# Tutorial #2 - Add Hits

## Overview

This tutorial will show a few examples of how to use AddHits to create some achievements in RAScript. [Streets of Rage 2](https://retroachievements.org/game/3) was chosen as the tutorial example to show how adding hits can be used to improve counting the number of enemies the player has knocked out.<br>

![Streets of Rage 2 Title Screen](Streets\_of\_Rage\_Title.png)<br>



## AddHits

Each comparison in an achievement's logic is evaluated every frame. By default each comparison is set to zero \*hits\* which means that the comparison will not remember its previous evaluations. For the achievement to be triggered a comparison with zero \*hits\* must evaluate true in the same frame as the rest of the comparisons in the achievement’s logic. If the comparison is set to one or more \*hits\*, then the comparison will keep track of how many frames that the comparison evaluated to true. Once number of \*hits\* equals the initial \*hits\* setting the comparison will remain true until either the achievement is triggered or reset.<br>

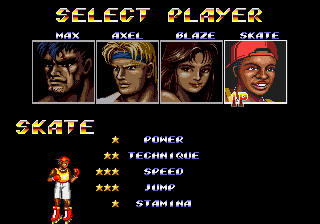
<br>

In some cases you’ll want to combine \*hits\* together from multiple comparisons. The following examples will show how to add \*hits\* together for tracking which characters the player has used and to count how many enemies the player has knocked out. This tutorial will use a few different type of RAScript \*hits\* related commands which can be found on the RATools wiki in the [Trigger-Functions](<https://github.com/Jamiras/RATools/wiki/Trigger-Functions>) section.

## Example 2A - Team Game

Streets of Rage 2 has four playable characters: Max, Axel, Blaze, and Skate. Each character has their own statistics and abilities that are significantly different enough to require a different strategy for each character. You are given the chance to select a character once at the beginning of the game and you may change your selection after continuing.<br>

![Character selection screenshot with Skate selected](Skate\_Select.png)<br>



The following code will track that you’ve played with each character at least once. The achievement will pause with the \*unless\* command whenever the screen mode is not in game. That way we don’t trigger \*hits\* during the character selection screen and the player must use the character in game. Using the \*once\* command the player selected comparison will latch on when the character is selected and stay on indefinitely. When using \*hits\* you would usually want include a \*never\* statement to reset the logic however, in this case we don’t need to reset the logic. Note that the \*hits\* would all be reset in a new session.

```

// Streets of Rage 2

// #ID = 3

// $FC02: Screen Mode- 0=segalogo, 4=pressstart, 8=demo, c=mainmenu, 10=options, 14=ingame, 18=charselect, 1c=ending, 24=introcards, 28=credits

function ScreenMode() => byte(0x00FC02)

// $FC50: Axel selected = 0xff

function AxelSelected() => word(0x00FC50)

// $FC51: Max selected = 0xff

function MaxSelected() => word(0x00FC51)

// $FC52: Skate selected = 0xff

function SkateSelected() => word(0x00FC52)

// $FC53: Blaze selected = 0xff

function BlazeSelected() => word(0x00FC53)

// Record a hit for each character active during gameplay

// This variation is the current core achievement

achievement(

title = "Team Game",

description = "Play once as every character",

points = 4,

trigger = unless(ScreenMode() != 20) &&

once(AxelSelected() == 255) &&

once(BlazeSelected() == 255) &&

once(MaxSelected() == 255) &&

once(SkateSelected() == 255)

)

```

The above works perfectly fine for tracking which character is used. A measure could be added to this function to give the player some feedback of how many characters they have used. To add the measure we will need to combine the \*hits\* using the \*tally\* function. Since the pause would disable the achievement while not in-game another modification was made to check the game mode with the selected player at the same time. When two or more comparisons are combined together like this in an \*once\* command they are grouped together with \*AndNext\* flags. So the result of this following code is a \*AddHits\* and \*AndNext\* chain that will sum up to four when every character has been selected.

```

// Record a hit for each character active during gameplay

// This variation of the using add hits to support using measure

achievement(

title = "Example 2A: Team Game",

description = "Play once as every character",

points = 0,

trigger = measured(

tally(4,

once(ScreenMode() == 20 && AxelSelected() == 255),

once(ScreenMode() == 20 && MaxSelected() == 255),

once(ScreenMode() == 20 && SkateSelected() == 255),

once(ScreenMode() == 20 && BlazeSelected() == 255)

)

)

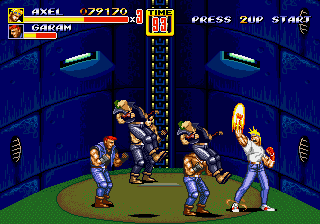
)

