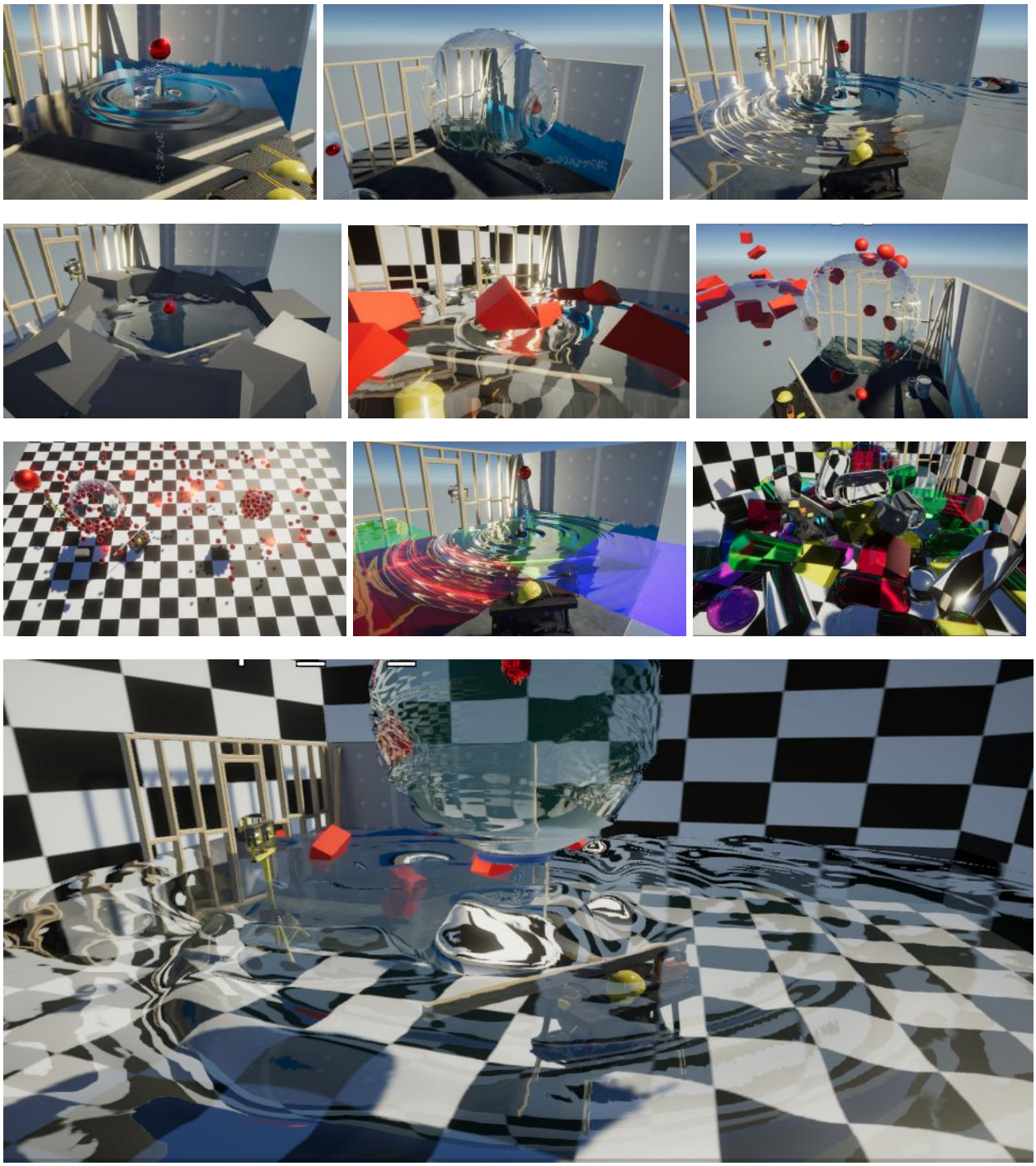
Simplestar-Game

Simple Interactive Water 3.0.2

VRでも破綻しない水の屈折表現と波のシミュレーション



Simple Interactive Water とは？

Unity の Universal RP と HDRP で利用できる、ユーザー操作によって波を発生させられる水面アセットです。コンピュートシェーダーによる軽快な波の計算と、水の屈折表現が **VR** の両眼視で破綻しないことが特徴です。そのほか、水面をタイル配置した場合に波が伝播しますので、タイル配置を工夫すると細長い水路や広い湖面を表現できます。波の障害物情報をマスクテクスチャに描き込むことで、任意の形状で波が反射する水面を作ることができます。また水面と一緒に使用されることが多い、浮力による水面に物が浮くサンプルも同梱しています。タイル配置のアイディアの発展として、6面で構成される球体メッシュオブジェクトと、その球面を波が伝播するサンプルを追加しました。合わせて、球体中心に向かう重力と浮力のサンプルを追加しました。任意の色を指定して、水面に色をつけることもできます。また、水面ではないですが、屈折する色付きガラスマテリアルが含まれています。加えて VR で Humanoid を動かし、コントローラのグリップとトリガーで水面とインタラクションするサンプルを同梱しています。水面の高さはリアルタイムに位置を指定して Script で取得することが可能です。

対象 **Unity** バージョン

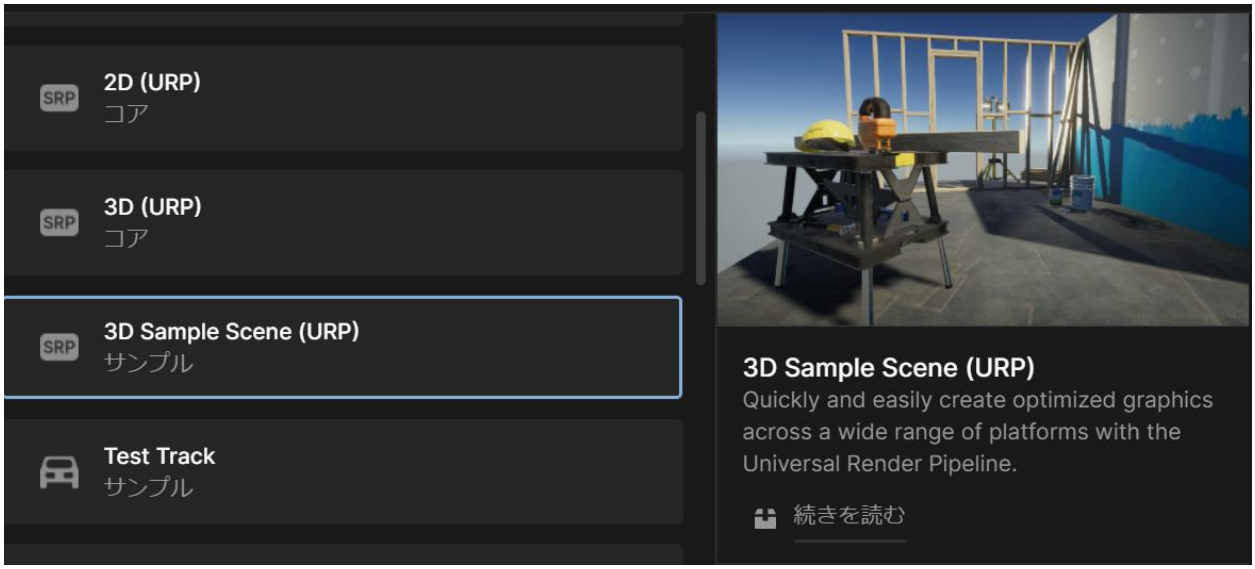
Unity 2021.3.9f1 で開発を行いました。Unity の Shader Graph を使用していますので、過去や未来のバージョンにおいても互換性をもって動作することが期待されます。

利用環境

PC 全般での利用を強く推奨します。GPU によるコンピュートシェーダによって波をシミュレーションしています。 GPU 計算量が大きい環境にて軽快に動作します。モバイル端末向けにビルドすることもできます。ただし、デバイスによって GPU 性能が期待できない場合、PCのように高いフレームレートで動作しません。WebGL は動作しません。これは使用しているコンピュートシェーダが WebGL ではサポートされないためです。

使い方（プロジェクトの初期設定）

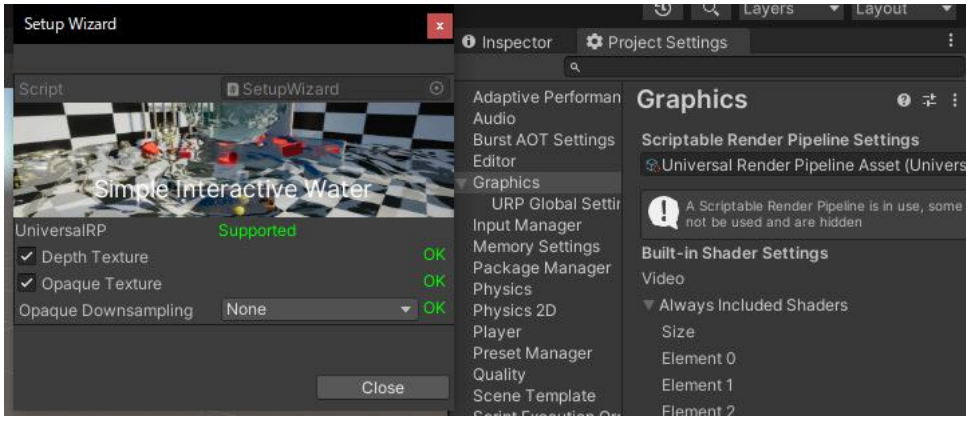
New Project から 3D Sample Scene (URP) を選んでプロジェクトを作成します。



その URP サンプルプロジェクトに本アセットを Package Manager からインポートします。

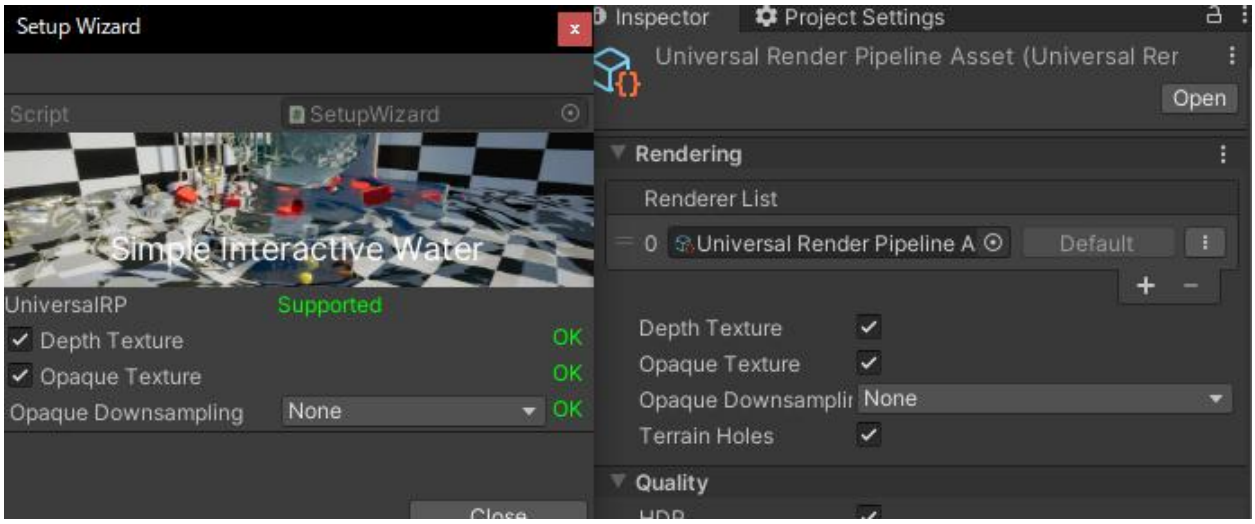
package の依存関係解決のため「This Unity Package has Package Manager dependencies.」ダイアログが表示されますので「Install/Upgrade」ボタンを押してください。パッケージインストールが完了すると新しい Input System を有効化するために Project の再起動が促されます。Project の再起動を行ってください。

プロジェクト再起動後にもう一度アセットのインポートを行います（二度目の再起動はありません）。その後はメニューの Window > Simple Interactive Water > Setup Wizard を選択して、セットアップウィザードを開いてください。



すべての項目が OK に切り替わるように、各項目の Fix Now ボタンを押してください。

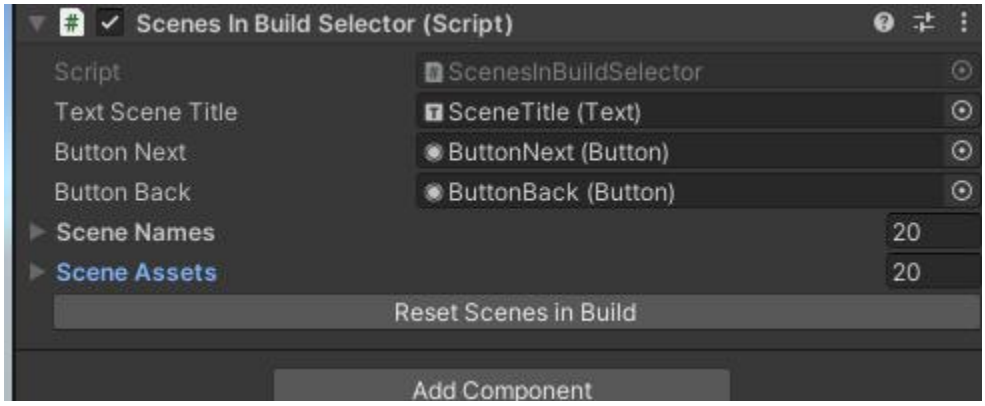
Setup Wizard が編集しているのは Project Settings の Graphics の Universal Render Pipeline Asset 以下、Depth Texture、Opaque Texutre です。



