STL DYNAMICS

Quick Start

Retro Water - Quick Start

From a clean URP project to profile-driven water, ocean LOD, buoyancy, caustics, and underwater effects.

This guide follows the actual editor menus and component workflow implemented by the package.

Part 1 - Project Setup

1. Import and verify URP setup

Open:

```
Tools > STL Dynamics > Retro Water > Setup & Diagnostics
```

In **Project Status**:

1. Confirm **Water shader** reports `Found`.
2. Confirm **Render pipeline** reports `URP`.
3. Click **Install / Repair Renderer Feature**.
4. Click **Enable Depth Texture** if it is disabled.

These actions use:

- `RetroWaterSetupWindow`
- `RetroWaterCausticsInstaller`
- `RetroWaterCausticsFeature`

The renderer feature is only required for projected caustics and the underwater camera pass. The surface shader can render without it, but depth-based shoreline and color effects still require Depth Texture.

2. Create starter profiles

Optional but recommended:

```
Tools > STL Dynamics > Retro Water > Create Starter Profiles
```

Profiles are generated under:

```
Assets/STLDynamics/RetroWater/Profiles/Starter
```

The generator creates:

- Calm Lake

- Windy Lake
- River
- Gentle Ocean
- Rough Ocean
- Swamp
- PS1 Extreme

Source: `Editor/RetroWaterStarterProfileGenerator.cs`

Part 2 - Surfaces and Oceans

---

3. Create a normal water surface

Fastest workflow

1. Select a `RetroWaterProfile` in the Project window.
2. Choose:

```
GameObject > STL Dynamics > Retro Water > Water Surface
```

A Unity Plane is created, its default collider is removed, `RetroWaterSurface` is added, and the selected profile is assigned automatically.

Source: `Editor/RetroWaterObjectFactory.cs`

Existing mesh workflow

1. Select a planar mesh object.
2. Add `RetroWaterSurface`.
3. Assign a profile.
4. Leave **Material Mode** on `Runtime Instance` for the simplest workflow.
5. Press **Apply Profile**.

For custom meshes:

- `Plane Axes = Auto` chooses the two largest local mesh dimensions.
- Use `XZ`, `XY`, or `YZ` when automatic detection is wrong.
- Keep `Finite Surface` enabled for ponds, pools, and rivers so sampling stops outside the mesh bounds.
- Use `Bounds Padding` to slightly expand or contract sampling coverage.

---

4. Create an ocean

1. Select an ocean-capable profile such as **Gentle Ocean**.
2. Choose:

```
GameObject > STL Dynamics > Retro Water > Ocean
```

3. Keep the ocean root Transform scale at:

```
1, 1, 1
```

4. Assign the gameplay camera or player camera transform as **Viewer**, or leave it empty with automatic main-camera lookup enabled.
5. Press **Build / Rebuild Ocean**.

Recommended seam settings:

```
Crack-Free Shared Borders: Enabled
Legacy Seam Skirts: Disabled
```

The generated grid is approximately:

```
ceil(Ocean Size X / Chunk Size) x ceil(Ocean Size Z / Chunk Size)
```

If that count exceeds **Maximum Chunk Count**, the runtime increases the effective chunk size while preserving the requested overall ocean size.

Source: `Runtime/Scripts/RetroWaterOcean.cs`

Part 3 - Physics and Effects

---

5. Add buoyancy

On the object that should float:

1. Add a non-kinematic `Rigidbody`.
2. Add at least one enabled, non-trigger Collider.
3. Add `RetroWaterBuoyant`.
4. Use `Automatic Collider Bounds`.
5. Press **Apply Recommended Defaults** in the inspector.
6. Test in water deep enough for the object to move without touching the bottom.

Useful starting values:

```
Buoyancy Strength: 1.25
Sink Depth: 0
Maximum Submersion Depth: 0.5
Vertical Damping: 0.85
Linear Water Drag: 0.35
Angular Water Drag: 0.08
Self Righting Strength: 2
Self Righting Damping: 0.75
```

Rules of thumb:

- `Buoyancy Strength > 1`: the object can float.
- `Buoyancy Strength ~ 1`: nearly neutral.
- `Buoyancy Strength < 1`: the object can sink.
- Positive `Sink Depth`: lowers the effective buoyancy waterline so the object rests deeper.
- `Sink Depth` is not a substitute for insufficient maximum lift. Use strength below `1` for true sinking.

Source: `Runtime/Scripts/RetroWaterBuoyant.cs`

6. Enable caustics

In the profile's **Effects** tab:

1. Enable **Caustics**.
2. Assign a caustics texture.
3. Set **Maximum Depth** deeper than the intended receiving geometry.
4. Confirm the renderer feature and Depth Texture are enabled.

Caustics affect opaque, depth-writing surfaces reconstructed from the camera depth texture. Transparent receivers generally cannot participate in the same way.

The renderer feature renders caustics after opaque geometry and before transparent water.

---

7. Enable underwater camera effects

In the profile's **Effects** tab:

1. Enable **Underwater Camera**.
2. Tune color, fog, visibility, distortion, vignette, and retro bands.
3. Move the camera below the actual animated surface.

No trigger volume is used. The camera is compared against `RetroWaterSurface.TrySample()` at its world position, so the transition follows the animated wave height and finite surface bounds.

The pass renders before URP post-processing.

Part 4 - Materials and Runtime

---

8. Choose a material workflow

Runtime Instance

Recommended default:

```
Material Mode: Runtime Instance
```

No `.mat` asset is required.

Baked Profile Material

On the profile inspector, press:

```
Create Material Asset
```

Then set the surface to:

```
Material Mode: Baked Profile Material
```

Use this when multiple surfaces should share one material asset.

Manual Material

Set:

```
Material Mode: Manual Material
```

Assign a normal material using:

```
STL Dynamics/Retro Water
```

The profile can still control surface sampling, caustics, underwater effects, and ocean settings, while the material owns shader values.

---

9. Validate the scene

Open **Setup & Diagnostics**, select the water object, and verify:

- A profile or manual material is assigned.
- The active material exists.
- Ocean scale is `1,1,1`.
- The ocean grid is built.
- Renderer feature is installed.
- Depth Texture is enabled.
- Buoyant objects have Rigidbody and colliders.

---

10. Runtime profile switching

```
using STLDynamics.RetroWater;
using UnityEngine;

public sealed class WaterStateController : MonoBehaviour
{
    [SerializeField] private RetroWaterSurface water;
    [SerializeField] private RetroWaterProfile calmProfile;
    [SerializeField] private RetroWaterProfile stormProfile;

    public void SetStorm(bool enabled)
    {
        water.SetProfile(enabled ? stormProfile : calmProfile);
    }
}
```

`SetProfile(profile)` applies the profile and rebuilds an attached ocean when required.