Prisoner Teaching System - Setup Guide

🎓 Overview

When rescuing a prisoner, they teach the party one of their skills with **diminishing returns** based on the student's current ability. Players must strategically choose which character receives the training.

🎮 Player Experience Flow

1. Find Prisoner

> "You found your old friend Aldric!"

2. Initial Message

> "Freed from their shackles, the Warrior tells you of their hard
> times in the dungeon, only surviving on their Force.
>
> Aldric is willing to share their knowledge..."
>
> Press Space to continue...

3. Character Selection

> Select a character to receive Force training from Aldric:
>
> 1. Gwendolyn
>   Force: 10 → 55 (+45)
>
> 2. Theron
>   Force: 50 → 75 (+25)
>
> 3. Elara
>   Force: 20 → 60 (+40)
>
> Click name or press number key (1-3)

4. Upgrade Result

> "Theron trained with Aldric!
>
> Force: 50 → 75 (+25)
>
> Aldric thanks you and leaves for the exit.

📐 Diminishing Returns Formula

Base Calculation

actualUpgrade = baseUpgrade × (1 - currentValue/100)

Examples

Prisoner: Warrior teaching Force (base 50)

Current Force	Calculation	Actual Upgrade
0	50 × (1 - 0%)	+50
10	50 × (1 - 10%)	+45
25	50 × (1 - 25%)	+37
50	50 × (1 - 50%)	+25
75	50 × (1 - 75%)	+12
90	50 × (1 - 90%)	+5
100	50 × (1 - 100%)	+1 (minimum)

Key Points

- ✅ **Low stats** get full benefit (noobs learn fast)
- ✅ **High stats** get reduced benefit (experts learn slowly)
- ✅ **Minimum +1** always guaranteed
- ✅ **Strategic choice** matters (who needs it most?)

🎯 Prisoner Types & Skills

Prisoner Type	Skill Taught	Base Value
Warrior	Force	50
Paladin	Force	45
Mage	Esprit	50
Cleric	Esprit	45
Druid	Esprit	45
Warlock	Esprit	55
Bard	Esprit	40
Thief	Reflexe	50

Prisoner Type	Skill Taught	Base Value
Ranger	Reflexe	45
Monk	Reflexe	55

🛠️ Setup Instructions

1. Create UI Structure

```
Canvas
└── CharacterSelectionPanel
    ├── InstructionText (TextMeshPro)
    └── CharacterButtonContainer (Vertical Layout Group)
```

2. Create Button Prefab

```
CharacterSelectButton (Prefab)
├── Button Component
├── TextMeshProUGUI (for character info)
└── CharacterSelectButton Script
```

3. Create PrisonerTeachingUI GameObject

```
Hierarchy:
- PrisonerTeachingUI (GameObject)
  └── PrisonerTeachingUI (Component)

Inspector Settings:
- Character Selection Panel: [Drag panel]
- Character Button Container: [Drag container]
- Character Button Prefab: [Drag prefab]
- Instruction Text: [Drag text component]
```

4. Configure Panel

```
yaml


```

CharacterSelectionPanel:
  - Start: Hidden (Active = false)
  - Position: Center screen
  - Size: 400x300 (adjustable)
  - Background: Semi-transparent dark

CharacterButtonContainer:
  - Layout: Vertical
  - Spacing: 10
  - Padding: 20
  - Child Alignment: Upper Center

5. Configure Button Prefab

```yaml
yaml

CharacterSelectButton Prefab:
  - Width: 360
  - Height: 60
  - Font Size: 18
  - Text Alignment: Left
  - Colors:
    - Normal: Light gray
    - Hover: Bright white
    - Pressed: Yellow
```

🎨 UI Example Layout

```
┌─────────────────────────────────────┐
│ Select character for Force training: │
│                  │
│ ┌─────────────────────────────────┐ │
│ │ 1. Gwendolyn            │ │
│ │    Force: 10 → 55 (+45)       │ │
│ └─────────────────────────────────┘ │
│                  │
│ ┌─────────────────────────────────┐ │
│ │ 2. Theron             │ │
│ │    Force: 50 → 75 (+25)       │ │
│ └─────────────────────────────────┘ │
│                  │
│ ┌─────────────────────────────────┐ │
│ │ 3. Elara              │ │
│ │    Force: 20 → 60 (+40)       │ │
│ └─────────────────────────────────┘ │
```

```
|                    |
|_____|
```

🧠 Strategic Decisions

Scenario 1: Weak Character

Warrior prisoner teaching Force (base 50)

Character: Gwendolyn (Force 10)
Upgrade: $50 \times (1 - 10\%) = +45$
Result: $10 \rightarrow 55$

Strategy: Great for catching up weak characters!