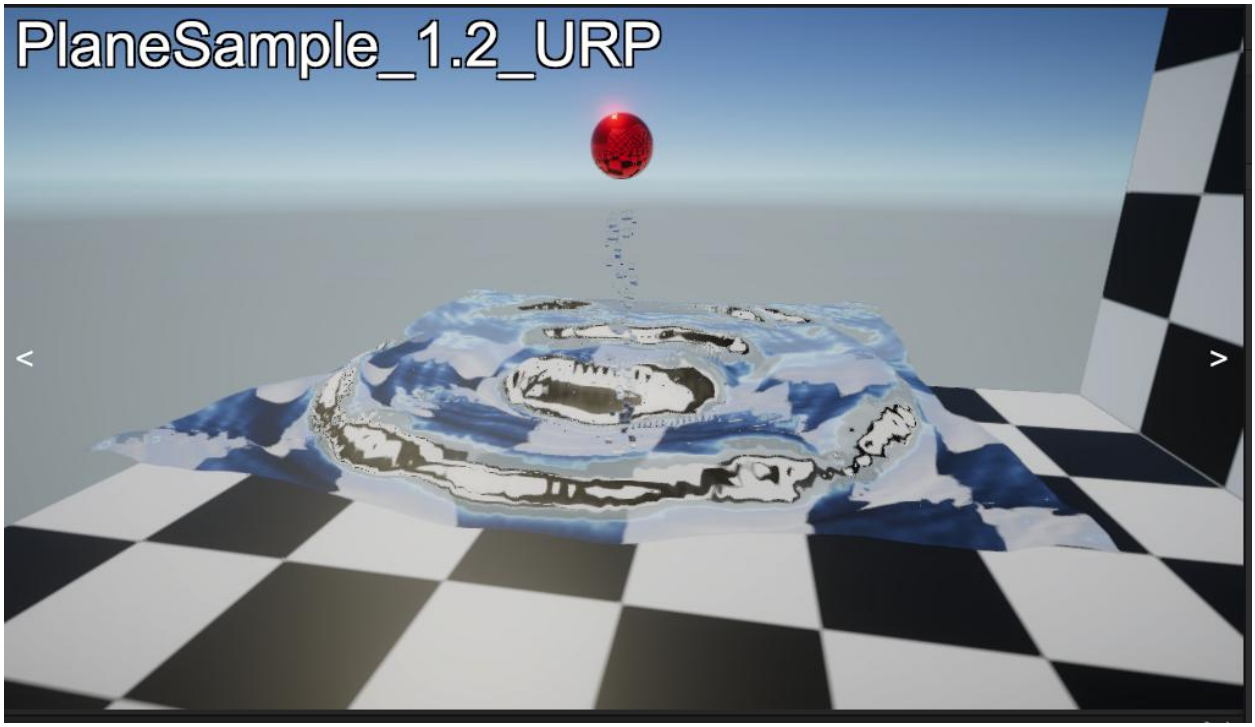
Universal RP Sample Scenes

プロジェクトの準備が整いましたら、サンプルシーン一覧を確認していきます。

Assets/SimplestarGame/SimpleInteractiveWater/Example/Scenes/URP/Sample Scenes for URP.unity を開いて、ヒエラルキーに存在する ScenesInBuildURP オブジェクトのインスペクタの Reset Scenes in Build ボタンを押します。Scene Names が 0 個から 20 個に切り替わったことを確認してください。



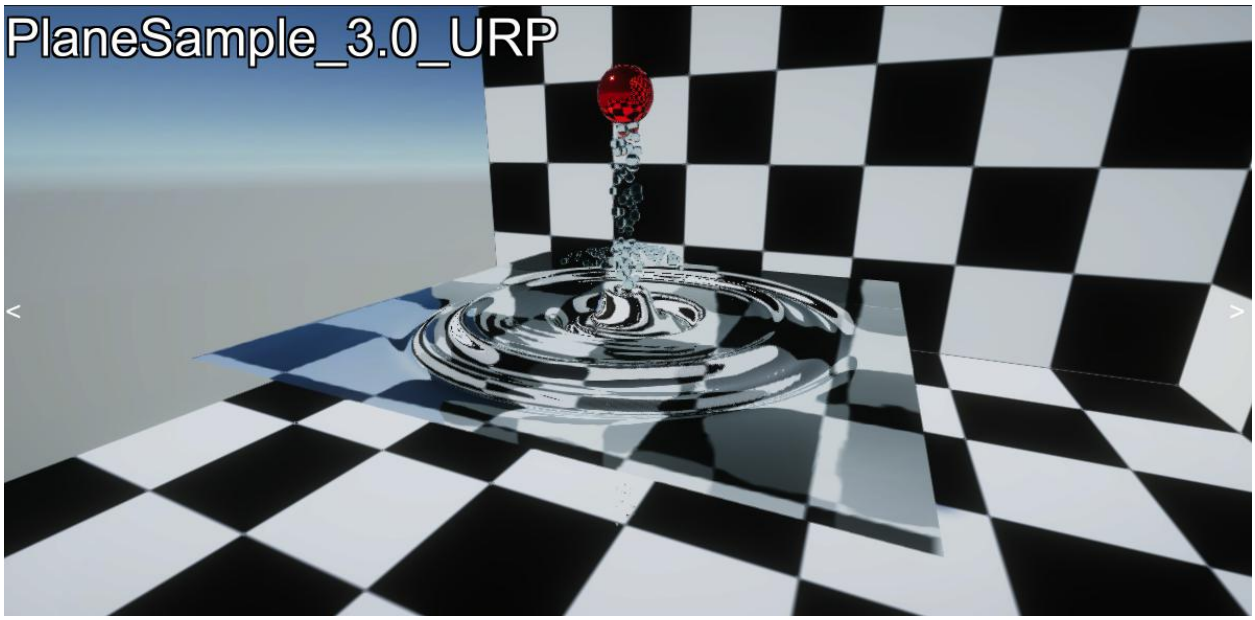
この状態で Play ボタンを押してゲームを再生します。Scene Names 先頭要素の「PlaneSample_1.2_URP」シーンの再生が始まります。



左右のアローキー or 画面内の左右のボタンを押してシーンを切り替えることができます。個別のシーンを確認したい場合は、Assets/SimplestarGame/SimpleInteractiveWater/Example/Scenes/URP 以下の各シーンを個別に開いて、それぞれで Play ボタンを押して動作を確認してください。

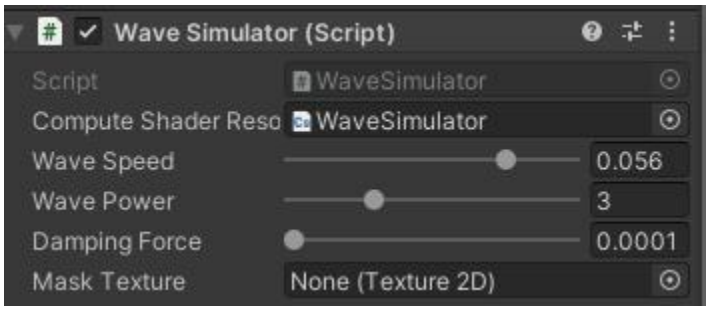
以降は、それぞれのサンプルシーンについて解説していきます。

0_Plane



ボールが水面を通過すると、水しぶきとともに波紋がひろがる本アセットの基本的なサンプルシーンです。過去バージョンの水面マテリアルを含めて、3つのマテリアルを適用したシーンが用意されています。それぞれ水面にアタッチされたマテリアルが異なるだけで、それ以外は同じ構成です。

シーンの構成要素を説明します。ヒエラルキーの QuadPlaneX.X_URP オブジェクトにアタッチされている Wave Simulator スクリプトコンポーネントのインスペクターをご確認ください。



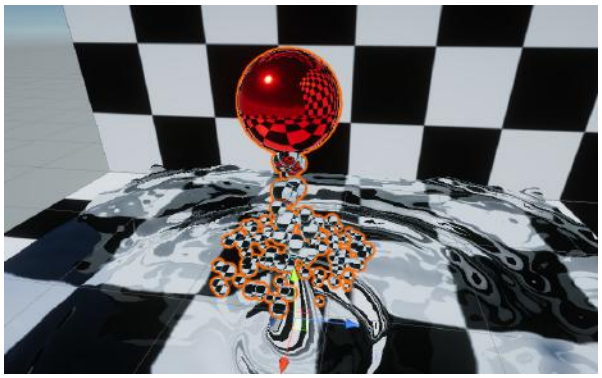
Wave Speed: 波の速さを調整します。(周囲の高さの取得頻度を上回ると波が共振・発散してしまうため、速さには限度があります。ご注意ください)
Wave Power: 発生する波の大きさを調整します。
Damping Force: 波の高さが収束する速度の係数です。小さい値ではなかなか収束しません。
Mask Texture は水面上の障害物で波が跳ね返るために利用する障害物情報です。(後述の

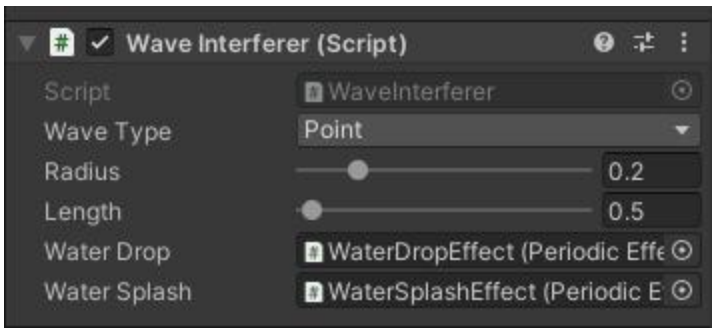
3_Mask シーンで作り方を説明します。

Disable Auto Resolution: ゲームオブジェクトの localScale を大きくした際、自動で水面の解像度が調整されますが、その調整を無効にするオプションです。チェックを入れると、localScale を大きくしても 128 x 128 の解像度で固定されます。

水面オブジェクトは localScale の変更に対応しています。水面をそのまま大きくすると波の速度が速くなりすぎたり、波の解像度が粗く見えてしまうため、自動で localScale に合わせた水の HeightMap テクスチャの解像度を内部にて調整しています。あまり大きくしすぎると高い解像度での処理となり、パフォーマンスが低下します。解像度を固定したい場合は Mask Texture に固定する解像度のテクスチャをアタッチしてください。Mask Texture の解像度を利用して水面が作られるようになります。または、前述の Disable Auto Resolution のチェックを入れてください。

Wave Interferer（波発生ギミック）





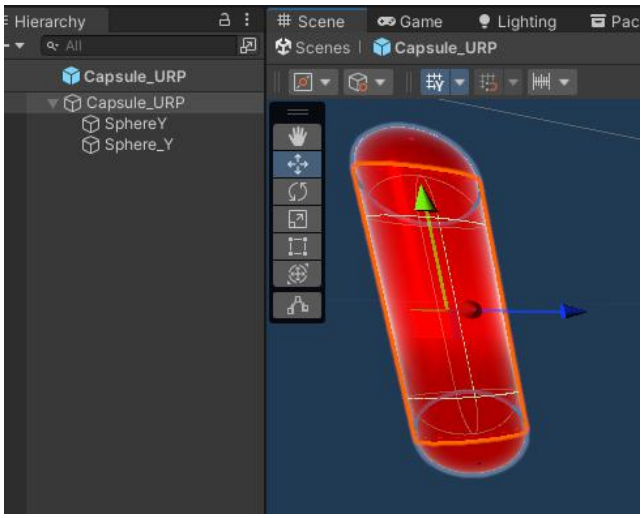
水面にインタラクティブに波を発生させているのが、ボールにアタッチされている Wave Interferer スクリプトコンポーネントです。水面の Collider と接触した際に OnTrigger イベントが発生し、そのときの速度と向きで水面に波を発生させています。

Water Drop: 水がボールから滴るパーティクルエフェクトです。
Water Splash: 水面に水しぶきが立つパーティクルエフェクトです。

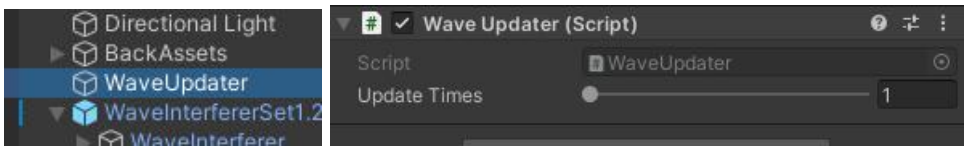
それぞれの参照に None を設定するとこれらのエフェクトを消すことができます。(演出を省略することでパフォーマンスが上がります)

どんなオブジェクトでも Rigidbody を持つオブジェクトに、この Wave Interferer スクリプトコンポーネントをアタッチすれば、水面に波を発生させることができます。その他のパラメータは次の通りです。

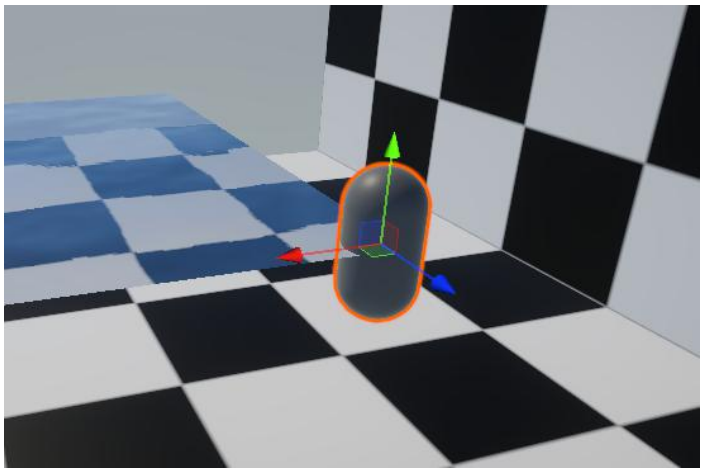
WaveType: Point、Line から選べます。Point は接触点にて波を発生させます。Line は子オブジェクトの SphereY、Sphere_Y の2点を結ぶ線分で波を発生させます。(※子オブジェクトに SphereY, Sphere_Y の名前のオブジェクトが無いと WaveType を Point から Line に切り替えることができないようになっています)
Radius: 水に突入するオブジェクトの半径です。調節するとオブジェクトの localScale も連動して切り替わります。(※逆に localScale の方を調整しないでください。連動を切りたい場合は OnValidate 関数の localScale 適用コードをコメントアウトしてください)
Length: Line Type にて Line の長さを意味します。調節するとオブジェクトの localScale も連動して切り替わります。Line Type は実装例を参考にしてください。
Assets/SimplestarGame/SimpleInteractiveWater/Prefabs/URP/Capsule_URP.prefab を参考に Line Type を利用してください。



さらに、シーンに必須の GameObject を説明します。全ての波の計算を同期させるため、シーンに必ず WaveUpdater オブジェクトを設置する必要があります。すべての Wave Simulator はこの WaveUpdater の更新イベントにて波の計算をしています。この WaterUpdater のインスペクタにて Update Times の値を 1から 2にすると、シーン内での波の計算量を 2倍にして波が伝わる速度が 2倍になります。

