MADNESS: PROJECT NEXUS CUSTOM ANIMATIONS

Modding isn't just new items and stat tweaks, you know! This guide will cover the process and everything you'll need to bring your animations into MADNESS: Project Nexus.

In order to create new animations for your M:PN mod, you'll need the following:

- **M:PN Blender Sample Project:** This project file for Blender includes the armature used by characters and their outfit pieces. Used to create new animations, or to weight the vertices of outfits and body parts.
    - **…and a copy of Blender v.2.79** to open it, plus some basic understanding of Blender and 3D modeling and animation.
- **The ModTools Project:** For applying Animation Events (see below) to your animation clips,
    - **…and a copy of Unity v2019.2.6f** to open it.

Once you've got these tools in hand, all you need is an idea. To help with that, this guide will show you what's possible and how to achieve it. Unfortunately it's 2024 and we still don't have any way to stop people from adding Fortnight memes to our game, so do whatever I guess.

PART 1: THE MADNESS ARMATURE

-Explain Root, Sync, all parts.
-Cannot change the armature or problems will ensue.

PART 2: CREATING AN ANIMATION

It all starts in Blender! Although there are many 3D applications out there that can export models in a format Unity understands and M:PN can use, the rig that moves a character model around was made in Blender v.2.79, so that's what we'll be demonstrating.

This guide isn't here to teach you how to animate in Blender, but you'll want to know how to export your finished work properly. Let's start from the beginning and work toward exporting an character animation for a M:PN mod.

- Unity versions
- Sync and Root nodes

- scale + 0.45 in Unity
- .fbx, NOT .blend, so give FBX instructions ONLY
- TYPES: Sync Anim, Attack (Warmup/Swing/Hit/End, Sweep), Entrance, etc

---

PART 3: THE ANIMATIONS MADCARD

-how to format your MadCard so that you can create a whole new attack, grapple, execute, etc

---

PART 4: ANIMATION EVENTS

Because of how the game is structured, there are two types of events you'll have access to, which we call "Run Events" and "Method Events". They're functionally the same, and both are called from event nodes located on the animation clip. But because they are accessed in different ways, the events you can use from each type will be listed separately.

RUN EVENTS

CATEGORY:

★

METHOD EVENTS

COMBAT [MELEE]:

★ **H2H_Warmup** - Lets the game know the character is preparing to swing a melee weapon. This should be placed at the start of an attack animation just when the character is pulling their weapon back and before they're ready to swing.
  - **INT**: 0 = Right hand, 1 = Left hand.

- ★ **H2H_Swing_M** - Used when a weapon is done warming up (see H2H_Warmup, above), at the moment it begins to swing toward the enemy. This is also when the game checks if the attack can be countered.
  - **INT**: 0 = Right hand, 1 = Left hand.
- ★ **H2H_Swing_U** - Like H2H_Swing_M, but forces the swing to use your unarmed weapon.
  - **INT**: “0” = Right hand, “1” = Left hand, “2” = Right foot, “3” = Left foot, “4” = Headbutt.
- ★ **H2H_Swing_M_Heavy** - Like H2H_Swing_M, but forces the swing to be heavy.
  - INT: (Same as H2H_Swing_M)
- ★ **H2H_Swing_U_Heavy** - Like H2H_Swing_U, but forces the swing to be heavy.
  - INT: (Same as H2H_Swing_U)
- ★ **H2H_SwingEnd** - Tell the game the swing has ended.
  - **INT**: 0 = Right hand, 1 = Left hand.
- ★ **H2H_Hit_M** - The moment the weapon should count as striking the enemy in front of the character.
  - **INT**: 0 = Right hand, 1 = Left hand.
- ★ **H2H_Hit_U** - Like H2H_Hit_M, but forces the hit to use your unarmed weapon.
  - **INT**: “0” = Right hand, “1” = Left hand, “2” = Right foot, “3” = Left foot, “4” = Headbutt.
- ★ **H2H_End** - Ends the attack. Place this event before the end of the animation, at the moment the attack would blend into the next swing in the combo. If the player doesn’t continue their combo then this event is ignored, but if they attempt to attack again, the next attack will begin at the moment the H2H_End event is reached.
- ★ **H2H_Stab_M** - Like H2H_Hit_M, but the damage type is converted to Piercing.
  - **INT**: (Same as H2H_Hit_M)
- ★ **H2H_Stab_U** - Like H2H_Stab_M, but forces the stab to use your unarmed weapon.
  - **INT**: (Same as H2H_Hit_U)
- ★ **H2H_Bite** - Applies a bite hit using your main unarmed weapon’s stats.
- ★ **H2H_Kick** - Like H2H_Hit_U, but applies special knockback rules.
  - **INT**: (Same as H2H_Hit_U)
- ★ **H2H_Sweep** - Used instead of H2H_Swing_M when the attack should hit everyone in the weapon’s swing arc. Heavy attacks do not need to use H2H_Sweep, as this is automated.
  - **INT**:
    - “0” = Right hand
    - “1” = Left hand
- ★ **H2H_SweepUnarmed** - Like H2H_Sweep, but forces the sweep to use your unarmed weapon.
  - **INT**:
    - “0” = Right hand
    - “1” = Left hand
    - “2” = Right foot
    - “3” = Left foot
    - “4” = Headbutt.
- ★ **H2H_SweepEnd** - Concludes the sweep so that no further enemies are struck.
- ★ **H2H_DashAttack** - The moment a dash attack would hit the opponent.
  - **INT**: 1 = Weak, 2 = Strong.
- ★ **H2H_Advance** - “Push” the character forward, toward either their target, or the player’s aim cursor otherwise.
  - **FLOAT**: The speed multiplier. Values of 0 or less will default to a 1.

