Dynamic Water Physics

For Unity3D

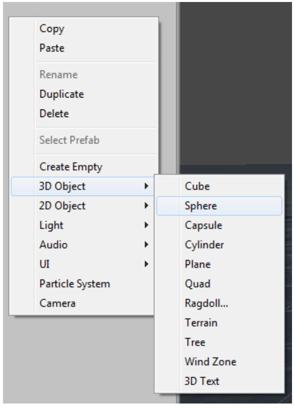
Manual for Version 1.0

NWH
::coding

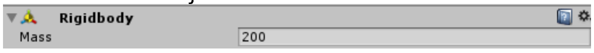
CONTENTS

QUICK START

1) Drag „DynamicWaterPhysics/Prefabs/Water/Water“ prefab into the scene.
2) Create a new 3D object (e.g. sphere).



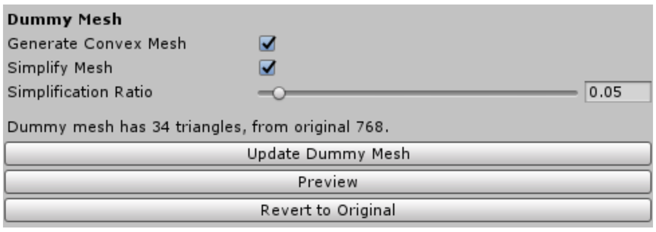
3) Add Rigidbody component to the object and set the mass to 200.


Rigidbody

Mass 200

4) Add FloatingObject component (or drag-and-drop it onto the sphere from „DynamicWaterPhysics/Scripts“)
5) Press play. Your object will now float.

[Recommended] In the FloatingObject inspector check the „Generate Convex Mesh“ and „Simplify Mesh“ options, set the simplification ratio to 0.05 and click „Update Dummy Mesh“ button. This will increase performance greatly. For simple objects 15-30 triangles is enough for simulation, while for ships and more complex objects 40-100 is adequate. Calculation is O(n), where n is number of triangles – so performance is proportional to number of triangles of the dummy mesh.


Dummy Mesh

Generate Convex Mesh ☑

Simplify Mesh ☑

Simplification Ratio 0.05

Dummy mesh has 34 triangles, from original 768.

[Optional] Add „DragObject“ script to the camera so you can drag the object around.

ADDING A NEW WATER SYSTEM

If there are no objects set up as water in the scene, any object that has FloatingObject script attached will assume water is at height 0.

If there are multiple object with „Water“ tag in the scene the nearest one will be used.

FLAT WATER SYSTEM

Any flat water system will work with this asset, including the water from Unity Standard Asset.

1) Drag your water prefab/object/model onto the scene.
2) Add „Water“ tag. FloatingObject will use the nearest object with „Water“ tag set when the scene loads.

WATER SYSTEM WITH WAVES

To work with this asset, non-flat water system must be able to provide water height y at world location x, z.

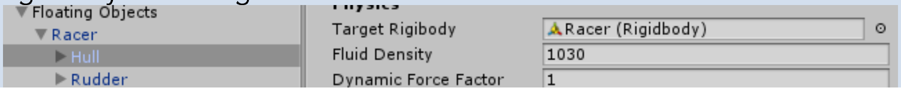
First drag the water object onto the scene and attach „Water“ tag. Next, add script „WaterInterface.cs“ to the water object. This script contains a water height function. When searching for the nearest water FloatingObject script will check if „WaterInterface“ component is assigned. If it is, FloatingObject will use the function inside the „WaterInterface.cs“ script to get the water height – example is provided inside the WaterInterface script. You will neeed to replace the example function with your water's.

FloatingObject has a delegate „WaterHeightFunction“ that is null by default. If this function is set instead of directly calculating water height from water transform's position, floating object will get the water height from that function. Delegate „WaterHeightFunction“ accepts two floats, x and z, as parameters and return float. Example is given in „WaterInterface.cs“ script.

FLOATING OBJECT.CS

FloatingObject.cs is the main script of this asset and contains most of the simulation code, making the object float – as long as that object has MeshRenderer and MeshFilter.