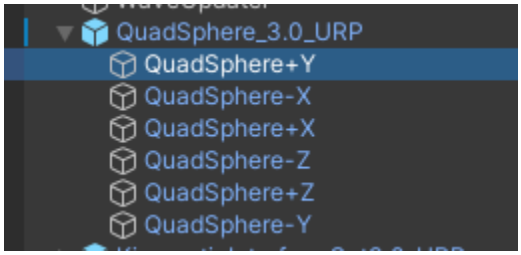


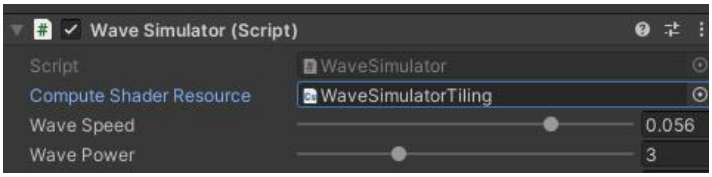
水面の高さをリアルタイムで取得することができます。PlaneSample_3.0_URP に水面に刺さったカプセルがありますが、WaveHeightChecker スクリプトコンポーネントがアタッチされています。スクリプト内にて Raycast Hit point を入力し、高さを取得しています。



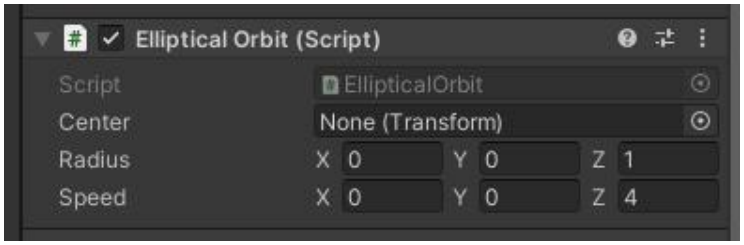
1_Sphere

QuadPlane を 6枚使って、6面体で表現される球体水面です。波は互いに接合する面で伝播し、球面を循環します。WaveSimulator には Tiling 用の WaveSimulator.compute ファイルが指定されています。指定しないと周囲に波が伝播しないので注意してください。



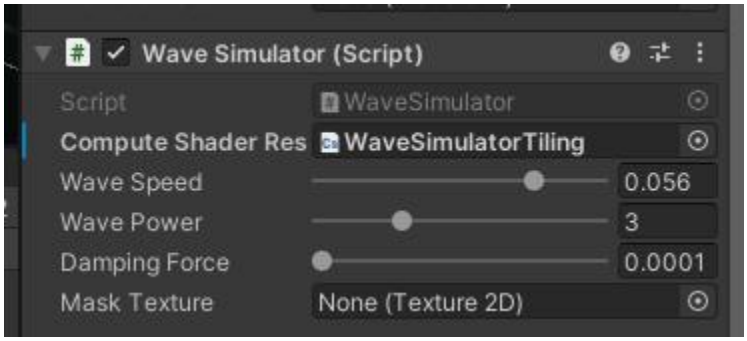


水平方向に振動するボールには楕円運動する Elliptical Orbit スクリプトコンポーネントがアタッチされています。デモ用のギミックです。半径と速度を各軸に指定して軌道を調節してテストにご利用ください。

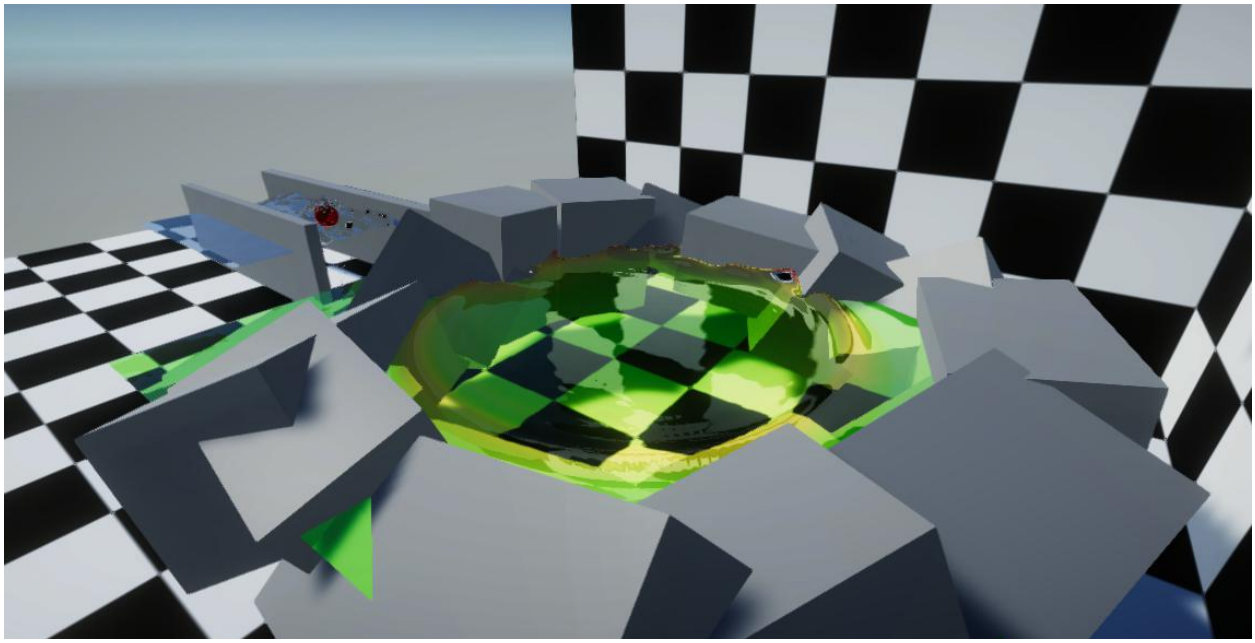


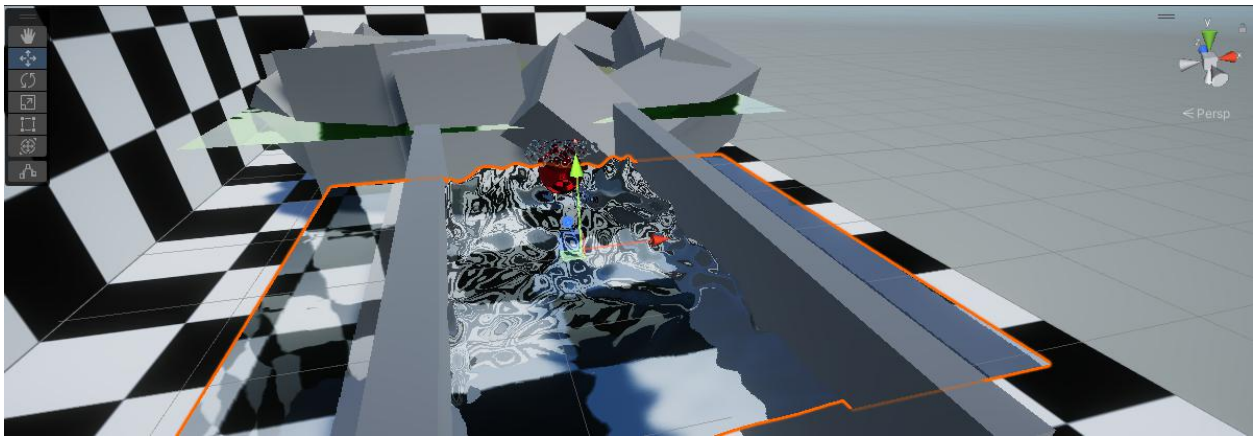
2_Tiling

細長い用水路や、より広い湖面など、タイリングによって表現できる水面の幅が広がります。タイリングにて気を付けることとして、すべての Plane オブジェクトの Wave Simulator の localScale が一致している必要があります。さらに、タイリングするためには ComputeShader Resource を WaveSimulatorTiling に切り替える必要があります。サンプルシーンを参考に、必要に応じてリソースを切り替えてください。



3_Mask





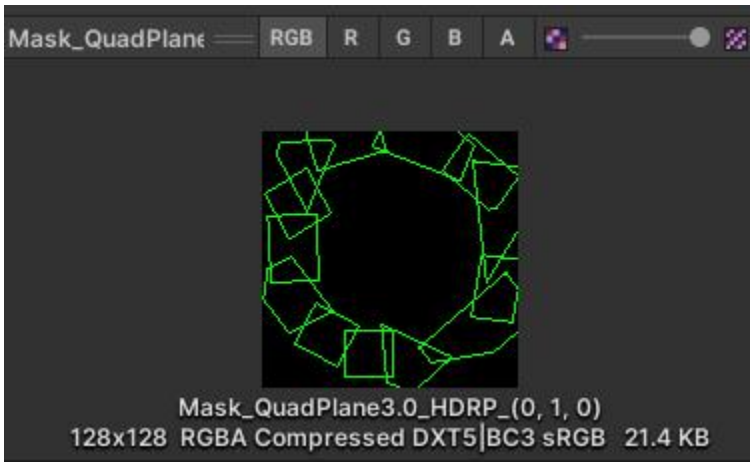
水面に波の障害物を配置した場合に、障害物で波が反射するサンプルシーンです。Wave Simulator スクリプトコンポーネントの Mask Texture に、波の障害物を描いた Texture を割り当てると波が反射するようになります。

QuadPlane オブジェクトにアタッチされている WaveMaskGen スクリプトコンポーネントの Generate ボタンを押すと Output Path に指定した先に QuadPlane オブジェクト名で Mask Texture ファイルを出力します。この出力された Texture ファイルを WaveSimulator の Mask Texture に設定すると、障害物で波が反射するようになります。

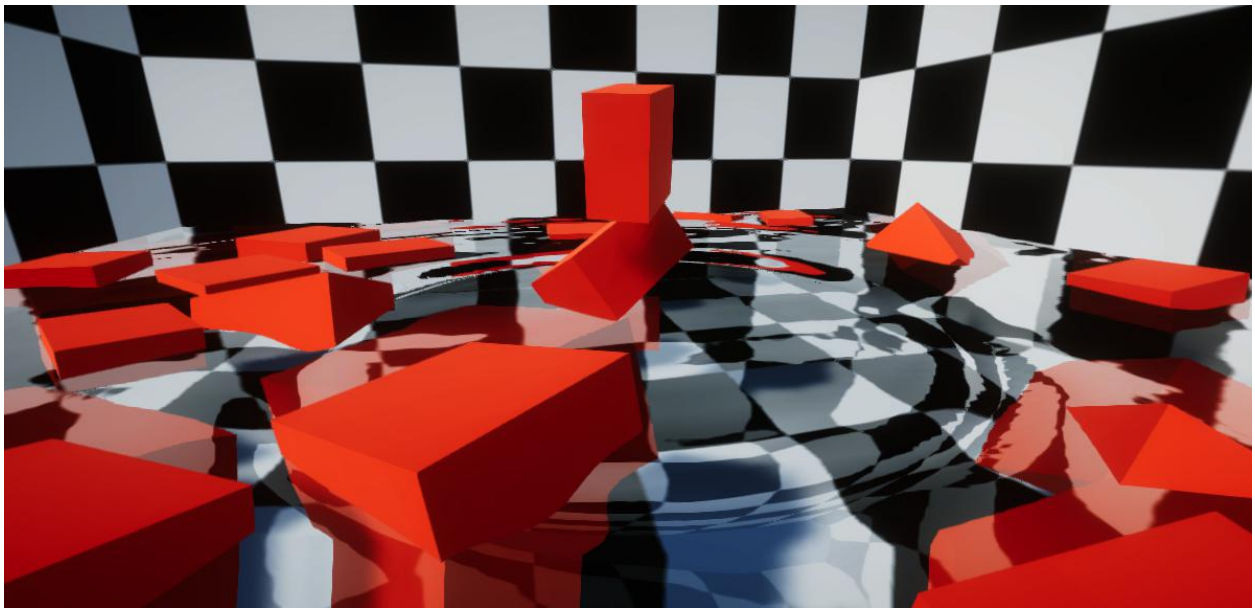
Mask Texture はご自身で作成することもできます。

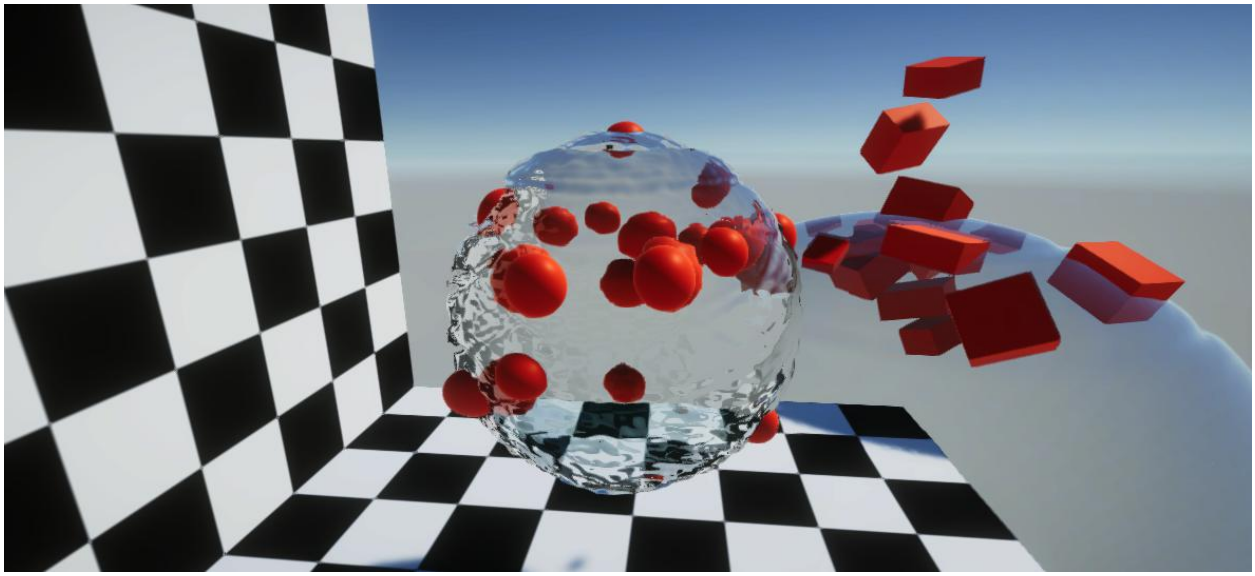
作り方：16bit RG フォーマットの G の値を 0 以外に設定すると、そこで波が反射します。Resolution の値は 2 の累乗数を表しており（7 は 128 x 128）、対象の水面に合わせて選択してから Generate ボタンを押します。水面は Scale に合わせて自動で解像度を設定しますが、Mask Texture が指定されている場合は Mask Texture の解像度に合わせて解像度が固定されます。

