

# Inventory Tab Filtering System

## Overview

The inventory system now uses **shared storage** with **tab-based filtering**:

- All 5 tabs share the same 50-slot inventory
- Tabs organize items by category
- "Misc" tab shows **all items**
- Other tabs filter by item type

## Tab Categories

### 1. Misc Tab

**Shows:** ALL items in inventory **Filter:** No filter - displays everything

### 2. Weap Tab

**Shows:** Weapons only **Filter:** `Item.ItemType.WEAPON`

Items shown:

- Wooden Short Sword
- Iron Sword
- Steel Longsword
- Mystic Staff

### 3. Arm Tab

**Shows:** Armor only **Filter:** `Item.ItemType.ARMOR`

Items shown:

- Leather Armor
- Chainmail
- Plate Armor

## 4. Acc Tab

**Shows:** Accessories only **Filter:** `Item.ItemType.ACCESSORY`

Items shown:

- Rune of Return
- Ring of Power
- Amulet of Protection
- Boots of Speed

## 5. Rune Tab

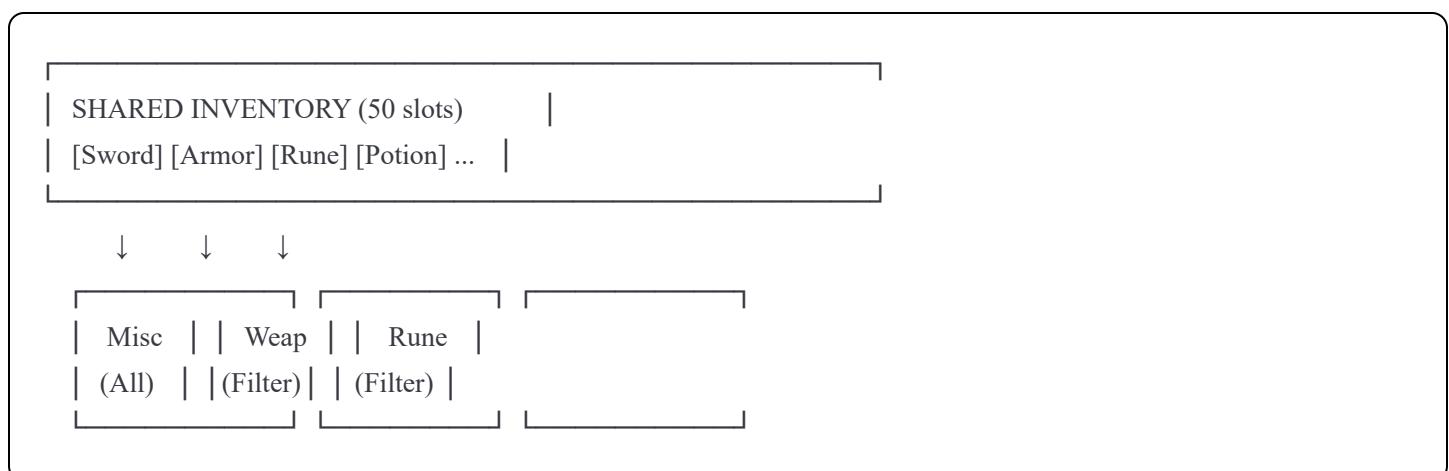
**Shows:** Runes and materials **Filter:** `Item.ItemType.MATERIAL` with "rune" in name

Items shown:

- Carved Wood
- Clay
- Carving Stone
- Fire Rune
- Ice Rune
- Lightning Rune

## How It Works

### Shared Storage Architecture



## Adding Items

```
java

// Items are added to shared storage
Item sword = ItemManager.createIronSword();
inventoryGrid.addItemToCurrentTab(sword);

// Item appears in:
// - "Misc" tab (shows all)
// - "Weap" tab (filtered view)
```

## Switching Tabs

```
java

// User clicks "Weap" tab
inventoryGrid.switchToTab("Weap");

// Display refreshes to show only weapons
// Scroll resets to top
// Item count shows: "3 / 50" (3 weapons out of 50 total slots)
```

## Item Count Display

Each tab shows: [filtered count] / [total capacity]

Examples:

- Misc tab: "15 / 50" (showing all 15 items)
- Weap tab: "3 / 50" (3 weapons)
- Arm tab: "2 / 50" (2 armor pieces)
- Acc tab: "4 / 50" (4 accessories)
- Rune tab: "6 / 50" (6 materials/runes)

