RPG MAKER MV

Action Sequences Guide

A helping manual about Yanfly's Action Sequences Plugins

Written by Yoshifull

04/02/2016

Holds all the lines used in Yanfly's plugins in relation with the Action Sequences Plugins.

Index Tables

Ctrl + click on the number of the page to go there.

Page Index

Author's note

Hello everyone, Yoshifull here. I'm a fellow user of the RPG Makers, and if you are here, it's because you are using RPG Maker MV. It is the most recent RPG Maker around, and after playing around a bit, I was amazed by the built-in sideview.

But after some time, I saw a big problem, which is the way the character does an animation. He simply walk forward, does a little gesture, the skill takes effect, and the character moves back. While it was quite correct for the magic spells, the physical ones were lacking.

My life changed when Yanfly released the Action Sequences plugins. While it took me some time getting used to it, it became fun to make my own full custom character movements.

There is already one PDF document stating all the commands from the Action Sequences, but it only displays the commands from the three Action Sequences. More commands came up with the others plugins, so here will be listed the commands of all the plugins Yanfly did until today.

I'm Mark, a tactician created by Yoshifull.
My job consist on making battle strategies, but I am also a decent warrior on the battlefield, using physical moves as well as magical. And to reach this feat, Yoshifull went creative on the Action Sequences.
I will provide hints on some commands uses.

There are a lot of commands if we regroup all the plugins that adds some. Let's get started.

I suggest you to have at least Action Sequences 1 & 2, as they are the two main plugins you need to make good actions. Yanfly's Battle Core is also needed for them to work.

Getting started on Action Sequences

Notetags patterns

Before anything, you must go to the Skill section in the Database, and see at the bottom right the « Note » section. That's where the magic happens.

Each skill and item consists of five different action sequences. They are as follows:

1. Setup Actions
They prepare the active battler before carrying out the bulk of the actionand its individual effects. Usually what you see here are things such as the active battler moving forward a bit, unsheathing their weapon, etc. This step will occur before the active battler expends their skill or item costs.

2. Whole Actions
These actions will affect all of the targets simultaneously. Although this section does not need to be used, most actions will use this for displaying animations upon all enemies. This step occurs after skill and item costs.

3. Target Actions
This section will affect all of the targets individually. Used primarily for physical attacks that will deliver more personal forms of damage. Actions that occur here will not affect other targets unless specifically ordered to do so otherwise.

4. Follow Actions
This section will dedicate towards cleanup work after the individual targeting actions. Here, it'll do things such as removing immortal flags, start up common events, and more.

5. Finish Actions
This section will have the active battler close up the action sequence. Usually stuff like running waits and holds at the last minute for skills and items, moving back to place, and others.

Now that you know each of the five steps each action sequence goes through, here's the tags you can insert inside of skills and items. Pay attention to each tag name.

1. <setup action>
action list
action list
</setup action>

2. <whole action>
action list
action list
</whole action>

3. <target action>
action list
action list
</target action>

4. <follow action>
action list
action list
</follow action>

5. <finish action>
action list
action list
</finish action>

They will do their own respective action sets.

Since each action sequence is a lot of command lines, you may want to avoid any overflooding that would come out copy/pasting the same sequence in another notetag. Instead, use the notetag :

<action copy :x :y> where x is « item » or « skill » depending of where you want the action to be copied from, and y the ID of the skill/item.

Each time you make a skill, you need to refer to these notetags, unless you use the action copy one or, of course, if you think there is no point in using the action sequence for this skill. You have your reasons to do this. Most common is the Attack and Guard skills.

Target typing

A lot of actions listed in this manual have « target » in their line of command. It determines who will do what, and possibly to who.

Target	Description
user	Selects the active battler
target, targets	Selects the active target in question
actors, existing actors	Selects living actors
all actors	Selects actors, alive and dead
dead actors	Select only dead actors
actors but user	Select all actors, except the user
actor x	Selects the actor in the slot x
character x	Selects the actor with the character ID x
enemies, existing enemies	Selects all living enemies
all enemies	Select all enemies, alive and dead
dead enemies	Selects only dead enemies
enemies not user	Selects all enemies but the user
enemy x	Selects enemy in the slot x
friends	Selects the battler's alive allies
all friends	Selects the battler's allies, alive and dead
dead friends	Selects only the battler's dead allies
friends not user	Selects the battler's allies except itself
friend x	Selects battler's friend in slot x
opponents	Selects battler's all alive opponents
all opponents	Selects battler's all opponents, alive and dead
dead opponents	Selects only battler's dead opponents
opponent x	Selects battler's opponent in slot x
all alive	Selects all alive actors & enemies
all members	Selects all alive and dead actors & enemies
all dead	Selects all dead actors & enemies
all not user	Selects all alive actors & enemies except the user
focus	Selects the user and its targets
not focus	Selects all but the user and it's targets.

Now that we have that cleared out, let's move on to the commands.

The three Action Sequencers Plugins

Those three plugins were the first ever made by Yanfly, and are the basis of any action sequence, except the third one.

Indeed, the third one controls the camera, while the first one control mechanic actions and the second characters actions.

The Action Sequence Pack 1

ACTION ANIMATION: (target), (mirror)

Plays the animation assigned to the skill/item. The animation will automatically select the skill's/item's assigned targets. If 'target' is used, it will specify a target to play the animation on. If 'mirror' is used, it will mirror the animation.

Usage Example: action animation
action animation: target
action animation: user, mirror

Action Animation is useful to make attacks that hits from behind using 'mirror'.

If you use 'mirror', you HAVE to use 'target' as well.

You can have the animation play on whoever you want that way.

==========

ACTION COMMON EVENT

Plays the common event found within the skill's/item's traits list. This will only play the last common event on the list, following the game engine's original process. Nothing else will continue on the action list until the common event is finished.