Mask Texture を確認すると、障害物の形を確認することができます。



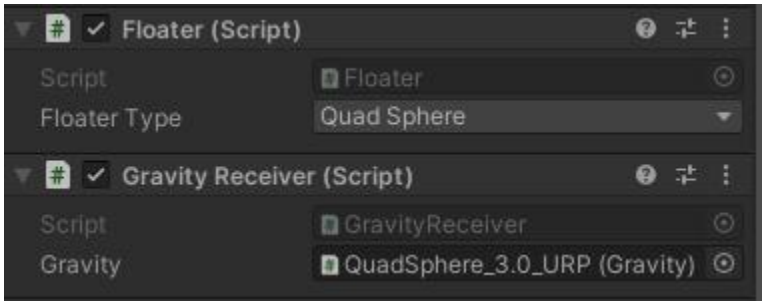
4_Floater



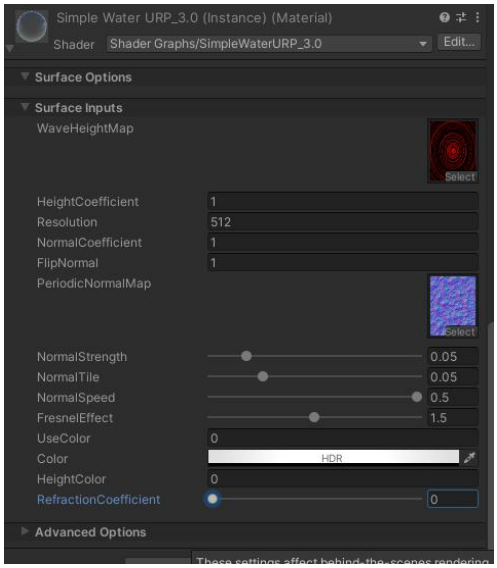


水面や球に箱やボールが浮くサンプルシーンです。浮く側に Floater スクリプトコンポーネントがアタッチされており、水面にどれだけ沈んでいるかの体積に比例して浮力が働きます。水に浮く軽いオブジェクトを表現したい場合は Reigidbody の Mass を小さくし、沈むオブジェクトを表現したい場合は Mass を大きくします。

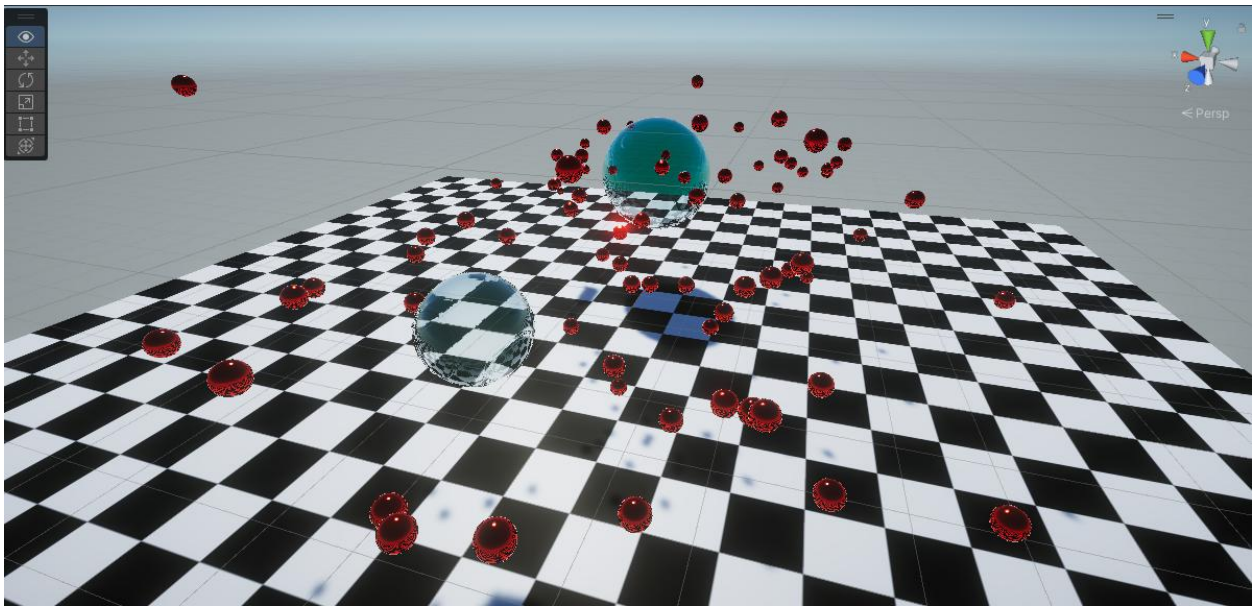
Floater には Quad Plane と Quad Sphere の2タイプがあり、浮く対象の水面に合わせて選択します。水球に引き寄せられる引力については次の 5_Gravity にて詳細を説明します。

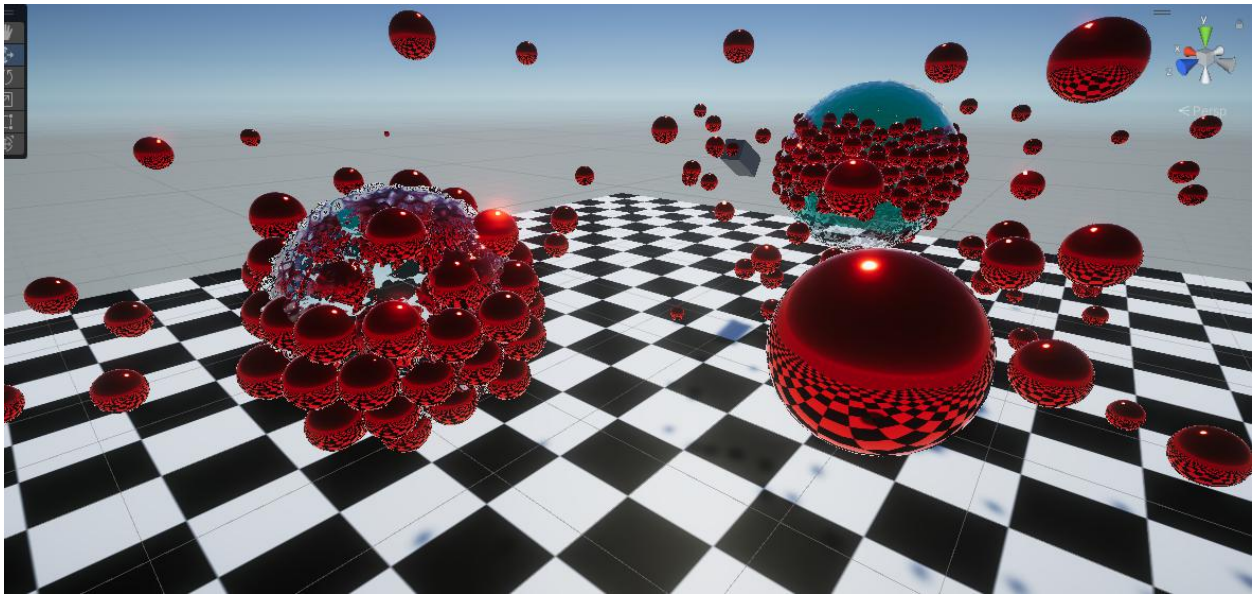


水の屈折表現は UV をずらした色参照となっているため、水面にオブジェクトが浮くシーンでは、おかしな描画結果が気になる方が多いと思います。水面にアタッチしているマテリアルの RefractionCoefficienet の値を 1 から 0 に切り替えることで、屈折表現は無くなりますが、通常の Transparent マテリアルとして、おかしな描画結果を消すことができます。

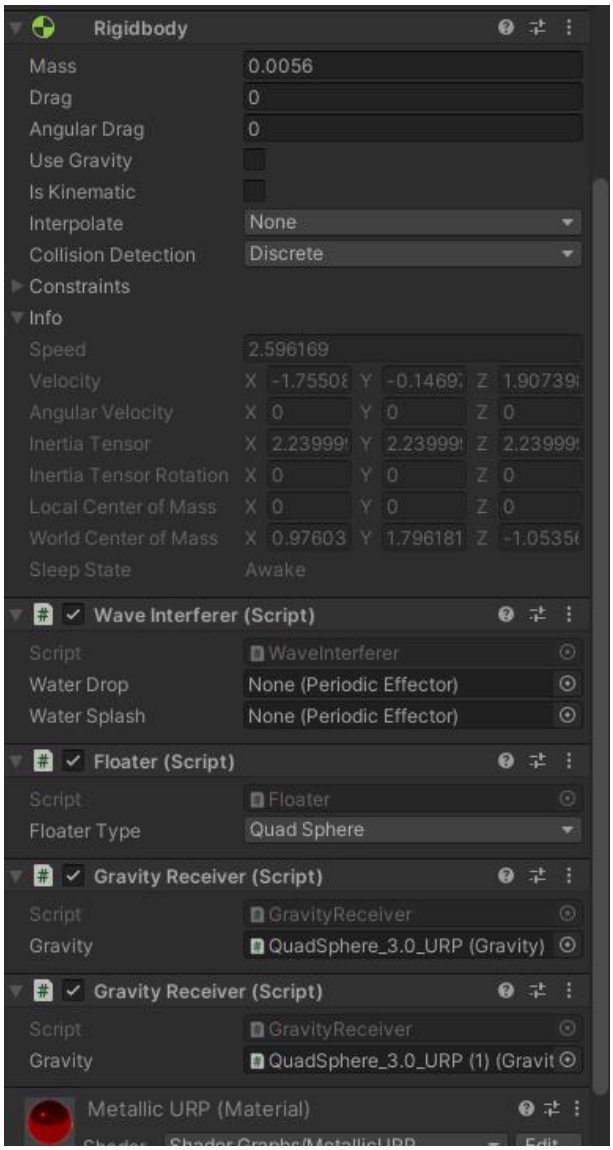


5_Gravity



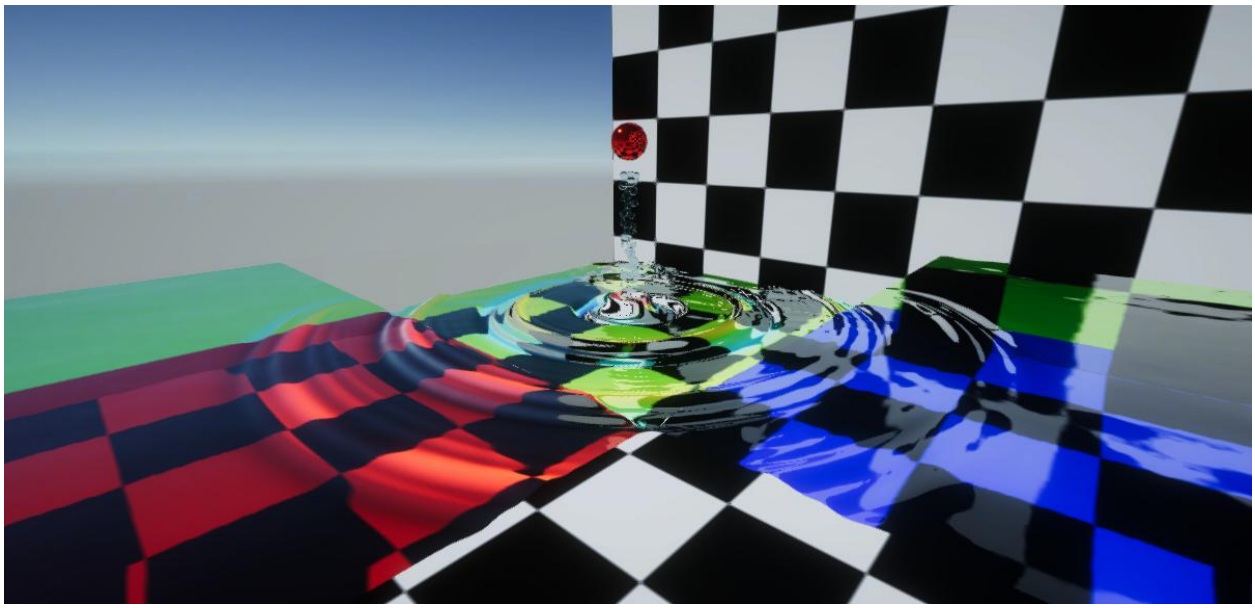


二つの水球に赤いボールが吸い寄せられる引力のサンプルシーンです。引力で吸い寄せられるためにはボールの Rigidbody の useGravity のチェックをはずして Gravity Receiver スクリプトコンポーネントをアタッチします。複数の重力源に吸い寄せられるようにするには、その数の Gravity Receiver スクリプトコンポーネントをアタッチします。Gravity Receiver が参照する Gravity スクリプトコンポーネントの Mass の大きさによって引力が変化します。また、ボール自身の Rigidbody の Mass の値も引力の計算に使われています。

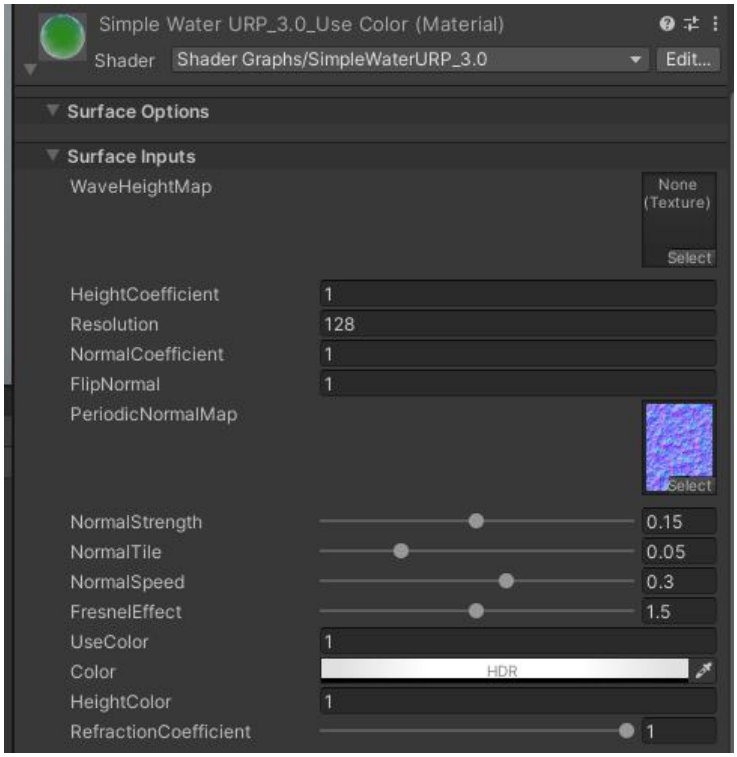


Assets/SimplestarGame/SimpleInteractiveWater/Example/Scenes/URP/5_Gravity/StarsSample_3.0_URP.unity では、互いに Gravity と GravityReceiver をアタッチすることで惑星運動をシミュレートしています。

