**MM3.2 Stats:**

Combat relies on the following values.

*Strength:*
Large integer value, between about 70 and 250. See creation spreadsheet.

*Agility:*
Float modifier of around 1. See creation spreadsheet.

*Constitution*:
Float modifier of around 1. See creation spreadsheet.

*Blood*:
Your current amount of blood, which is either rising or falling because you're bleeding or healing.

*Max_blood:*
Does not increase as you level – basically determined from your size and stats at creation.

*Endurance:*
Your current stamina. Call it stamina?

*Max_Endurance:*
We either have to use this for both mental and physical endurance, or we can break it up again. I know you said breaking it up was going to be a pain. Physical endurance is based on IC levels, IC kills, WM, and talents. Mental endurance is based on IC levels, weaves woven, CM, and talents.

*HP*:
Your current amount of HPs.

*Max_HP:*
Same equation as blood, but call it Max_HP in case we need to scale one or the other. It's easiest to balance combat by modifying these stats rather than editing weapon damage values and form damage values.

*Encumbrance:*
Sum of inventory weight, concealed item weight, get_worn_weight, get_wielded_weight, gold.

*Weight*:
Naked character bodyweight in lbs. See creation spreadsheet.

*Height:*
Selected at creation, no charge.

*WM:*

Mod based on WM level.

*Aware* (an affect, shown in prompt, but not listed in score):
Aware drains your stamina but acts as a bonus to the initial round of combat while active. The idea is that if you suspect you're about to get attacked codely, you can type 'aware'. This gives you a much better chance of successfully reacting to the first initial attack (such as a backstab), which would normally have a large chance of landing before you even react.

*Readiness:*

Usually a mod of 1, though it goes down as the result of critical failures and some successful attacks against you. These are specified in the affects portion of attack structures.

*Intimidation:*
Your current intimidation level. Goes down from fear weaves, getting stunned, and from facing vastly more intimidating opponents.

*Max_Intimidation:*

(goes up permanently the more IC kills you get, settable for FS, SS, bookies,etc, also based on WM/CM)

Note to self: Each mob should have an intimidation integer associated in its stat sheet. When you kill a chicken, you gain no intimidation points, but if you kill a fade, you gain several. Killing several fades IC makes you a scary mo-fo. For later.

Eventually I'd like to customize the score sheet so you can either bring up the standard MM2-3 info or additional combat info, as somebody suggested on the IM board a while ago.

Aggressiveness and followthrough:

"aggressiveness"

This stance option is just like in MM2, where you picked how reckless and offensively your character was fighting. Options are defensive, normal, aggressive, and reckless. The higher the setting, the more chance you have to go 'next' each turn. This data is used in the function "get_next_attacker".

Defensive:
You never attack (mod of 0).
Normal cost per turn (mod of 1).
Normal:
Normal chance to attack (mod of 1).
Normal cost per turn (mod of 1).

Aggressive:
Increased chance to attack (mod of 1.3).
Increased cost per turn (mod of 1.5).
Decreased chance to defend (mod of .8).
Reckless:
Increased chance to attack (mod of 1.7).
Increased cost per turn (mod of 1.85).
Decreased chance to defend (mod of .4).

"follow-through"

This sets one of the damage mods. As we noted in an earlier conversation, just because a player is fighting quickly doesn't mean he is fighting to deal substantial wounds - he could be haulting his blade just before touching the other person, such as in a practice match. The options are [practice, normal, extreme]. This data is used in the damage calculation equation and in "get_next_attacker".

Practice:
Damage is reduced to nearly zero (random chance of doing a couple hps damage).
Normal chance to attack (mod of 1).
Normal:
Damage is normal (mod of 1).
Normal chance to attack (mod of 1).
Extreme:
Damage is maximum (mod of 1.6).
Decreased chance to attack (mod of.8).

We need a syntax to accommodate all the different options for fighting. For players that don't want to use many options, the less they have to type the better. But all fields should be available to players that are expert with the system and want to specify everything, such as if they're using this to decide the outcome of an emote attack.

It's your turn!
>Attack (if only one person not in your group)
-or-
>Attack <victim>
-or-
>Attack <victim> <target>
-or-
>Attack <victim> <form name> <target>
-or-
>Attack <victim> <form name> <target> <tier>
-or-
>Attack <victim><form name><target><tier><aggressiveness><followthrough>

Specifying aggressiveness and/or followthrough should execute that form with those settings and not the ones specified in options. Should we make this also change the options settings themselves, or not?

Functions:

"get_next_attacker" (only once combat has begun – otherwise "get_attack")

> "get_next_attacker" determines whose turn it is to attack. It calculates an integer Value for
> each legal combatant, and the highest value (with some random chance) goes next. If the player
> suffered a 'stumble/recover' failure (see Critical Failure section under Structures for Attack Form) or a 'fall' failure, the player has probably lost his turn and may not attack this turn, because his readiness is so low.
>
> Value = 100 * [readiness] * [agility] * [endurance/max_endurance]
> * [blood/max_blood] * [hp/max_hp] * [encumb_mod (maybe 4*str)/
> (5+weight+get_worn_weight+get_wield_weight)] + [aggressiveness mod] * [follow-through mod] * [WM mod] * [intimidation mod - see 'intimidation' stat – mod for players that are stunned or scared] *[random number between .9 and 1.1]

We should note that for players that are channeling, encumb wouldn't play any part in modififying
their chance to weave an attack. Any idea how to account for that? I guess this isn't that big of a deal - what do you think?