COMBAT [RANGED]:

- ★ **ShootEvent** - Fire your weapon and instantly hit your sync partner. Fails if not in a sync animation with another character.
  - **INT**: 0 = Right hand weapon, 1 = Left hand weapon
- ★ **ShootEvent_Head** - Like ShootEvent, but targets their head.
  - **INT**: (Same as ShootEvent)
- ★ **ShootEvent_Body** - Like ShootEvent, but targets their head.
  - **INT**: (Same as ShootEvent)
- ★ **ShootEvent_Sourced_ByName**- Like ShootEvent, but fires as if from a specified MadCard_Ranged or _Thrown.
  - **MadCard_Ranged/Thrown**: The weapon to fire instead of what is equipped.
- ★ **WeaponEvent_Degrade** - Deals damage to the melee weapon's condition.
  - **INT**: Amount of damage.

COMBAT [SYNC]:

- ★ **H2H_GrappleSynch_ByName** *[v2.13.i]* - If sync'd with a partner, the character will begin a grapple with them.
  - **MadCard_Animations**: The Animations MadCard to use for this grapple.
- ★ **ExecuteEvent** - The moment when a takedown/execution is applied.
  - **STRING**: The category of execution to apply.
    - "execute" =

<u>EXPLOSIONS</u>:

- ★ **SplodeEvent_ByName**- Create an explosion from character's feet
  - **MadCard_Explosion**: The explosion to create.
- ★ **SplodeEvent_Friendly_ByName**- Like SplodeEvent, but cannot harm allies.
  - **MadCard_Explosion**: (Same as SplodeEvent)
- ★ **SplodeEvent_Friendly_Hand_ByName** *[v2.13.i]* - Like SplodeEvent_Friendly but from your right hand.
  - **MadCard_Explosion**: (Same as SplodeEvent)
- ★ **SlamEvent** - Create the character's SlamExplode from the specified hand.
  - **INT**: The hand producing the slam. 0 = Right, 1 = Left
- ★ **SlamEvent_Sourced** - Create an explosion from the character's right hand.
  - **MadCard_Explosion:** The explosion to create.
- ★ **SplodeDeath** - Kill and remove character, and then create an explosion where they were.
  - **MadCard_Explosion**: (Same as SplodeEvent), or SlamExplode if empty
- ★ **GibSplode** - Same as GibMe, followed by the character's SlamExplode from their location.
- ★ **GibEvent** - Runs GibMe followed by SplodeEvent.
  - **MadCard_Explosion**: (Same as SplodeEvent).

<u>EFFECT</u>:

- ★ **ChatterEvent** - Speaks a line of text chatter.
  - **INT**: 0 = Wakeup, 1 = Spawned, 2 = Chasing, 3 = Waiting, 4 = Attacking, 5 = Wounded, 6 = Died, 7 = FriendDied, 8 = EnemyDown, 9 = Taunt
- ★ **SpeakEvent** - Speak a custom line of text.
  - **STRING**: The line to speak.

