**Shared component data**

Shared components are a special kind of data component that you can use to subdivide entities based on the specific values in the shared component (in addition to their archetype). When you add a shared component to an entity, the EntityManager places all entities with the same shared data values into the same chunk.

Shared components allow your systems to process like entities together. For example, the shared component Rendering.RenderMesh, which is part of the Hybrid.rendering package, defines several fields, including **mesh**, **material**, and **receiveShadows**. When your application renders, it is most efficient to process all of the 3D objects that have the same value for those fields together. Because a shared component specifies these properties, the EntityManager places the matching entities together in memory so that the rendering system can efficiently iterate over them.

**Note:** If you over-use shared components, it might lead to poor chunk utilization. This is because when you use a shared component it involves a combinatorial expansion of the number of memory chunks based on archetype and every unique value of each shared component field. As such, avoid adding any fields that aren't needed to sort entities into a category to a shared component. To view chunk utilization, use the [Entity Debugger](https://docs.unity3d.com/Packages/com.unity.entities@0.9/manual/ecs_debugging.html).

If you add or remove a component from an entity, or change the value of a shared component, The EntityManager moves the entity to a different chunk, and creates a new chunk if necessary.

You should use [IComponentData](https://docs.unity3d.com/Packages/com.unity.entities@0.9/api/Unity.Entities.IComponentData.html) for data that varies between entities, such as storing a World position, agent hit points, or particle time-to-live. In contrast, you should use [ISharedComponentData](https://docs.unity3d.com/Packages/com.unity.entities@0.9/api/Unity.Entities.ISharedComponentData.html) when a lot of entities share something in common. For example in the Boids demo in the DOTS package, a lot of entities instantiate from the same [Prefab](https://docs.unity3d.com/Manual/Prefabs.html) and as a result, the RenderMesh between many Boid entities is exactly the same.

[System.Serializable]

public struct RenderMesh : ISharedComponentData

{

public Mesh mesh;

public Material material;

public ShadowCastingMode castShadows;

public bool receiveShadows;

}

ISharedComponentData has zero memory cost on a per entity basis. You can use ISharedComponentData to group together all entities that have the same InstanceRenderer data, and then efficiently extract all matrices for rendering. The resulting code is simple and efficient because the data is laid out as ECS accesses it.

For an example of this, see the RenderMeshSystemV2 file Packages/com.unity.entities/Unity.Rendering.Hybrid/RenderMeshSystemV2.cs.

**Important notes about SharedComponentData:**

* ECS groups entities with the same SharedComponentData together in the same [chunks](https://docs.unity3d.com/Packages/com.unity.entities@0.9/manual/chunk_iteration.html). It stores the index to the SharedComponentData once per chunk, not per entity. As a result, SharedComponentData has zero memory overhead on a per entity basis.
* You can use [EntityQuery](xref:Entities.EntityQuery) to iterate over all entities with the same type. You can also use [EntityQuery.SetFilter()](https://docs.unity3d.com/Packages/com.unity.entities@0.9/api/Unity.Entities.EntityQuery.html" \l "Unity_Entities_EntityQuery_SetSharedComponentFilter_) to iterate specifically over entities that have a specific SharedComponentData value. Because of the data layout, this iteration has a low overhead.
* You can use EntityManager.GetAllUniqueSharedComponents to retrieve all unique SharedComponentData that is added to any alive entities.
* ECS automatically [reference counts](https://en.wikipedia.org/wiki/Reference_counting) SharedComponentData.
* SharedComponentData should change rarely. If you want to change a SharedComponentData, it involves using [memcpy](https://msdn.microsoft.com/en-us/library/aa246468(v=vs.60).aspx) to copy all ComponentData for that entity into a different chunk.