Eight Directional Sprite System (EDSS)

by Healthbar Games

LEGAL INFO AND LINKS

Licence:	https://unity3d.com/legal/as_terms
Asset Store link:	https://assetstore.unity.com/packages/slug/134177
Project page:	http://healthbargames.pl/projects/eight-directional-sprite-system
General contact:	contact@healthbargames.pl
Support contact:	support@healthbargames.pl

INTRODUCTION

The creators of the first computer games implemented in 3D technology, encountered many problems related to the efficiency of the equipment at the time. The mere rendering of the three-dimensional world of the game absorbed most of the computing power of the computer without leaving it much for the rest of the calculations. Therefore, to display characters and other models, it was decided to use the so-called sprites (ghosts) or two-dimensional images representing a given character or model. It was so-called 2.5D characters. The character's animation took place through a fast, continuous replacement of the images of a given sprite. For the character, eight separate animations were prepared - one for each of the eight different directions in which the character could be oriented. All pictures for a given character or game model (all animations) were stored in one texture - so-called atlas. The clever system of calculating the image number allowed to display the picture appropriate for a given frame of animation and the angle resulting from the direction of camera observation and the direction in which the given model was returned.

EDSS is just such a system that allows you to display 2-dimensional sprites in the 3D world in the way used in those games.

It should be noted that due to the high demand for images, some of the animations of a given character were prepared only as 1-directional. During such animation, the given actor was always facing the camera, no matter what position in the world and what

direction the camera had. With this animation, it was not possible to look at the given character from the side or from the back. EDSS supports both: 8- and 1-directional animations as well as 1-frame model positions without any animation.

DESCRIPTION

Each EDSS actor consists of two separate objects. The main object is responsible for the position and orientation of the actor in the scene. The second, subordinate object is responsible for the graphical representation of the actor. It must have the *Sprite Renderer* and *Actor Billboard* components.

The *Actor Billboard* component deals with determining which image from a given animation should be displayed by the *Sprite Renderer* component and additionally it deals with setting the sprite so that it always faces the camera. The *Actor Billboard* component must know the current orientation of the actor in the world (scene). This information can be provided to him in two ways. The first way is on calling the `SetActorForwardVector(Vector3 actorForward)` function passing the direction of the actor with the *actorForward* parameter. This should be done after changing the direction of the actor so that the *Actor Billboard* component is up to date. Then just before rendering by the current camera, the *Actor Billboard* component will use the current actor vector to determine the image to be displayed. The second method is to indicate in the *Actor Transform* field of the *Actor Billboard* component the object from the scene whose forward vector will be responsible for the direction of the actor. In this case, the *Actor Billboard* will read the actor's orientation automatically just before the sprite for the current camera is rendered.

In order to display the actor on the screen, the *Actor Billboard* component should be supplied with the animation to play. Animation in EDSS contains a sequence of frames (images) to display in the right order and at the appropriate speed. In the case of 8-directional sprites, each frame contains a definition of eight pictures (one for each of the 8 directions of the character). However, in the case of 1-directional sprites, from each frame, only one (the first) image is used. For static actors which do not have any animations, you still need to create the *Actor Animation* with only one frame.

Animation is created from the menu: [Assets / Create / EDSS / Actor Animation] or from the context menu (right mouse button): [Create / EDSS / Actor Animation]. When creating an animation, specify the number of frames to display in order, the speed of display and whether it is an 8-directional or 1-directional animation. You should also assign sprites to each frame and each direction of the actor.

To set the current animation to play in the *Actor Billboard* component, use the `PlayAnimation(ActorAnimation animation)` function. There are also functions to:

- pause animation - `PauseAnimation()`,
- stop animation - `StopAnimation()` and
- resume current animation - `PlayAnimation()`

It is best to collect animations in a component that supports a given actor and send it to the *Actor Billboard* component as needed. The same as it was used in the *Demo Actor* component (DemoActor.cs script).