Scenario 2: Strong Character

Warrior prisoner teaching Force (base 50)

Character: Theron (Force 60)
Upgrade: $50 \times (1 - 60\%) = +20$
Result: $60 \rightarrow 80$

Strategy: Diminished returns, but still useful

Scenario 3: Optimal Choice

You have:
- Tank: Force 70 (diminished)
- DPS: Force 30 (good gains)
- Healer: Force 10 (best gains)

Best choice: Healer gets +45
Good choice: DPS gets +35
OK choice: Tank gets +15

Scenario 4: Role Matching

Mage prisoner teaching Esprit (base 50)

Warrior: Esprit 5 → +47 (wasted on tank?)
Mage: Esprit 60 → +20 (makes sense!)
Thief: Esprit 15 → +42 (could be useful!)

Consider: Role vs. efficiency

📊 Progression Examples

Early Game (Days 1-5)

Party Stats: 10-30 range

Prisoner teaches: +40-45 each

Impact: Significant power boost

Mid Game (Days 6-12)

Party Stats: 30-60 range

Prisoner teaches: +20-35 each

Impact: Solid progression

Late Game (Days 13+)

Party Stats: 60-90 range

Prisoner teaches: +5-20 each

Impact: Diminishing but valuable

🎯 Balancing Guide

Base Values

Challenge Level	Base Upgrade
Easy	60-70
Normal	40-50
Hard	25-35
Very Hard	15-25

Adjusting Per Type

```csharp
// In PrisonerData.AssignSkillBasedOnType()

case "warrior":
    skillToTeach = PrisonerSkill.Force;
    baseUpgradeValue = 50; // ← Adjust this
    break;
```

Custom Diminishing Returns

```csharp
// In PrisonerData.CalculateUpgradeValue()

// Current formula (linear)
float reductionPercent = currentAttributeValue / 100f;

// Harsher diminishing (exponential)
float reductionPercent = Mathf.Pow(currentAttributeValue / 100f, 1.5f);

// Gentler diminishing (square root)
float reductionPercent = Mathf.Sqrt(currentAttributeValue / 100f);
```

🔧 Customization Options

Add New Skills

1. Add to enum:

```csharp
public enum PrisonerSkill
{
    Force,
    Reflexe,
    Esprit,
    Vitality, // ← New
    Luck      // ← New
}
```

2. Update GetCurrentAttributeValue():

```csharp
case PrisonerSkill.Vitality:
    return character.maxHealthPoints;
```

3. Update ApplyAttributeUpgrade():

```csharp
case PrisonerSkill.Vitality:
    character.maxHealthPoints += amount;
    break;
```

Multiple Skills Per Prisoner

```csharp
public List<PrisonerSkill> skillsToTeach; // Instead of single skill

// Player chooses skill, then character
```

Skill Specialties

```csharp
public float specialtyMultiplier = 1.5f;

// Warriors get 1.5x bonus when teaching Force
if (character.characterClass == "Warrior" && skill == Force)
    upgradeAmount = (int)(upgradeAmount * specialtyMultiplier);
```

Party-Wide Bonuses

```csharp
public bool teachAllCharacters = false;

// Everyone gets smaller upgrade instead of one big
```

🐛 Debug Tools

Show Calculations

```csharp
if (Input.GetKeyDown(KeyCode.F4))
{
    foreach (var character in party)
    {
        int current = character.force;
        int upgrade = prisoner.CalculateUpgradeValue(current);
        Debug.Log($"{character.characterName}: {current} → {current + upgrade} (+{upgrade})");
    }
}
```

Test All Upgrades

```csharp
```

```
[ContextMenu("Test Diminishing Returns")]
void TestDiminishingReturns()
{
    for (int stat = 0; stat <= 100; stat += 10)
    {
        int upgrade = CalculateUpgradeValue(stat);
        Debug.Log($"Stat {stat} → +{upgrade}");
    }
}
```

⚠️ Common Issues

Buttons don't appear

- Check button prefab assigned
- Verify container reference
- Check party members array populated

Wrong upgrade values

- Verify diminishing returns formula
- Check base upgrade values
- Test with Debug.Log

UI doesn't hide

- Ensure callback invokes
- Check panel activation
- Verify button cleanup

Keyboard input not working

- Check Update() runs when panel active
- Verify key codes match (Alpha1, Alpha2, etc.)
- Test with Debug.Log

✅ Testing Checklist

☐ PrisonerTeachingUI GameObject exists
☐ UI panel and components assigned
☐ Button prefab created and assigned
☐ Find prisoner shows teaching message
☐ Press Space shows character selection

- ☐ Buttons display correct calculations
- ☐ Click button applies upgrade
- ☐ Keyboard numbers work (1-9)
- ☐ Upgrade result message shows
- ☐ Stats actually increase
- ☐ Progress counter updates
- ☐ Movement re-enables after
- ☐ Diminishing returns work correctly
- ☐ Low stat = high gain
- ☐ High stat = low gain
- ☐ Multiple rescues work

💡 Design Tips

For Players:

- **Early characters** benefit most

- **Specialists** vs. generalists strategy

- **Save strong prisoners** for important upgrades

- **Plan ahead** - skills are limited resource

For Designers:

- **Balance base values** for progression pace

- **Match skills to types** thematically

- **Consider party composition** needs

- **Test edge cases** (stat 0, stat 100)

- **Tutorial prisoners** could have higher values

Knowledge is Power 📚 - Choose wisely who receives each prisoner's teachings!