Unity 2022.3.4f1 – Condensed Cheatsheet

A balanced, streamlined reference with C# commands and key Unity Editor workflows.

Editor Overview

- **Hierarchy**: Scene objects.
- **Scene View**: 3D/2D workspace.
- **Inspector**: Adjust properties.
- **Project**: Asset management.
- **Console**: Logs & errors.
- **Settings**: `Edit ▸ Project Settings…`.
- **Package Manager**: `Window ▸ Package Manager`.

C# Essentials

Basic Script

```
using UnityEngine;

public class Mover : MonoBehaviour
{
    [SerializeField] float speed = 5f;
    void Update()
    {
        float h = Input.GetAxis("Horizontal");
        float v = Input.GetAxis("Vertical");
        Vector3 dir = new Vector3(h, 0f, v);
        transform.Translate(dir * speed * Time.deltaTime, Space.World);
    }
}
```

Lifecycle

- `Awake()`, `Start()`, `Update()`, `FixedUpdate()`, `LateUpdate()`, `OnDestroy()`.

Common Actions

```
// Get components
Rigidbody rb = GetComponent<Rigidbody>();
var cam = GetComponentInChildren<Camera>();

// Instantiate / Destroy
var b = Instantiate(prefab, pos, rot);
Destroy(b, 3f);
```

```
// Coroutine
StartCoroutine(MyRoutine());
IEnumerator MyRoutine() { yield return new WaitForSeconds(1f); }

// Physics
rb.AddForce(Vector3.up * 5f);
if (Physics.Raycast(transform.position, transform.forward, out var hit, 10f))
    Debug.Log(hit.collider.name);

// Math & rotation
transform.position = Vector3.Lerp(a, b, 0.5f);
transform.rotation = Quaternion.RotateTowards(transform.rotation, targetRot,
90f * Time.deltaTime);

// Scene load
SceneManager.LoadScene("Level1");
```

Editor Workflows

Scripts & Prefabs

- **Add Script**: Create ▸ C# Script ▸ Drag to object.
- **Prefab**: Drag GameObject to Project. Edit via Prefab Mode.

Materials & Lighting

- **Materials**: Create ▸ Material, assign shader and drag to mesh.
- **Lighting**: `Window ▸ Rendering ▸ Lighting`. Adjust Environment and Bake settings.

Physics Setup

- **Rigidbody** for movement.
- **Colliders** for collisions.
- **Project Settings ▸ Physics** to manage Layer Collision Matrix.

Animation & Animator

1. `Window ▸ Animation ▸ Animation`.
2. Create Clip, record keyframes.
3. Use Animator Controller, states, and transitions with parameters.

UI (uGUI)

- Create Canvas ▸ Add Button/Text (TMP).
- Adjust anchors with RectTransform.
- Button OnClick: assign target method in Inspector.

UI Toolkit

- `Window ▸ UI Toolkit ▸ UI Builder`.
- Add **UI Document** component.
- Query elements in code:

```
var root = GetComponent<UIDocument>().rootVisualElement;
var playBtn = root.Q<Button>("PlayButton");
playBtn.clicked += () => Debug.Log("Play!");
```

Input System

- Install via Package Manager.
- Create Input Actions asset.
- Add **PlayerInput** component to player.
- Implement callbacks in code.

Audio

- Add AudioSource, assign clip.
- Use Audio Mixer for groups.
- Adjust via `audioMixer.SetFloat("Volume", val);`.

Navigation & AI

- `Window ▸ AI ▸ Navigation` ▸ Bake.
- Add **NavMeshAgent**, set destination via script.

Cameras

- Install **Cinemachine**.
- Add Virtual Camera, set Follow/Look At targets.

Build

- `File ▸ Build Settings...` ▸ Add Scenes ▸ Build.
- Adjust Player Settings (resolution, icon, etc.).

Inspector Reference

- **Transform**: Reset pos/rot/scale.
- **Rigidbody**: Mass, Gravity, Kinematic.
- **Collider**: Enable Trigger.
- **Light**: Intensity, Color, Shadows.
- **Camera**: Projection, FOV.

Shortcuts

- **F**: Frame selected.
- **Q/W/E/R/T**: Select/Move/Rotate/Scale/Rect.
- **Ctrl+D**: Duplicate.
- **Alt+Drag**: Orbit view.
- **Shift+Space**: Maximize panel.
- **Ctrl+P**: Play/Stop.

Tips

- Multiply movement by `Time.deltaTime`.
- Physics in FixedUpdate, input in Update.
- Use SerializeField for private Inspector fields.
- Cache components in Awake/Start.
- Avoid expensive `Find` calls in Update.

End of Condensed Cheatsheet.