Usage Example: action common event

==========

ACTION EFFECT: target

Causes the target(s) to take damage/healing from the skill/item and incurs any changes made to the target(s) such as buffs and states.

Usage Example: action effect

If you don't have that line anywhere on your notetag, the damage/healing won't be applied. You don't want to make something flashy that deals no damage at the end, do you ?

==========

ADD stat BUFF: target, (turns), (show)

Affects the target with 'stat' buff. Replace 'stat' with 'hp', 'mp', 'atk','def', 'mat', 'mdf', 'agi', or 'luk'. If you include a number after the target, it will buff the target by that many turns. Include 'show' and it will show the target getting the buff applied in the battle log.

Usage Example: add atk buff: user, 3, show
add def buff: target, 8

==========

ADD stat DEBUFF: target, (turns), (show)

Affects the target with 'stat' debuff. Replace 'stat' with 'hp', 'mp', 'atk', 'def', 'mat', 'mdf', 'agi', or 'luk'. If you include a number after the target, it will debuff the target by that many turns. Include 'show' and it will show the target getting the debuff applied in the battle log.

Usage Example: add atk debuff: user, 3, show
add def debuff: target, 8

==========

ADD STATE X: target, (show)
ADD STATE X, Y, Z: target (show)

Affects the target with X state (including Y and Z if used in that format).
If 'show' is included, it will display any state related messages.

Usage Example: add state 5: target
add state 6, 7, 8: user, show

==========

ANIMATION X: target, (mirror)

Plays animation X on target. 'Mirror' will cause the animation to appear mirrored. Keep in mind that animations played on actors will automatically be mirrored and setting the mirror option will reverse it and have it appear unmirrored.

Usage Example: animation 5: user
animation 6: target, mirror

The main use of this would be to **play another animation than the one bound to the skill.**
That way, you could play an animation that shows a charging skill, then the skill's animation, or put up a healing animation on the user to show a drain effect… What you can do with that line only has the animations as limits !

==========

ANIMATION WAIT: X

- -

Waits x animaiton frames. Each frame for an animation does not last one game frame, but instead, several. To make life easier, you can use this to have the game wait x frames played for the animation.

- -

Usage Example: animation wait 5: user
animation wait 6: target, mirror

==========

BGM: STOP
BGM: MEMORIZE
BGM: MEMORY
BGM: filename, (volume), (pitch), (pan)

Changes the current background music at hand. 'Stop' will stop any BGM from playing. 'Memorize' will memorize the current BGM. 'Memory' will replay the memorized BGM if there is one playing. If you choose a filename (without the filename extensions), the game will play that BGM instead. Using this option opens up access to the volume, pitch, and pan control, all of which are optional to use. If no values are inputed for volume, pitch, and pan, the game will use the settings in this plugin's parameters.

Usage Example: bgm: stop
bgm: memorize
bgm: memory
bgm: Battle7
bgm: Theme2, 80, 100, 0

==========

BGS: STOP
BGS: MEMORIZE
BGS: MEMORY
BGS: filename, (volume), (pitch), (pan)

Changes the current background sound at hand. 'Stop' will stop any BGS from playing. 'Memorize' will memorize the current BGS. 'Memory' will replay the memorized BGS if there is one playing. If you choose a filename (without the filename extensions), the game will play that BGS instead. Using this option opens up access to the volume, pitch, and pan control, all of which are optional to use. If no values are inputed for volume, pitch, and pan, the game will use the settings in this plugin's parameters.

Usage Example: bgs: stop
bgs: memorize
bgs: memory
bgs: City
bgs: Darkness, 80, 100, 0

==========

BREAK ACTION

This will force the remainder of the action sequences for the part of the skill/item to shut down and be skipped.

Usage Example: break action

You don't have many reasons to use it as it is. A skill that work shouldn't have BREAK ACTION anywhere, unless you use them in a Conditional Branch.

See IF … ELSE STATEMENT for that.

==========

CAST ANIMATION

Plays an animation on the skill's user. Will not occur if the action is an item or the user's default normal attack. If the skill has the notetag <Cast Animation: x>, then that animation will be played.

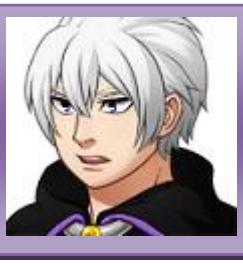
Usage Example: cast animation

==========

CLEAR BATTLE LOG

Clears all the messages at the top of the screen.

Usage Example: clear battle log



Let me insist on the point that you should ALWAYS put that in the <finish action> part. If you don't, the message will stay until you use another skill.

It should become like a reflex. A new skill ? Put CLEAR BATTLE LOG in the notetag. Same goes for DISPLAY ACTION.

==========

CHANGE SWITCH X: on/off/toggle/switch z
CHANGE SWITCH X..Y: on/off/toggle/switch z
CHANGE SWITCH X TO Y: on/off/toggle/switch z

Changes Game Switch X to on, off, toggle (switching between on/off), or to whatever value the switch y is.

Usage Example: change switch 1: on

change switch 2..4: off
change switch 5 to 8: toggle
change switch 9: switch 5
==========
CHANGE VARIABLE X = Y
CHANGE VARIABLE X += Y
CHANGE VARIABLE X -= Y
CHANGE VARIABLE X *= Y
CHANGE VARIABLE X /= Y
CHANGE VARIABLE X %= Y

Changes variable X in the middle of the action sequence to be modified by value Y. Y can be either an integer or a piece of code.

Usage Example: change variable 1 = 2
change variable 3 += 4
change variable 5 -= 6
change variable 7 *= 8
change variable 9 /= 10
change variable 11 %= 12

Due to the nature of variables, make sure you use a variable you can easily keep track of. If you're using a variable here, then make extra sure that this variable is only battle-related to avoid any confusion, and don't forget to give it a name.

