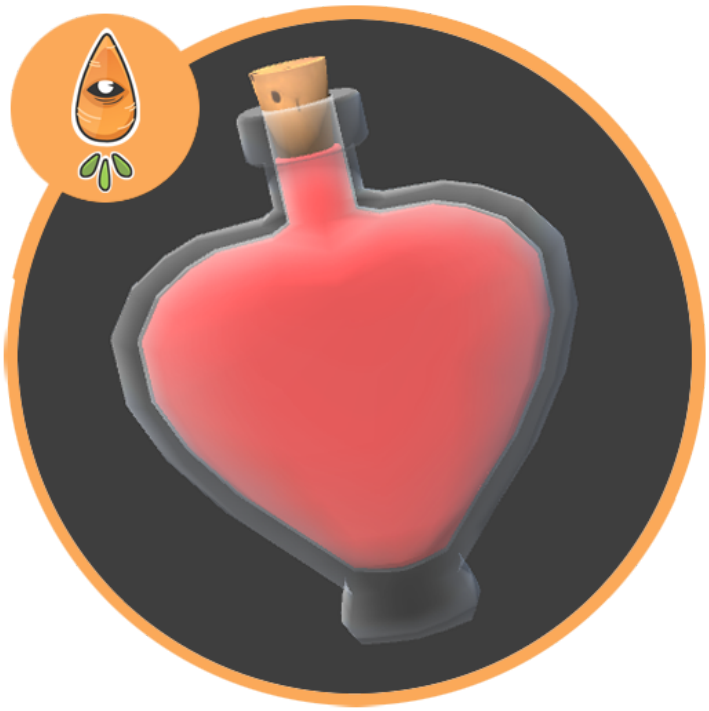


Magic Potions with Liquid Simulation

V1.1 - July 2017

Index

Magic Potions - Overview

This asset pack has been developed to allow you to easily integrate potions with liquid simulation, shattering, pouring and animated surface textures into your next Unity game project. Taking the prefabs we have developed you will be able to quickly drag and drop these potions into your project as environmental props, or interactable items for players to collect.

Secondly, this pack is designed to allow you to easily create your own potions making use of the shaders and scripts we have developed.

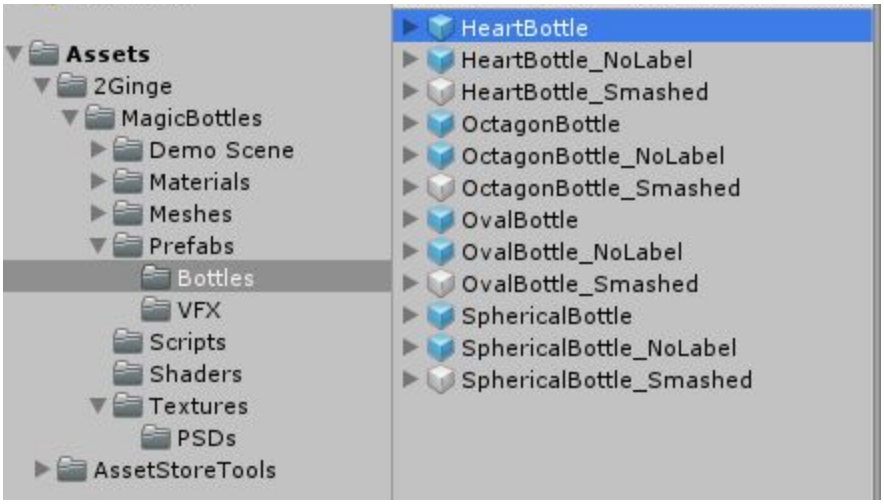
Included in v0.1 of this pack is:

- A tall ‘oval’ potion bottle with liquid mesh simulation, shattering, pouring, a cork and a label
- A ‘heart’ shaped potion bottle with liquid mesh simulation, shattering, pouring, a cork and a label
- An octagonal bottle with liquid mesh liquid simulation, shattering, pouring, a cork and a label
- A spherical bottle with liquid mesh simulation, shattering, pouring, a cork and a label
- A short rectangle bottle with mesh liquid simulation, shattering, pouring, a cork and a label
- A wide cylinder bottle with mesh liquid simulation, shattering, pouring, a cork and a label
- An ink bottle with mesh liquid simulation, shattering, pouring, a cork and a label
- Easy to use liquid shader - allows users to define liquid fill level and draws a ‘fill face’ where the liquid mesh is culled - possesses a texture and animated texture channel to add detail to liquid surfaces as well as emission, smoothness and metallic sliders and colour pickers
- A glass shader with metallic, gloss, texture and normal channels to allow for unique bottle designs
- A ‘liquid volume animator’ script which allows users to simulate approximate liquid physics and define the behavior of the liquid within the predefined parameters
- A bottle smash script that allows users to manage a bottle smash event, determining which particle effects play at the time of breakage and how the liquid puddle renders on the ground plane below the smash location
- A liquid color script that allows the users to have their bottles automatically inherit liquid color blending when liquids are poured into one another, or inherit an overall liquid color set by the user (updates particles, liquid and liquid surface to match one another)

