

Animated Timer Manual - Version: 240810 - AT Version: 1.0.1

Table of content

General : Introduction	Page 2
General : Features	Page 2
Basics : Controlling the timer	Page 2
Basics : Manually setting time	Page 3
Basics : Timer modes	Page 3
Advanced : Configuration	Page 4
Advanced : Configuration : Render types.....	Page 5
Advanced : Configuration : Style types.....	Page 6
Advanced : Configuration : Colors	Page 7
Advanced : Configuration : Animations	Page 7
Advanced : Configuration : Style configuration	Page 9
Advanced : Configuration : Visibility.....	Page 10
Advanced : Configuration : Triggers.....	Page 10
Advanced : Profiles	Page 11
Advanced : Removing AT	Page 13
Details : Function namespaces	Page 14
Details : Scoreboards	Page 14
Details : NBT storage	Page 15
Modification : Custom animations	Page 16
Modification : Integration	Page 16
Modification : Integration : Configuration	Page 16
Modification : Integration : Extraction	Page 17

Animated Timer Manual - Version: 240810 - AT Version: 1.0.1

General : Introduction

Animated Timer (AT) data pack adds an in-game timer via the actionbar field of Minecraft's HUD. A wide range of functions ensures a simple yet versatile configuration, allowing you to customize your timer according to your likings and needs.

General : Features

In its original and unmodified state, features include:

- Count-up & count-down
- Multiple render (display) types
- Both static (never changing) and animated (always changing) color configuration
- General style configuration (bold & italic text)
- Pre-built animations
- Profiles (ability to save current configuration & load at a later time)
- Trigger scoreboards for complete configuration & interaction even if cheats are disabled
- Advanced features for map makers (facilitated embedding into external projects)

Basics : Controlling the timer

AT can be controlled the following way:

Description	Cheats enabled	Without cheats
Continue / resume the timer in the mode in which it was last active OR start it initially	<code>/function at:continue</code>	<code>/trigger continue</code>
Pause the timer	<code>/function at:pause</code>	<code>/trigger pause</code>
If counting up: Set time to zero and start counting up immediately If counting down: Restore last manual time setting and start counting down from there	<code>/function at:reset</code>	<code>/trigger reset</code>
Set time to zero and stay paused – next continue will always start counting upwards	<code>/function at:reboot</code>	<code>/trigger reboot</code>

Animated Timer Manual - Version: 240810 - AT Version: 1.0.1

Basics : Manually setting time

In some situations, you may want to alter the current time manually. This can be done either with a function in the *at-time* namespace or with */trigger*.

Description	Cheats enabled	Without cheats
Configure seconds	<code>/function at-time:add/second</code> <code>/function at-time:remove/second</code> <code>/function at-time:set/seconds/10</code>	<code>/trigger seconds set 10</code>
Configure minutes	<code>/function at-time:add/minute</code> <code>/function at-time:remove/minute</code> <code>/function at-time:set/minutes/15</code>	<code>/trigger minutes set 15</code>
Configure hours	<code>/function at-time:add/hour</code> <code>/function at-time:remove/hour</code> <code>/function at-time:set/hours/3</code>	<code>/trigger hours set 3</code>
Configure days	<code>/function at-time:add/day</code> <code>/function at-time:remove/day</code> <code>/function at-time:set/days/7</code>	<code>/trigger days set 7</code>

Note: The above functions that include *at-time:set* are just examples. In fact, a multitude of *set* functions are available for use.

Basics : Timer modes

AT is counting up by default but count-down is also available. You can toggle between both count modes with a single command:

Description	Cheats enabled	Without cheats
Switch between timer modes	<code>/function at-conf:mode</code>	<code>/trigger mode</code>

Animated Timer Manual - Version: 240810 - AT Version: 1.0.1

Advanced : Configuration

Part of what makes AT so unique is its versatile configuration, which you can print out at any time.

Description	Cheats enabled	Without cheats
View current configuration	<code>/function at-conf:show</code>	<code>/trigger show</code>

The output will look something like this, depending on what settings you currently use:

```
Running function at-conf:show
[AT] Current timer configuration:
Timer mode: DOWN
Render type: spaces / specifiers and zeros [SAZ]
Style type: animated
Animation: teal-yellow
Bold: true
Italic: false
Speed (ticks): 2
Direction: right-to-left
1h 46m 8s (paused)

<name> [<arguments>]with]
/function at-conf:show
```

Here is another example, this time using the static style type:

```
Triggered [show]
[AT] Current timer configuration:
Timer mode: DOWN
Render type: colons and zeros [CAZ]
Style type: static
Primary color: #f81b3e
Primary bold: false
Primary italic: false
Secondary color: #f81b3e
Secondary bold: true
Secondary italic: false
01:46:00 (paused)

<objective> [add|set]
/trigger show_
```

Animated Timer Manual - Version: 240810 - AT Version: 1.0.1

Advanced : Configuration : Render types

The render type defines the shape of your timer. There are three render types in total:

Colons and zeros [CAZ]

- Looks like this: DD:HH:MM:SS
- Every timer unit is always padded to two characters
- If an unit matches zero, it is shown



Spaces / specifiers and zeros [SAZ]

- Looks like this: DDd HHh MMm SSs
- No padding
- If an unit matches zero, it is shown



Spaces / specifiers no zeros [SNZ]

- Looks like this: DDd HHh MMm SSs
- No padding
- If an unit matches zero, it is hidden



To toggle between them, use one of these commands:

Description	Cheats enabled	Without cheats
Toggle render type	<code>/function at-conf:render</code>	<code>/trigger render</code>

Animated Timer Manual - Version: 240810 - AT Version: 1.0.1

Advanced : Configuration : Style types

The style type determines what color your timer has and what type of text design settings can be applied. There are two style types in total, static and animated.