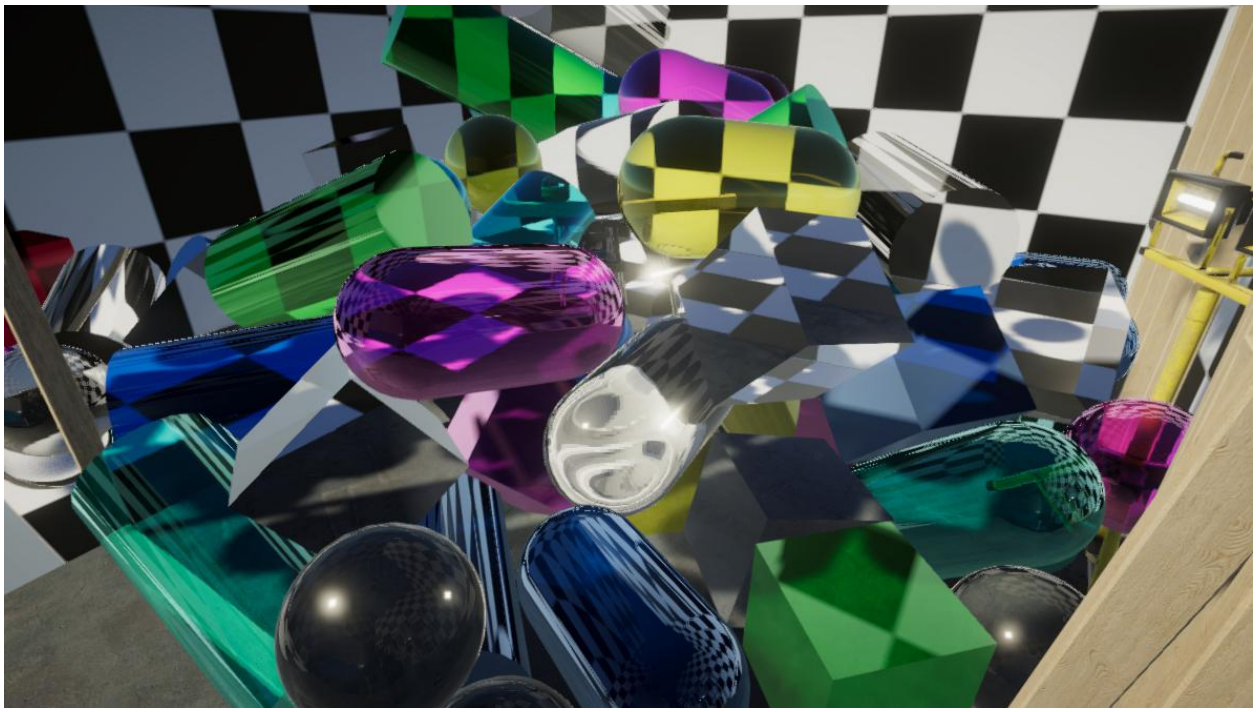
6_Colors



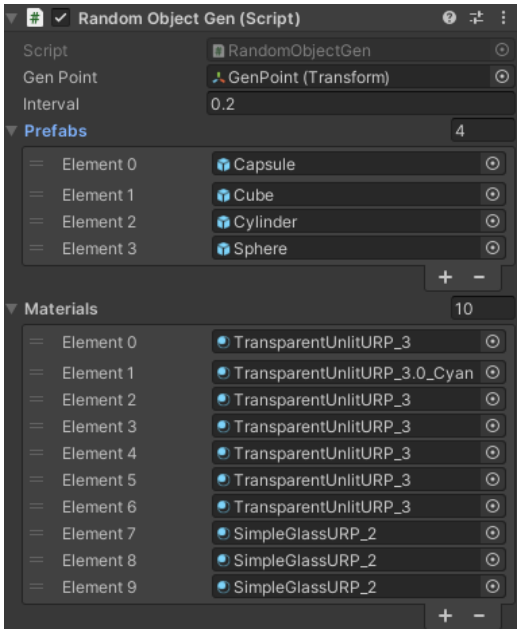
水に色を与えたり、波の高さによって色を変化させられるサンプルシーンです。それぞれの色のついた水面オブジェクトの Material の Color を確認してください。波の高さに合わせて色を変化させる場合は UseColor の値を 0 から 1 に上げます。



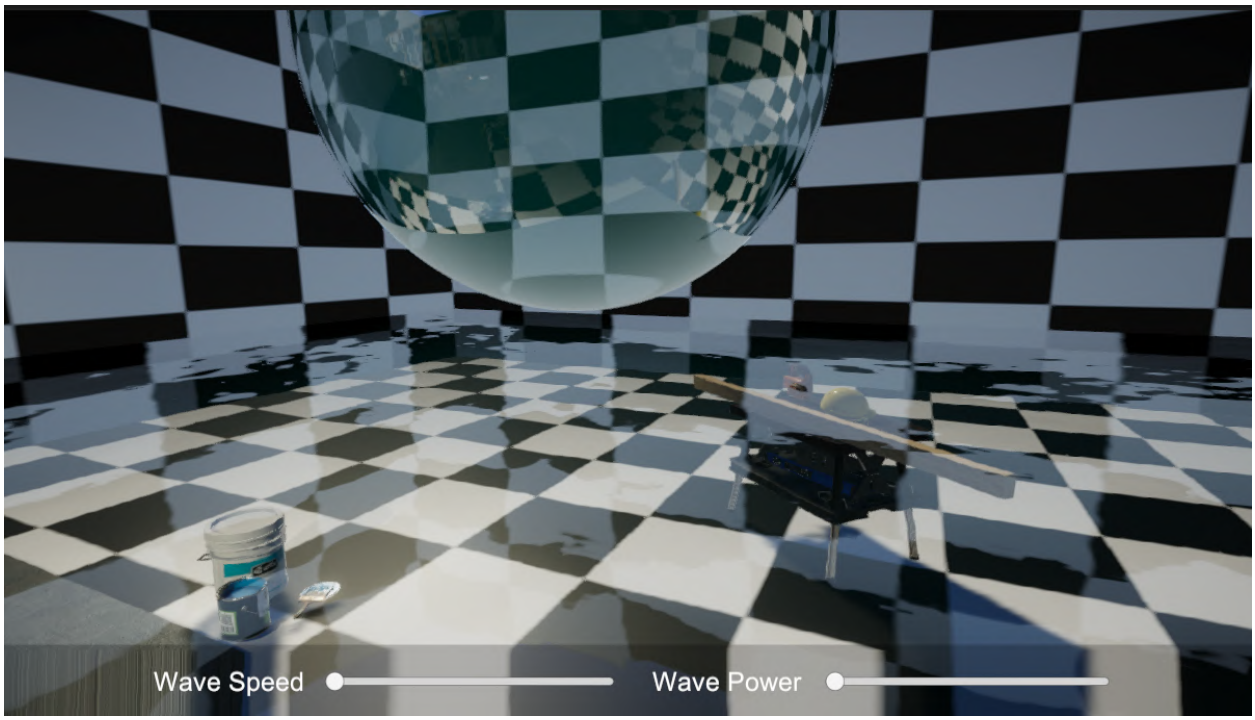
7_Transparent

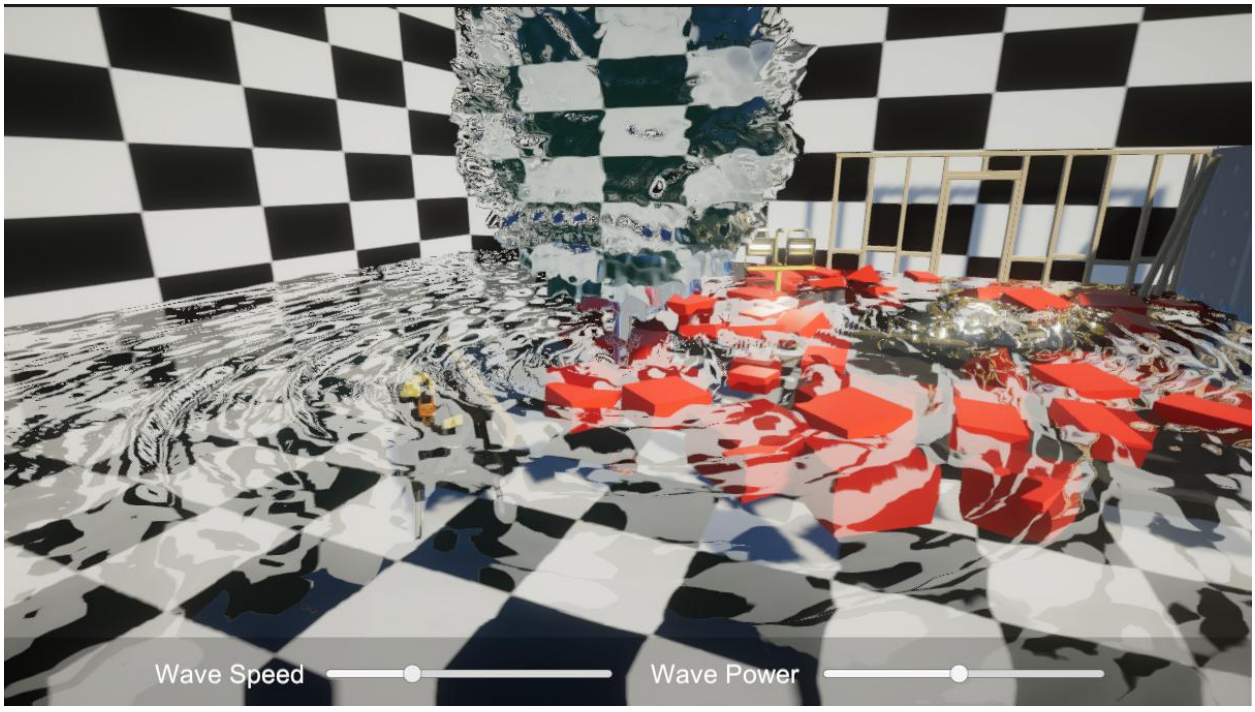


水面の屈折表現で作られた色ガラスマテリアルのサンプルです。シーンに存在する Random Object Gen スクリプトコンポーネントに列挙されている Materials をご利用ください。



8_Params

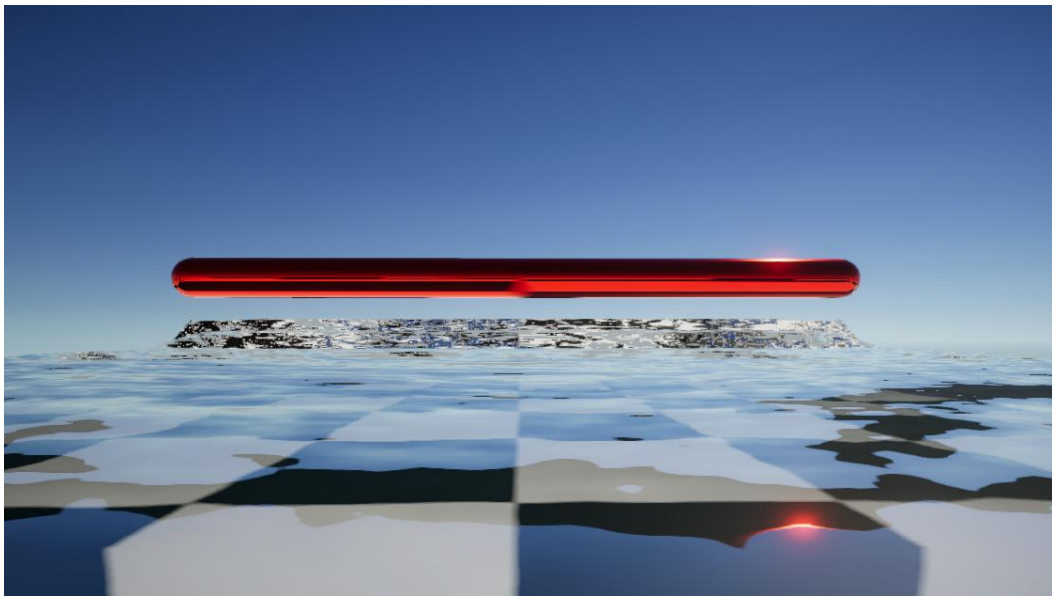




水面の波の速度や発生させる波の大きさを調節できるサンプルシーンです。GUIのスライダーをつまんで、調節しながらランタイムに変化する波の速度や波の大きさを確認できます。実際に調節してい

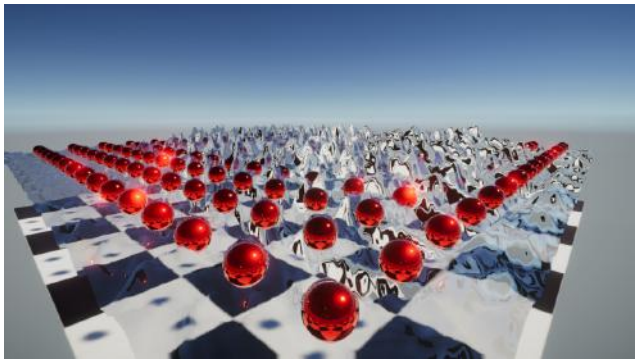
るのは、Wave Simulator スクリプトコンポーネントの Wave Speed と Wave Power の値です。Damping Force は波の減衰力のことです。小さくすると波はおさまらず、大きな値にするとすぐに波はおさまります。

9_BigWave



大きな Capsule の Wave Interferer オブジェクトが大波を発生させるシーンです。一つの水面のスケールを大きくして、広大な水面にできることと、Line Type の Wave Interferer による線分による波起こしができることを確認できます。水面を広く使いたい場合や、線状の波を発生させたい場合の参考資料としてご利用ください。