How to use Magic Potions - first time users

Importing pre made bottles

First time users can expect to find prefab bottles within the 2Ginge folder following import from the Unity Asset Store. These bottles are already set up to break when the change in velocity of the bottle object is greater than the predefined break velocity and simulate stylised liquid physics when in an unbroken state. Simply drag these prefabs into your scene and you're ready to go!



If you are creating your own bottles you will require five components to mimic the effects and functionality of our pre made assets. You will require a bottle mesh, a liquid mesh, a cork mesh a label mesh and finally a series shattered glass objects. You can create these shattered segments manually, or perhaps look at implementing a mesh fracturing tool to do this for you.

Once you have the objects described above you should import them into your project and set them up in the scene using the hierarchy shown to the left. Ensure the smashed particles are turned off by default as the bottle smash script will enable these when the break conditions are met. Use the 'glass_potion' shader for the bottle mesh and the 'Liquid_Potion' shader for the liquid. Both of these can be found in the shader drop down under 2Ginge > Potion.

Most of you will be aware, but it bears mentioning that it is best practice to center your parent objects in world space while organising the initial hierarchy to ensure all objects are aligned correctly with one another.

At this point ensure that the necessary objects have rigidbodies and colliders set up. The bottle parent object should have a rigidbody attached, the bottle object should have a mesh collider and the smashed bottle parts and cork should have their own rigidbodies and mesh colliders in order to resolve their own physics after they are turned on. The liquid object and label do not require any colliders or rigidbodies as they will despawn on the smash event.

If you are generating your own smash particle effects, also include them in the setup described above underneath the bottle parent object, positioned correctly. The particle system will be activated by the bottle smash script and should remain turned on by default.

Once you have set up your objects in a hierarchy similar to that outlined above you are ready to attach the provided scripts and shaders to ensure correct functionality. The 'BottleSmash.cs' script should be attached to the bottle parent object and the 'LiquidVolumeAnimator.cs' script should be attached to the liquid object within your bottle.

Script Overview

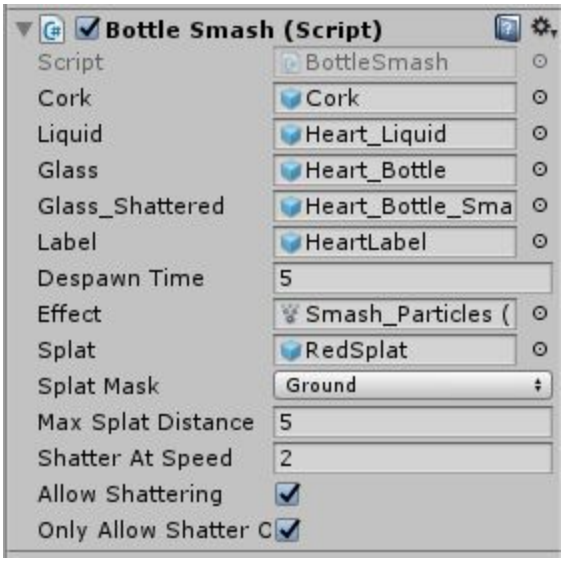
Described below are the functions of the scripts within the asset pack. If you are unclear on the functionality of these scripts and their many components please do not hesitate to contact us for assistance. You can find our support contact details at the bottom of this document.

BottleSmash.cs

'Cork', 'Liquid', 'Glass' and 'Label' are all gameobjects to be destroyed. Cork is unique in that it will despawn after the despawn timer has completed, so be sure to have a disabled mesh collider with a rigidbody (with kinematic ticked) on it to take full effect.

Glass_Shattered is a disabled container full of the glass shards (each with their own rigidbody and mesh collider).

Despawn Time is how long parts of the mesh will stay around for (including the shattered glass).



Effect is the particle effect at the center of the bottle (splash effect essentially).
Splat is the mesh that gets instantiated on the ground.

Mask is the layer that the 'ground' exists on.