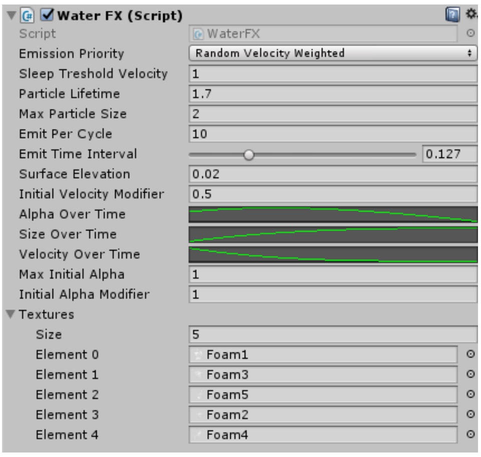
Skipped next to the Physics means that the force calculation for that frame was skipped due to the object being still and so forces from the previous frame were used.

Target Rigidbody	A rigidbody to which forces will be applied. Rigidbody can be from the same object the script is on in which case script will auto-detect it, but can also be on one of the parent objects. Must be set to self or parent rigidbody (not child). Example from the demo scene for the „Racer" boat, hull has Racer's (parents) rigidbody set as target.  Floating Objects Racer Hull Rudder Target Rigibody: Racer (Rigidbody) Fluid Density: 1030 Dynamic Force Factor: 1
Material Density	If the object that has Floating Object also contains a rigidboy, setting this value to other than 0 will set that rigidbody's mass to the value calculated from the set density and volume of the mesh. Only available from editor. Also check out MassFromChildren.cs documentation for cases where FloatingObject and Rigidbody are not attached to the same object.
Fluid Density	Determines intensity of buoyant force on the object. If a cube that is 1x1x1 (1m^3) is created weighting less than fluid density it will float, otherwise it will sink. **It is assumed that one Unity unit represents one meter in all the calculations.**
Dynamic Force Factor	Multiplier for the hydrodynamic forces acting upon the floating body.
Convexify Mesh	Generates mesh where all the surfaces are convex. Can help with polygon reduction. Do not use for shapes where holes might be of importance, such as catamarans. If the mesh is missing triangles in one of its parts, convexify will fill in the empty spaces and make it work properly with the script. This can happen with hulls which are empty on the top (invisible) side.
Simplify Mesh	Generates a simplified mesh with a number of triangles set by „Simplification Ratio". This is an **important step as it increases performance.** Can be combined with [GenerateConvexMesh]
Simplification Ratio	Percentage of the original triangles that will be present in the dummy mesh. Set the simplification ratio as low as possible while perserving important features of the mesh.
Update Dummy Mesh	Runs the simplification or convex mesh generation algorithms on the mesh. Can take some time with exceedingly large meshes. For meshes above 30k triangles use of external tools such as Blender (free) to generate a dummy mesh might be faster: https://docs.blender.org/manual/ja/dev/modeling/modifiers/generate/decimate.html **If you change the original mesh, do not forget to update dummy mesh.**
Preview	Shows / hides the dummy mesh used for simulation.
Debug	Draws the gizmos representing forces, triangles and other variables.
Reuse Forces	If set to true and object is still enough, forces from the last frame will be reused instead calculating new ones.
Reuse Position Treshold	Position change of floating object needed for forces to be recalculated.
Reuse Angle Treshold	Angle change of floating object needed for froces to be recalculated.

WATERFX.CS

WaterFX.cs script only works with flat water surfaces. It works by emitting 2D water particles just above the surface of the water based on the parameters of the simulation from the FlloatingObject. Object it is attached to has to have FloatingObject.cs script as it depends on it. Information contained in FloatingObject.waterLines that is generated during simulation steps is used.

[IMPORTANT] For particles to appear properly in the build, „WaterFX/WaterParticle" shader from „DynamicWaterPhysics/Scripts/Effects" needs to be added under „Edit>ProjectSettings>Graphics", to the „Always Included Shaders" field.



