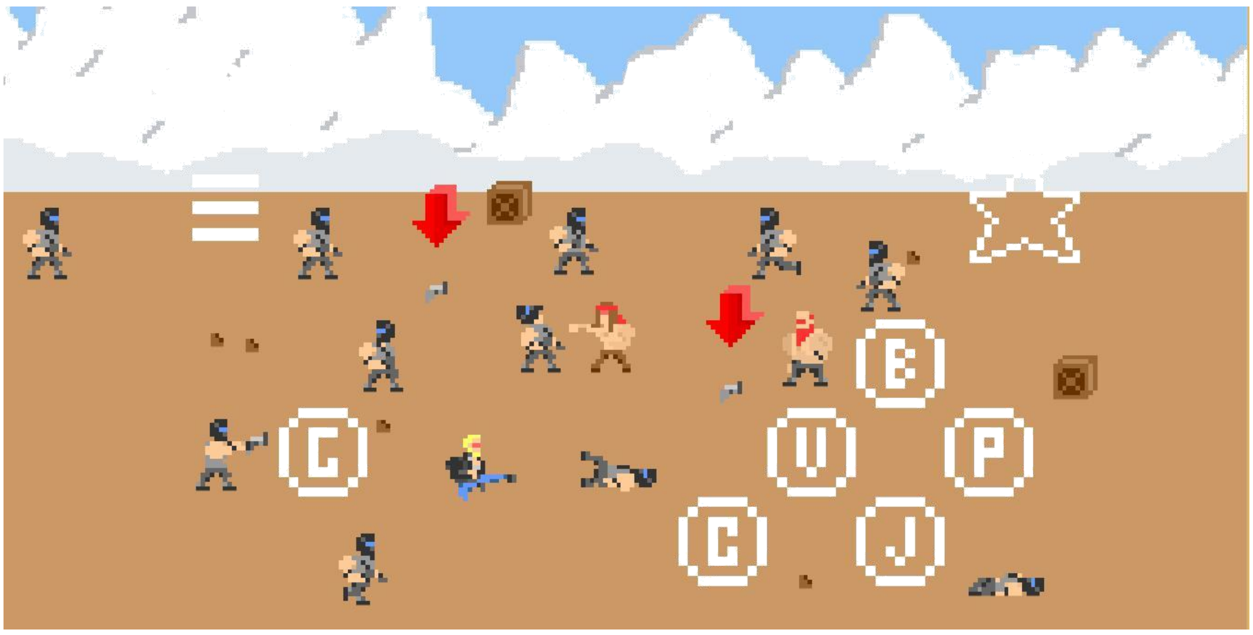

BEAT'EM UP 2D ultimate kit

by Fome development

Hello, dear Developer!

Thank you for purchasing our asset “Beat’emUp 2D Ultimate Kit”. Fome development team hopes you will get to get the maximum possible from our work!

The “Beat’em Up 2D Ultimate Kit” is a beat’em up game template, which is constructed using best practices of beat’em up genre. We hope that you will be the one who’ll bring the genre back to life.

If you have any questions, remarks or just want to chat, feel free to write us on fomedevelopment@gmail.com

If you want this asset to be updated or if you want our team to create new assets – help us by rating this asset.

Content

Folder structure

In ourasset you can find such folder structure:

- Sprites
- Scripts
- Scenes
- Materials
- Prefabs
- Animation
- Audio
- Facebook SDK
- UI
- Mobile UI

Sprites

Despite our main goal is to provide ready-to-go game for reskin, “Beat’em Up 2D Ultimate Kit” comes with premade sprite sheet, that fits gameplay and adds little specific. You can use it for any purposes.

Package includes UI spritesheet, which you can use as an universal UI for all your projects.

Aggregation all sprites in one file improves the performance of the game.

Scripts

All scripts are written in C Sharp and fully commented.

Scenes

You can findherethree scenes: game (Mobile UI/Scenes folder), level selector and demo level.