> **Include some random chance of going to the #2 person in the lineup. 1/5 chance that the 2nd highest value goes next, and they have a 1/5 chance of being bypassed by the 3rd, etc. If this system returns a person with a 0 Value, everybody receives a message: "You find yourself pausing, along with everyone else. Only a moment passes…" Resolve bleeding/affects. New combat turn.**

The encumb mod isn't quite right, but I'm still working on it. We can use this for now.

"get_attack" (formerly "is_a_hit")

> Does the attacker have this form? (ch)
> Does the attacker have this form disabled in form options? (ch – only for the auto feature)
> Does the attacker have the proper weapon equipped? (ch and attack structure)
> Does the defender have the proper weapon equipped? (vict and /attack/ structure)
> Is the attacker in the proper stance? (ch and attack structure)
> Is the defender in the proper stance? (vict and defense structure)
> Are the combatants in the proper area? (either vict or ch, they're both in the same room)
> Is the bodypart target legal for this form? (attack structure)
> Does the bodypart target exist?

It should print the proper failure message if it failed any of the above checks. “You don’t have a sword to disarm with, stupid!” No need to resolve combat, just let the person pick again.
Does the attacker suffer a critical failure?

If it passes everything and there is no critical failure, call “is_aim_ok”

“get_critical_failure_attack”

Select and print a critical failure (attack structure). If the critical failure is of type penalty-free, loss of weapon, stumble/recover, or fall, call “resolve_turn”. If the critical failure is of type counter, recalculate readiness *=.93 and let the victim attack immediately without resolving the turn.

“is_aim_ok”
Calculate chance to hit that body part based on size of body part, skill with the attack form, WM, and random chance.

If aim is not ok, call “get_new_bodypart” to select a more likely target.

Output initial attacker text from attack structure (determined by aggressiveness). Now that “get_new_bodypart” has selected the real target (or the target was acceptable in the first place, we can print the attack.

“get_new_bodypart”
“is_aim_ok” determined the attacker can’t hit the body part he was aiming for.

Output ‘bad aim’ text. “Your aim is poor – you’re not going to hit <body part> after all.”

Which body part does it hit? (attack direction, body part size and orientation data from Bpedit)

This function is always going to pick something – presumably, if you were going to miss completely even if the defender does nothing, it was covered in critical failures.

Return to “is_aim_ok”

“calculate_attacker_value”
Takes the raw form value and mods it appropriately (haven’t finalized).

“get_defense”
Defender selects defensive form.
Does the defender have this form? (ch)
Does the defender have the proper weapon equipped? (ch)
Does the attacker have the proper weapon equipped? (vict)
Is the defender in the proper stance? (ch)
Is the attacker in the proper stance? (vict)

Are combatants in the proper area? (ch)

**Print appropriate error message if it failed any of the above checks. "Yeah, that only works on ships, idiot."**

Does the defender have this form disabled in form options? (ch – only for the auto feature)

Does the defense have a high mod to protect the target with the given attack direction? (defense structure, attack structure) (only for auto – if the person picked it manually, let them die. Otherwise, pick another defense).

Does the defender suffer a critical failure?

"get_critical_failure_defense"

Options are 'fall', 'stumble/recover' (botch form), and 'loss of weapon', and 'petrified'. Output appropriate text message from defense structure.

Falling, stumble/recover and loss of weapon are chosen randomly. Petrified is determined by the discrepancy between defender's intimidation mod and the attacker's intimidation mod.

Falling goes to 'resolve turn' (skip the rest of the exchange) since it's no longer possible to determine what the attacker is going to hit when the defender is on the ground. Print that attacker missed as defender falls to the ground. Set readiness*=.93, set them to 'on ground', and attacker gets to attack again for free.

'Stumble/recover' decreases defender value in "calculate_defender_value" – readiness*=.93, and lowers defense tier by 1 (unless already at 0) then proceeds normally.

'Loss of weapon' fakes the rest of the combat function by printing that the attacker's attack sends the defender's weapon flying, but no damage is done. Proceed to 'resolve turn'.

Petrified does not defend at all. Readiness *= .01. Combat continues normally.

"calculate_defender_value"

Takes the raw defensive form value and mods it by readiness and other factors like endurance, aggressiveness, etc. Equation to be finalized later.

"compare/exchange"

Compares "calculate_attacker_value" and "calculate_defender_value" – the larger value indicates the winner of the exchange.

"resolve turn"

If attacker won, damage is calculated based on attacker hit/dam, force mod, strength, etc, defender's armor, etc. The damage and attack type (slash, crush or pierce) determines the wound. Wounds are applied and stats (such as blood, endurance are updated). Counter damage from the defense is taken into account, and all affects are resolved (such as

stunning, knockdown, code loss of weapon) The appropriate text message for the wound is given.

If defender won, calculate counter damage from defense structure, apply affects and resolve everything. Print appropriate defense message.

Recalculate readiness= MAX[readiness*=1.5, 1.1]

Lastly, calculate chance for everybody to bleed to death or pass out, and have them do so. Terminate combat when only one conscious combatant/group remains. Otherwise, proceed to “get_next_attacker”.