```
/// <summary>
/// Equip the selected sword on the player's hand
/// </summary>
void EquipOnHand()
{
    if (list_of_sword_can_echipeOnPlayer[selector_sowrd] == null) return;

    echipeSword = Instantiate(list_of_sword_can_echipeOnPlayer[selector_sowrd].gameObject);
    echipeSword.transform.SetParent(Hand_Echipe[0].transform);
    echipeSword.transform.localPosition = Hand_Echipe[0].transform.localPosition;
    echipeSword.transform.rotation = mainCamera.transform.rotation;

    UpdateInventoryIcons(1);
    SetPlayerAttackStats(selector_sowrd);

    sword_echipe_now = echipeSword.GetComponent<swordManager>();
    playerAttack_.echipeSword = true;

    PlayRandomSwordSound();
}

/// <summary>
/// Handle sword switching and visual trigger
/// </summary>
void EquipSwordCycle()
{
    DestroyCurrentSwordIcon();
    Destroy(echipeSword);
    EquipOnHand();
    selector_sowrd = (selector_sowrd + 1) % 2;
    playerAttack_.swordAnimator.SetTrigger("echipe");
}

/// <summary>
/// Cycle through available magic incantations
/// </summary>
void SelectNextIncantation()
{
    if (all_magic_incantation.Count == 0) return;
    selector_incantation = (selector_incantation + 1) % all_magic_incantation.Count;
    UpdateInventoryIcons(2);
}


/// <summary>
/// Cycle through available potions
/// </summary>
void SelectNextPotion()
{
    if (all_potions.Count == 0) return;
    selector_potion = (selector_potion + 1) % all_potions.Count;
    UpdateInventoryIcons(3);
}

/// <summary>
/// Use the currently selected potion on the player
/// </summary>
void UsePotion()
{
    if (all_potions.Count == 0 || statPlayer_.getBool_whait_to_drink_nexPotion()) return;

    DestroyCurrentPotionIcon();
    potions_Manager_drink_now = all_potions[selector_potion];
    statPlayer_.drink_Potion(all_potions[selector_potion]);
    all_potions.RemoveAt(selector_potion);

    statPlayer_.generate_voidSound(thePlayer.transform, statPlayer_.drinking_potion_clip, 0.3f);
}

/// <summary>
/// Update the visual icons for swords, incantations and potions
/// </summary>
void UpdateInventoryIcons(int type)
{
    switch (type)
    {
        case 1: UpdateSwordIcon(); break;
        case 2: UpdateIncantationIcon(); break;
        case 3: UpdatePotionIcon(); break;
    }
}

/// <summary>
/// Equip a sword from the main inventory to the player slots
/// </summary>
public void EquipSwordFromInventory(int mainInventoryIndex)
{
    list_of_sword_can_echipeOnPlayer[iandex_for_echipe_sword] =
sword_mainInventory[mainInventoryIndex];
    if (!firstEhipe || iandex_for_echipe_sword != selector_sowrd)
        EquipSwordCycle();
    firstEhipe = true;

    iandex_for_echipe_sword = (iandex_for_echipe_sword + 1) % 2;
}

/// <summary>
/// Add a new item to the appropriate inventory list
/// </summary>
public void AddItem(int type, GameObject item)
{
    switch (type)
    {
        case 1:
            sword_mainInventory.Add(item.GetComponent<swordManager>());
            break;
        case 2:
            all_magic_incantation.Add(item.GetComponent<magic_incantation_Manager>());
            UpdateInventoryIcons(2);
            break;
        case 3:
            all_potions.Add(item.GetComponent<potions_Manager>());
            UpdateInventoryIcons(3);
            break;
    }
}

/// <summary>
/// Play a random sword equip sound effect
/// </summary>
void PlayRandomSwordSound()
{
    statPlayer_.generate_voidSound(
        transform,
        all_soundEffect_take_sword_on_hand[UnityEngine.Random.Range(0,
all_soundEffect_take_sword_on_hand.Count)],
        0.5f
    );
}

/// <summary>
/// Destroy the currently equipped sword icon
/// </summary>
void DestroyCurrentSwordIcon()
{
    if (icoSword_inventoryPLayer != null) Destroy(icoSword_inventoryPLayer.gameObject);
}

/// <summary>
/// Destroy the currently selected potion icon
/// </summary>
void DestroyCurrentPotionIcon()
{
    if (icoPotion_inventoryPLayer != null) Destroy(icoPotion_inventoryPLayer.gameObject);
}

/// <summary>
/// Update player attack stats according to equipped sword
/// </summary>
void SetPlayerAttackStats(int index)
{
    playerAttack_.setDMG(list_of_sword_can_echipeOnPlayer[index].dmg);
}

/// <summary>
/// Update sword icon in the UI
/// </summary>
void UpdateSwordIcon()
{
    if (sword_mainInventory.Count == 0) return;
    icoSword_inventoryPLayer = Instantiate(list_of_sword_can_echipeOnPlayer[selector_sowrd].ico);
    icoSword_inventoryPLayer.transform.SetParent(inventory[2].transform);
    icoSword_inventoryPLayer.transform.localPosition = Vector3.zero;
    icoSword_inventoryPLayer.transform.localScale = Vector3.one;
    CreateSelectorEffect(2);
}

/// <summary>
/// Update incantation icon in the UI
/// </summary>
void UpdateIncantationIcon()
{
    if (all_magic_incantation.Count == 0) return;
    if (icoIncantation_inventoryPLayer != null)
Destroy(icoIncantation_inventoryPLayer.gameObject);
    icoIncantation_inventoryPLayer =
Instantiate(all_magic_incantation[selector_incantation].ico_incantation);
    icoIncantation_inventoryPLayer.transform.SetParent(inventory[1].transform);
    icoIncantation_inventoryPLayer.transform.localPosition = Vector3.zero;
    icoIncantation_inventoryPLayer.transform.localScale = Vector3.one;
    CreateSelectorEffect(1);
}

/// <summary>
/// Update potion icon in the UI
/// </summary>
void UpdatePotionIcon()
{
    if (all_potions.Count == 0) return;
    if (icoPotion_inventoryPLayer != null) Destroy(icoPotion_inventoryPLayer.gameObject);
    icoPotion_inventoryPLayer = Instantiate(all_potions[selector_potion].ico_Potion);
    icoPotion_inventoryPLayer.transform.SetParent(inventory[3].transform);
    icoPotion_inventoryPLayer.transform.localPosition = Vector3.zero;
    icoPotion_inventoryPLayer.transform.localScale = Vector3.one;
    CreateSelectorEffect(3);
}

/// <summary>
/// Instantiate a brief selection effect for inventory icons
/// </summary>
void CreateSelectorEffect(int inventoryIndex)
{
    GameObject effect = Instantiate(ico_effectSelector);
    effect.transform.SetParent(inventory[inventoryIndex].transform);
    effect.transform.localPosition = Vector3.zero;
    effect.transform.localScale = Vector3.one;
    Destroy(effect, 0.2f);
}
```