```

## Example 2B – Untouchable

The main goal of Streets of Rage 2 is to fight your way through eight different stages full of enemies trying to stop you from getting to their boss, Mr X. As such, you are constantly beating up and knocking out enemies as you play through the game. The game memory tracks how many enemies you have knocked out overall from stage one. You can use this information to add challenges like tracking how many enemies the player has knocked out in one life. <br>

![Screenshot of Axel knocking out multiple enemies](Axel\_Uppercut.png)<br>



Since the number of total knockouts from stage one might not be the same value as how many knockouts the player has done in one life we must track the value differently. The following code will increment one hit every time the Player1KO() value increases. The knockout count will reset whenever the Player1Lives() value decreases.

```

// Streets of Rage 2

// #ID = 3

// $EF4E: Player one, number KOs

function Player1KO() => word(0x00EF4E)

// $EF82: Player one lives

function Player1Lives() => word(0x00EF82)

// $FC02: Screen Mode- 0=segalogo, 4=pressstart, 8=demo, c=mainmenu, 10=options, 14=ingame, 18=charselect, 1c=ending, 24=introcards, 28=credits

function ScreenMode() => byte(0x00FC02)

// Count how many enemies where knocked out in one life

// This variation is the current core achievement

achievement(

title = "Untouchable",

description = "Defeat 200 enemies without dying",

points = 25,

trigger = repeated(200, Player1KO() > prev(Player1KO())) &&

never(Player1Lives() < prev(Player1Lives()))

)

```

Using \*AddHits\* to count like this has a potential issue that would throw the count off. If the player knocks out two or more enemies at once the count would only increment by one. To remedy this you would need to add additional \*hits\* when the player knocks out two enemies, three enemies, four enemies, and five enemies. Since Streets of Rage 2 has a maximum of five active enemies at once you would only need to track up to five simultaneous knock outs. The following code uses a combination of \*AddHits\* and \*AddSource\* to count how many knock outs have occurred:

``` // Count how many enemies where knocked out in one life

// This variation counts instance of multiple enemies getting knocked out at once.

achievement(

// Count how many enemies where knocked out in one life

// This variation counts instance of multiple enemies getting knocked out at once.

achievement(

title = "Example 2B: Untouchable",

description = "Defeat 200 enemies without dying",

points = 0,

trigger = measured(

tally(200,

prev(Player1KO()) < Player1KO(),

prev(Player1KO()) + 1 < Player1KO(),

prev(Player1KO()) + 2 < Player1KO(),

prev(Player1KO()) + 3 < Player1KO(),

prev(Player1KO()) + 4 < Player1KO()

)

) &&

never(ScreenMode() != 20) &&

never(Player1Lives() < prev(Player1Lives()))

)

```

The above code might be easier to understand if we put some numbers to it. Let say that the previous number of knockouts ```prev(Player1KO())``` is 100 and the player just knocked out three enemies at once raising the current number knockouts ```Player1KO()``` value to 103. The following comparisons for that frame would evaluate to:

```

prev(Player1KO()) < Player1KO() ➡ 100 < 103 ➡ true add hit

prev(Player1KO()) + 1 < Player1KO()➡101 < 103 ➡true add hit

prev(Player1KO()) + 2 < Player1KO()➡102 < 103 ➡true add hit

prev(Player1KO()) + 3 < Player1KO()➡103 < 103 ➡false no hit

prev(Player1KO()) + 4 < Player1KO()➡104 < 103 ➡false no hit

```

The result of the above evaluation for that frame would add three \*hits\* to the total. This may seem like an edge case scenario however, multiple knockouts occur often since the enemies will clump in front of the player. Using the code Example 2B you can see the hit distribution that occurred during a test run. Remember that if multiple simultaneous knockouts occur one hit will be added to all of the previous comparisons. Therefore to figure out the occurrence of each type of multi-knockout you must subtract the \*hits\* from the next comparison below. Out of 200 total knock outs there were:

\* 0 Quintuple knock outs

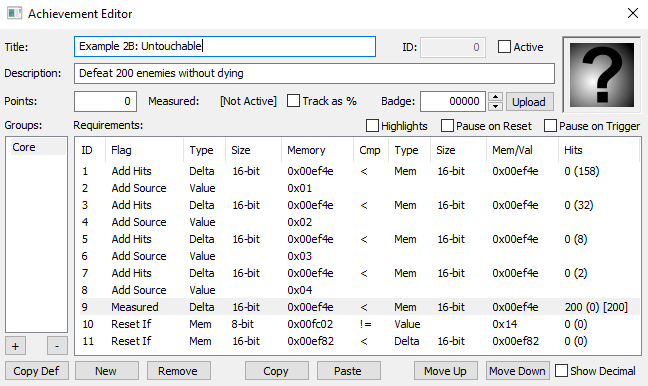
\* 2 Quadruple knock outs

\* 6 Triple knock outs

\* 24 Double knock outs

\* 126 Single knock outs

![Screenshot of hits counts for Example 2B](Untouchable\_Hits.PNG)<br>

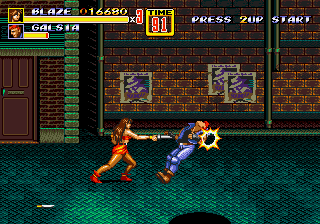


If we were just using the condition ```prev(Player1KO()) < Player1KO()``` then \*\*42 knock outs would have been missed!\*\* I wasn’t able to get a quintuple knock out during the test however, you can sleep well knowing that if a player does get five knock outs simultaneously all five will be counted.<br>

## Example 2C - Steel Grip

Besides beating up enemies with your bare knuckles the player can pick up and use weapons that the enemies drop. Weapons significantly increase your damage so once you get a weapon, it is best to hold on to it for as long as possible. The Steel Grip achievement rewards the player for holding on to the same weapon for ten or more knockouts.

![Screenshot of Blaze knocking an enemy out with a knife](Blaze\_Knife.PNG)<br>



```

// Streets of Rage 2

// #ID = 3

// $EF33: 8-bit - Holding Weapon

function HoldingWeapon() => byte(0x00EF33)

// $EF37: 8-bit - Weapon Type

function WeaponType() => byte(0x00EF37)

// $EF4E: Player one, number KOs

function Player1KO() => word(0x00EF4E)

// $FC02: Screen Mode- 0=segalogo, 4=pressstart, 8=demo, c=mainmenu, 10=options, 14=ingame, 18=charselect, 1c=ending, 24=introcards, 28=credits

function ScreenMode() => byte(0x00FC02)

// Count how many enemies where knocked out while holding a weapon

// This variation is the current core achievement

achievement(

title = "Steel Grip",

description = "Defeat 10 enemies without dropping your weapon",

points = 10,

trigger = once(HoldingWeapon() == 1) &&

repeated(10, Player1KO() > prev(Player1KO())) &&

never(WeaponType() != prev(WeaponType())) &&

never(HoldingWeapon() != 1)

)

```

## Homework 2

A) Adapt the Steel Grip achievement above to use \*tally\* similar to Example 2B so that multiple knock outs are counted.<br>

[Tutorial #2 Solution](Solution/readme.md)<br>

# Tutorial #2 Solution

## Solution A

Using the same method found in Example 2B we use a \*tally\* to add \*hits\* for multiple knockouts. The \*hits\* are reset if the player is not holding a weapon, the weapon type changed, or if the game mode changes from in game.

```

// Streets of Rage 2

// #ID = 3

// $EF33: 8-bit - Holding Weapon

function HoldingWeapon() => byte(0x00EF33)

// $EF37: 8-bit - Weapon Type

function WeaponType() => byte(0x00EF37)

// $EF4E: Player one, number KOs

function Player1KO() => word(0x00EF4E)

// $FC02: Screen Mode- 0=segalogo, 4=pressstart, 8=demo, c=mainmenu, 10=options, 14=ingame, 18=charselect, 1c=ending, 24=introcards, 28=credits

function ScreenMode() => byte(0x00FC02)

// Count how many enemies where knocked out while holding a weapon

// This variation counts instance of multiple enemies getting knocked out at once.

achievement(

title = "Example 2C: Steel Grip",

description = "Defeat 10 enemies without dropping your weapon",

points = 0,

trigger = measured(

tally(10,

prev(Player1KO()) < Player1KO(),

prev(Player1KO()) + 1 < Player1KO(),

prev(Player1KO()) + 2 < Player1KO(),

prev(Player1KO()) + 3 < Player1KO(),

prev(Player1KO()) + 4 < Player1KO()

)

) &&

never(ScreenMode() != 20) &&

never(WeaponType() != prev(WeaponType())) &&

never(HoldingWeapon() != 1)

)

```