ITENS DENTRO DE BLOCOS

Prof. Me. Hélio Esperidião

O que é transform.parent?

No Unity, todo GameObject tem um Transform, que representa posição, rotação e escala no espaço.

transform.parent define quem é o pai daquele objeto na hierarquia.

- - Jogador (Transform)
- └── Câmera (Transform)

O parente de câmera é o jogador

- Ou seja, a câmera **está "presa" ao jogador**, e **vai se mover junto** com ele.

transform.parent

transform.parent é uma propriedade no Unity que permite acessar ou definir o pai de um Transform.

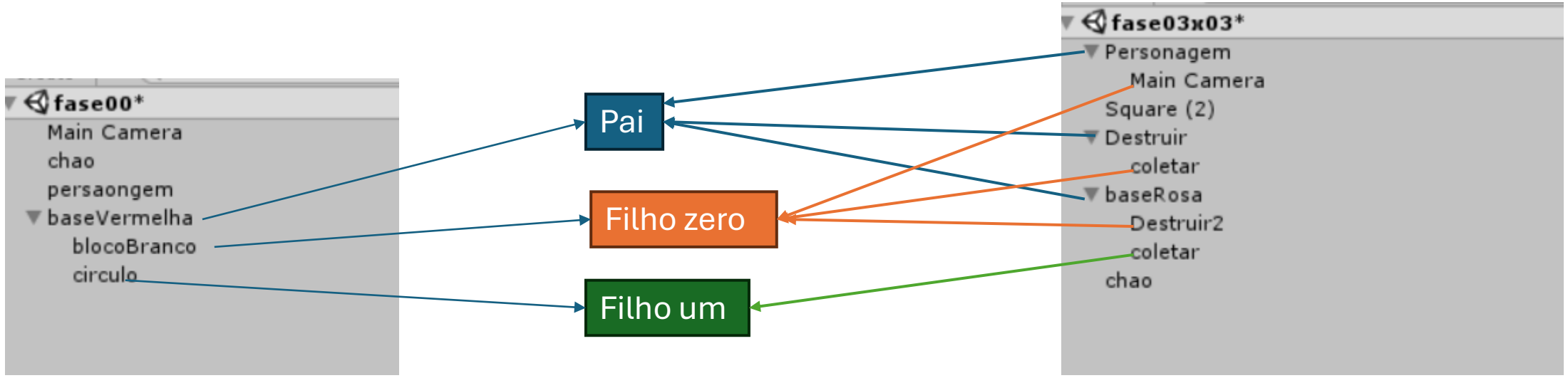
A hierarquia de objetos no Unity é organizada em uma estrutura de árvore, onde cada objeto pode ter zero ou mais filhos e, opcionalmente, um pai.

A propriedade transform.parent é usada para navegar e modificar essa hierarquia.

Analise de parent.

- Personagem é parent de main Camera.
- Coletar é parent de Destruir
- baseRosa é parente de Destruir2 e coletar