Static

- Unchanged, constantly has the same color
- Splits timer output into two individually configurable zones, primary and secondary



Animated

- Gets its color from an animation
- Timer output is treated as one style zone



Again, if you want to switch between them, use one of these commands:

Description	Cheats enabled	Without cheats
Toggle style type	<code>/function at-conf:style</code>	<code>/trigger style</code>

Note: Most of the configuration commands below only work with one of the two style types. Be sure to first select your style type before configuring colors, animations, etc.

Animated Timer Manual - Version: 240810 - AT Version: 1.0.1

Advanced : Configuration : Colors

As mentioned before, a static style splits the timer into two areas that can be configured separately. This level of separation applies to colors as well as to other style settings. Changing the color works as follows:

Description	Cheats enabled	Without cheats
Change color of primary zone	<code>/function at-conf:color1 {color:aqua}</code>	<code>/trigger color1 set 2</code>
Change color of secondary zone	<code>/function at-conf:color2 {color:"#C63C51"}</code>	<code>/trigger color2 set 13</code>



Updating the color via `/trigger` uses Minecraft's in-game color codes. An overview of said colors can be found here: https://minecraft.wiki/w/Formatting_codes#Color_codes

Advanced : Configuration : Animations

By default, AT comes with 14 different animations that are available for you to use. Changing the animation can be achieved like this:

Description	Cheats enabled	Without cheats
Change animation	<code>/function at-conf:animation {name:red-peach}</code>	<code>/trigger animation set 2</code>

Animated Timer Manual - Version: 240810 - AT Version: 1.0.1

As with colors, */trigger* does not accept any text input. Despite this, selection of your favorite animation is still possible due to the scores being mapped to animations 1-11. See below:

Animation	Trigger value
red-peach	1
red-dark	2
red-purple	3
blue-pink	4
pastel-purple-blue	5
blue-mint	6
purple-mint	7
green-blue	8
teal-yellow	9
teal-lime	10
rainbow	11
red-yellow	12
white-purple	13
white-gray	14



Additionally, animation speed and direction may also be configured:

Description	Cheats enabled	Without cheats
Modify animation speed	<code>/function at-conf:speed {ticks:4}</code>	<code>/trigger speed set 2</code>
Modify animation direction	<code>/function at-conf:direction</code>	<code>/trigger direction</code>

Note: A higher speed value means that the animation runs more slowly; the fastest animation runs at speed 1.

Animated Timer Manual - Version: 240810 - AT Version: 1.0.1

Advanced : Configuration : Style configuration

Both the bold and italic design setting belong to this category. Depending on the current style type, the following commands behave differently when executed.

Description	Cheats enabled	Without cheats
Static style type: Modify bold style of primary zone Animated: Modify bold style of current animation	<code>/function at-conf:bold1</code>	<code>/trigger bold1</code>
Static style type: Modify bold style of secondary zone Animated: Modify bold style of current animation	<code>/function at-conf:bold2</code>	<code>/trigger bold2</code>
Static style type: Modify italic style of primary zone Animated: Modify italic style of current animation	<code>/function at-conf:italic1</code>	<code>/trigger italic1</code>
Static style type: Modify italic style of secondary zone Animated: Modify italic style of current animation	<code>/function at-conf:italic2</code>	<code>/trigger italic2</code>

```

Running function at-conf:bold1
[AT] Changed primary bold style to true
Function at-conf:bold1 returned 1
22h 30s

<name> [<arguments>]with]
/function at-conf:bold1_
    
```

```

Triggered [italic2]
[AT] Changed secondary italic style to true
22h 30s

<objective> [add|set]
/trigger italic2
    
```

Animated Timer Manual - Version: 240810 - AT Version: 1.0.1

Advanced : Configuration : Visibility

In some cases, you may want to temporarily disable AT's actionbar output. This can be done by issuing one of these two commands:

Description	Cheats enabled	Without cheats
Show / hide actionbar	<code>/function at-conf:visibility</code>	<code>/trigger visibility</code>

Advanced : Configuration : Triggers

By default, AT can be used both with and without cheats, but there are certain use cases where not every player should be able to alter the configuration and timer settings. As trigger scoreboards are part of the configuration, any priveleged player will be able to control their state (enabled / disabled) with ease:

Description	Cheats enabled	Without cheats
Toggle trigger scoreboards on / off	<code>/function at-conf:triggers</code>	-

