1. Transform 变换

Transform 允许你将对象放置到游戏世界中。它是对象的“平移”（位置/坐标）、“旋转”和“缩放”的组合。

你可以通过修改 translation 来移动对象，通过修改 rotation 来旋转对象，通过修改 scale 来缩放对象。

```
fn transform_red_rectangle(
    mut right: Local<bool>,
    mut rectangle: Query<&mut Transform,
With<RedRectangle>>,
    time: Res<Time>,
) {
    let Ok(mut transform) = rectangle.get_single_mut() else
{
        return;
    };

    // 平移方向
    if transform.translation.x < -300. {
        *right = false;
    } else if transform.translation.x > 300. {
        *right = true;
    }

    // 平移和缩放
    let velocity = Dir3::X * 200. * time.delta_seconds();
    let scale = time.delta_seconds() / 3.;
    if *right {
        transform.translation -= velocity;
        transform.scale -= scale;
    } else {
        transform.translation += velocity;
        transform.scale += scale;
    }

    // 旋转
    transform.rotate_z(time.delta_seconds());
}
```

2. Transform 组件

在 Bevy 中，变换由两个组件表示：Transform 和 GlobalTransform。

任何代表游戏世界中对象的实体都需要有这两个组件。如果你要创建自定义实体，可以使用 TransformBundle 来同时添加这两个组件，当然你也可以分别添加 Transform 和 GlobalTransform。

```
commands.spawn((
    Mesh2dHandle(meshes.add(Capsule2d::new(40., 100.))),
    materials.add(Color::Srgba(css::BLUE)),
    TransformBundle {
        local: Transform::from_xyz(0., -120., 0.),
        global: GlobalTransform::default(),
    },
    VisibilityBundle::default(),
    Name::new("Blue Capsule 2d"),
));
```

2.1 Transform

Transform 是你经常使用的内容。它是一个包含平移、旋转和缩放的 struct 。要读取或操作这些值，请使用查询从你的系统访问它。

如果实体有父级，则 Transform 组件是相对于父级的。这意味着子对象将与父对象一起移动/旋转/缩放。

```
let circle = commands
    .spawn((
        MaterialMesh2dBundle {
            mesh:
Mesh2dHandle(meshes.add(Circle::new(50.))),
            material:
materials.add(Color::Srgba(css::ORANGE)),
            transform: Transform::from_xyz(220., 0., 0.),
            ..default()
        },
        OrangeCircle,
        Name::new("Orange Circle"),
    ))
    .id();

commands
    .spawn((
        MaterialMesh2dBundle {
            mesh:
Mesh2dHandle(meshes.add(Rectangle::new(300., 150.))),
            material: materials.add(Color::Srgba(css::RED)),
            transform: Transform::from_xyz(0., 185., 0.),
            ..default()
        },
        RedRectangle,
        Name::new("Red Rectangle"),
    ))
    .add_child(circle);
```

2.2 GlobalTransform

GlobalTransform 是相对于全局空间的，如果实体没有父级，Transform 与 GlobalTransform 相等。

GlobalTransform 的值由 Bevy （“变换传播”）在内部计算/管理的。

与 Transform 不同，平移/旋转/缩放不能直接访问。数据以优化的方式存储（使用 Affine3A），并且可以在层次结构中进行无法表示为简单变换的复杂变换。

如果你想尝试将 GlobalTransform 转换回可用的平移/旋转/缩放表示，你可以尝试以下方法：

- .translation()
- .to_scale_rotation_translation()
- .compute_transform()

3. Transform 传播

这两个组件由一组内部系统（“变换传播系统”）同步，该系统在 PostUpdate Schedule 中运行。

注意：当你改变 Transform 时，GlobalTransform 不会立即更新。在变化传播系统运行之前，它们将不会同步。

如果你需要直接使用 GlobalTransform。你应该将你的系统添加到 PostUpdate Schedule 中，并在 TransformSystem::TransformPropagate 之后进行排序。

```
.add_systems(
        PostUpdate,
        circle_global_transform_info
            .after(TransformSystem::TransformPropagate),
    )
```