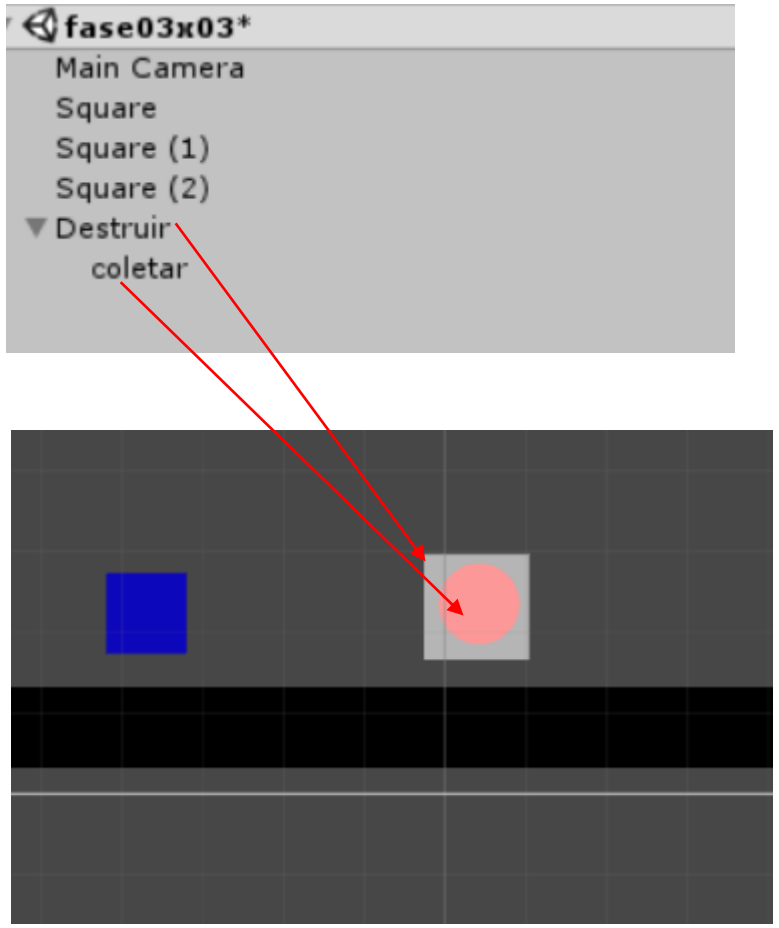
Pais e Filhos



Desanexar um objeto da hierarquia

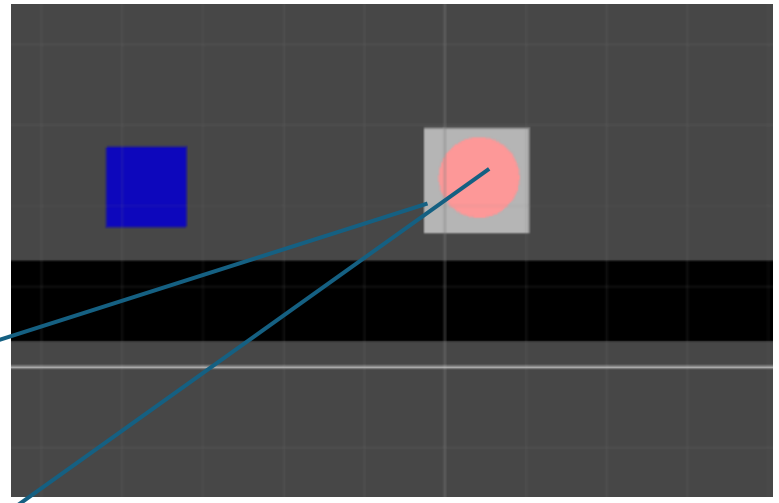
- transformObjeto.parent = null;

Configuração inicial



- Posicione dentro do objeto Destruir o elemento que será coletado.
  - O objeto Destruir deve ser configurado com a tag: destrutivel1
  - É importante definir o order in layer corretamente
- Esse lemento pode ser configurado com alguma tag.
- O bloco destruir foi configurado com transparência para fins didáticos, dentro do seu jogo não é necessária a utilização de transparência.

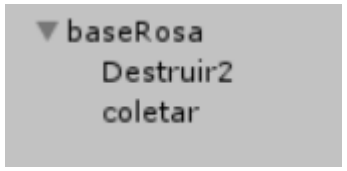
Toque em qualquer lugar de: destrutivel1