## Testing the System

### Test 1: Add Items to Inventory

```
java
```

```

// Add various items
uiManager.addItemToInventory(ItemManager.createIronSword());
uiManager.addItemToInventory(ItemManager.createLeatherArmor());
uiManager.addItemToInventory(ItemManager.createFireRune());
uiManager.addItemToInventory(ItemManager.createHealthPotion());
uiManager.addItemToInventory(ItemManager.createPowerRing());

// Result:
// - Misc tab: 5 items
// - Weap tab: 1 item (Iron Sword)
// - Arm tab: 1 item (Leather Armor)
// - Acc tab: 1 item (Ring of Power)
// - Rune tab: 1 item (Fire Rune)

```

## Test 2: Switch Between Tabs

```

java

// Start on "Misc" tab - see all 5 items
inventoryGrid.switchToTab("Misc");

// Switch to "Weap" - see only weapons
inventoryGrid.switchToTab("Weap");

// Switch to "Rune" - see only materials/runes
inventoryGrid.switchToTab("Rune");

```

## Test 3: Remove Item (Updates All Tabs)

```

java

// Remove sword from inventory
inventoryGrid.removeItemFromSlot(0);

// Item disappears from:
// - "Misc" tab (all items view)
// - "Weap" tab (filtered view)

```

## Test 4: Fill Inventory

```

java

```

```
// Add 50 items to test capacity
for (int i = 0; i < 10; i++) {
    uiManager.addItemToInventory(ManagerInterface.createIronSword());
    uiManager.addItemToInventory(ManagerInterface.createLeatherArmor());
    uiManager.addItemToInventory(ManagerInterface.createFireRune());
    uiManager.addItemToInventory(ManagerInterface.createHealthPotion());
    uiManager.addItemToInventory(ManagerInterface.createPowerRing());
}

// Result:
// - Misc tab: 50 items (scrollable)
// - Weap tab: 10 items
// - Arm tab: 10 items
// - Acc tab: 10 items
// - Rune tab: 10 items
// - Consumables show in Misc tab
```

## Integration Examples

### Example 1: Equip Item from Inventory

java

```
// In UIInventorySlot.onRightClick()
@Override
public boolean onRightClick() {
    if (item != null) {
        if (item.isWeapon()) {
            boolean equipped = uiManager.equipItem(
                UIGearSlot.SlotType.WEAPON,
                item
            );

            if (equipped) {
                // Remove from shared inventory
                UIScrollableInventoryPanel inventory =
                    uiManager.getInventoryGrid();
                inventory.removeItemFromSlot(slotIndex);

                System.out.println("Equipped " + item.getName());
                return true;
            }
        }
    }
    return true;
}
```

## Example 2: Loot Drop System

```
java
```

```

public void onMonsterDeath(Entity monster) {
    // Drop random items
    List<Item> drops = generateLoot(monster);

    for (Item item : drops) {
        boolean added = uiManager.addItemToInventory(item);

        if (added) {
            System.out.println("Looted: " + item.getName());
            // Item automatically appears in appropriate tabs
        } else {
            System.out.println("Inventory full!");
        }
    }
}

private List<Item> generateLoot(Entity monster) {
    List<Item> drops = new ArrayList<>();

    // Random weapon drop
    if (Math.random() < 0.3) {
        drops.add(ItemManager.createIronSword());
    }

    // Random material drops
    if (Math.random() < 0.5) {
        drops.add(ItemManager.createCarvedWood());
    }
    if (Math.random() < 0.4) {
        drops.add(ItemManager.createClay());
    }

    return drops;
}

```

### Example 3: Quest Item Collection

java

```

// Check if player has quest items
public boolean hasQuestItems(String questId) {
    UIScrollableInventoryPanel inventory =
        uiManager.getInventoryGrid();

    // Switch to Rune tab to check materials
    inventory.switchToTab("Rune");

    int carvedWood = 0;
    int clay = 0;
    int carvingStone = 0;

    // Count quest items in shared inventory
    for (int i = 0; i < 50; i++) {
        Item item = inventory.getItemAtSlot(i);
        if (item == null) continue;

        if (item.getName().equals("Carved Wood")) carvedWood++;
        if (item.getName().equals("Clay")) clay++;
        if (item.getName().equals("Carving Stone")) carvingStone++;
    }

    // Switch back to Misc tab
    inventory.switchToTab("Misc");

    return carvedWood >= 1 && clay >= 1 && carvingStone >= 1;
}

```

## Advanced Features

### Get Item Counts by Category

java

```

// Add to UIScrollableInventoryPanel
public int getItemCountByType(Item.ItemType type) {
    int count = 0;
    for (Item item : sharedInventory) {
        if (item != null && item.getType() == type) {
            count++;
        }
    }
    return count;
}

// Usage
int weaponCount = inventoryGrid.getItemCountByType(Item.ItemType.WEAPON);
int armorCount = inventoryGrid.getItemCountByType(Item.ItemType.ARMOR);

