```
--- Handle Jump events
function Jump:CheckJump(hum, humanoid, moveVector, jumpKeys, character)

	-- Initiliaze values
	local jumpType = ""
	local jumpVect = Vector3.new(0,0,0)
	local jumpCount = 0
	local startPos = Vector3.new(0,0,0)

	-- Normal Jump
	if jumpKeys.space and not character.jumpStatuses.doubleJump then
		if not character.jumpStarted.doubleJumped and not character.jumpStarted.dashed then
			jumpType = "normalJumped"
			character:DoNormalJump(hum, character, moveVector.Unit)
		end
	-- Double Jump
	elseif jumpKeys.space and character.jumpStatuses.doubleJump then
		if character.normalJumpCount < character.maxNormalJumpCount then
			jumpType = "doubleJumped"
			jumpVect, jumpCount = doubleJumpEvent:Invoke(hum, character, moveVector.Unit)
			character.currentNormalJumpCount = jumpCount
		end
	-- Dash Jump
	elseif jumpKeys.shift and not character.jumpStarted.dashed and character.jumpStatuses.dashJump
	then
		jumpType = "dashed"
		jumpVect, startPos = dashEvent:Invoke(hum, humanoid, moveVector.Unit, character)
		character.dashStartPos = startPos
	-- Super Jump
	elseif jumpKeys.ctr and not character.jumpStarted.superJumped and
	character.jumpStatuses.superJump then
		jumpType = "superJumped"
		jumpVect = superJumpEvent:Invoke(hum, humanoid, moveVector.Unit, character)
	end

	-- Set jumping values
	if jumpType ~= "" then
		character.jumpStarted[jumpType] = true

		hum:ChangeState(Enum.HumanoidStateType.Jumping)

		if jumpVect ~= Vector3.new(0,0,0) then
			character.savedJumpVelocity = jumpVect
		end
		-- Save velocity to base character class
		character.savedNormalVelocity = Vector3.new(character.savedNormalVelocity.X, 0,
		character.savedNormalVelocity.Z) + character.savedJumpVelocity
	end
end
```