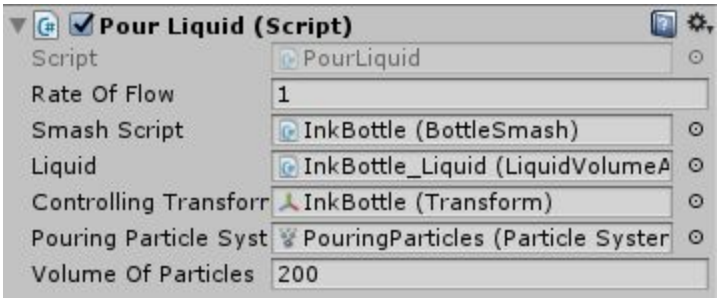
Splat Distance means that any ground that is greater distance than this, the splat will not occur .

Shatter At Speed is the speed at which the potion needs to be travelling in (subtracting its previous velocity) in order to 'break', think stopping very quickly or starting very quickly.

Allow Shattering will disable any normal shattering logic (however it can still be shattered through code.

Only AllowShatter On Collision means that there is a small window after colliding with another collider that the potion can shatter (0.2 seconds).

PourLiquid.cs



Rate of flow: Multiplier of particles per second. (Particle emission base rate set in the 'Emission' tab of the 'PouringParticles' particle system.)

Smash Script: A reference to the BottleSmash.cs script attached to the parent bottle object.

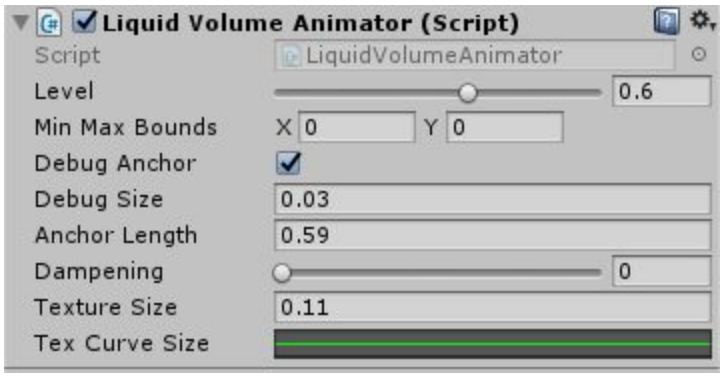
Liquid A reference to the LiquidVolumeAnimator.cs script attached to the liquid object of the bottle being set up.

Controlling Transform A reference to the parent object transform.

Pouring Particle System Reference to the particles that the bottle can emit. (The particle system that is used to make the bottle look as if it is pouring out its contents).

Volume of Particles How many particles the bottle 'consists' of, or the total 'amount' of particles that can be emitted relative to the liquid mass.

LiquidVolumeAnimator.cs



Level is the slider which fills/empties the mesh (0, being empty, 1 being full).

Min Max Bounds used to debug if the volume is approximated correctly.

Debug Anchor allows you to see the physicality of the liquid as a pendulum object.

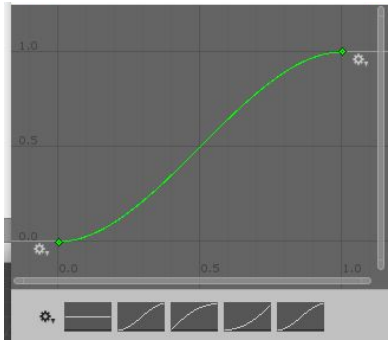
Debug Size allows the user to scale the debug handle objects while editing.

Anchor Length will increase and decrease the effect of rotation and movement on the liquid volume. Closer to '0' the bottle 'physics' will be more erratic and appear like a thinner liquid. A longer pendulum will cause the liquid to appear thicker and behave in a more static manner.

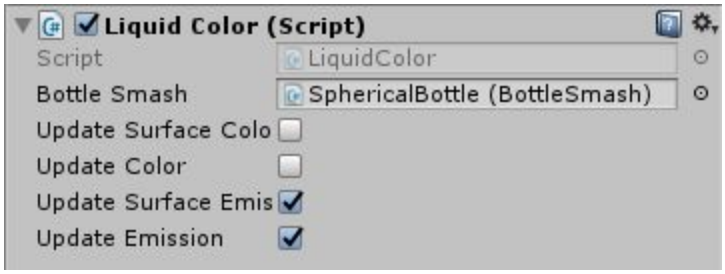
Dampening allows the user to set the liquid volume movement “speed” during simulation numerically. This will net you similar effects to the anchor length scaling.

Texture Size sets the overall texture scale on the liquid volume.

Tex Curve Size is an editable graph that users can manipulate (add keys, move keys, edit curves) to ensure that the texture visualises correctly across the entire surface fill level (0,1). For example a linear curve will not suit a spherical bottle liquid as the surface size starts small when empty, increases towards the middle and then becomes smaller again as the bottle nears full. Some default curves are available, as is the ability to save your own curves once you have made edits.