In game like in battle testing, « F9 » opens the debugger. You can see the varibles there.

==========
COLLAPSE: target, (force)

If the target is to be dead at this point, this will be the point in the action sequence where you can promt the game to kill the target as long as the target has 0 HP. If you want to force the death of the target, include the 'force' command after the targets.

Usage Example: collapse: user
collapse: target, force
==========
COMMON EVENT: X

Plays common event X at that point in the action sequence. Nothing else will continue until the common event is finished. If you're using Gab Window, you can use them too, they will appear in battle.

Usage Example: common event: 1

==========

DEATH BREAK

If a user were to die for any reason during the middle of the skill (either via counter attack or reflection), this will force the remainder of the action sequences for the part of the skill/item to shut down and be skipped. Best used at the end of <target action>

Usage Example: death break

==========

DISPLAY ACTION

Displays the action's name at the top of the battle log. It will remain there until the battle log is cleared.

Usage Example: display action

This must be placed in <start action> and on each skill, for the sake of clarity and to not confuse the player.

It may not sound like much, but going back to the skill just to add that line gets annoying over time. Just like the Poison state.

==========

EVAL: code

For those who'd like to do something that the current Battle Engine doesn't support, you can use an eval function to have a piece of code occur. Users beware, for those unfamiliar with JavaScript should avoid handling this action sequence command.

Usage Example: eval: $gameParty.loseItem($dataItems[3], 10)

==========

GAIN ITEM X: Y LOSE ITEM X: Y
GAIN WEAPON X: Y LOSE WEAPON X: Y
GAIN ARMOR X: Y LOSE ARMOR X: Y

Your party will gain/lose item x, weapon x, or armor x in the amount of y. If you choose to omit y, it will default to 1.

Usage Example: gain item 1: 20
lose weapon 2
gain armor 3: 50

==========

GOLD +x
GOLD -x

Your party will gain/lose gold in the middle of battle by x amount.

Usage Example: gold +2000
gold -500

==========

IF ... ELSE STATEMENTS

For those familiar with programming, you can use if...else statements to perform different actions based on different conditions. Use 'if' to specify a block of code to be executed, if a specified condition is true. Use 'else' to specify a block of code to be executed, if the same condition is false. Use 'else if' to specify a new condition to test, if the first condition is false. Use 'end' to specify where the conditions are to end.

Usage Example:

```
if $gameSwitches.value(1)
   action effect
else if $gameSwitches.value(2)
   action effect
   action effect
else
   action effect
   action effect
   action effect
end
```

*Note: You do not have to indent the code in between to work. It just looks better that way in your action sequences.

==========

IMMORTAL: targets, true/false

Sets the targets to a state of immortality so that they don't die in the middle of an attack. This is to ensure every action effect goes through.

Usage Example: immortal: targets true

If you set IMMORTAL to be true and forget to put it to be false at the end of the sequence, **the enemy will stay immortal.**

Put IMMORTAL : target, true in <start action> and IMMORTAL : target , false in <follow action>/<finish action>

==========
HP +X: target, (show)
HP -X: target, (show)
HP +X%: target, (show)
HP -X%: target, (show)
HP +VARIABLE X: target, (show)
HP -VARIABLE X: target, (show)
HP +VARIABLE X%: target, (show)
HP -VARIABLE X%: target, (show)

Target(s) gains HP equal to X values. To show the popup, insert 'show' after the target in the action sequence line. Including 'show' is entirely optional. If 'show' is omitted, no popup will be displayed.

Usage Example: hp +500: user
hp -variable 5: target
hp +25%: target
hp -variable 7: user
==========
ME: STOP
ME: filename, (volume), (pitch), (pan)

Causes the battle to play a music fanfare. 'Stop' will stop any ME from playing. If you choose a filename (without the filename extensions), the game will play that ME instead. Using this option opens up access to the volume, pitch, and pan control, all of which are optional to use. If no values are inputed for volume, pitch, and pan, the game will use the settings in this plugin's parameters.

Usage Example: me: stop
me: Victory1
me: Darkness, 80, 100, 0
==========
MOTION WAIT: target

Makes the game wait 12 frames if the target(s) performing the action is an actor. If the target(s) is not an actor, no waiting will be done.

Usage Example: motion wait: user

==========

MP +X: target, (show)
MP -X: target, (show)
MP +X%: target, (show)
MP -X%: target, (show)
MP +VARIABLE X: target, (show)
MP -VARIABLE X: target, (show)
MP +VARIABLE X%: target, (show)
MP -VARIABLE X%: target, (show)

Target(s) gains MP equal to X values. To show the popup, insert 'show' after the target in the action sequence line. Including 'show' is entirely optional. If 'show' is omitted, no popup will be displayed.

Usage Example: mp +500: user
mp -variable 5: target
mp +25%: target
mp -variable 7: user

==========

PERFORM ACTION

Causes actors to step forward and swing their weapon, thrust it, however the motion that is determined will be automatically done by the game.

Usage Example: perform action

==========

PERFORM FINISH

Causes actor to move back to its home spot.

Usage Example: perform finish

PERFORM FINISH may be the best thing to put in <finish action>.

It will move your character to your home spot, regardless where it is on the screen.

Feel free to use and abuse it !

==========

PERFORM START

Causes actor to move forward from its home spot.

Usage Example: perform start

ATTACK ANIMATION: target, (mirror)

Displays the active battler's attack animation on the target(s). This will be the animation determined by the actor's weapon(s). If it's an enemy, it will be determined by the enemy's attack animation. If 'mirror' is used, the animation will be flipped.

Usage Example: attack animation: target

==========

ENEMY EFFECT: target, effect-type

This affects enemies only. Makes the target display either a 'whiten' effect or a 'blink' effect.