Fonts

We can’t include Google fonts in out package, but it’s free and we recommend to use Google font called “VT323-Regular” in your game, you can find it here https://www.google.com/fonts/specimen/VT323

Materials

In our game we use at two materials: pixel snap (to improve pixel graphics performance) and hitted material (as a classic beat’em up effect). Also, we created 3 materials to apply colors to fonts.

Prefabs

Here you can find characters and usable objects, which you can customize, reskin and place in a game scene. Also, you can find a prefab of a level button. It is used in level selector.

Animation

Animator controllers and animations are distributed by names.

Audio

Audio folder stores oneShot audio clips, such as score, win, fail and punch sounds. All audio tracks are made in Ableton live with default synthes.

Gameplay

Player controls one of playable fighters. The goal is to get through tens of enemies to a usable object that triggers “win”. Player can beat, block, jump, use usable objects (shoot weapon, collect med kits, score points), and select other playable fighters. Ally and enemy have AI, which lets them patrol area, follow, attack opponent, randomly block attacks, randomly make super punches, pick weapon and shoot. You can create different characters or usable objects as many, as you want. You can fully customize any parameter (chance of block, chance of super punch and so on) to create your own original gameplay.

Keyboard control:

W/A/S/D – to move
E – punch/shoot
Q – use
Space – jump
B - block

Touch input control:

Put one finger in a left side of a screen and move it to move fighter. Use second finger to press buttons in a right side of a screen.
P – punch/shoot
U – use
J – jump
B– block

User interface

Main menu allows player to choose character, earn coins by watching ads or sharing game link in a social network.

Game scene contains of two basic buttons (rate button, which leads to an URL you'll set, and button, which opens in-game menu), and five gameplay buttons (P - punch, B - block, J – jump, C – change fighter, U -use)

In-game menu contains of restart, back to main menu and next level buttons.

Also, as non-intractable UI elements, there are score bar and health bar.

GameObjects' system

We have gameManager, which is responsible for the basic functions, such as pause game, open/close in-game menu, load level, restart,go to rate page and other.

We have gamePlayManager, which responsible for organization game process.

- It counts score and refreshes score bar.
- It counts health and refreshes health bar
- It tells message bar what text to display
- It plays win/fail/score audio clips

We have keyboard input manager.

We have touch input manager.
We have premade fighters (ally (playable) and enemy)

We have main camera, with a cameraBehaviour script. In camera settings we can put camera's height, but in our case, we need to set camera's width. We do it in cameraBehaviour script. Put your specific size in script settings.We put box collider to camera to make camera always stay in playable area while following controlled fighter.

Canvas contains of

- In-game menu
  In-game menu is a game object, that contains 3 UI buttons
  - Restart
  - Menu
  - Next level (appears if it is not the last or the only level)

- Score bar

Score bar is a text UI element, whose text is controlled by gameplay manager.

- Rate button

  Rate button is an UI button, which leads to at link (https://play.google.com/ as default)
- Message bar

  Message bar displays text in a middle of a screen. If you want to display text there, call displayText("text", time) method in gamePlayManager script.
- Health bar

  Health bar shows how much health controlled fighter has.

You can create fighters. Use premade prefabs, set fighter's parameters as you decide and select it's friend's and enemy's layers.

If you want to make fighter playable – put it in array of playable fighters in game play manager.

Technically, fighter is a game object, that overlaps surrounding area to find enemy. If there are enemy, it goes to closest one and attacks it. If there are no enemy around, fighter overlaps area to find a friend. If there are no friends, fighter patrols position.

You can create usable objects/weapon. Use premade prefabs, define, what usable object should do by setting parameters in usableObjectScript.

Also we have environment game object, which stores barriers for fighters. We have an edge collider all over game area. It is made to make camera not leave the playable area while following controlled fighter.

SocialMediaManager stores your game and personal links, so player share your game with friends and follow you on twitter and tumblr.

AdManager used to add ads in your game (Interstitials and rewarded videos)

CutsceneManager triggers dialogs.

