

○ ○ ○

```
///  
///<Summary>  
/// Handles the main interaction with the door: open, close, lock, or jumpscare.  
///  
public void Interact()  
{  
    if (Cooldown) return;  
  
    if (_jumpscareOn)  
    {  
        TriggerJumpscareInteraction();  
        return;  
    }  
  
    if (_hasKey) HandleKeyDoorInteraction();  
    else HandleLockedDoor();  
  
    Cooldown = true;  
}  
  
///  
///<Summary>  
/// Triggers a jumpscare with monster spawn and player repositioning.  
///  
private void TriggerJumpscareInteraction()  
{  
    Cooldown = true;  
    LockDoorVisuals();  
    TriggerJumpscare();  
    _jumpscareOn = false;  
}  
  
///  
///<Summary>  
/// Handles opening or closing when the player has a key.  
///  
private void HandleKeyDoorInteraction()  
{  
    if ((_doorAnimator.GetBool("open") || _doorAnimator.GetBool("scary_open")) && !_onlyOpen)  
    CloseDoor();  
    else if (!_openDoorWithKey) OpenDoor();  
    else StartCoroutine(OpenDoorWithKey());  
  
    _lastSoundPointRepeatCount += _gameEngineScript.CheckDollDistance(transform,  
    _dollHearingRange, _dollHearingRange + _lastSoundPointRepeatCount);  
}  
  
///  
///<Summary>  
/// Handles door interaction when it is locked or no key is present.  
///  
private void HandleLockedDoor()  
{  
    LockDoorAction();  
    _lastSoundPointRepeatCount += _gameEngineScript.CheckDollDistance(transform,  
    _dollHearingRange, _dollHearingRange + _lastSoundPointRepeatCount * 0.5f);  
}  
  
///  
///<Summary>  
/// Opens the door with animation and sound.  
///  
private void OpenDoor()  
{  
    CheckTriggerEvent(DoorState.OpenDoor);  
    SetDoorState(open: true, scary: false, drop: false);  
    StartCoroutine(CooldownForInteract(_cooldownForOpen));  
    PlaySoundEffect(_openClip, 1f, 1f);  
}  
  
///  
///<Summary>  
/// Closes the door with animation and sound.  
///  
private void CloseDoor()  
{  
    SetDoorState(open: false, scary: false, drop: false);  
    StartCoroutine(CooldownForInteract(_cooldownForClose));  
    PlaySoundEffect(_closeClip, 0.5f, 1f);  
}  
  
///  
///<Summary>  
/// Locks the door with animation and sound.  
///  
private void LockDoorAction()  
{  
    LockDoorVisuals();  
    StartCoroutine(CooldownForInteract(_cooldownForLock));  
    PlaySoundEffect(_lockClip, 1f, 1f);  
}  
  
///  
///<Summary>  
/// Common method to set door animator states for open, scary, and drop.  
///  
private void SetDoorState(bool open, bool scary, bool drop)  
{  
    _doorAnimator.SetBool("open", open);  
    _doorAnimator.SetBool("scary_open", scary);  
    _doorAnimator.SetBool("drop_door", drop);  
}  
  
///  
///<Summary>  
/// Coroutine for jumpscare sequence including monster instantiation and player reposition.  
///  
private IEnumerator JumpscareCooldown()  
{  
    yield return new WaitForSeconds(0.2f);  
    _gameEngineScript.PlayerLocation.position = _playerJumpscarePosition.position;  
    _gameEngineScript.PlayerLocation.localRotation = Quaternion.Euler(0, -90f, 0);  
    _gameEngineScript.PlayerScript.MainCameraAnimator.transform.localRotation =  
    Quaternion.Euler(0, 0, 0);  
    yield return new WaitForSeconds(0.5f);  
    _gameEngineScript.InCinematic = false;  
    Destroy(_monster);  
    if (_jumpscareTask) _jumpscareTask.SetActive(true);  
    Cooldown = false;  
}  
  
///  
///<Summary>  
/// Coroutine to open the door with key object visible temporarily.  
///  
private IEnumerator OpenDoorWithKey()  
{  
    _gameEngineScript.IsOpenDoor = true;  
    _keyOfDoor.SetActive(true);  
    yield return new WaitForSeconds(1);  
    _gameEngineScript.IsOpenDoor = false;  
    _openDoorWithKey = false;  
    OpenDoor();  
}  
  
///  
///<Summary>  
/// Coroutine for cooldown between door interactions.  
///  
private IEnumerator CooldownForInteract(float time)  
{  
    yield return new WaitForSeconds(time);  
    if (!_onlyOpen || !_doorAnimator.GetBool("open")) Cooldown = false;  
}  
  
///  
///<Summary>  
/// Opens door in a scary way with custom sound.  
///  
public void ScaryOpen()  
{  
    _hasKey = true;  
    Cooldown = true;  
    SetDoorState(open: true, scary: true, drop: false);  
    StartCoroutine(CooldownForInteract(_cooldownForScaryOpen));  
    PlaySoundEffect(_scaryOpenClip, 1f, 0.4f);  
}  
  
///  
///<Summary>  
/// Drops the door down with sound effect.  
///  
public void DropDoor()  
{  
    if (!_doorAnimator.GetBool("open") && !_doorAnimator.GetBool("scary_open")) return;  
    Cooldown = true;  
    SetDoorState(open: false, scary: false, drop: true);  
    StartCoroutine(CooldownForInteract(0.5f));  
    PlaySoundEffect(_dropClip, 1f, 1f);  
}  
  
///  
///<Summary>  
/// Plays a sound effect with volume and pitch.  
///  
private void PlaySoundEffect(AudioClip clip, float volume, float pitch)  
{  
    _audioSource.volume = volume;  
    _audioSource.pitch = pitch;  
    _audioSource.clip = clip;  
    _audioSource.Play();  
}  
  
///  
///<Summary>  
/// Triggers the door event if available.  
///  
private void CheckTriggerEvent(DoorState doorState)  
{  
    if (doorState == DoorState.OpenDoor && _openDoorEvent) _openDoorEvent.SetActive(true);  
}
```