Usage Example: enemy effect: targets, whiten
enemy effect: targets, blink

This won't work if the enemy is a Sideview Animated Enemy, from the plugin of the same name.

==========

FACE target: args

FACE target1: FORWARD
FACE target1: BACKWARD
FACE target1: HOME
FACE target1: AWAY FROM HOME
FACE target1: POINT, x coordinate, y coordinate
FACE target1: AWAY FROM POINT, x coordinate, y coordinate
FACE target1: target2
FACE target1: AWAY FROM target2

This will cause the battler to face a certain direction. Arguments can be used in the above formats. This action sequence command will cause target1 to face any of those directions. If target2 is used, then target1 will face directions relative to target2.

Usage Example: face user: forward
face target: backward
face enemies: home
face allies: away from home

face target: point, 20, 40
face target: away from point, 500, 600
face user: target
face target: away from user

==========

FADE OUT: (frames)
FADE IN: (frames)

Fades the screen out and fades the screen in respectively. You can set the amount of frames for the fading process. If you omit frames, 60 frames will be used by default.

Usage Example: fade out
fade in: 10

==========

FLASH SCREEN: args

FLASH SCREEN: WHITE, (frames)
FLASH SCREEN: RED, (frames)
FLASH SCREEN: ORANGE, (frames)
FLASH SCREEN: YELLOW, (frames)
FLASH SCREEN: GREEN, (frames)
FLASH SCREEN: BLUE, (frames)
FLASH SCREEN: PURPLE, (frames)
FLASH SCREEN: MAGENTA, (frames)
FLASH SCREEN: BLACK, (frames)
FLASH SCREEN: (red), (green), (blue), (intensity), (frames)

Causes the game screen to flash a set color. If for the arguments, you use a color name, it will use a premade flash setting. If you choose to use your own settings, use the red, green, blue, intensity format to determine what color flash you would like. Red, green, blue, and intensity settings range from 0 to 255. If frames are used, that will be the duration of the screen flash. If omitted, the default frame count will be 60 frames.

Usage Example: flash screen: white
flash screen: red, 45
flash screen: 128, 170, 214, 170
flash screen: 68, 68, 68, 170, 45

==========

FLOAT target: (height), (frames)
FLOAT target: (height%), (frames)

Causes the target to float into the air above the ground by height%. The height is relative to the floating target. Using 100% means the target will float above the ground 100% higher than its height. If no '%' sign is used, the target will float that many pixels rather than a percentage of the

target's height. The frames determine how many frames it will take for the target to reach that height. Using 0% for the height will bring the target back to the ground.

Usage Example: float user: 200%
float enemies: 500, 30
float target: 0%, 30

==========

HIDE BATTLE HUD

Hides the battle hud to not obstruct any animations being played. You can reveal the battle hud again using 'show battle hud'.

Usage Example: hide battle hud

==========

JUMP target: (height), (frames)
JUMP target: (height%), (frames)

Causes the target to jump a height relative to the target itself. If the target jumps a height of 200%, the height will be 200% of the target's height. If no '%' sign is used, the target will jump that many pixels rather than a percentage of the target's height. The frame count is how long the target will be in the air. You can use this with the 'Move' action sequence to make the target appear like it is jumping a distance.

Usage Example: jump user: 150%
jump target: 300, 60

If you want to make the user jump forward over a certain distance, you'll have to use the following :

JUMP user : 120, 30
MOVE user : FORWARD, x, 30

While x can be anthing, try to make the frame duration match for more realism.

==========

MOTION type: target, (no weapon)

MOTION WALK: target
MOTION STANDBY: target
MOTION CHANT: target
MOTION GUARD: target
MOTION DAMAGE: target
MOTION EVADE: target
MOTION ATTACK: target
MOTION THRUST: target
MOTION SWING: target
MOTION MISSILE: target
MOTION SKILL: target
MOTION SPELL: target
MOTION ITEM: target
MOTION ESCAPE: target
MOTION VICTORY: target
MOTION DYING: target
MOTION ABNORMAL: target
MOTION SLEEP: target
MOTION DEAD: target

Forces the target to perform the specific type of action in sideview. If you issue an action sequence command for the target to perform 'attack', the target will automatically determine based on the weapon it has equipped to use either a thrust, swing, or missile motion. Attack, thrust, swing, and missile will also display the target's weapon if the target has one.

If 'no weapon' is used after the target, no weapons will be displayed. This effect will only work with the Thrust, Swing, and Missile motions.

Usage Example: motion walk: user
motion thrust: user, no weapon

==========

MOVE target: args

MOVE target1: HOME, (frames)
MOVE target1: RETURN, (frames)
MOVE target1: FORWARD, (distance), (frames)
MOVE target1: BACKWARD, (distance), (frames)
MOVE target1: POINT, x coordinate, y coordinate, (frames)
MOVE target1: target2, BASE, (frames)
MOVE target1: target2, CENTER, (frames)
MOVE target1: target2, HEAD, (frames)
MOVE target1: target2, FRONT BASE, (frames)
MOVE target1: target2, FRONT CENTER, (frames)
MOVE target1: target2, FRONT HEAD, (frames)
MOVE target1: target2, BACK BASE, (frames)
MOVE target1: target2, BACK CENTER, (frames)
MOVE target1: target2, BACK HEAD, (frames)

This is a move command. Arguments can be used in the above formats. This action sequence command will move target1 to any of those locations listed in the arguments. If it's towards target2, you must include what location relative to target2 for target1 to travel to.

Usage Example: move user: home, 20
move target: forward, 48, 12
move enemy 1: point, 400, 300
move actor 2: front base, 20

This is the very core of any physical animation !

Most everything depends on where to move your character, and more importantly, how fast he will.