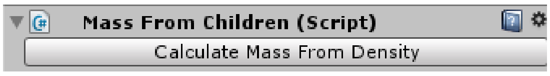
☑ Water FX (Script)

Script	WaterFX
Emission Priority	Random Velocity Weighted
Sleep Treshold Velocity	1
Particle Lifetime	1.7
Max Particle Size	2
Emit Per Cycle	10
Emit Time Interval	0.127
Surface Elevation	0.02
Initial Velocity Modifier	0.5
Alpha Over Time	
Size Over Time	
Velocity Over Time	
Max Initial Alpha	1
Initial Alpha Modifier	1
Textures	
Size	5
Element 0	Foam1
Element 1	Foam3
Element 2	Foam5
Element 3	Foam2
Element 4	Foam4

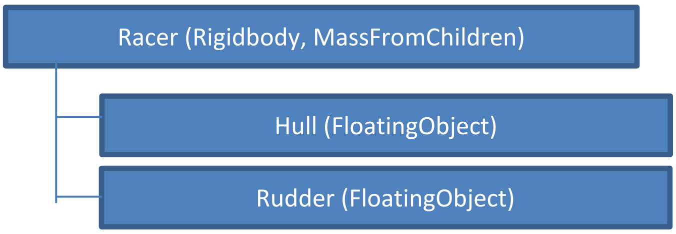
Emission Priority	Determines which points of the water line will emit water particles: Random – Chooses [EmitPerCycle] points randomly from the water line. Highest Velocity – Chooses top [EmitPerCycle] points from the list of water lines sorted by highest velocity. Highest Force – Chooses top [EmitPerCycle] points from the list of water lines sorted by highest force. Similar to highest velocity but surface of the triangle related to the water line is taken into consideration. Random Velocity Weighted – Selects one quater of points from water lines sorted by velocity and selects [EmitPerCycle]. This makes water effects more random when using small [EmitPerCycle] number. Random Force Weighted – Selects one quater of points from water lines sorted by force and selects [EmitPerCycle].
Sleep Treshold Velocity	Velocity object has to have to emit particles. If set to a too small value it will cause objects to constantly emit which will result in a major slowdown.
Particle Lifetime	Time in seconds after which the particle will be destroyed.
Max Particle Size	Maximum particle size a particle can achieve. Particle size depends on velocity and force of the triangle adjecent to the water line.
Emit Per Cycle	How many particles will be emitted every cycle?
Emit Time Interval	Particles will be emitted every [EmitTimeInterval] seconds.
Surface Elevation	How hight above the water will water particles be emitted? Too low a value will cause particles to flicker or be invisible.
Initial Velocity Modifier	Determines how much velocity of the object will affect initial particle speed.
Max Initial Alpha	Limit initial alpha to this value.
Initial Alpha Modifier	Multiplies initial alpha by this value. Alpha cannot be higher than maxInitialAlpha.
Foam Textures	Foam textures. At leas one has to be assigned. Textures will be picked randomly.

MassFromChildren.cs

Editor script that when attached to the parent object checks self and all the children for the component <FloatingObject> and calculates mass from all the densities. Rigidbody mass will not be changed until the button is pressed. Requires Rigidbody to be attached to the object.



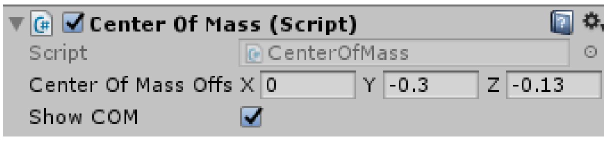
Example hierarchy from the demo scene:



Where both FloatingObjects have Racer's rigidbody as a target rigidbody. FloatingObject's target rigidbody always has to be attached to self or parent rigidbody, never children rigidbody. In the above example both FloatingObjects will affect Racer's rigidbody mass after clicking „Calculate Mass From Density“.

CenterOfMass.cs

A helper script for setting center of mass of a rigidbody. Has to be attached to the object that contains rigidbody. X, Y and Z represent offset from the Unity calculated center of mass.