Animation

Every fighter should has his own animation controller. But you can copy premade animation controller and fill it with new fighter's animations. In default, every fighter (except boss), has 6 strike animations and 2 block animations. If you want to add more animations, add it to animation controller and connect it to other states just like it is made with other strike/block animations.

Scenes

You can find 3 scenes in a “Scenes folder”.

- game

  This is a first game scene, that player will see. It has 5 buttons:

  Play (loads level select scene),

  Rate (opens your link in default browser),
  Open character selector,
  Open settings,
  Open earn menu (appears only if there are available earn options)

  Scene has MenuManager script to provide those functions.
  You are absolutely free to add your own buttons. Just put them inside the canvas.

- levelSelector

  This is a second scene, where player selects level to load. To prepare levels select LevelManager and set number of levels. Script will do the rest.
  First level is opened. If it is finished, second level becomes opened and so on.

- level_1

  This is a demo game level. In this level you can see an example of a simple level.

Scripts

- cameraBehaviour

This scripts sets camera width. According to gameplay specific, we need fixed width of an area, instead of height, to keep game process the same on different-size devices.

- gameManager

This script contains non-specific methods, that pauses/unpauses/loads level/opens URL/closes game. game player manager, UI buttons refer to gameManager. The only two game objects, that gameManager refers, are in-game menu and in-game menu button.

- gamePlayManager

This script organizes game process.

It counts health, score points, refreshed UI elements, plays audio tracks, displays text in text bar.

In "UI elements" section we put message bar, score table, health bar, gameplay buttons and next level button in a proper variables. Message bar shows intro text (which is set in gamePlayManager) at the beginning.

When player fails, game play manager tells game manager to disable menu button and plays one shot fail sound.

- fighterScript

This script makes fighters move, fight, patrol, look for enemies and friends. With this script you can fully customize your fighter and select his side (ally or enemy).

- usableObjectScript

This script makes usable objects trigger applying score, health, win game or take weapon.

- weaponBehaviour

This script inherits from usableObjectScript and stores such variables, as ammo prefab, number of ammo and index number of an animation.

- boxBehaviour

This script inherits from usableObjectScript and stores such variables, as hit direction and throw power.

- ammoBehaviour

This script makes ammo damage fighters in case of collision

- CutsceneManager

This script allows you to create cutscenes with dialogs.
Cutscene manager checks every frame if any cutscene is triggered. If it is triggered, it starts a dialog.
This is a foundament for building more advanced cutscenes.

- MenuManager

This script organizes menu (first scene). It manages flow between sub menus and counts player's coins

- SocialMediaManager

This script allows share and post game links on facebook and twitter and allows player get a reward for following developer on twitter.

- AdManager

This script is used to manage work of your ad network service. We advice you to use Appodeal (https://www.appodeal.com).
It contains of show ad functions and callbacks.

- Audio

This script is a class for an audio clip, which allows to set unique volume for every audio clip.

Preparing game

Setting tags

You need to check if gamePlayManager and gameManager have their tags on.

Reskin

You need to make steps to reskin this game:

- Draw your game.
- Put your atlas in “Sprites” folder, instead of original. Call it Spritesheet.
- Put all sprites in one atlas, arrange sprites to all gameObjects.
- Create/download audio clips and put it in proper variables ingamePlayManager.
- Manage animators and animations
- Create/download font and put it to all UI elements.
- Put proper links to a social manager.
- Design levels. Use level_1 scene as a start point in creating levels, because gamePlayManager, gameManager and menus are already prepared for a game.
- Set number of levels in level selector.
- Publish your game.

-

Integrated SDKs

We’ve integrated Facebook SDK.
(https://developers.facebook.com/docs/unity/)
With it you can socialize your game with every feature facebook has.
In this template with allow player to make post with game’s link to get reward.

Credits

Fome development team wants to thank

- Unity

  Thanks for making it all happen https://unity3d.com/

If you want this asset to be updated or if you want our team to create new assets – help us by rating this asset.