You should try moving around your target and use ANIMATION x : target, (mirror) to see what wonders you could make !

==========

OPACITY target: x, (frames)
OPACITY target: x%, (frames)

Changes the opacity of the target to x (0-255) or x% (0% to 100%). If you use 'frames', that will be the frame duration for the change in opacity for the target.

Usage Example: opacity user: 50%, 30
opacity not focus: 0

==========

SHOW BATTLE HUD

If the battle hud was hidden using 'hide battle hud', use this to show the battle hud back within the player's screen.

Usage Example: show battle hud

==========

SHAKE SCREEN: (power), (speed), (frames)

Causes the game screen to shake. Adjust the power from 0-9, speed from 0-9, and the frames to alter the duration of the screen shaking. If those values are omitted, they will default to 5 power, 5 speed, and 60 frames.

Usage Example: shake screen
shake screen: 9
shake screen: 3, 9, 30

==========

TINT SCREEN: args

TINT SCREEN: NORMAL, (frames)
TINT SCREEN: DARK, (frames)
TINT SCREEN: SEPIA, (frames)
TINT SCREEN: SUNSET, (frames)
TINT SCREEN: NIGHT, (frames)
TINT SCREEN: (red), (green), (blue), (gray), (frames)

Tints the battle screen. If using the arguments 'normal', 'dark', 'sepia', 'sunset', or 'night' the screen will be be given a premade tint. If not, then the arguments for red, green, blue, and gray values must be inputted for the tint. Red, green, and blue can range from -255 to 255 while gray will range from 0 to 255. If frames are used, that will be the duration for which the screen will change to the tint. If omitted, the default amount of frames used will be 60 frames.

Usage Example: tint screen: normal
tint screen: sepia, 30
tint screen: 68, -34, -34, 0
tint screen: 68, -68, 0, 68, 45

TINT SCREEN doesn't only affect the background, but also the battlers.

Make sure you return the tint to normal after you don't need it for the skill.

==========

WAIT FOR FLOAT

Waits for all battler float changes to finish before going on to the next action in the action sequence.

Usage Example: wait for float

==========

WAIT FOR JUMP

Waits for all battler jumps to finish before going on to the next action in the action sequence.

Usage Example: wait for jump

==========

WAIT FOR OPACITY

Waits for all battlers to finish changing opacity before going on to the next action in the action sequence.

Usage Example: wait for opacity

WAIT : x

Waits for x game frames, where 1 second is 60 frames.

Usage Example : wait : 30

WAIT is what you WILL abuse in those notetags. There are no skills that have no WAIT tag, whatever comes after (Like WAIT FOR ANIMATION).

Timing is everything in battle as well as in sequence making.

Action Sequence Pack 3

The third pack contains every command related to the camera action

Please note that the Battle Camera option in the Parameters can be turned off in-game.

==========

CAMERA CLAMP ON
CAMERA CLAMP OFF

By default, the camera clamp is on, which forces the camera to never pan outside of the battlefield's boundaries. However, in the event you wish to turn this off, use 'camera clamp off' to shut off the clamp. The clamp, however, will be turned back on at the end of each 'perform finish' action.

Usage Example: camera clamp on
camera clamp off

==========

CAMERA FOCUS: target, (location), (frames)

CAMERA FOCUS: target, FRONT BASE, (frames)
CAMERA FOCUS: target, BASE, (frames)
CAMERA FOCUS: target, BACK BASE, (frames)
CAMERA FOCUS: target, FRONT CENTER, (frames)
CAMERA FOCUS: target, CENTER, (frames)
CAMERA FOCUS: target, BACK CENTER, (frames)
CAMERA FOCUS: target, FRONT HEAD, (frames)
CAMERA FOCUS: target, HEAD, (frames)
CAMERA FOCUS: target, BACK HEAD, (frames)

This will focus on a target(s) (refer to target typing) and a location. If the location is omitted, the camera will focus on the target(s)'s center.
Note: The camera will not shift past screen boundaries.

Usage Example: camera focus: user
camera focus: target, front, 40
camera focus: enemies, center, 30

CAMERA FOCUS will stay on the defined target.

As such, i twill follow every movement the target does. Move him, the camera will follow.
Focus on the user for the casting animation, then to the target. That's the most common yet effective use.

==========

CAMERA OFFSET: DIRECTION, DISTANCE

CAMERA OFFSET: LEFT, distance
CAMERA OFFSET: RIGHT, distance

CAMERA OFFSET: UP, distance
CAMERA OFFSET: DOWN, distance

Offsets the camera a direction by (distance) amount.

Usage Example: camera offset: left, 200
camera offset: right, Graphics.boxWidth / 4
camera offset: up, 300
camera offset: down, $gameVariables.value(3);
==========

CAMERA PAN

CAMERA PAN: LEFT, distance, (frames)
CAMERA PAN: RIGHT, distance, (frames)
CAMERA PAN: UP, distance, (frames)
CAMERA PAN: DOWN, distance, (frames)

Pans the camera a direction a certain distance in pixels. You can use a
combination of left/right and up/down to perform a diagonal camera pan.
Using 'frames' will allow you to adjust the duration of the camera pan.
Omitting 'frames' will set the camera pan duration to 30 frames.
Note: The camera will not shift past screen boundaries.