- **CreationEvent_ByName** *[v2.13.i]*- Create the specified object at character's feet.
    - **STRING**: ParticleSystem or ModTools GameObject to be created.
- **CreationEvent_Center_ByName** *[v2.13.i]*- Like CreationEvent, but created from center of character.
    - **STRING**: (Same as CreationEvent)
- **CreationEvent_Head_ByName** *[v2.13.i]*- Like CreationEvent, but created from character's head.
    - **STRING**: (Same as CreationEvent)
- **CreationEvent_Chest_ByName** *[v2.13.i]*- Like CreationEvent, but created from character's chest.
    - **STRING**: (Same as CreationEvent)
- **CreationEvent_HandR_ByName** *[v2.13.i]*- Like CreationEvent, but created from character's right hand.
    - **STRING**: (Same as CreationEvent)
- **CreationEvent_HandL_ByName** *[v2.13.i]*- Like CreationEvent, but created from character's left hand.
    - **STRING**: (Same as CreationEvent)
- **CreationEvent_Center_ParentCapsule_ByName** *[v2.13.i]*- Like CreationEvent_Center, but parents the object to the character's collider..
    - **STRING**: (Same as CreationEvent)
- **CamShakeEvent** - Rattles the camera.
    - **FLOAT**: The amount of shake (0-25)
- **SlowMoEvent_PlayerSquad** - Applies the slow motion effect, but only if the animation was played by a Player or their squadmates.

WORLD:
- **LandEvent** - Create a puff of dust and play a ground impact sound.
    - **INT**: 0 = Hard fall, 1 = Light fall, 2 = Normal fall, 3 = Hurt fall
- **Footstep** - Play the footstep sound and place a footprint if required.
    - **INT**: 0 = Right foot, 1 = Left foot

SPAWNING:
- **SpawningComplete** - Releases an entrance before an animation is done so it can let the next character use it. Optional.
    - **INT**: Are we touching the floor? 0 = True, 1 = False
- **EnterComplete** - Character has finished going into the entrance. Optional.
- **TriggerEntranceAddonEffects** - Tell my entrance to do its special effect if it has one (such as glass breaking when swinging in on a rappel line).
- **EndAnimUpdateRoot** - If an animation ends with the Hips node outside its starting position, this will stop them sliding back to where they started.
- **SpawnEndUpdateRoot** - Runs EndAnimUpdateRoot followed by SpawningComplete**.**

SOUND:
- **SoundEvent** - Plays the sound effect provided.
    - **STRING**: The FMOD event name of the sound to play.
- **SoundEvent_Hands** - Like SoundEvent, but plays from the hands, unless the character's hands are restrained.
    - STRING: (Same as SoundEvent)
- **VoiceEvent** - Character speaks a voice line.

    - **INT**: 0 = Angry, 1 = Attack, 2 = BigAttack, 3 = Cheer, 4 = Laugh, 5 = Grunt, 6 = Death, 7 = Pain, 8 = Affirmative, 9 = Dialogue
- ★ **VoiceEvent_20** - Twenty percent chance character speaks a voice line
    - **INT**: (Same as VoiceEvent)
- ★ **JumpEvent** - Plays the jump sound.
- ★ **WeaponSoundEvent** - Play the sound of a weapon held in the main hand.
    - **INT**: 0 = Equip, 1 = Stow, 2 = Drop, 3 = Block, 4 = Hit, 5 = Swing

MISC:

- ★ **AnimEvent** - Play a new animation.
    - **AnimationClip**: The animation clip to play.

- ★ **GibMe** - Burst into a bloody mess and die.

BACKUP

- ★ H2H_Swing - Used when a weapon is done warming up (see H2H_Warmup, above), at the moment it begins to swing toward the enemy. This is also when the game checks if the attack can be countered.
    - STRING: Category of the swing, written as a letter followed by a number. It will look like “m0”, “u3”, and so on, to designate this swing as melee or unarmed, followed by the hand or location doing the swinging.
        - “u” = Unarmed weapon, “m” = Melee weapon (or Unarmed if no weapon equipped)
        - “0” = Right hand, “1” = Left hand, “2” = Right foot, “3” = Left foot, “4” = Headbutt.