

BitDash Product Document

*Alasdair Cross, Cameron Roberts, Danny Demarco, Shuai Ning, Xiaoyan Zhou, Xuan Wang,
Ye Zhu, and Zhiyuan Tan*

Master of Science in Computer science

University of Bath

Dec 2020

Contents

BitDash User Manual	1
What is BitDash?	1
Minimum Recommended System Requirements:	1
Installation of BitDash:	1
Playing Bitdash:	1
Jump:	2
Dash:	2
Troubleshooting:	2
Installation Guide	3
Windows	3
Mac	3
Maintenance Guide.....	4
Part1 A detailed signposting of how to navigate the code.....	4
Part2 Where and how to make extensions.....	6

BitDash User Manual

Thank you for your interest in the game BitDash. We hope you have a great time playing the game and find it as entertaining as the development team has to produce it.

What is BitDash?

BitDash is 2d, Parkour inspired running game in which you as a player needs to run, jump and dash your way through the game. The theme of the game is dungeon inspired so is dark, scary, and full of horrors like monsters, and death traps. Use your speed, agility, and quick reactions to get as far through the levels as you can before you are killed off.

Minimum Recommended System Requirements:

OS: Windows 7 and above/ MacOS Catalina and above

CPU: Intel i5 or newer @4.0GHz

RAM: 4GB and above

Installation of BitDash:

Please refer to attached document 'Installation Guide'

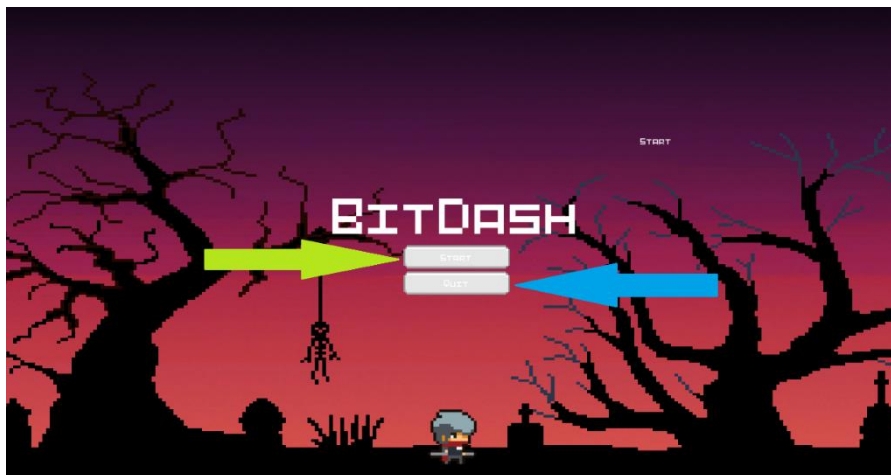


Figure 1: Main Menu

Playing Bitdash:

Bitdash is designed to be as user friendly as possible, and is so easy to play, that all ages can enjoy the game. After following the instructions to install the game, you will be met with the Main Menu (See Figure 1). From this main menu you can either exit the game, by choosing 'QUIT' indicated by the BLUE arrow in Figure 1, or you can start the gameplay by choosing 'START' indicated by the GREEN arrow in Figure 1. Upon choosing start, you will immediately be thrust into the game and your character will start to run automatically. Gameplay is now controlled with just 2 key choices: Jump or Dash.

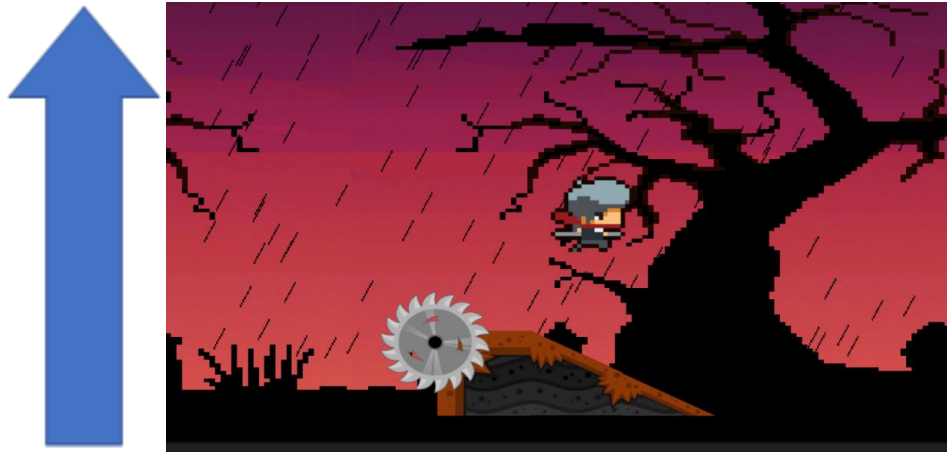


Figure 2: Jump

Jump:

As indicated in Figure 2, before you reach an obstacle, you can use the upwards arrow on your keyboard to initiate the character's jump.