Usage Example: camera pan: left, 200
camera pan: up, 250
camera pan: right, 500, 60
camera pan: down: 300, 60
==========

CAMERA SCREEN

CAMERA SCREEN: TOP LEFT, (frames)
CAMERA SCREEN: FAR LEFT, (frames)
CAMERA SCREEN: BOTTOM LEFT, (frames)
CAMERA SCREEN: TOP CENTER, (frames)
CAMERA SCREEN: CENTER, (frames)
CAMERA SCREEN: BOTTOM CENTER, (frames)
CAMERA SCREEN: TOP RIGHT, (frames)
CAMERA SCREEN: FAR RIGHT, (frames)
CAMERA SCREEN: BOTTOM RIGHT, (frames)
CAMERA SCREEN: POINT, x, y, (frames)
CAMERA SCREEN: target, FRONT, (frames)
CAMERA SCREEN: target, BASE, (frames)
CAMERA SCREEN: target, BACK, (frames)
CAMERA SCREEN: target, FRONT CENTER, (frames)
CAMERA SCREEN: target, CENTER, (frames)
CAMERA SCREEN: target, BACK CENTER, (frames)
CAMERA SCREEN: target, FRONT TOP, (frames)

CAMERA SCREEN: target, TOP, (frames)
CAMERA SCREEN: target, BACK TOP, (frames)

Moves the camera to a certain part of the screen. If you choose a target, the camera will lock to that part of the target. Using (frames) will determine the duration of the time the camera will move over to the target location. Omitting (frames) will set the camera pan duration to 30 frames.
Note: The camera will not shift past screen boundaries.

Usage Example: camera screen: top left
camera screen: far right, 30
camera screen: point, 400, 300, 60
camera screen: user, base
camera screen: targets, base, 60

==========

RESET CAMERA: (frames)

Resets the camera location back to default location, which is the center of the battlefield. Using (frames) will allow you to adjust the duration in which the camera resets. Omitting 'frames' will set the camera to reset in 30 frames.
Note: The camera will not shift past screen boundaries.

Usage Example: reset camera
reset camera: 30

==========

RESET ZOOM: (frames)

Resets the camera zoom back to default zoom, which is 100%. Using (frames) will allow you to adjust the duration in which the zoom resets. Omitting 'frames' will set the zoom to reset in 30 frames.
Note: The camera will not shift past screen boundaries.

Usage Example: reset zoom
reset zoom: 30

Use RESET ZOOM and RESET CAMERA together in the <finish action>.

Using the (frames) part will help you making an effect of your liking.

Sudden camera reset or progressive ? It is up to you to make an effect.

==========

WAIT FOR CAMERA

Waits for the camera to finish panning before going on to the next action in the action sequence.

Usage Example: wait for camera

==========

WAIT FOR ZOOM

Waits for the zoom to finish changing before going on to the next action in the action sequence.

Usage Example: wait for zoom

==========

ZOOM: x%, (frames)
ZOOM: x.y, (frames)

Zooms to x% or x.y rate. Using (frames) will allow you to adjust the duration in which the zooming occurs. Omitting 'frames' will set the zoom duration to 30 frames.

Usage Example: zoom: 200%
zoom: 1.5, 45

==========

Battle system related Plugins

Of course, this doesn't end here. With the new mechanics affecting your game, new lines can be added to add special effects to your (already flashy) action sequences.

Active Time Battle Plugin

Most fans of Final Fantasy games will recognize that mechanic. Based of ticks and a gauge, you can only attack when the gauge is full. The ATB Plugin provides ways to control the ATB Gauge.

==========

ATB CHARGE: target, X
ATB CHARGE: target, X%
ATB CHARGE: targets, +X
ATB CHARGE: targets, +X%

Usable only for ATB. Sets the target's ATB charge to X or X%. This only applies when the target is in the ATB charge phase. This will not affect the user to prevent mechanical issues.

Usage Example: atb charge: targets, +5000
atb charge: target, -50%

==========

ATB GAUGE: target, X
ATB GAUGE: target, X%
ATB GAUGE: targets, +X
ATB GAUGE: targets, +X%

Usable only for ATB. Sets the target's ATB speed or charge (whichever the user is currently filling up) to X or X%. This only. This will not affect the user to prevent mechanical issues.

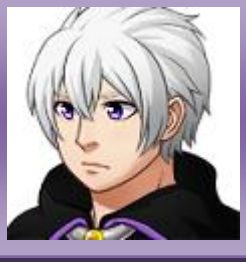
Usage Example: atb gauge: targets, +5000
atb gauge: target, -50%

==========

ATB INTERRUPT: target

Usable only for ATB. If the target is in the charging phase, interrupt it.

Usage Example: atb interrupt: targets



The difference between using the ATB INTERRUPT command and the <ATB Interrupt> notetag is that the Action Sequence command will always do the interruption.

If you want the skill to have a certain percent chance to interrupt, put the <ATB Interrupt : x%> notetag instead.

==========

ATB SPEED: target, X
ATB SPEED: target, X%
ATB SPEED: targets, +X
ATB SPEED: targets, +X%

Usable only for ATB. Sets the target's ATB speed to X or X%. This only applies when the target is filling up its ATB gauge. This will not affect the user to prevent mechanical issues.

Usage Example: atb speed: targets, +5000
atb speed: target, -50%
==========

Charge (Conditionnal) Turn Battle Plugin

This plugin adds a battle system mechanic akin to Final Fantasy X.

==========
CTB ORDER: target, +X
CTB ORDER: target, -X

Usable only for CTB. Moves target's position in the turn order by +x or -x.
+x will make the target having to wait more before getting their turn while -x will make the target having to wait less. The effect is minimal and will only last for the current turn cycle.
* Note: If you use this for multiple targets, each target will shift turns individually at a time.

Usage Example: ctb order: target, -2
ctb order: target, +3

Use that for a skill that delays the enemy's turn.

A skill that boosts your allies' turn order is also an idea.

CTB ORDER gives a lot of tactic value. Use it well.

==========
CTB SPEED: target, X
CTB SPEED: target, X%
CTB SPEED: targets, +X
CTB SPEED: targets, +X%

Usable only for CTB. Sets the target's CTB speed to X or X%. This only applies when the target is filling up its CTB gauge. This will not affect the user to prevent mechanical issues.