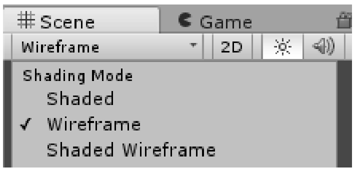
☑ Center Of Mass (Script)

Script CenterOfMass

Center Of Mass Offs X 0 Y -0.3 Z -0.13

Show COM ☑

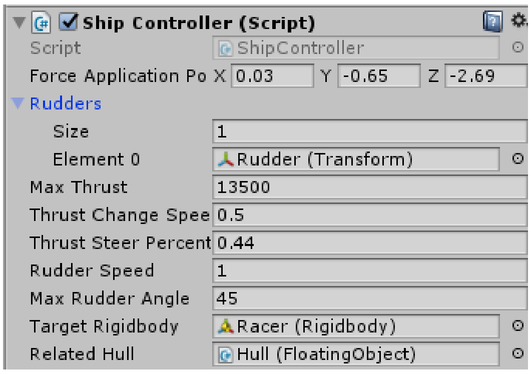
Center of mass will be displayed as a green sphere. To see it more easily select wireframe shading in the editor:



For normal objects such as boxes, barrels, planks, etc. default center of mass provided by Unity will be sufficient, but ships, buoys and similar objects will need center of mass changed (usually lowered).

SHIPCONTROLLER.CS

ShipController is a simple example script for – controlling ships. Needs to be attached to the parent ship object.



☑ Ship Controller (Script)

Script	ShipController
Force Application Po	X 0.03 Y -0.65 Z -2.69
Rudders	
Size	1
Element 0	Rudder (Transform)
Max Thrust	13500
Thrust Change Spee	0.5
Thrust Steer Percent	0.44
Rudder Speed	1
Max Rudder Angle	45
Target Rigidbody	Racer (Rigidbody)
Related Hull	Hull (FloatingObject)

Force Application Point	Offset from the center of the ship's transform at which thrust force will be applied. Represented with a red sphere. Switch editor shading to wireframe to see it more easily.
Rudders	Rudder transforms that have FloatingObject attached to them. Can be simple flattened cube primitives such as in the example. Rudders are actually used to steer so their size affects ship's behavior greatly.
Max Thrust	Max thrust force that can be applied at the [ForceApplicationForce]
Thrust Change Speed	Speed of change of thrust. Larger ships take some time to switch from full forward to reverse and this approximates that.
Thrust Steer Percent	Amount of thrust that will steer the ship. Another component of ship steering is that propeller is throwing water at the rudder which makes ship able to steer even while standing near to still. Ships such as the Racer in the scene actually have propeller mounted on the rudder, so [ThrustSteerPercent] is set to 1 as thrust force will always point in the same direction as rudder.
Rudder Speed	Time it takes for rudder to change from 0 to [MaxRudderAngle]
Max Rudder Angle	Max angle rudder transforms will be rotated to.
Target Rigidbody	Rigidbody of the ship thrust and steer forces will be applied to. Usually the rigidbody and this script will be on the same (parent) object.
Related Hull	Hull's FloatingObject component. Needed to check if the [ForceApplicationPoint] is in water or otherwise no force will be applied.

Setting up a new FloatingObject / Ship (Step-By-Step)

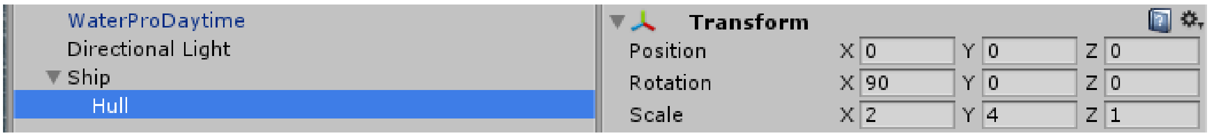
This chapter will explain start-to-finish setup of a Racer boat as seen in the demo scene.