```

## Search Inventory

```

java

// Add to UIScrollableInventoryPanel
public Item findItemByName(String name) {
    for (Item item : sharedInventory) {
        if (item != null && item.getName().equalsIgnoreCase(name)) {
            return item;
        }
    }
    return null;
}

// Usage
Item sword = inventoryGrid.findItemByName("Iron Sword");
if (sword != null) {
    System.out.println("Found: " + sword);
}

```

## Get All Items of Type

```

java

```

```

// Add to UIScrollableInventoryPanel
public List<Item> getAllItemsOfType(Item.ItemType type) {
    List<Item> items = new ArrayList<>();
    for (Item item : sharedInventory) {
        if (item != null && item.getType() == type) {
            items.add(item);
        }
    }
    return items;
}

// Usage
List<Item> allWeapons = inventoryGrid.getAllItemsOfType(Item.ItemType.WEAPON);
System.out.println("You have " + allWeapons.size() + " weapons");

```

## UI Behavior

### Tab Switching

1. User clicks "Weap" tab
2. `switchToTab("Weap")` called
3. Scroll resets to top (`scrollOffsetY = 0`)
4. Display refreshes to show only weapons
5. Item count updates: "3 / 50"

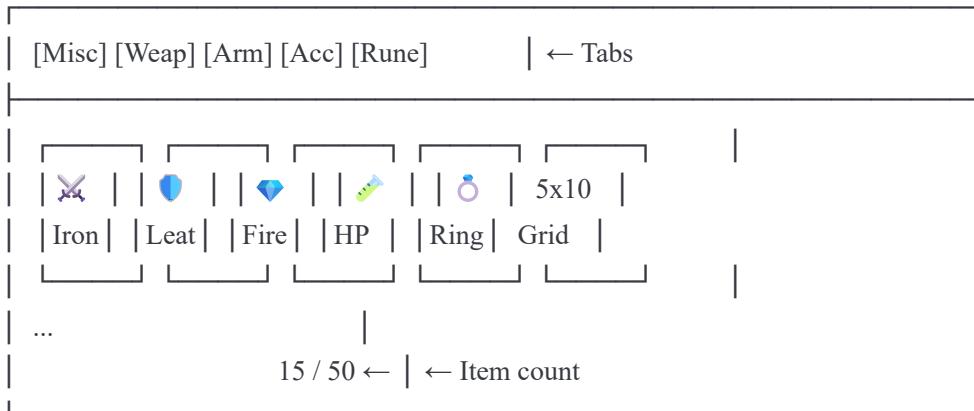
### Scrolling

- Each tab has independent scroll position
- Scroll resets when switching tabs
- Scrollbar size adjusts based on filtered item count

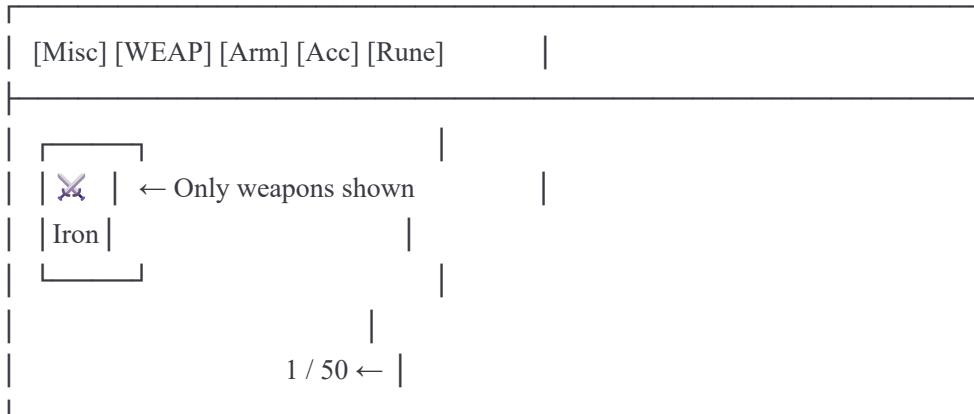
### Item Removal

1. Right-click item to equip/use
2. Item removed from shared inventory
3. All tab displays auto-update
4. Item counts update

## Visual Reference



Click "Weap" tab:



## Benefits

- Shared Storage** - All tabs use same 50 slots
- Organized** - Find items quickly by category
- Flexible** - "Misc" shows everything, others filter
- Efficient** - No duplicate storage or copying
- Auto-Update** - Adding/removing updates all tabs
- Visual Feedback** - Item counts per tab
- Scalable** - Easy to add new categories/filters