LiquidColor.cs



BottleSmash is a reference to the parent object to centralize the color of the liquid

Update Surface Color
Turns the surface color update feature on or off. When on, it inherits the color values from the liquid or liquids that are poured into the bottle, unless the bottle already has a liquid volume level, in which case it blends between the added liquid and the current liquid.

Update Color
Turns the liquid color update on or off.

Update Surface Emission
Turns the surface emission value update on or off.

Update Emission
Turns the liquid emission value update on or off.

PS_Color.cs / ParticleColor.cs

Bottle Smash is a reference to the Bottle Smash Script to tie the particle system color to the unified bottle color.

LiquidAbsorbtion.cs:

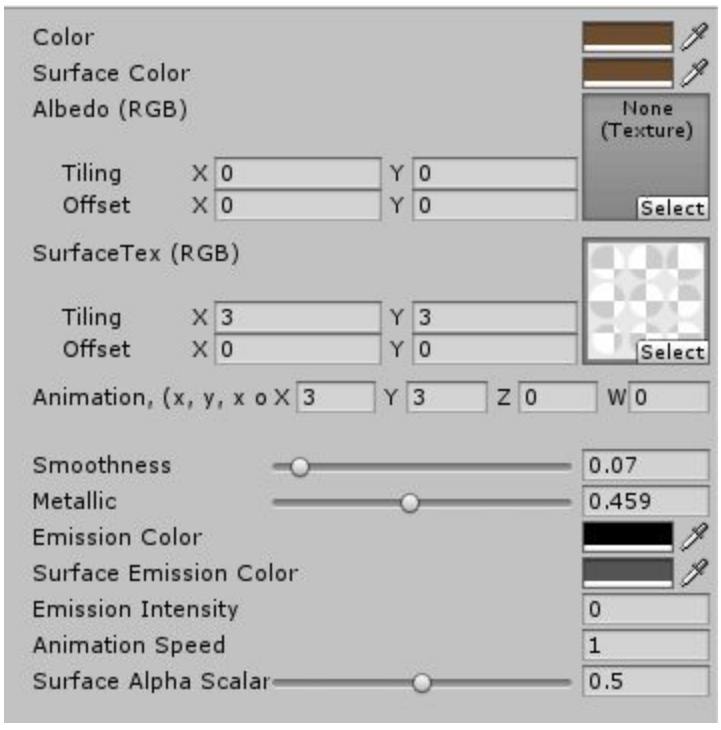
Current Color shows the current color of the unified bottle.

Particle Value is the amount that each particle that collides with the ‘absorber’ collider transfers to the level of the bottle (and the color based on the ratio of what's in the bottle).

LVA is reference to the LiquidVolumeAnimator.cs script.

Shader Overview

Liquid_Potion_Texture



Color sets the liquid body color

Surface Color sets the liquid surface color

Albedo (RGB) is a texture space for the liquid body

SurfaceTex (RGB) is the surface texture space

Animation x,y indicates how many ‘cells’ there are in the sprite sheet provided. For example ‘3,3’ will tell the system there are 9 cells to move through during the animation.

Animation z,w sets the UV space scrolling

Smoothness / Metallic allows the user to set the smoothness and metallic values for the liquid object

Emission Color/ Surface Emission Color lets the user set the emission color for the body and surface of the liquid respectively.

Emission Intensity increases the emission value past 1

Animation Speed sets the speed at which the shader cycles through the animation. This time value is not synced with the engine's overall time value.

Surface Alpha Scalar affords the user the ability to fade out the sprite/ texture on the surface of the liquid.

Liquid_Potion

Liquid_Potion.Shader is simply a liquid shader minus the surface texture components. Useful for objects that do not require a textured surface.

Glass_Potion

Glass_Potion is simply a shader that will allow you to set up stylised glass, control opacity, emission and textures for any glass object.

Additional Help/ Contact

If you are having any issues in setting up your own bottles with the provided scripts and shaders, please watch the video demonstration which can be found on our Youtube channel. There you will also be able to find a video detailing the modelling practices put into place when creating the meshes required to create your own bottle assets.

Feel free to contact us with any issues you may be having via any channel. We are always happy to support our customers and will address bug fixes as soon as possible. Please do not hesitate to contact us with feature requests either! We'd love to continue to make our tools and assets better wherever possible.

Email: contact@2ginge.com
Website: www.2ginge.com
Twitter: @TwoGinge | @PezzSp | @JairMcBain

If you'd like to hear about our other projects and tools, please find our newsletter signup form at www.2ginge.com or check out our Unity Asset Store developer profile.