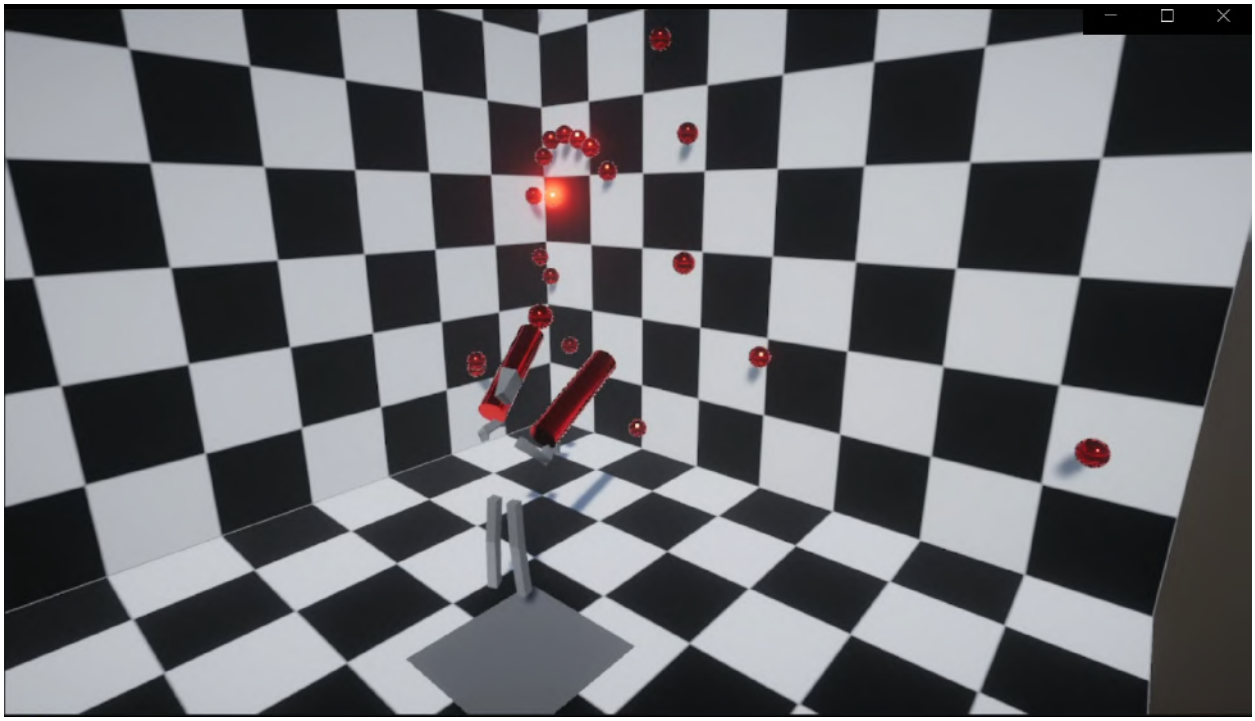
MultiWaves



1フレームに発生できる最大数の波を確認するテストシーンです。この数で一斉に波を発生させてもパフォーマンスは低下しません。

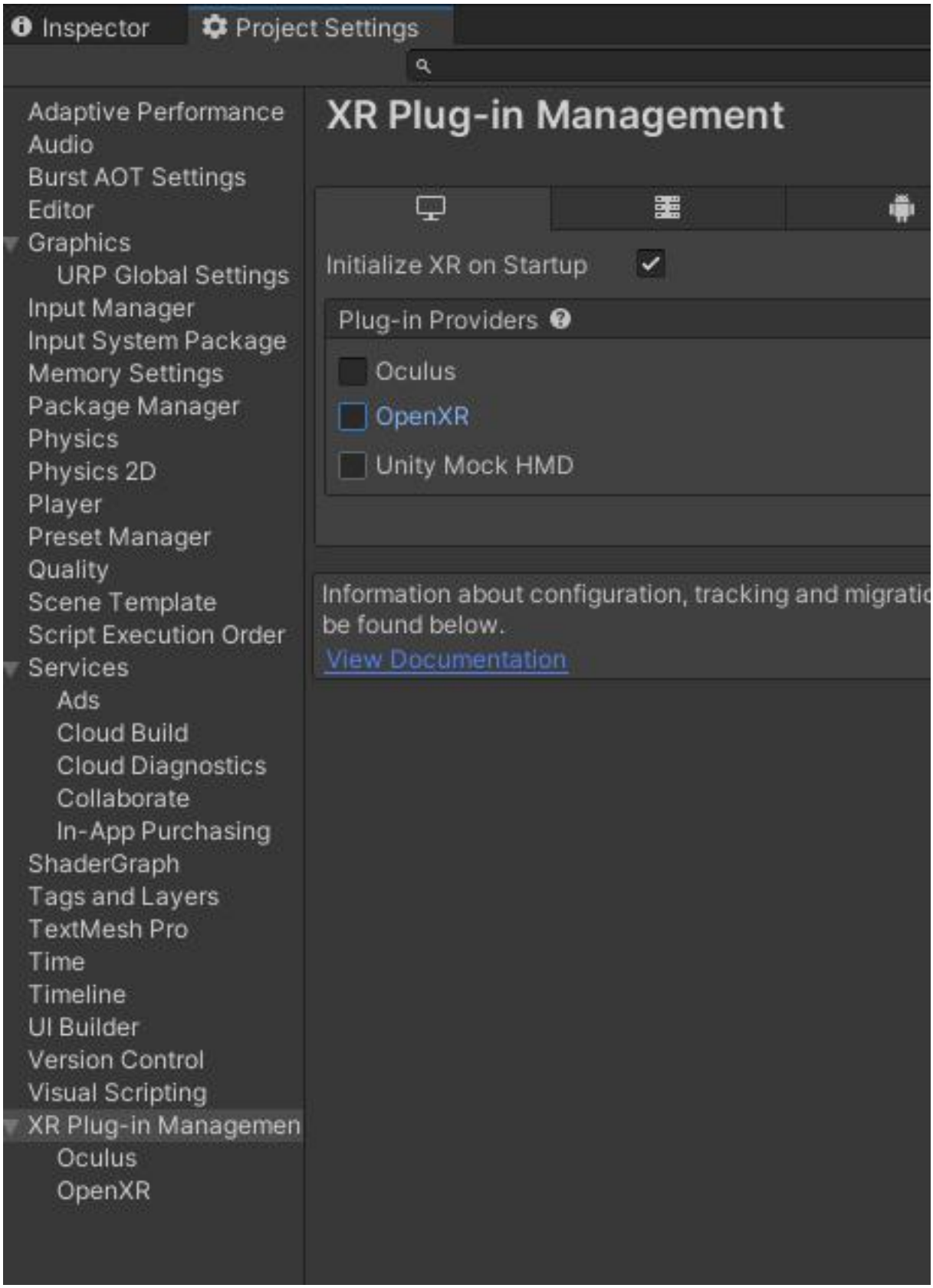
SimpleXR

XRAvatar



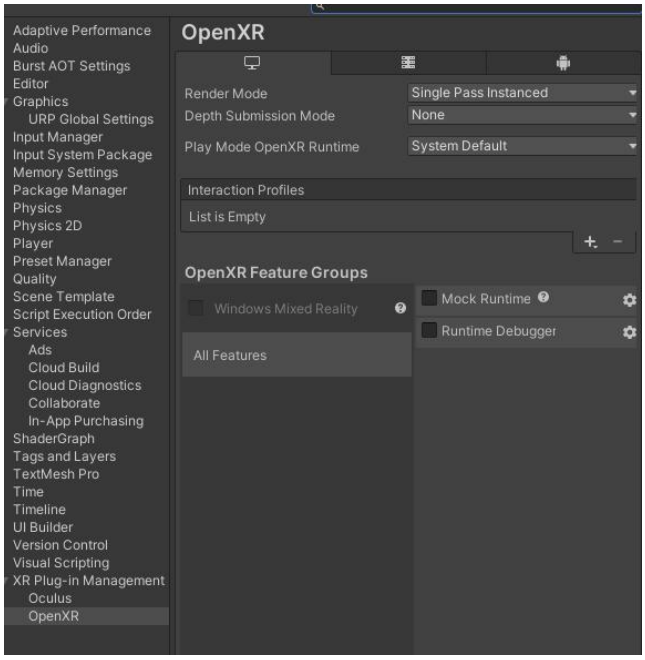
VR 環境にて、Humanoid Avatar をコントロールする UniversalRP 専用のデモです。（Unity の制約で HDRP では VR 使用のパフォーマンスが出ないため Universal RP 専用としています）

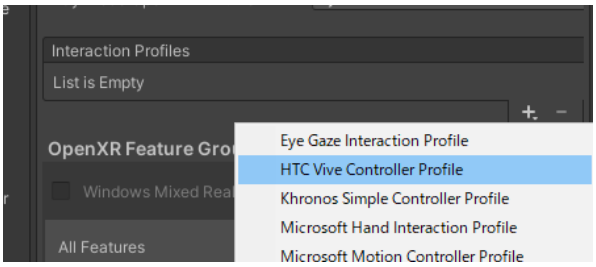
準備のため Project Settings の XR Plug-in Management の Plug-in Providers から、開発者が対象とする VR 環境を選択してください。今回は HTC Vive を対象とした OpenXR にチェックを入れます。

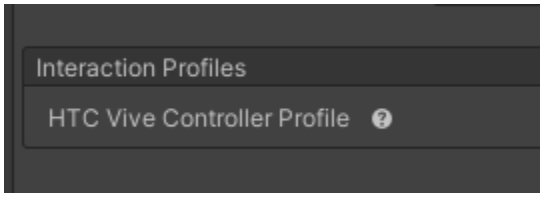

Inspector
Project Settings
Adaptive Performance
Audio
Burst AOT Settings
Editor
Graphics
URP Global Settings
Input Manager
Input System Package
Memory Settings
Package Manager
Physics
Physics 2D
Player
Preset Manager
Quality
Scene Template
Script Execution Order
Services
Ads
Cloud Build
Cloud Diagnostics
Collaborate
In-App Purchasing
ShaderGraph
Tags and Layers
TextMesh Pro
Time
Timeline
UI Builder
Version Control
Visual Scripting
XR Plug-in Managemen
Oculus
OpenXR
XR Plug-in Management
Initialize XR on Startup
Plug-in Providers
Oculus
OpenXR
Unity Mock HMD
Information about configuration, tracking and migrati
be found below.
View Documentation

チェックを入れると警告が出ます。

画像のように Project Settings から OpenXR を選択し、Interaction Profiles に HTC Vive Controller Profile を追加します。

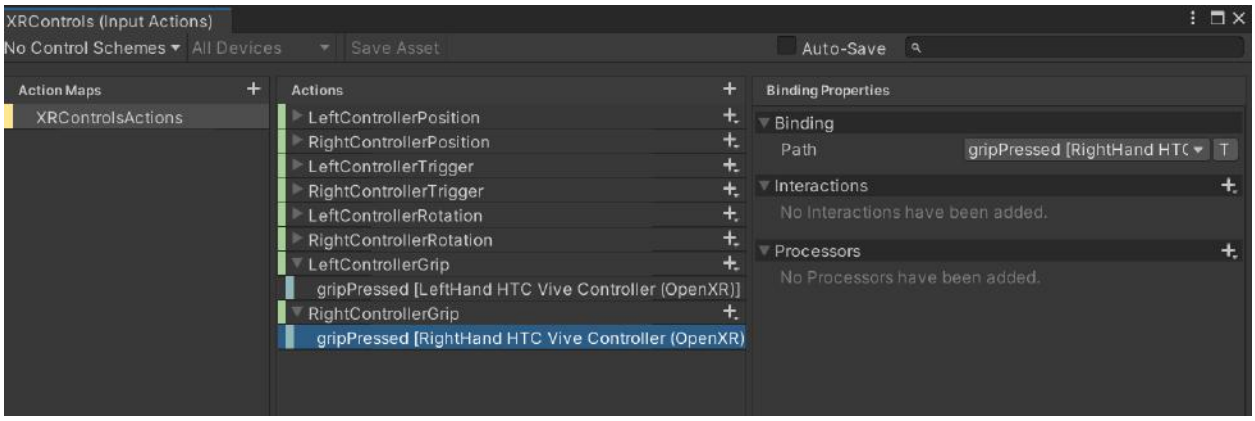






対象 VR 環境を選んだら、次は Input Actions を確認します。

Assets/SimplestarGame/SimpleXRAvatar/Input/XRControls.inputactions ファイルを開いてください。



このようにサンプルシーンで利用している VR 機器の入力 Action 一覧を確認することができます。それぞれの Binding Properties を開き、お使いの VR 機器の入力が足りていない場合はバインドを追加してください。（HTC Vive の OpenXR 環境の場合は、すでに設定済みです。）

以上の準備が整いましたら

Assets/SimplestarGame/SimpleXRAvatar/Scenes/XRAvatar.unity シーンを開いてください。

再生すると VR の HMD と左右のコントローラーの3つのトラッキングで Humanoid が IK を使って動き出し、コントローラーのグリップ操作で赤い筒を手に持つことができ、さらにトリガーを引くと赤い球を発射できます。

シーンのヒエラルキーの説明

StaticCamera : VRビューとは別のカメラです。Camera の Priority が高いことから VR で遊んでいる姿を第三者視点でモニターに映し出すことができます。

AvatarController: XRAvatarController スクリプトコンポーネントにて、Humanoid にアタッチする Animator コントローラーやトラッキングで追従移動する Head, Left Controller, Right Controller オブジェクトの参照、スタート位置のオフセット姿勢となる XRRig を設定しています。ゲームを開始すると AvatarActivator スクリプトコンポーネントにて、指定している Humanoid に、XRAvatarController の設定を与えて IK により Avatar を追従させることができるようになります。

XRActions: Input System の Action 一覧のイベント登録と、Grip によるオブジェクトのつかみ処理、トリガーによるオブジェクトのトリガー処理が記述されています。Grip できるオブジェクトは Tag

に “Grip” を指定する必要があります。また、グリップした後に位置を調整できる機能があり、これをオブジェクトごとにアタッチした GripOffset スクリプトコンポーネントにて指定しています。

対象の Grip オブジェクトに ITrigger を継承したクラスがアタッチされている場合、その OnTrigger イベントをトリガーイベントで呼び出して、手に持ったオブジェクトの Trigger 処理を実行します。例としてシーン内の ShooterL, ShooterR オブジェクトには Shooter スクリプトコンポーネントがアタッチされており、これが ITrigger クラスを継承していることから、Sphere オブジェクトを勢いよく発射する OnTrigger 機能が実装され、機能しています。