Water

1) Create new scene
2) Add „DynamicWaterPhysics/Libs/Water/Prefabs/WaterProDaytime.prefab“ to the scene. This is water from Unity's standard assets and you can also import it from there.
3) Set water transform's position to e.g. (0,0,0) and scale to e.g. (1000,1,1000)
4) Change tag to „Water“. If there is no Water tag you will have to add it by clicking on the dropdown menu next to „Tag“ at the top of the inspector and selecting „Add Tag…“

Floating Object

1) Create a new empty game object named „Ship“ that will contain all the ship parts. Set its position to (0,0,0).
2) Add Rigidbody and ShipController components to the „Ship“ object. Set mass of the rigidbody to 2000, drag to 0.05 and angular drag to 0.1. Drag is optional.
3) Right click on ship and create new 3D object: Capsule.
4) Change the scale to (2,4,1) and rotation to (90,0,0) and rename it to „Hull“. Scene should now look like this:

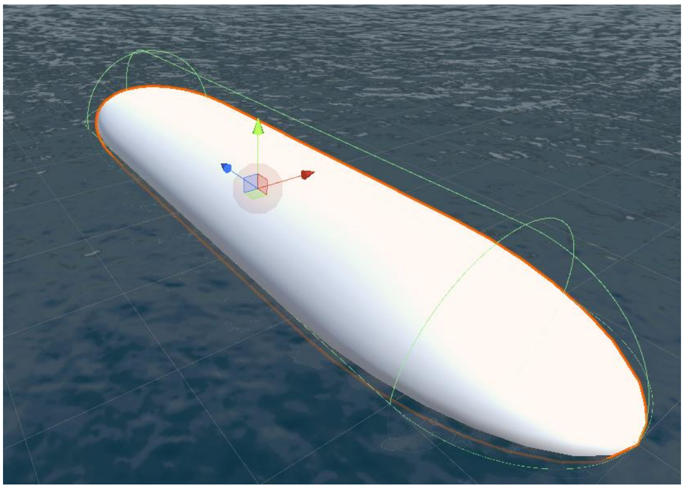


WaterProDaytime
Directional Light
▼ Ship
Hull

▼ Transform

Position	X	0	Y	0	Z	0
Rotation	X	90	Y	0	Z	0
Scale	X	2	Y	4	Z	1

5) Add FloatingObject script to the „Hull“ object.
6) Drag the „Ship“ object onto the „TargetRigidbody“ field. If you press play Ship will now float.
7) [Optional] To optimize performance mesh can be simplified by enabling Simplify Mesh in the FloatingObject inspector. Set the simplification ratio to the wanted percentage (by default 0.1 or 10%) and click „Update Dummy Mesh“. This will reduce the triangle count of the dummy mesh used for simulation. You can preview the generated mesh by clicking „Preview“. It is important that the general shape stays the same, but details are not as important. This step can take some time on very large meshes. You could also use convexify mesh to further reduce poly count on some objects – such as ships that have hull and interior detail as a single object. Convexify will get rid of all the interior triangles since they are not important for the water simulation.



RESULT SO FAR. BLUE ARROW REPRESENT POSITIVE DIRECTION OF Z AXIS AND WILL BE USED AS SHIP'S FORWARD DIRECTION

Additional Components

Now that the hull is floating, it's time to add other a rudder and a keel.

Rudder

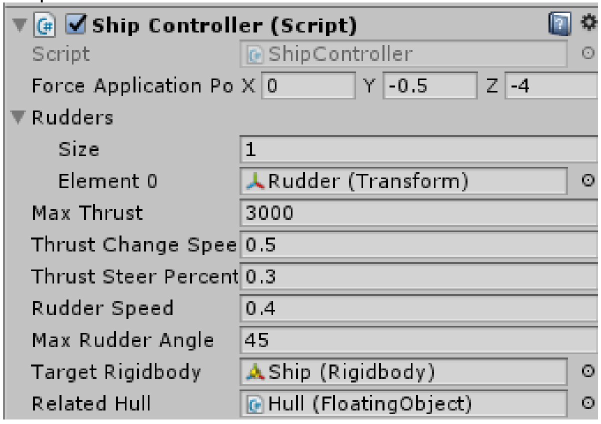
1) Create a new Cube by right clicking on the „Ship“ object and name it „Rudder“. Set scale to (0.05,1,1) and position to (0, 0, -4.5) – or just at the end of the ship.
2) Add FloatingObject component to „Rudder“ and drag „Ship“ to the TargetRigidbody field.

