevelScript. If all of this is done, the game should be able to flow from one state to the next until all the levels are completed.

To create a new piece type, the developer should copy one of the existing prefabs for both the piece and the break particles. The piece prefab will have a Piece script, which can be used to vary the behaviour of different materials. The main things the developer should look at when changing values is the breakThreshold, which controls how fast an object must be moving to break the piece, and the mass (on the rigidbody not the script) which changes how easy it is to move the piece by hitting it. The breakParticles reference should also be updated with the new particles.

### Level Design:

Each level should have at least one building. The exception to this is the first level which introduces the concept of attacking the pig. This building should be made up of pieces, that either protect the pig or in some way prevent the player from successfully attacking the pigs/completing the level. There should be at least one pig in each level. The maxScore for the level should be achievable with the number of bombs available, but not easily so. This maxScore doesn’t have to require everything in the level to be destroyed and shouldn’t require absolute perfection to achieve. The number of bombs available to the player should be enough to kill the pigs with an optional bomb left over, especially if the level has lots of pieces or lots of steel pieces. This makes the maxScore more achievable for the target playerbase of causal players and children.

### Localization:

The game is not very text heavy. All the text is in the UI, primarily the buttons and title. The game is in English, but it should be trivial to localize the game. In total there are 4 text buttons that are reused, the “score” text, and the title text. To actually localize the game there are lots of options available, the most obvious and easiest one is Unity’s localization package. This lets developers localize text, strings, and other assets such as PNGs. This would be used to localize all the text in the game.

None of the sprites need localization as they don’t contain any text, flags, or any other country specific imagery. The music, similarly doesn’t contain any lyrics, and as such doesn’t need any localization.

### Monetization:

The game is designed to be as casual as possible, and as such is not very compatible with traditional monetization. For example, if the player were able to pay to get extra bombs, it would be ineffective. This is because each level can be restarted in a few seconds and takes less than a minute to complete. Furthermore, each bomb the player places will decrease the score, so having more bombs than initially intended would likely leave the player unable to achieve 3 stars. This makes that method of monetization even less viable, because who would pay to be blocked from a high score. Another avenue of monetization is skins. This would likely work quite well with the game, as the birds use a collider, independent from the actual sprite. This lets the designer use virtually any design, and long as it has a primarily circular shape. The pieces would be tough to design alternate skins for, as they need to be readable for the player (glass vs steel strength). The enemies, however, are just as versatile as the birds in terms of skins available.

### Game Objects:

The game objects visible varies greatly depending on which scene is currently active. For the main menu there are just 11 game objects, only 5 of which are visible. The first game object to mention is the Camera. This is a unity game object and is used for rendering and is in most scenes. In this scene, the child of this object is a so called Empty. It is simply there to hold other components. Here it is used to hold the audio source, which is used to play the background music. The next object is an Empty that holds the MainMenuManager script. This script controls the background animation, and the functions of the buttons. The next object is the Canvas parent object, this is another Unity object and simply controls rendering of the canvas through the camera. The next object is a TitleLabel object that handles rendering of the title UI text. The next two objects are the buttons that appear in the scene. They both have button components and call on the MainMenuManager script when they are clicked. The children of both of these objects, are the text on the button. The next object EventManager, is controlled by Unity. The final object is a background image. This image has an animator component to allow it to play the background shake animation.

For the next scene, the gameplay scene, there are some familiar objects, for example: the text labels, the canvas, the event manager, and the Camera which are all basically the same as the main menu. There is an Empty object that holds GameManager script. This script handles the spawning of the bombs, moving of the logs, the flow of the game states etc. The Bird game object is an Empty that holds the rigidbody, collider, and the script. The bird script handles respawning itself and also adding the rigidbody to the game manager’s list. The bird has a sprite renderer as the child object. This is to allow it to be moved without offsetting the collider. The Pig object is very similar to the bird object. The difference is the sprite and the use of the Pig script as opposed to the Bird script. There are also some values that are changed such as the collider radius. The Log object is a basic sprite renderer and collider that is moved by the game manger. The Building1 empty has all the Pieces as its children. These pieces are built in much the same way, with a rigidbody, collider, and script, however, it has the sprite renderer on the main object instead of the child. There is an Empty object Background that holds the multiple background images that make up the background.