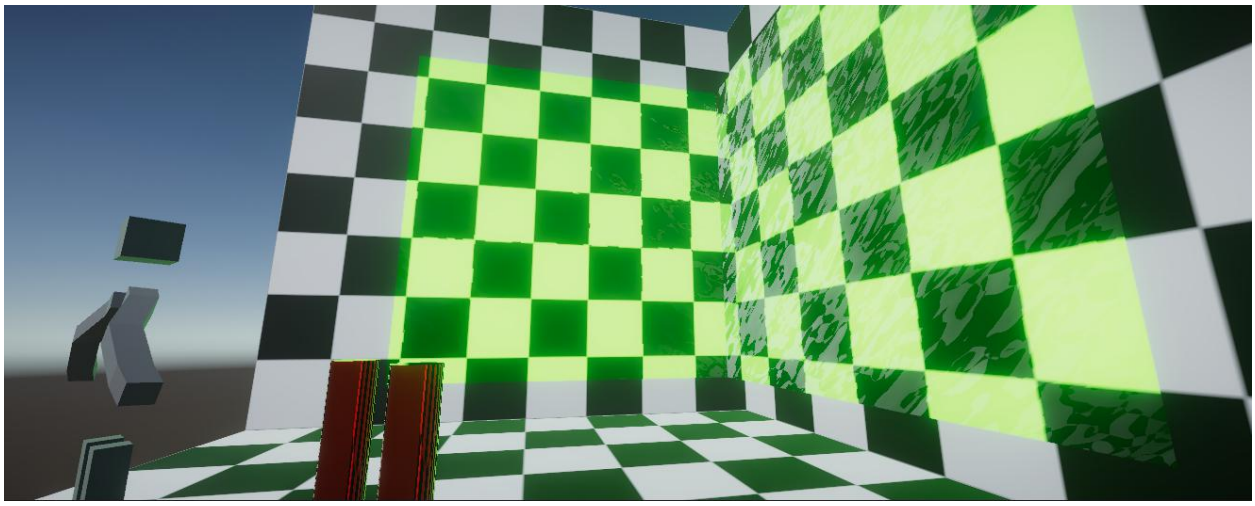
BoxHumanoid は非常に単純な Humanoid です。ご自身のゲームやアバターの Humanoid と差し替えてこのシーンを、自身のプロダクトにご活用ください。

XRWaterInteraction

水面に浮かぶ2つのボールをグリップ操作でつかんで、水面に打ち付けて波を発生させるデモシーンです。もっとも基本的な水面を使った VR シーンを体験することができます。

XRInterfererShooter

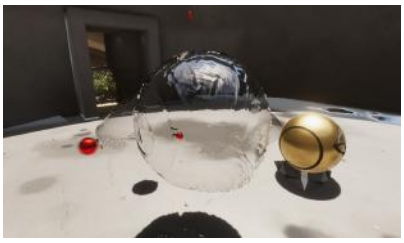
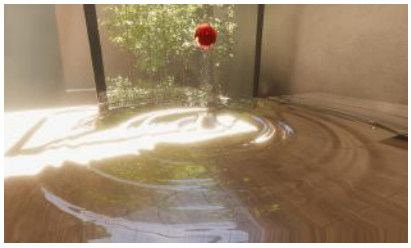
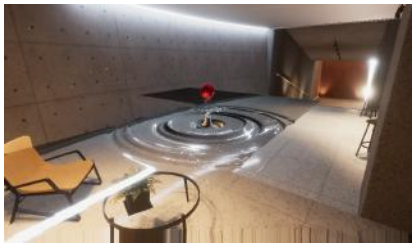
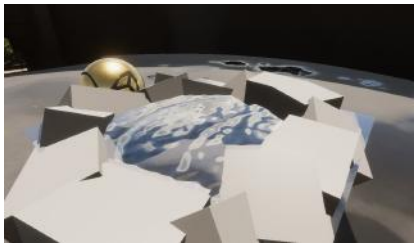
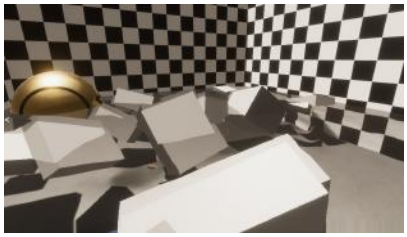
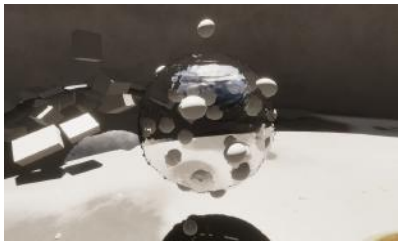
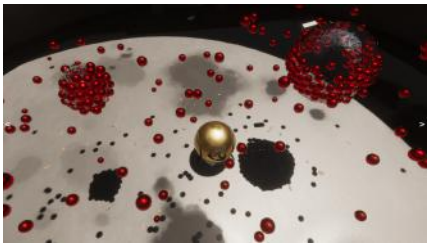
UseColor の水面を垂直に二枚配置して、水面に Wave Interferer を撃ち込むサンプルシーンです。

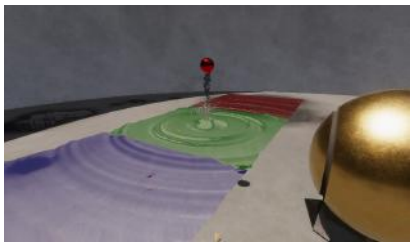
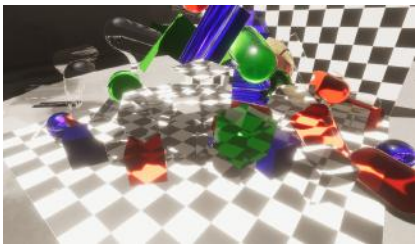
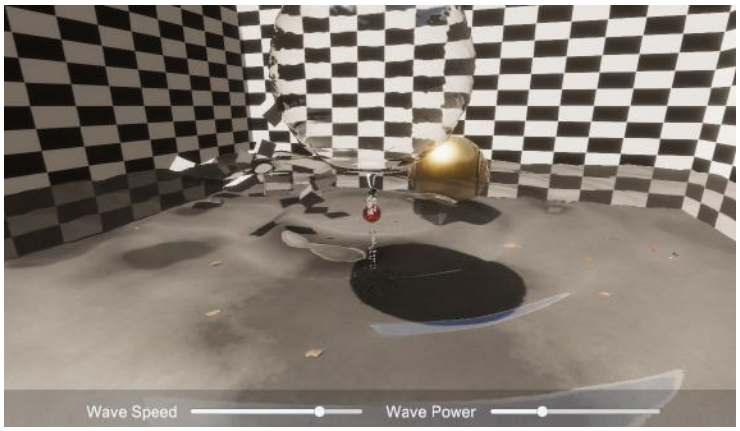


水面を垂直配置しても機能するテストシーンとなっています。

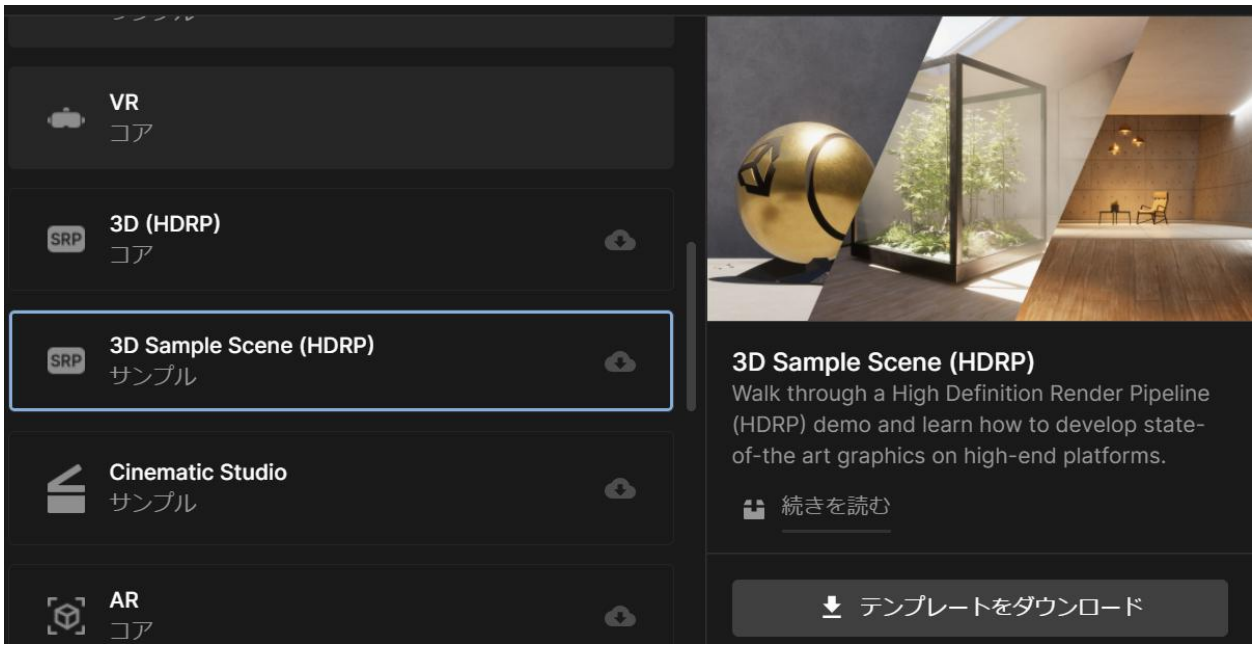
HDRP Sample Scenes

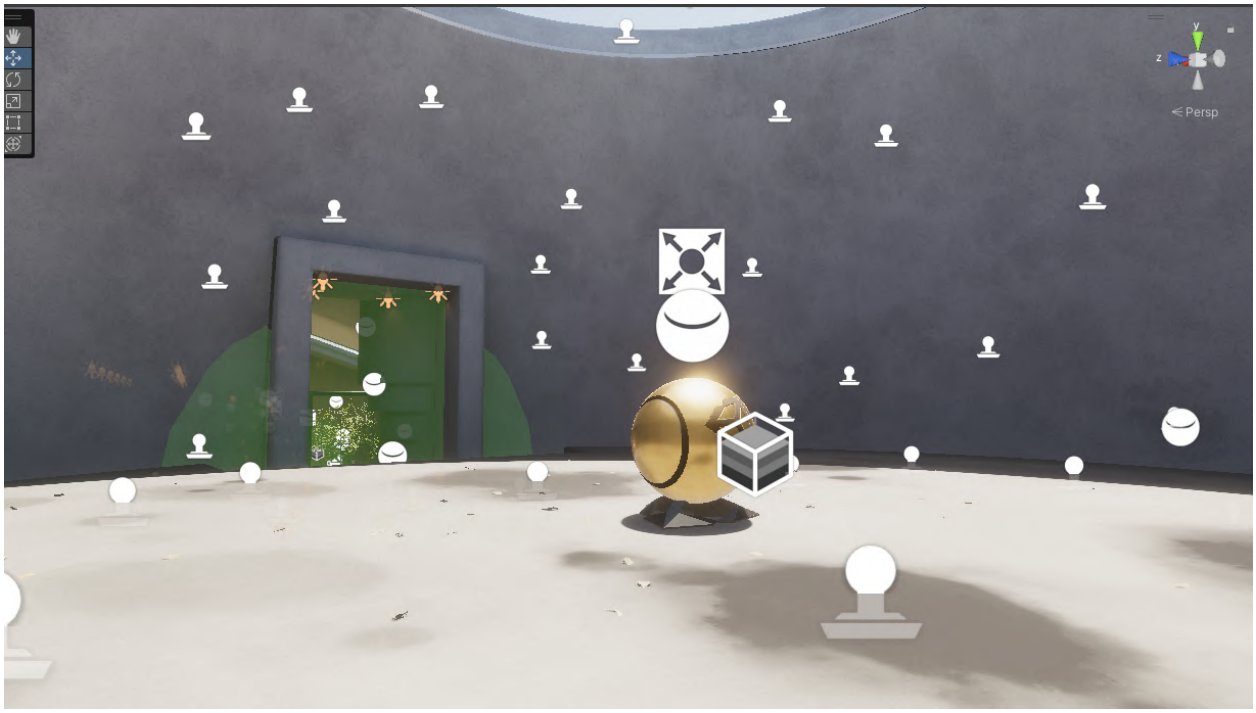


新しい Unity プロジェクトを HDRP のテンプレートシーンを選んで作成します。

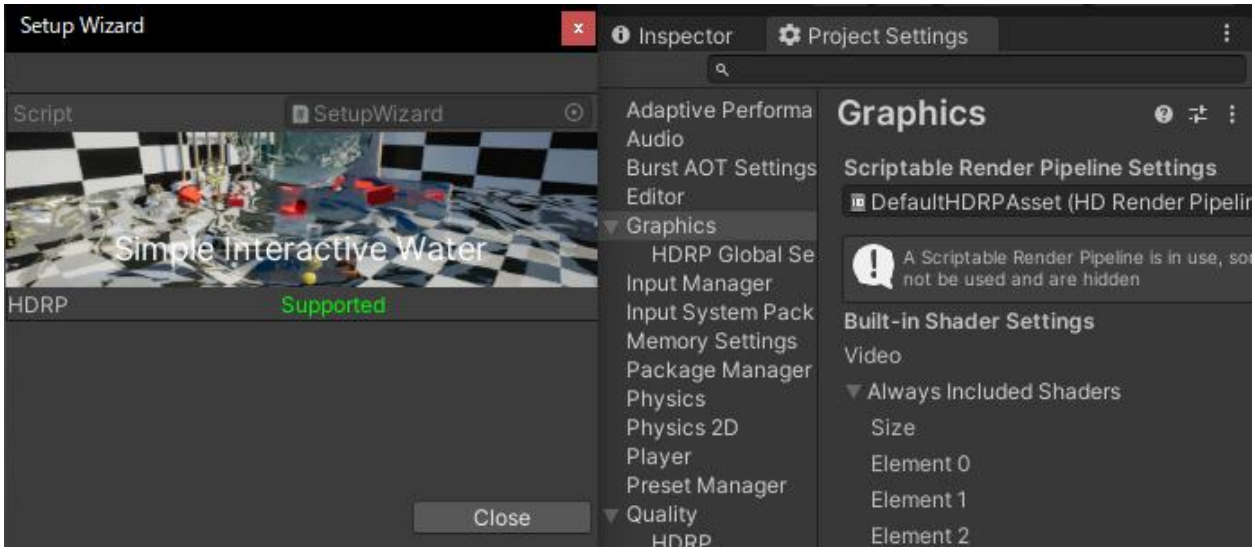




次に Simple Interactive Water アセットをインポートします。

package の依存関係解決のため「This Unity Package has Package Manager dependencies.」ダイアログが表示されますので「Install/Upgrade」ボタンを押します。

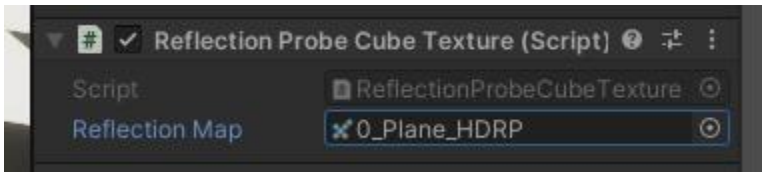
HDRP テンプレートは初めから新しい Input System がインストールされているので、このまま続けてアセットのインポートを行います。その後はメニューの Window > Simple Interactive Water > Setup Wizard を選択して、セットアップウィザードを開いてください。



