Chapter 1: Pursuing Performance Problems
Chapter 2: Scripting Strategies

![Profiler](image)

EmptyClassComponents Enabled

EmptyCallbackComponents Enabled
A new enemy was created! Harry
UnityEngine.Debug.Log(Object)

A new enemy was created! Tom
UnityEngine.Debug.Log(Object)

A new enemy was created! Dick
UnityEngine.Debug.Log(Object)

A new enemy was created! Harry
UnityEngine.Debug.Log(Object)

A new enemy was created! Tom
UnityEngine.Debug.Log(Object)

A new enemy was created! Tom
UnityEngine.Debug.Log(Object)
Chapter 3: The Benefits of Batching
Chapter 4: Kickstart Your Art

<table>
<thead>
<tr>
<th>Texture Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>Texture Shape</td>
<td>2D</td>
</tr>
<tr>
<td>sRGB (Color Texture)</td>
<td>Yes</td>
</tr>
<tr>
<td>Alpha Source</td>
<td>Input Texture Alpha</td>
</tr>
<tr>
<td>Alpha Is Transparency</td>
<td>No</td>
</tr>
</tbody>
</table>

**Advanced**

<table>
<thead>
<tr>
<th>Non Power of 2</th>
<th>ToNearest</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read/Write Enable</td>
<td>No</td>
</tr>
<tr>
<td>Generate Mip Maps</td>
<td>Yes</td>
</tr>
<tr>
<td>Border Mip Maps</td>
<td>No</td>
</tr>
<tr>
<td>Mip Map Filtering</td>
<td>Box</td>
</tr>
<tr>
<td>Mip Maps Preset</td>
<td>No</td>
</tr>
<tr>
<td>Fadeout Mip Map</td>
<td>No</td>
</tr>
<tr>
<td>Wrap Mode</td>
<td>Repeat</td>
</tr>
<tr>
<td>Filter Mode</td>
<td>Bilinear</td>
</tr>
<tr>
<td>Aniso Level</td>
<td>1</td>
</tr>
</tbody>
</table>

**Default**

<table>
<thead>
<tr>
<th>Max Size</th>
<th>2048</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resize Algorithm</td>
<td>Mitchell</td>
</tr>
<tr>
<td>Compression</td>
<td>Normal Quality</td>
</tr>
<tr>
<td>Format</td>
<td>Auto</td>
</tr>
<tr>
<td>Use Crunch Compress</td>
<td>No</td>
</tr>
</tbody>
</table>
ATLASING

128x128 (each) → 512x512

16 Textures  →  1 Texture
16 Materials →  1 Material
16 Draw Calls →  1 Draw Call
Chapter 5: Faster Physics

Two objects in motion

Discrete Collision Detection

COLLISION DETECTED!
Two very fast objects in motion

Discrete Collision Detection

Continuous Collision Detection

Convex

Concave
Chapter 6: Dynamic Graphics

CPU

Game → Graphics API (OpenGL, DirectX) → Hardware Driver

CPU-GPU Boundary

Front End

Rasterization ← Primitive Assembly ← Vertex Shaders

Back End

Pixel (Fragment) Shaders → Frame Buffer

GPU
Enable Instancing

Batches: 3052

Enable Instancing

Batches: 16

Scene View

Overdraw

Occlusion Culling Disabled

Occlusion Culling Enabled
Automatic culling is disabled because:
- Local space simulation is not being used.
- Trails module is enabled.
Chapter 7: Virtual Velocity and Augmented Acceleration

![Graph showing the lifecycle of a technology innovation process, including stages like Peak of Inflated Expectations, Plateau of Productivity, and Trough of Disillusionment.]

- **Multi-Pass Stereo Rendering**: Two Frame Buffers, unique screenspace coordinates.
- **Single-Pass Stereo Rendering**: Shared Frame Buffer, shared screenspace coordinates.
Chapter 8: Masterful Memory Management

1. Empty Heap Space

2. Four 64-byte objects allocated

3. Objects A and C deallocated

4. New 128-byte object allocation

Must be allocated in the next contiguous block large enough to fit the new object.

Reserved Total: 241.4 MB  Unity: 199.0 MB  Mono: 10.7 MB  GfxDriver: 15.8 MB  FMOD: 1.3 MB  Video: 224 B  Profiler: 16.0 MB
Total System Memory Usage: 0.78 GB
11 Orcs (5 active, 6 inactive)
8 Trolls (3 active, 5 inactive)
5 Ogres (2 active, 3 inactive)
1 Dragon (1 active)

New Orc is spawned
1. Determine which Pool corresponds to the given Prefab
2. The first inactive Orc in the Inactive Group (Orc7) is activated - the corresponding object in the Heap is therefore activated
3. Newly-spawned Orc is moved to Active group
**Ogre3 is despawned**
1. Determine which Pool corresponds to the given Object.
2. Deactivate Ogre3 - the corresponding object in the Heap is therefore deactivated.
3. Move Ogre3 to Inactive group.

**Heap Memory**

**Pooling System**

**Inactive**
- Orc3
- Orc4
- Orc5
- Orc10
- Troll1
- Troll2
- Troll3
- Troll4
- Troll5
- Troll6

**Active**
- Orc1
- Orc5
- Orc2
- Orc6
- Orc10
- Troll1
- Troll2
- Troll3
- Troll4
- Troll5

---

**New Dragon is spawned**
1. Determine which Pool corresponds to the given Prefab.
2. Inactive group is empty, so a new Instance of Dragon must be created.
3. Instantiate a new Dragon from the Prefab on the heap.
4. Add the newly-created Dragon to the Active list.

**Heap Memory**

**Pooling System**

**Inactive**
- Orc3
- Orc4
- Orc5
- Orc10
- Troll1
- Troll2
- Troll3
- Troll4
- Troll5
- Troll6

**Active**
- Orc1
- Orc5
- Orc2
- Orc6
- Orc10
- Troll1
- Troll2
- Troll3
- Troll4
- Troll5

---
Chapter 9: Tactical Tips and Tricks
<table>
<thead>
<tr>
<th>Layers</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Everything</td>
<td>[ ]</td>
</tr>
<tr>
<td>Nothing</td>
<td>[ ]</td>
</tr>
<tr>
<td>0: Default</td>
<td>[ ]</td>
</tr>
<tr>
<td>1: TransparentFX</td>
<td>[ ]</td>
</tr>
<tr>
<td>2: Ignore Raycast</td>
<td>[ ]</td>
</tr>
<tr>
<td>4: Water</td>
<td>[ ]</td>
</tr>
<tr>
<td>5: UI</td>
<td>[ ]</td>
</tr>
<tr>
<td>8: Reflection_Probes</td>
<td>[ ]</td>
</tr>
<tr>
<td>9: Lighting</td>
<td>[ ]</td>
</tr>
<tr>
<td>10: Props</td>
<td>[ ]</td>
</tr>
<tr>
<td>11: ReflectedInWater</td>
<td>[ ]</td>
</tr>
<tr>
<td>12: Ground</td>
<td>[ ]</td>
</tr>
<tr>
<td>13: AccessibleVolume</td>
<td>[ ]</td>
</tr>
</tbody>
</table>

This is a very specific kind of log message.