Abschlusspräsentation

Moncif Souhali, Xue Ming Chen

Ergebnsisse

- Level 1 - Level 3
- Fähigkeiten Eis, Feuer, Stein, Wasser
- Mit ihrer Ultimativen Fähigkeiten
- Wall Jumping
- Monster Design und Animationen
- Sounds

Behavior Tree

1. Ergebnisse
2. Vor- und Nachteile
3. Problemen und Lösungen
4. Erkenntnisse und Verbesserungen
5. Einstiegsempfehlung

Ergebnisse — Behaviour Tree

RangeMonster

- Attack
- Chasing
- Patrol

HealMonster

- Heal enemy
- Distance
- Attack
- Chasing
- Patrol

Ergebnisse - Behavior Tree

MidRangeMonster

- Guard
- Attack
- Chasing
- Patrol

Ergebnisse – Behavior Tree

Boss 1

- 2 Phase
- Kann Schießen und Nahkampf
- Phase 2 Dashangriff und Riesenschuss als Spezialattacken

Boss 2

- Dodge(Jump)
- 3 Phasen
- 1. Phase: normaler Schuss
- 2. Phase: verfolgbare Schuss
- 3. Phase: Laserangriff

Ergebnisse - Behavior Tree

Boss 3

- Teleport
- 3 Phasen
- 1. Phase: Nahkampfangriff
- 2. Phase: Flamme am Gegner
- 3. Phase: Regen voller Eis

Behavior Tree Vor- und Nachteile

Vorteile	Nachteile
Modularität	Leistungsproblem
Kompakte Strukturen	hoher Speicherbedarf
Übersichtlich (z. B. Behavior Designer)	unübersichtlich (eigene Implementierung)
Flexibilität	

Probleme & Herausforderungen (BT)

Freeze

- MoveTowards
- ein kompletter Teilbaum für jedes Monster
- Animation Stop und Gegner bleibt an der Stelle

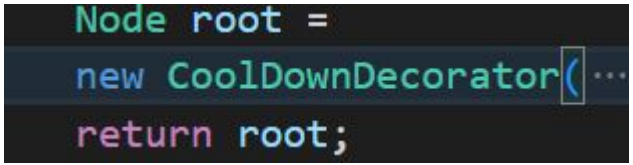
```
new Selector(
    new List<Node>{
     new Sequence(
        new List<Node>{
        new CheckFreezed(transform),
        new TaskToFreezed(transform)
        }
     ),
      new Sequence(
        new List<Node>{
        new CheckUnfreezed(transform),
        new TaskToUnfreezed(transform)
        }
     ),
```

Herausforderungen & Lösung (BT)

Boss Cooldown

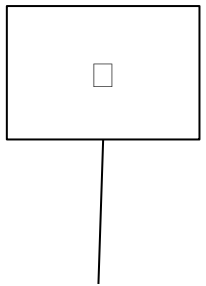
- Decorator Node
- Als Wurzel
- gibt das zurück, was in dem

  Kind des Knotens als State hatte



Exkurs: Decorator Node

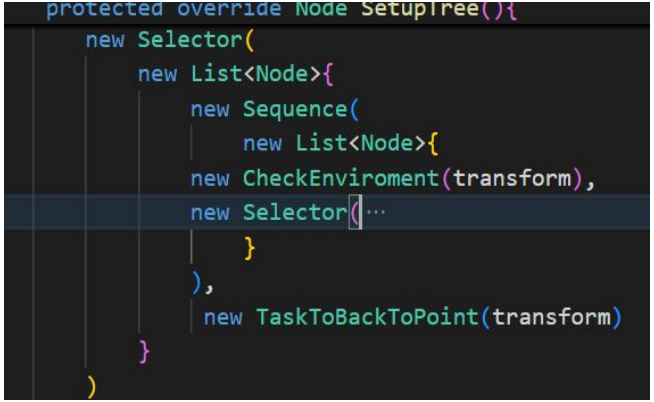
- Composite Node
- hat nur ein Kind
- Design Pattern Decorator



Herausforderungen & Lösung (BT)

Boss range

- Über den eigentlichen Baum Zusatzbedingung
- ganz am Ende des Baumes nach einem Task, der zurück zum Startpunkt geht



Herausforderungen & Lösung (BT)

Animationen mit BT

- in dem Animation-Controller schauen, wo fixed Time an oder aus sein sollte
- Timer in den Knoten für das Triggern der Animation
- bei manchen Animationen Loop ausschalten

Erkenntnisse & Verbesserungen

Erkenntnisse

- Speicherung von Gegnerpositionen durch eine Dictionary
- Generischer Baum und Composite Nodes bleiben gleich

Verbesserung

- bei gleichen Knoten, die generisch zu halten Bsp.: CheckToGo und TaskToGo

Empfehlung für den Einstieg (BT)