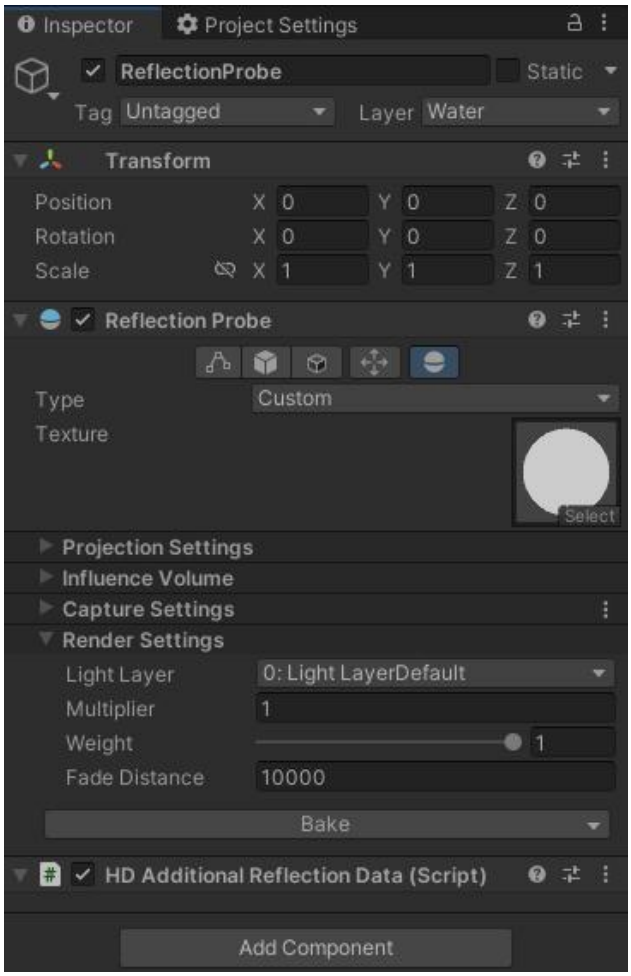
Supported の表示を確認したら、次にサンプルシーン一覧を確認します。

Assets/SimplestarGame/SimpleInteractiveWater/Example/Scenes/HDRP/Sample Scenes for HDRP.unity を開いて、ヒエラルキーに存在する ScenesInBuildHDRP オブジェクトのインスペクタの Reset Scenes in Build ボタンを押します。Scene Names の個数が 14 となれば準備完了です。

Play ボタンを押してゲームを再生すると、Scene Names 先頭要素の PlaneSample_2.2_HDRP のサンプルが再生されます。左右のアローキーか、画面内の左右のボタンを押してシーンを切り替えることができます。URP 用に説明してきたものと内容は同じなので、各シーンの詳細説明は省きます。一点、HDRP 版の 3.0 シェーダーにて、Reflection Probe の値が Shader Graph で取得できないため、一度テクスチャに Cube Map を書き出して、これをシェーダーが利用することで水面の反射表現を実現しています。もし水面の配置場所を移動させた場合は、Plane オブジェクトの子オブジェクトである Reflection Probe にアタッチされている Custom Reflection Probe の Bake ボタンを押してください。次の Reflection Probe Cube Texture で参照している Cube Texture が更新されます。

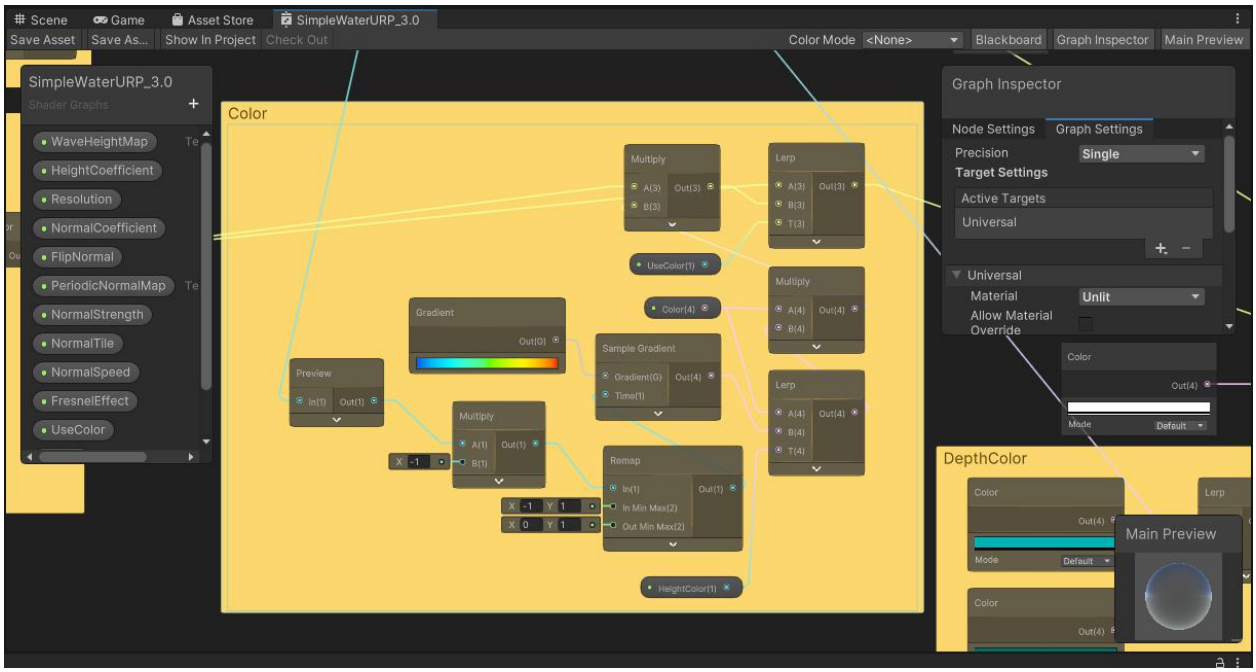


手間が 2.2 に比べて増えています。(そのかわり、反射の強度などが 2.2 と比べて調整できます)



Shader Graph

ここからは、より水面の見た目を変更したい方向けの情報です。それぞれの水面にアタッチされているマテリアルの Edit ボタンを押すと、対応する Shader Graph の編集画面を開くことができます。シェーダー内容を解析して独自の水面表現にカスタマイズしたい方は、こちらの Shader Graph の内容を理解して編集してください。

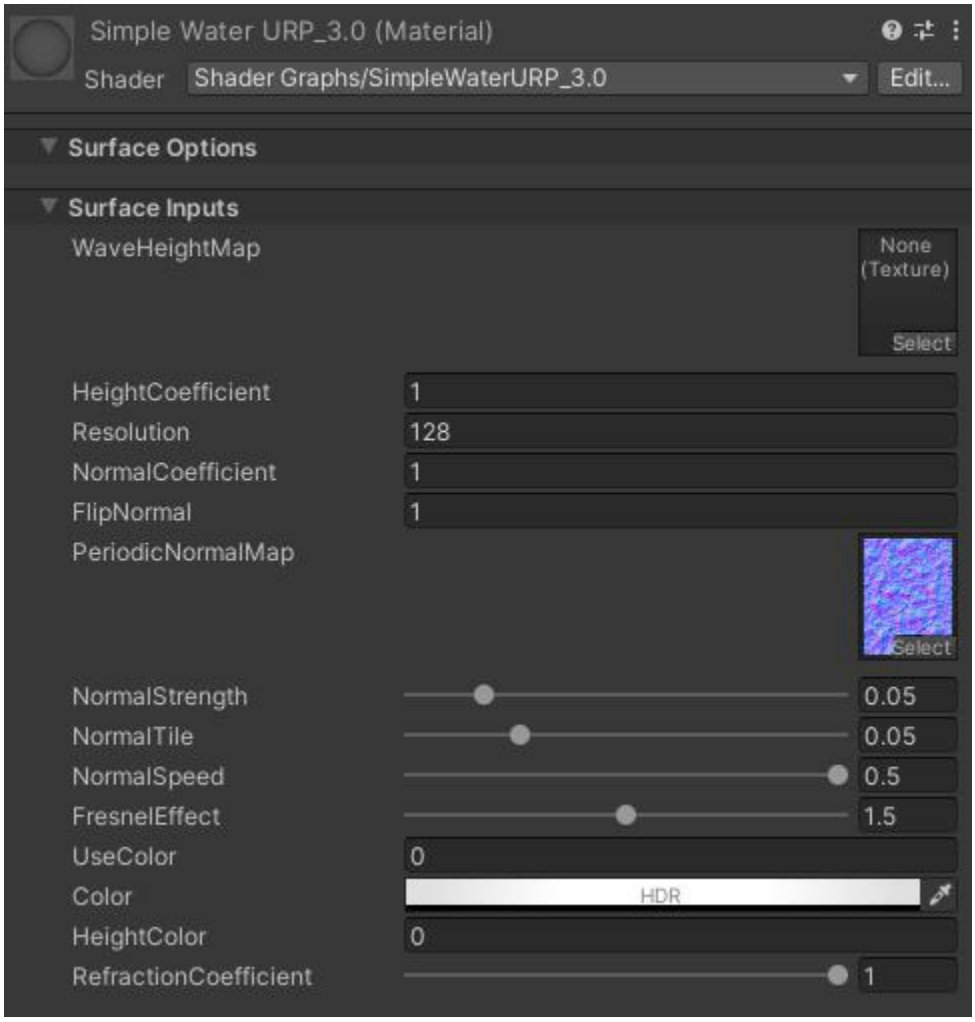


Uninstall

Asset/SimplestarGame/SimpleInteractiveWater と SimpleXRAvatar フォルダを削除することでアンインストールが完了します。

Material Parameter

水面表現の調整パラメータを説明します



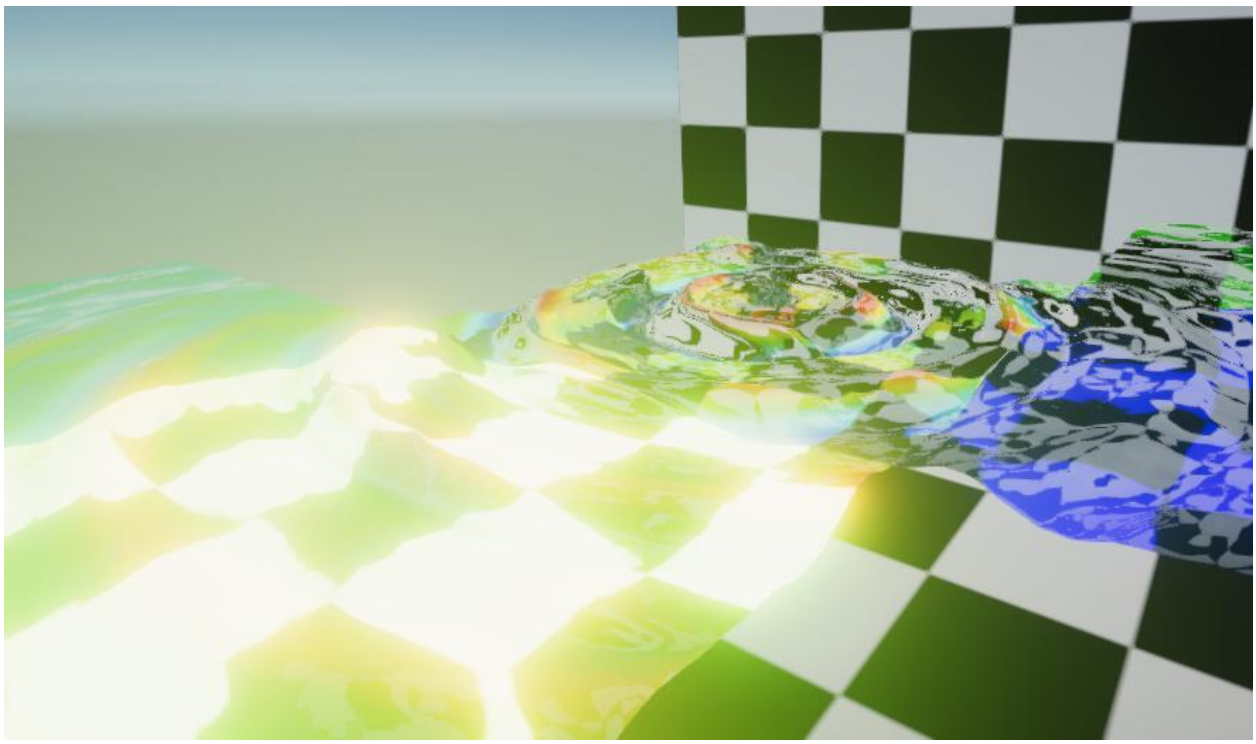
WeveHeightMap: ゲームプレイ時に Wave Simulator が作成した解像度の波の HeightMap がここに設定されます。

HeightCoefficient: シミュレーションとは別で、見た目に関する波の高さを調整できます。

Resolution: HeightMap の解像度が表示されます。調整すると計算結果に影響は出ますが、表示

が崩れるだけなので編集しないでください。
NormalCoefficient: 波の法線強度を調整できます。より見た目にインパクトのある水面にしたい場合は大きな値を設定すると効果的です。
FlipNormal: 水中にカメラが潜った際に全反射表現に切り替える係数です。1 から -1 に設定すると全反射する水面表現が行えます。これには水面を両面マテリアルにするなどの調整が必要です。
PereodicNormalMap: 定常状態における水面に起きている小さな波に使われる法線マップ Texture を指定します。タイリングするテクスチャであることが望ましいです。指定を None にすると、定常状態での波を非表示にできます。
NormalStrength: 定常状態の波の法線強度を調整できます。
NormalTile: 定常状態の波のタイリング密度を調整できます。
NormalSpeed: 定常状態の波の速度を調整できます。
FresnelEffect: 法線と視線角度によって映る反射強度を調整できます。

UseColor: 1 に設定すると次の Color の値を乗算します。
Color: 水面に乗算する色です。光る水などを表現したい場合は HDR カラーとして明るい色を設定します。



HeightColor: 波の高さによってグラデーションが起こるようになります。主にデバッグ用の表示オプションです。
RefractionCoefficient: 水面の光の屈折表現の強度を調整できます。水面に浮かぶオブジェクトが

水面の屈折表現に使われると見た目がおかしいことがあるので、シーンによっては 0 に設定することで、おかしな見た目を回避することができます。