Keel

1) To create a keel, duplicate „Rudder“ object or repeat the steps for creating a rudder. Name it „Keel“
2) Position it to (0, -0.6, 0) – just under the ship – and set the scale to (0.05, 0.3, 6)

Ship Controller

1) Add ShipController script to the „Ship“ object.
2) Set it up as in the picture below and press play. Details about this script are explained in the ShipController.cs section.



☑ Ship Controller (Script)

Script: ShipController

Force Application Po: X 0 Y -0.5 Z -4

Rudders

Size: 1

Element 0: Rudder (Transform)

Max Thrust: 3000

Thrust Change Spee: 0.5

Thrust Steer Percent: 0.3

Rudder Speed: 0.4

Max Rudder Angle: 45

Target Rigidbody: Ship (Rigidbody)

Related Hull: Hull (FloatingObject)

Changing Center of Mass

To change center of mass of the ship add the „CenterOfMass“ script to the „Ship“ object. Set the value to e.g. (0, -0.5, 0). New center of mass will be indicated as a green sphere.

Mass from Density

You might have have noticed that the Material Density field was left at 0, which means it will get ignored, and that the mass in the first step was manually set to 2000. If the object that has a FloatingObject script also has a Rigidbody, setting this value to something other that 0 will change the mass of that Rigidbody according to the object's mesh volume and density. Since our „Hull“, „Rudder“ and „Keel“ objects do not have a rigidbody but rather refer to the „Ship“ rigidbody, MassFromChildren script has to be used.

1) Attach MassFromChildren script to the „Ship“ object.
2) Set hull, rudder and keel densities to 600.
3) Go back to „Ship“ object and click „Calculate Mass From Density“ on the MassFromChildren script.

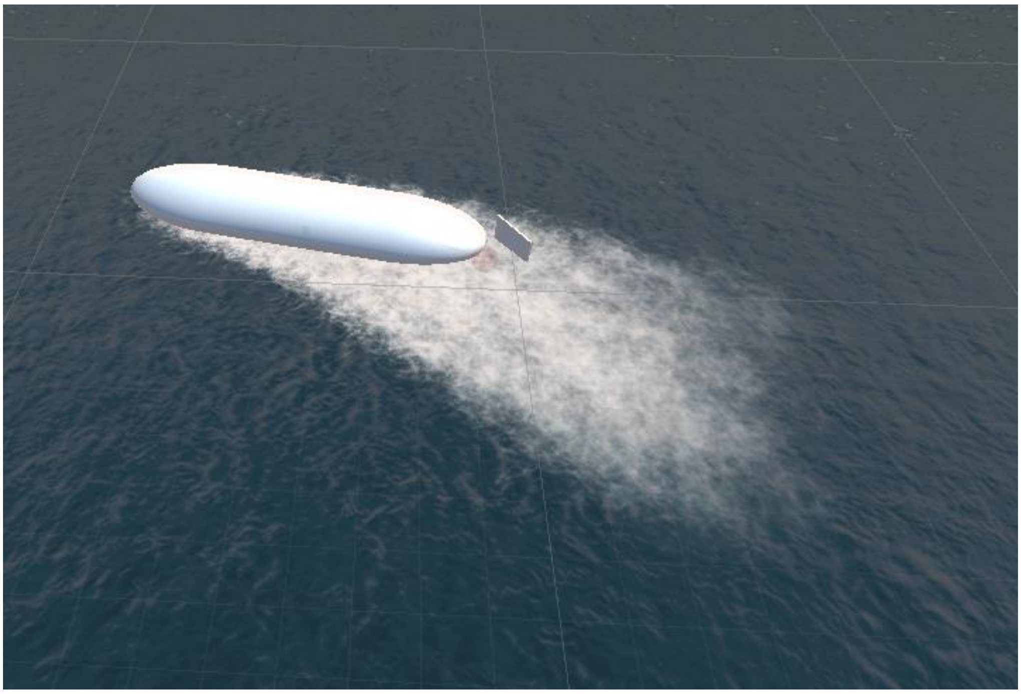
Adding Water Effects

To add water effects to the ship created above, attach WaterFX script to the „Hull“. This component is explained in more detail in section WaterFX.cs.