Usage Example: ctb speed: targets, +5000
ctb speed: target, -50%
==========

Skill Core related Plugins

Skill Cooldowns Add-on

While you can use the proper notetags, the Skill Cooldowns Add-On also gives you the possibility to manage the Cooldowns using the sequencer.

A good usage of the IF … ELSE would put that to its optimal use.

==========

GLOBAL COOLDOWN: targets, +X
GLOBAL COOLDOWN: targets, -X
GLOBAL COOLDOWN: targets, X

Sets the cooldown for all of the targets to be adjusted by X value. This applies to every skill that doesn't bypass cooldowns.

Usage Example: global cooldown: target, +5
global cooldown: user, -3
global cooldown: enemies, 10

==========

SKILL X COOLDOWN: targets, +Y
SKILL X COOLDOWN: targets, -Y
SKILL X COOLDOWN: targets, Y

Causes skill X to be adjusted by Y value for the targets. This only applies the specific skill x's cooldown.

Usage Example: skill 10 cooldown: target, +5
skill 12 cooldown: user, -3
skill 15 cooldown: enemies, 10

==========

SKILL TYPE X COOLDOWN: targets, +Y
SKILL TYPE X COOLDOWN: targets, -Y
SKILL TYPE X COOLDOWN: targets, Y

Causes skill type X skills to be adjusted by Y value for the targets. This only applies the specific skill type x skill's cooldown.

Usage Example: skill type 1 cooldown: target, +5
skill type 2 cooldown: user, -3
skill type 5 cooldown: enemies, 10

==========

Damage Core Plug-ins

The Damage Core plugin and its add-ons gives more control over damage done and healing. Lines in the Action Sequences are also added.

==========

BYPASS DAMAGE CAP

This will override all damage caps. This is applied to healing, too.

Usage Example: bypass damage cap

On my journey around the world, I met up with two mage twins. Using Twincast, they casted a spell in sync.

That spell was so powerful, it messed with the rules of natures and had bypassed the damage cap of 9999…

I learned that together with the ones you love, you are stronger than alone.

==========

DAMAGE CAP: x
HEALING CAP: x

This sets the action's damage cap to x, overriding all over damage caps in play except its own. This will also apply to healing, too.

Usage Example: damage cap: 999
healing cap: 999999

==========

DAMAGE RATE: x%
DAMAGE RATE: x.y
DAMAGE RATE: VARIABLE x

This changes the damage rate across all types of damage (physical, magical, and certain hit). The damage rate is reset at the end of each action sequence. If you use a variable, it is treated as a percentage.

Usage Example: damage rate: 50%
damage rate: 8.667
damage rate: variable 3

This part is the most handy of all. You can make combos with it. I explain. If you put ACTION EFFECT several times, the damage formula will be applied. But if you use the DAMAGE RATE, you can be more accurate on what hit you want to be the strongest. The last hit of a three hit combo is usually the strongest, after the two first weakened the foe's guard.

==========

FLAT DAMAGE: +x
FLAT DAMAGE: -x
FLAT DAMAGE: VARIABLE x

This adds a flat damage across all types of damage (physical, magical, and certain hit). The flat damage is reset at the end of each action sequence. If you use a variable, it is added onto the damage.

Usage Example: flat damage: +100
flat damage: -250
flat damage: variable 3

==========

FLAT GLOBAL: +x
FLAT GLOBAL: -x
FLAT GLOBAL: VARIABLE x

This adds a flat global damage and heal across all types of damage (physical, magical, and certain hit). The flat damage and heal is reset at the end of each action sequence. If you use a variable, it is added onto the damage and heal.

Usage Example: flat global: +100
flat global: -250
flat global: variable 3

==========

FLAT HEAL: +x
FLAT HEAL: -x
FLAT HEAL: VARIABLE x

This adds a flat heal across all types of damage (physical, magical, and certain hit). The flat heal is reset at the end of each action sequence. If you use a variable, it is added onto the heal.

Usage Example: flat heal: +100
flat heal: -250
flat heal: variable 3

==========

GLOBAL RATE: x%
GLOBAL RATE: x.y
GLOBAL RATE: VARIABLE x

This changes the damage and healing rates across all types of damage (physical, magical, and certain hit). The damage and healing rates are reset at the end of each action sequence. If you use a variable, it is treated as a percentage.

Usage Example: global rate: 50%

global rate: 8.667
global rate: variable 3
==========

HEAL RATE: x%
HEAL RATE: x.y
HEAL RATE: VARIABLE x

This changes the healing rate across all types of damage (physical, magical, and certain hit). The healing rate is reset at the end of each action sequence. If you use a variable, it is treated as a percentage.

Usage Example: heal rate: 50%
heal rate: 8.667
heal rate: variable 3

A magician friend of mine was our only healer. We had to often rely on potions and on her healing spells.
When she was given a certain accessory, her HEAL RATE increased by 35%.
I'm pretty sure there was a IF… ELSE in the Action Sequence.

What was the name of that accessory again… Medical Code ? No…

==========

RESET DAMAGE CAP

This will reset the damage cap implemented by the Damage Cap action sequence. This will also reset the effects of the Bypass Damage Cap action sequence.

Usage Example: reset damage cap
==========

RESET DAMAGE MODIFIERS

This will cause all damage and healing modifiers caused by action sequences to reset. While they normally reset at the end of each action sequence, this will allow you to do it manually.

Usage Example: reset damage modifiers
==========

Critical Control

Critical hits are a lucky moment when you deal more damage than usual. By default in RPG Maker MV, the damage done is multiplied by three, but with this plugin, you can set out your own. The Action Sequence also got new lines added to manage the critical hits and damage better.