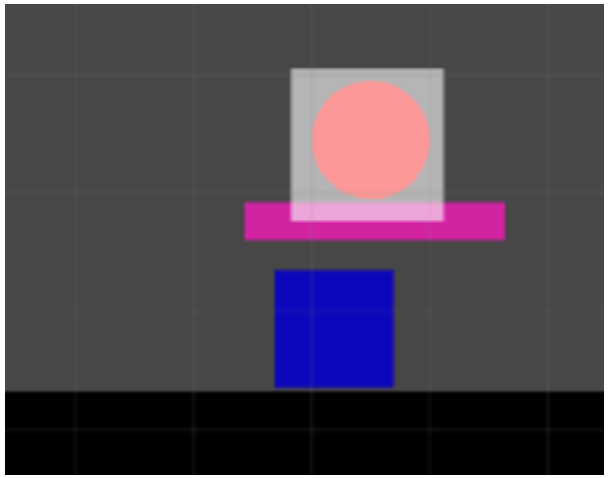


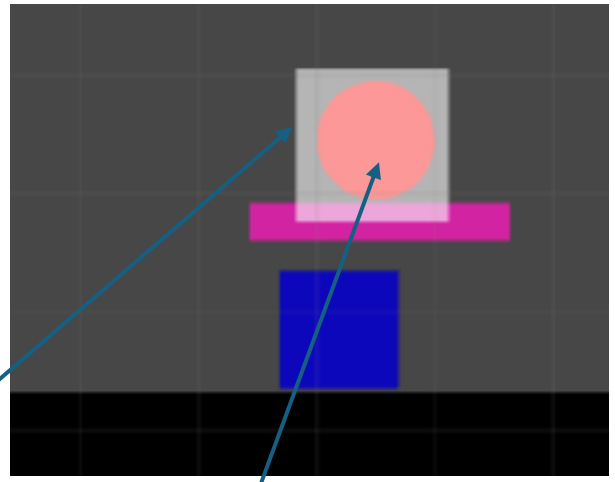
```
    void OnCollisionEnter2D(Collision2D objetoTocado ){
        TagObjetoTocado = objetoTocado.gameObject.tag;


        if (TagObjetoTocado.Contains("destrutivel1"))  {

                GameObject circulo = objetoTocado.transform.GetChild(0).gameObject; // Pega o filho um


                circulo.transform.parent = null; // Desvincula os filhos

                circulo.AddComponent<BoxCollider2D>(); // Adiciona um boxCollider2d ao circulo
                circulo.AddComponent<Rigidbody2D>();
                circulo.GetComponent<Rigidbody2D> ().velocity = new Vector2 (0, 10);
                circulo.tag = "Bonus1";
                //circulo.GetComponent<BoxCollider2D>().isTrigger = true;

            Destroy(objetoTocado.gameObject);


        }
}
```

Configuração Inicial

- Posicione dentro da barra vermelha o blocoBranco e depois o circulo vemelho.
- É importante definir o order in layer corretamente
- A ordem dos elementos faz diferença.
- Observe o exemplo de hierarquia.



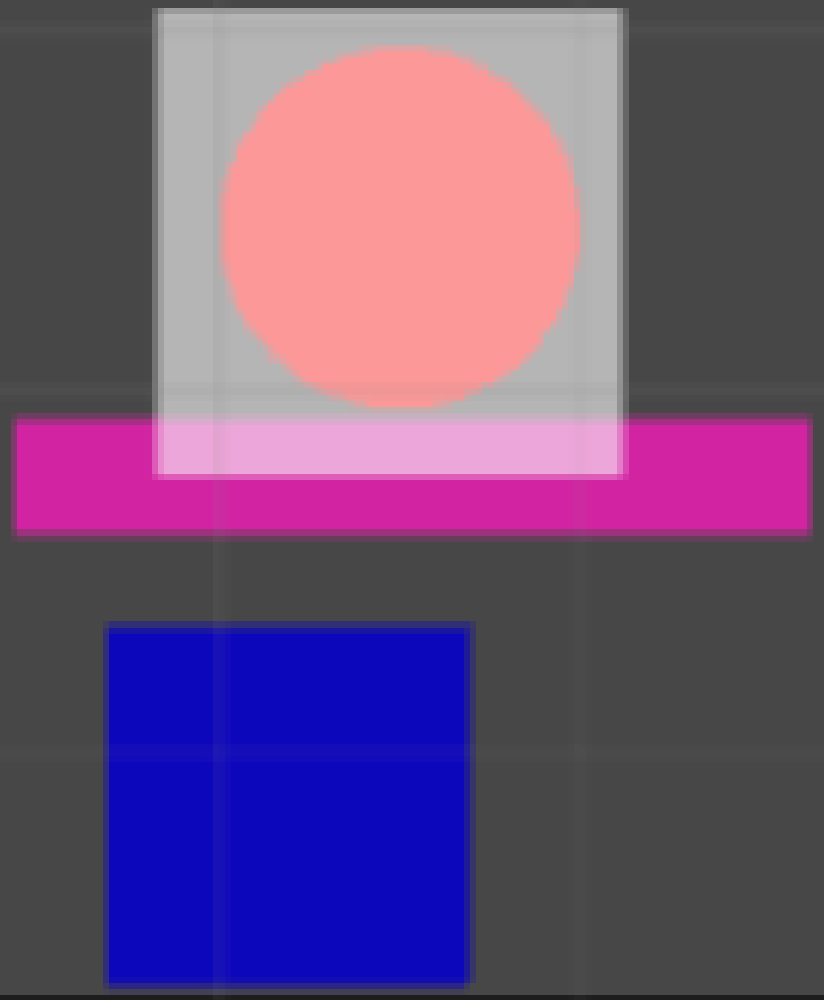




```
void OnCollisionEnter2D(Collision2D objetoTocado ){
    TagObjetoTocado = objetoTocado.gameObject.tag;
    if (TagObjetoTocado.Contains("destrutivel2"))  {
        GameObject quadradoBranco = objetoTocado.transform.GetChild(0).gameObject; // Pega o filho um
        GameObject circulo = objetoTocado.transform.GetChild(1).gameObject; // Pega o filho um

        circulo.transform.parent = null; // Desvincula os filhos
        quadradoBranco.transform.parent = null;  // Desvincula os filhos

        circulo.AddComponent<BoxCollider2D>(); // Adiciona um boxCollider2d ao circulo
        circulo.AddComponent<Rigidbody2D>();
        circulo.GetComponent<Rigidbody2D> ().velocity = new Vector2 (0, 10);
        circulo.tag = "Bonus1";
        //circulo.GetComponent<BoxCollider2D>().isTrigger = true;

        Destroy(quadradoBranco.gameObject);
        Destroy(objetoTocado.gameObject);

    }
}
```