Since AnimationCurves cannot be serialized they are not set out of the box, but if left empty on entering play mode the script will generate default curves by itself.

The only field that needs to be changed for water effects to work is „Foam Textures“. There needs to be at least one foam texture attached. If there are more textures attached, each particle will have a random texture.

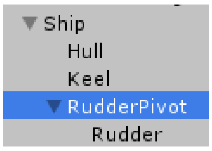
WaterFX script can be attached to any object that has FloatingObject component.



FINAL RESULT

Fixing Rudder Pivot Point

In the image above rudder pivots around its center. To set another pivot point create a new empty game object that is a parent of „Ship“, position it at the position you want the rudder to pivot around and move the pivot to be a child of the newly created game object. Now drag the RudderPivot to the field „Rudders“ of the ShipController to replace the old rudder transform. Example hierarchy:



```
▼ Ship
    Hull
    Keel
  ▼ RudderPivot
      Rudder
```

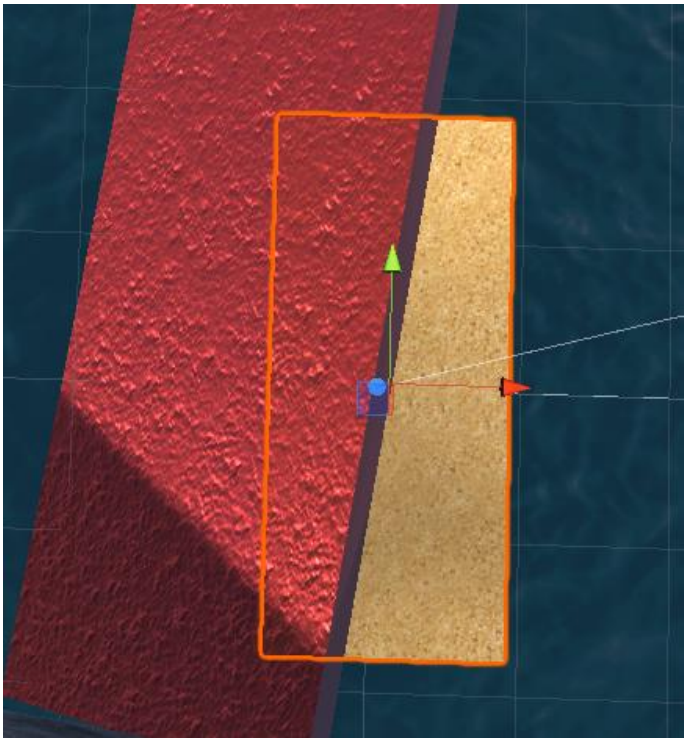
Fixing Model Rotation

Some models have incorrect rotation when imported. FloatingObject script will work with any rotation, but ShipController expects Z to be forward, X to be right and Y to be up. More info on that: https://docs.unity3d.com/Manual/HOWTO-FixZAxisIsUp.html

Masking Water

To prevent water showing through the interior of the boat, use the WaterMask prefab located in „DynamicWaterPhysics/Prefabs/WaterMask“ or create your own by using „DepthMask“ material provided in „DynamicWaterPhysics/Scripts/DephtMash“ and dragging it onto your own quad or plane.

By default, geometry render queue is set to 2000 in Unity. To make water invisible inside the boat set water's render queue to 2200. By placing an object that uses DepthMask material (e.g. WaterMask prefab) over the area in which water should be hidden makes that area render is such order that water will be rendered over by the geometry underneath (in the image below that is the red surface and terrain floor).



TROUBLESHOOTING / FAQ

WATERFX PARTICLES APPEAR AS WHITE TRIANGLES IN BUILD.

Add „WaterParticleShader“ to the Always Included Shaders under „Edit>ProjectSettings>Graphics“.

FLOATING OBJECT SEEMS TO JITTER / BOB UP AND DOWN.

This is due to forces not always being recalculated. Under the Floating Object > Additional Options uncheck „Reuse Forces“ or reduce the treshold values to the smaller number. Also, extremely light object will tend to jump so try setting rigidbody's mass to a higher value.

CHANGE LOG

V1.0 - Initial