==========

CRITICAL MULTIPLIER: x%
CRITICAL MULTIPLIER: x.y
CRITICAL MULTIPLIER: VARIABLE x

This changes the critical multiplier for the skill/item until the end of the action sequence. This will only take effect if there is a critical hit. If you use a variable, it is treated as a percentage.

Usage Example: critical multiplier: 50%
critical multiplier: 8.667
critical multiplier: variable 3

==========

FLAT CRITICAL: +x
FLAT CRITICAL: -x
FLAT CRITICAL: VARIABLE x

This changes the flat critical increase for the skill/item until the end of the action sequence. This will only take effect if there is a critical hit. This will automatically adjust for damage or healing.

Usage Example: flat critical: +100
flat critical: -250
flat critical: variable 3

==========

FORCE CRITICAL
FORCE NO CRITICAL
NORMAL CRITICAL

This forces the following action effects in the action sequence to either always be a critical hit or not a critical hit ignoring all other factors. Using 'normal critical' will reduce the following action effects to use the regular critical hit rate calculation.

Usage Example: force critical
Force no critical
Normal critical

I never understood why my samurai friend was dealing more crits than me. It turned out that when his HP were hitting below 25% of their maximum, the skill would always be critical.

I found out later that this skill had IF (user.hp < user.mhp / 4) … ELSE and a FORCE CRITICAL somewhere… I still owe him one.

==========

ARMOR PENETRATION: X
ARMOR PENETRATION: X%

Causes the skill to have a global armor pentration property of either x or x% rate. This property is reset at the end of the action sequence.

Usage Example: armor penetration: 20
armor penetration: 30%

==========

ARMOR REDUCTION: X
ARMOR REDUCTION: X%

Causes the skill to have a global armor reduction property of either x or x% rate. This property is reset at the end of the action sequence.

Usage Example: armor reduction: 20
armor reduction: 30%

==========

RESET ARMOR PENETRATION

Resets global set armor penetration properties.

Usage Example: reset armor penetration

==========

RESET ARMOR REDUCTION

Resets global set armor reduction properties.

Usage Example: reset armor reduction

I had a hard time realizing what were the differences between ARMOR REDUCTION and ARMOR PENETRATION. REDUCTION happens first and is provided by the target and PENETRATION happens last and is provided by the user.

Don't overestimate your defense, if your enemy have <Armor Penetration : 50%>, you'll suffer at a high level. 50% of 500 DEF is 250, which is huge.

Element Core

The Element Core plugin was released the 10/06/2016. The Melody Sequencer allows you to bring more versatility into your actions.

This plugin has more interest in being used in conditionnal branches.

A skill's element is better to be preset in your skill's settings, for clarity's sake.

ADD ELEMENT: X
ADD ELEMENT: X, X, X
ADD ELEMENT: NAME
ADD ELEMENT: NAME, NAME, NAME
- -
This will add more elements to the currently existing elements. This will not include forced elements.
- -
Usage Example: add element: 4
add element: 5, 6, 7
add element: fire
add element: ice, wind, water
==========

==========
CLEAR ELEMENT
- -
This will clear any ___ Element action sequence settings and revert element calculation back to normal.
- -
Usage Example: clear element
==========

==========
FORCE ELEMENT: X
FORCE ELEMENT: X, X, X
FORCE ELEMENT: NAME
FORCE ELEMENT: NAME, NAME, NAME
- -
This will override elemental settings for elemental damage rate calculations except for customized calculations that aim at specific elements. This will ignore all other elements.
- -
Usage Example: force element: 4
force element: 5, 6, 7

force element: fire
force element: ice, wind, water
==========

==========
NULL ELEMENT
- -
This will force the elements to null. Calculated elemental damage will always return neutral rate. Using this will clear the Force Element action sequence effect.
- -
Usage Example: null element

Action Sequence example : The “Cleave” Skill

With everything above, you’re ready to make an animated skill. Huuzah!

You have everything to do your own Action Sequence. All you have to do is to find the correct delaying between the actions and the waits. I’m going to give you a freebie : my home-made Cleave skill! There are some like it, but this one’s my version.

<setup action>
OPACITY not focus: 0
MOVE user: POINT, 540, 400
IMMORTAL: target, true
WAIT: 50
DISPLAY ACTION
CAST ANIMATION
WAIT FOR ANIMATION
</setup action>

Let’s see here… OPACITY not focus: 0 will make the rest of the battler’s allies transparent. That effect is to show who will be getting the hits. You can omit it if you want. The battler will then move at a given pint of the battlefield. Since we don’t want the targets to die exactly when their HP reaches 0, we’ll make them immortal. Some time is waited, the “Cleave” name will show, and the cast animation will be played, and nothing else will happen while the animation is still playing.

Now, since the skill hits the whole enemy party, we will use <whole action>.

<whole action>
MOVE user: FORWARD, 450, 24
MOTION ATTACK: user
ACTION ANIMATION
Wait: 60
JUMP user: 180, 35
MOVE user: BACKWARDS, 450, 30
WAIT FOR MOVEMENT
</whole action>

The battler will move forward while swinging his weapon, making the skill animation play on all enemies at the same time. A second is waited, then the user will jump backwards. We will wait for the movement to be played…

<target action>
ACTION EFFECT
</target action>
…Then the damage will be dealt.

<follow action>
CLEAR BATTLE LOG
OPACITY not focus: 255
WAIT: 25
</follow action>
The “Cleave” message will vanish, and the other battlers will show up again. We wait before the next actions.

<finish action>
FACE user: HOME
perform finish
IMMORTAL: target, false
</finish action>
The user faces his home location if he isn’t already. He will move back there, while at the same time, any enemy with 0 HP will now die.

Most of the fun comes out by trying things. You can do a lot with the Action Sequence, the only limit is your imagination!

Credits page

The Action Sequence was made by Yanfly. That fellow makes plug-ins for us all to use. You can visit his page here.