Organize os assets

- Diminua o tamanho da base rosa
- Deixe a base rosa completamente transparente.
- Não deixe o quadrado branco transparente.



Organização.

- Crie um bloco de código exclusivo para verificação dos itens destrutíveis.

```
void VerificarDestrutivel(Collision2D objetoTocado ){
    //TagObjetoTocado = objetoTocado.gameObject.tag;
    if (TagObjetoTocado.Contains("destrutivel2"))  {
        GameObject quadradoBranco = objetoTocado.transform.GetChild(0).gameObject; // Pega o filho um
        GameObject circulo = objetoTocado.transform.GetChild(1).gameObject; // Pega o filho um


        circulo.transform.parent = null; // Desvincula os filhos
        quadradoBranco.transform.parent = null;  // Desvincula os filhos

        circulo.AddComponent<BoxCollider2D>(); // Adiciona um boxCollider2d ao circulo
        circulo.AddComponent<Rigidbody2D>();
        circulo.GetComponent<Rigidbody2D> ().velocity = new Vector2 (0, 10);
        circulo.tag = "Bonus1";
        //circulo.GetComponent<BoxCollider2D>().isTrigger = true;
        Destroy(quadradoBranco.gameObject);
        Destroy(objetoTocado.gameObject);
    }
    if (TagObjetoTocado.Contains("destrutivel1"))  {
        GameObject circulo = objetoTocado.transform.GetChild(0).gameObject; // Pega o filho um
        //blocoBranco.transform.parent = null; // Desvincula os filhos
        circulo.transform.parent = null; // Desvincula os filhos
        circulo.AddComponent<BoxCollider2D>(); // Adiciona um boxCollider2d ao circulo
        circulo.AddComponent<Rigidbody2D>();
        circulo.GetComponent<Rigidbody2D> ().velocity = new Vector2 (0, 10);
        circulo.tag = "Bonus1";
        //circulo.GetComponent<BoxCollider2D>().isTrigger = true;
        Destroy(objetoTocado.gameObject);
    }
}
```

Onde chamar o `VerificarDestrutivel()`?

- Chame dentro de:

```
void OnCollisionEnter2D(Collision2D objetoTocado ){

        TagObjetoTocado = objetoTocado.gameObject.tag;
        VerificarDestrutivel (objetoTocado);

}
```