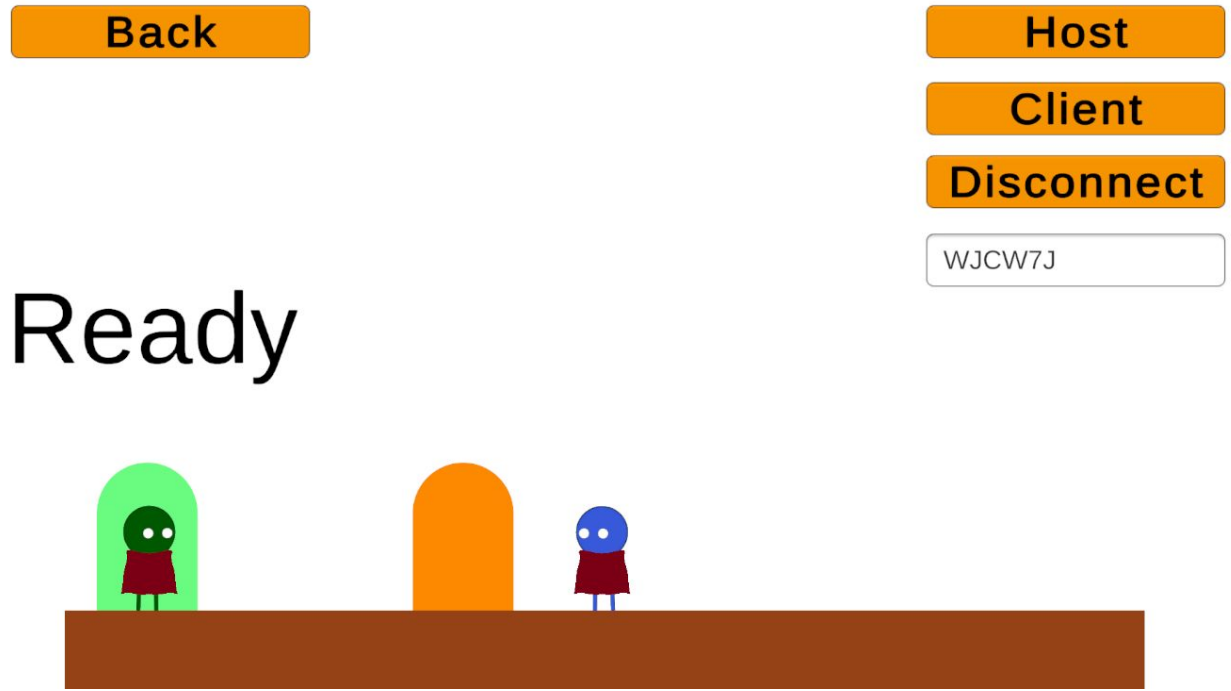
Multiplayer

1. Lobby
2. Synchronisierung
3. Multiplayer-Aktionen

Lobby

Unity for GameObjects

Unity Relay



Synchronisierung (am Beispiel “Dash-Attacke”)

```
void OnCollisionEnter2D(Collision2D other){
    if(dash && other.gameObject.CompareTag("Enemy")) {
            if (multiplayer) {
                float enemyID = other.gameObject.GetComponent<Health>().myid;
                if (OwnerClientId == 0) {
                    dashDamagaeClientRpc(enemyID);
                }
                else {
                    dashDamagaeServerRpc(enemyID);
                    other.gameObject.GetComponent<Health>().takeDamage(damage);
                }
            }
            else {
                other.gameObject.GetComponent<Health>().takeDamage(damage);
            }
    }
```

```
[ClientRpc]
1 usage  Moncif
private void dashDamagaeClientRpc(float enemyID) {
    GameObject[] enemies;
    enemies = GameObject.FindGameObjectsWithTag("Enemy");

    foreach (GameObject enemy in enemies)
    {
        if (enemy.GetComponent<Health>().myid == enemyID) {
            enemy.GetComponent<Health>().takeDamage(damage);
        }
    }
}
```

Multiplayer-Aktionen

- Partner-Kamera
- Fähigkeit abgeben
- Partner mit Wasser-Fähigkeit heilen

```
[ServerRpc]
Frequently called  2 usages  Moncif
private void dropItemServerRpc(float offset) {
    GameObject[] players;
    players = GameObject.FindGameObjectsWithTag("Player");
    foreach (GameObject player in players)
    {
        if (!IsOwner) {
            dropItemLocal(offset);
        }
    }
}
```

Multiplayer: Vor- und Nachteile

Vorteile

- Erhöhter Spielspaß
- Mehr Möglichkeiten für kreative Interaktionen

Nachteile

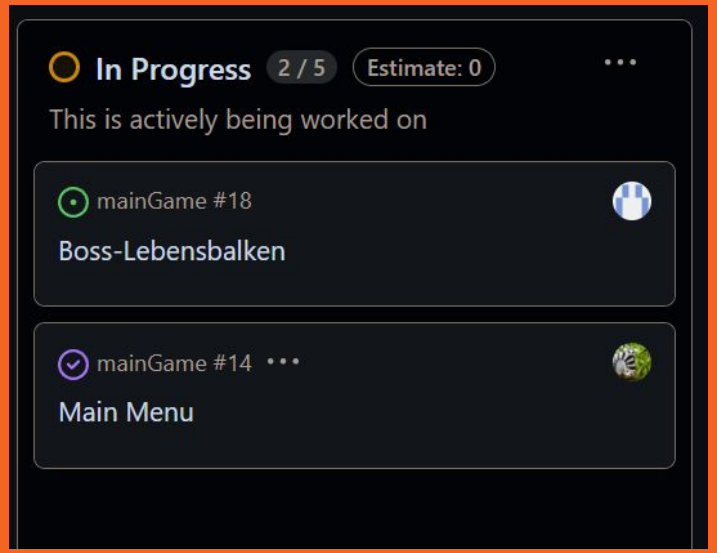
- Synchronisierung aller Zustände und Interaktionen sehr aufwändig

Ausbau des Projektes

1. Mehrere Level und Gegner
2. neue Fähigkeiten für den Spieler
3. Level 3 als eigenen Modus
4. Auswahl von Leveln, wenn man die geschafft hat

Projekt-management

- Gantt-chart
- Wöchentliche Meetings
- Kanban-Board



Noch Fragen?