1. 摄像机

Camera 在 Bevy 中驱动所有渲染。它们负责配置绘制内容、绘制方式和绘制位置。

你必须至少拥有一个摄像机实体，才能显示任何内容！如果忘记生成摄像机，会看到一个空的黑屏。

在最简单的情况下，可以使用默认设置创建摄像机。只需使用 Camera2dBundle 或 Camera3dBundle 生成一个实体。它会绘制所有可见的可渲染实体。

实用建议：始终为你的摄像机实体创建标记组件，这样可以方便地查询摄像机。

```
#[derive(Component)]
struct MyGameCamera;

fn setup(mut commands: Commands) {
  commands.spawn((
    Camera2dBundle::default(),
    MyGameCamera,
  ));
}
```

2. 摄像机 Transform

摄像机拥有 Transform，可以用于定位或旋转摄像机。这就是移动摄像机的方法。

3. 缩放摄像机

不要使用 Transform 来缩放摄像机！这只是拉伸图像，并不是真正的缩放。这可能还会导致其他问题和不兼容性，应该使用 Projection 来缩放。

对于正交投影，改变缩放比例。对于透视投影，改变视场（FOV）。视场模拟镜头缩放效果。

```
fn scale_camera(mut projection: Query<&mut Projection,
With<MyCamera>>) {
    let Ok(projection) = projection.get_single_mut() else {
        return;
    };
    match projection.into_inner() {
        Projection::Orthographic(projection) => {
            if (projection.scale - 0.15).abs() <=
f32::EPSILON {
                projection.scale = 0.05;
            } else {
                projection.scale = 0.15;
            }
        }
        Projection::Perspective(projection) => {
            if (projection.fov - 0.785).abs() <=
f32::EPSILON {
                projection.fov = 1.2;
            } else {
                projection.fov = 0.785;
            }
        }
    }
}
```

4. Projection（投影）

摄像机投影负责将坐标系映射到视口（通常是屏幕/窗口）。它配置坐标空间以及图像的任何缩放/拉伸。

Bevy 提供两种投影：正交投影和透视投影。它们是可配置的，以服务于各种不同的使用场景。

正交投影意味着无论物体距离摄像机多远，大小始终相同。

透视投影意味着物体距离摄像机越远，看起来越小。这是为 3D 图形提供深度和距离感的效果。

2D 摄像机始终是正交的。

3D 摄像机可以使用任一种投影。透视是最常见（也是默认）的选择。正交投影适用于如 CAD 和工程等应用，在这些应用中，你希望准确表示物体的尺寸，而不是创造逼真的 3D 空间感。一些游戏（尤其是模拟游戏）出于艺术选择使用正交投影。

可以实现自定义摄像机投影。这可以让你完全控制坐标系统。不过，请注意，如果违反 Bevy 的坐标系统约定，可能会导致行为异常！

```
fn toggle_perspective_orthographic(mut projection:
Query<&mut Projection, With<MyCamera>>) {
    let Ok(mut projection) = projection.get_single_mut()
else {
        return;
    };
    if let Projection::Perspective(_) = projection.as_ref()
{
        *projection =
Projection::Orthographic(OrthographicProjection {
            scale: 0.15,
            ..default()
        });
    } else {
        *projection =
Projection::Perspective(PerspectiveProjection::default());
    }
}
```

5. 渲染层

RenderLayers 是一种过滤哪些实体应该由哪些相机绘制的方法。将此组件插入到您的实体上，以将它们放置在特定的“层”中。

将此组件插入到相机实体上可以选择该相机应该渲染哪些层。将此组件插入到可渲染实体上可以选择哪些相机应该渲染这些实体。如果相机的层和实体的层之间有任何重叠（它们至少有一个共同的层），则该实体将被渲染。

如果实体没有 RenderLayers 组件，则假定它属于第 0 层（仅此一层）。

您还可以在实体生成后修改其渲染层

```
fn toggle_render_layers(mut render_layers: Query<&mut
RenderLayers>) {
    for mut render_layer in &mut render_layers {
        if render_layer.iter().next().unwrap() == 0 {
            *render_layer = RenderLayers::layer(1);
        } else {
            *render_layer = RenderLayers::layer(0);
        }
    }
}
```

6. 禁用摄像机

您可以在不销毁相机的情况下停用它。这在您想保留相机实体及其携带的所有配置，以便以后可以轻松重新启用时非常有用。

```
fn toggle_camera_active(mut camera: Query<&mut Camera,
With<MyCamera>>) {
    let Ok(mut camera) = camera.get_single_mut() else {
        return;
    };
    if camera.is_active {
        camera.is_active = false;
    } else {
        camera.is_active = true;
    }
}
```