Dash:

The dash function makes the character sprint forward at speed so as to avoid any of the horrors lurking around trying to kill you off. As the character dashes, he reduces in height somewhat and so this can be used to run under low objects or hazards. To initiate the dash, you should press the downwards arrow on your keyboard as indicated in Figure 3 where you can see the character dashing under a particularly terrifying monster!



Figure 2: Dash

That is all you need to know to get started with BitDash! Please enjoy.

Troubleshooting:

If the game is not running properly on your system, please first check that your system meets the minimum recommended specifications listed above. Also please ensure that all your local drivers are up-to-date and perform a system restart.

Installation Guide

Windows

1. Download the BitDash_Win.zip folder.
2. Extract the contents of the zip folder to your chosen location on your computer.
3. Open the BitDash_Win folder.
4. Open the BitDash application. The game window will open on the start screen of the game. You can now play the game.

Mac





1. Download the BitDash_Mac.zip folder.
2. Extract the contents of the zip folder to your chosen location on your computer.
3. Open the BitDash_Mac application. The game window will open on the start screen of the game. You can now play the game. (Most of the Mac would forbid open the applications from the unknown developers, if so, please hold the control button to open the application.)

IMPORTANT: The BitDash application can only be opened when in the same folder as the game data. If you want to change the location of the application, please move all of the contents in the folder to the same location.








Maintenance Guide

Part1 A detailed signposting of how to navigate the code.















First click on the BitDash folder.

 BitDash	final version upload
 .gitattributes	update gitattributes
 .gitignore	Update .gitignore
 README.md	Initial commit

Second select the Assets folder.

..	
 .idea/.idea.BitDash	update
 Assets	update
 Library	final version upload
 Logs	The initial game for SE. Version 1.0
 Packages	The initial game for SE. Version 1.0
 ProjectSettings	update
 obj/Debug	update

Third find the Scripts folder.

..	
 Animations	update
 Backgroud music	Adding BGM
 Material	The initial game for SE. Version 1.0
 Prefabs	update
 Scenes	update
 Scripts	update
 Sprites	update
 Animations.meta	The initial game for SE. Version 1.0
 Backgroud music.meta	Adding BGM
 Material.meta	The initial game for SE. Version 1.0
 Prefabs.meta	The initial game for SE. Version 1.0
 Scenes.meta	The initial game for SE. Version 1.0
 Scripts.meta	The initial game for SE. Version 1.0
 Sprites.meta	The initial game for SE. Version 1.0

As can be seen from the screen shot below, there are four folders named Castle, Cave, Forest, Forest1. These four folders contain all the code of the project. The code in each folder corresponds to a different game scene, but the functions implemented are roughly the same. The only difference between these three folders is that only the Start method of the GameManger class in the Forest folder adds the mouse monitoring event of the start button, which allows the player to manually start the game. The start method in the other three folders does not have this mouse monitoring event, which means that all scene switching is automatic.

..	
Castle	update
Cave	update
Forest	updated with scene 2
Forest1	update
Castle.meta	update finish castle
Cave.meta	update finish castle
Forest.meta	update
Forest1.meta	update

Take the Castle folder as an example. Each different scene contains three classes, namely CameraManager1.cs, GameManager1.cs and Player1.cs. The language used for all the code of the project is C#.

..	
CameraManager1.cs	update finish castle
CameraManager1.cs.meta	update finish castle
GameManager1.cs	update
GameManager1.cs.meta	update finish castle
Player1.cs	update finish castle
Player1.cs.meta	update finish castle

Part2 Where and how to make extensions

The extensions can be divided into two parts, namely the function of the game and the ability of the characters in the game. The former can be extended in GameManager class, and the latter can be extended in Player class.

Extensions of game function

The game currently includes four functions: Start, Quit, Restart and Game over. As can be seen from the screen shot below, The Game over function is implemented by adding GameOver() method to GameManager class. So developers can add a different method to the same class to implement a different game function, such as Game Pause.

```
//GameOver function
Frequently called 1 usage TommyTanZzz +1
public void GameOver()
{
    //Player is dead
    if (player.playerDeath == true)
    {
        title.text = "Game Over";
        title.gameObject.SetActive(true);
        restartButton.gameObject.SetActive(true);
        gameState = GameState.Stop;
        GameStart = false;
        timelock = true;

        restartButton.onClick.AddListener(Restart);
    }

    if (player.playerWin == true)
    {
        SceneManager.LoadScene(1);
    }
}
```

Expand the ability of the game character

The hero of the game can currently run, jump and dash. Simply put, these three abilities are implemented by adding three methods and other useful judgment conditions to Player class, so developers can create new abilities for the game protagonist in Player class, such as killing monsters.

```
public void Dash()
{
    animator.SetBool( name: "dash", value: false);
    GetComponent<BoxCollider2D>().size = new Vector2(bodySize.x, bodySize.y);
    GetComponent<BoxCollider2D>().offset = new Vector2(bodyCenter.x, bodyCenter.y);
    dashSwitch = true;
}
```