COMPONENTS

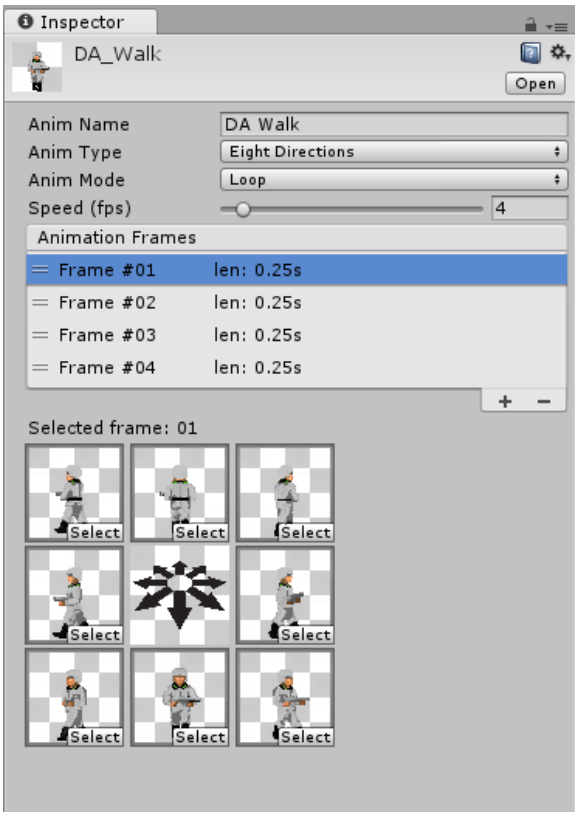
Actor Animation

Animation is created using functions from the [Assets / Create / EDSS / Actor Animation] menu. You can also use the function from the context menu (right mouse button) in the project window. Animation data is saved in the project - it is worth creating a separate folder for this purpose.

Animation is used by the *Actor Billboard* component to calculate the sprite to be displayed for a given actor at a given moment.

It's best to create a *MonoBehaviour* script for the actor with *ActorAnimation* fields. Then assign actor's animations from the project to these fields. The script should set the appropriate animation in the *Actor Billboard* component. Look at *DemoActor.cs* script for reference.

In the case of a single (static) sprite, create an animation containing only one frame.

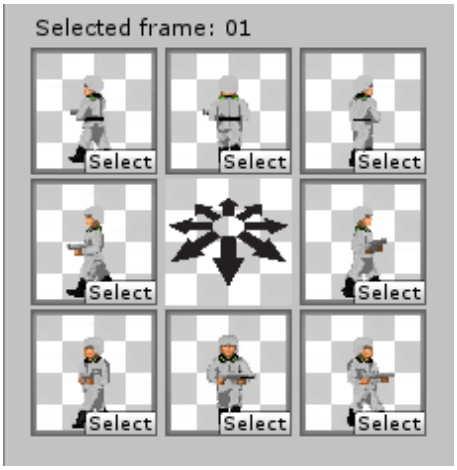


Property	Function
Anim Name	Name of the animation
Anim Type	Type of the animation
One Direction	1-directional animation. Each frame contains one picture representing the actor, which is always displayed regardless of which side the camera is looking at.
Eight Directions	8-directional animation. Each frame contains eight pictures representing the actor - one for each of the eight directions of the world. *Actor Billboard* komponent, playing the animation, calculates the appropriate picture for display based on the direction of the actor and the camera.
Anim Mode	Mode of the animation (behavior when it's finished)
Once	The animation will be played exactly once. After completion, the actor will be represented by the last frame of the animation.
Loop	The animation will be played all the time in the loop. After the last frame, the animation will be played back again from first frame and so on.
Ping Pong	The animation will be played all the time in the loop. After the last frame, the animation will be played back to the first frame. Then, after the first frame is displayed, the animation will be played back again and so on.
Speed (fps)	How fast the animation is playing (in frames per second)

Animation Frames – section where you can create frames for animation. Frames are numbered and can be rearranged by dragging icons on the left side of the list. Elements of the list can be added using button with plus sign and selected element of list can be deleted using button with minus sign. Each frame on this list show its number and calculated duration (in seconds). Currently there is no functionality to create frames with different durations.

When you select a frame in the list, its contents appear underneath. For 1-directional animation, this is one field to indicate the sprite, while for 8-way animation it will be 8 fields to indicate the sprites for each direction.

In the case of 8-directional animations, it is very important to place the sprites in the appropriate fields. The lower middle field should contain the image of the actor from the front. Then, moving through the fields in a clockwise direction, place subsequent images of the actor turning to his right. In the fields on your left side there should be pictures of the actor looking in your left side. In the upper field there should be a picture of an actor standing back to you. Finally, in the fields on your right side should be pictures of the actor looking in your right side.



At the very bottom of the inspector there is a window with a preview of the animation. In the upper part of the window there are small buttons that allow you to play animation and - in the case of 8-directional animations - rotate the actor. Thanks to this, you can view your animation without having to create the actor in the scene.

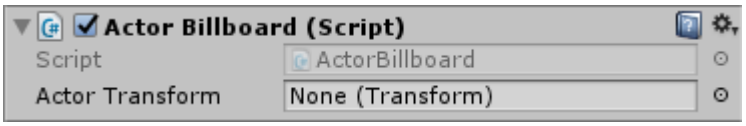
Actor Billboard

Provides two functionalities.

First, it orientates the object to which it is attached towards the camera through which this object is currently being rendered.

Second, it calculates and sets the right sprite in *Sprite Renderer* depending on the actor's orientation, the direction of the camera and the current frame of the animation.

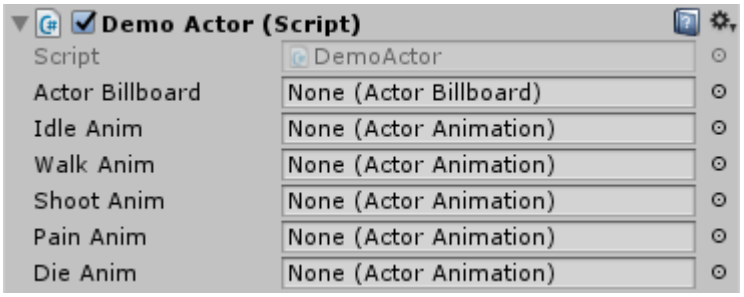
Requires a *Sprite Renderer* component that will automatically be included if it is not on the object when the *Actor Billboard* component is attached.



Property	Function
Actor Transform	An object whose transformation will determine the direction of the actor in the world. If this field is empty, in order to correctly calculate the right sprite for display, the current vector of the actor's direction must be passed to the component using the `SetActorForwardVector()` function. If this field is filled, then the component itself will read the current direction of the actor from the indicated object.

Demo Actor

Implementation of a simple actor for demonstration purposes. When enabled, sets the initial animation (Idle Anim) in the *Actor Billboard* component. By pressing the spacebar you can change the animation to the next one from the list.



Property	Function
Actor Billboard	Reference to *Actor Billboard* component for this actor. If set, the script will automatically transfer data about the current direction of the actor (the object to which it is assigned) to the referenced *Actor Billboard component*. If it is not set, the direction of the actor is not sent to the *Actor Billboard*. In this case, the *Actor Transform* field should be set in the *Actor Billboard* component indicating the object with the actor's direction.
Idle Anim	Reference to the actor's idle animation.
Walk Anim	Reference to the actor's walk animation.
Shoot Anim	Reference to the actor's shoot animation.
Pain Anim	Reference to the actor's pain animation.
Die Anim	Reference to the actor's die animation.

Healthbar Games
contact@healthbargames.pl
www.healthbargames.pl
www.twitter.com/HealthbarGames
www.facebook.com/HealthbarGames
www.youtube.com/channel/UCiCdifoEyr7FVLtELkuM_Qw