Animated Timer Manual - Version: 240810 - AT Version: 1.0.1

Advanced : Profiles

Profiles allow you to store the current configuration and load it at a later time. By default, AT comes with 10 different profiles - the maximum amount of profiles one can have is 64. Sufficient profile management is achieved with the the following commands:

Description	Cheats enabled	Without cheats
Save current configuration at profile <name>	<code>/function at-prof:save {name:foo}</code>	<code>/trigger save set 1</code>
View configuration of profile <name>	<code>/function at-prof:view {name:foo}</code>	<code>/trigger view set 3</code>
Load profile <name>	<code>/function at-prof:load {name:foo}</code>	<code>/trigger load set 5</code>
Remove profile <name>	<code>/function at-prof:delete {name:foo}</code>	<code>/trigger delete set 7</code>
List existing profiles	<code>/function at-prof:list</code>	<code>/trigger list</code>



```
Running function at-prof:save  
[AT] Saved current configuration at profile bar  
[<arguments>]  
/function at-prof:save (name:bar)
```



```
Running function at-prof:view  
[AT] Contents of profile bar:  
Render type: colons and zeros [CAZ]  
Style type: animated  
Animation: purple-mint  
Bold: false  
Italic: false  
Speed (ticks): 2  
Direction: right-to-left  
[<arguments>]  
/function at-prof:view (name:bar)
```

Animated Timer Manual - Version: 240810 - AT Version: 1.0.1

```
22:00:30
Running function at-prof:load
[AT] Loaded profile bar
[arguments]
/function at-prof:load (name:bar)_
```

```
22:00:30
Running function at-prof:delete
[AT] Removed profile bar
[arguments]
/function at-prof:delete (name:bar)_
```

```
Running function at-prof:list
[AT] The following 10 profiles are available:
- galaxy [0]
- peach [1]
- lush [2]
- cloud [3]
- motley [4]
- breeze [5]
- herbs [6]
- desert [7]
- justpink [8]
- neon [9]
Function at-prof:list returned 1
22:00:30
<name> [arguments][with]
/function at-prof:list_
```

While profile management using */function* is consistent, meaning that you can save, load, etc to / from the same name, and will always access the same profile, management using */trigger* is not. Its designed this way to give players without cheats the ability to access existing (pre-made) profiles.

With triggers it works like this: The score you enter at the *save* gets appended behind "trigger", forming a unique name like "trigger0" or "trigger42", so saving works as expected.

Loading is a bit different though. To load the profile, you first need to know its index, which you can obtain by entering the *list* command. (the number in square brackets []) If you have decided what profile you want to have, run the *load* command with the index behind your profile. The correct profile should now be loaded. The same principle also applies to the *view* and *delete* command.

Animated Timer Manual - Version: 240810 - AT Version: 1.0.1

This functionality may change at a later time, if / when I have a better idea for solving this issue.

```
22:00:30  
Triggered [save] (set value to 9)  
[AT] Saved current configuration at profile trigger9  
  
    <value>  
/trigger save set 9
```

```
Triggered [list]  
[AT] The following 11 profiles are available:  
- galaxy [0]  
- peach [1]  
- lush [2]  
- cloud [3]  
- motley [4]  
- breeze [5]  
- herbs [6]  
- desert [7]  
- justpink [8]  
- neon [9]  
- trigger9 [10]  
  
    <objective> [add/set]  
/trigger list_
```

```
22:00:30  
Triggered [load] (set value to 10)  
[AT] Loaded profile trigger9  
  
    <value>  
/trigger load set 10
```

Advanced : Removing AT

If you are unhappy with AT, you may uninstall it at any given time. Use the following instructions to remove all scoreboards and NBT storages:

1. Run `/function at-main:uninstall`
2. Remove AT from data pack folder
3. Reload your world

Animated Timer Manual - Version: 240810 - AT Version: 1.0.1

Details : Function namespaces

AT implements 11 namespaces, 10 of them are AT-exclusive.

Namespace	Class	Description / purpose
at	UI	Timer controls
at-abar	internal	Generate & print actionbar
at-conf	UI	Timer / actionbar configuration
at-dcfg	UI / external	Direct Configuration - instantly modify configuration without messing around with config toggles from <i>at-conf</i>
at-dext	external	Do External - Collection of functions that run when specific things happen within the timer (useful for external projects)
at-main	internal	Timer backend
at-misc	internal	Stuff that does not fit into other namespaces
at-prof	UI	Profile management
at-time	UI	Timer controls
at-trig	internal	Trigger backend
minecraft	internal	Minecraft namespace; running tick and load

Details : Scoreboards

At load, AT adds 34 scoreboards, most of them are triggers.

Objective	Criteria	Description / purpose
at_s_state	dummy	Current & initial timer mode, version, parts of configuration
at_s_config	dummy	Most of current (running) config - style related
at_s_animation	dummy	Needed for rotating through the animation
at_s_profiles	dummy	Literally just the current profile count
at_c_time	dummy	That's the timer
at_t_interact	dummy	Temporary storage for UI interaction
at_t_profout	dummy	Temporary storage for viewing profiles

Trigger scoreboards (27 total):

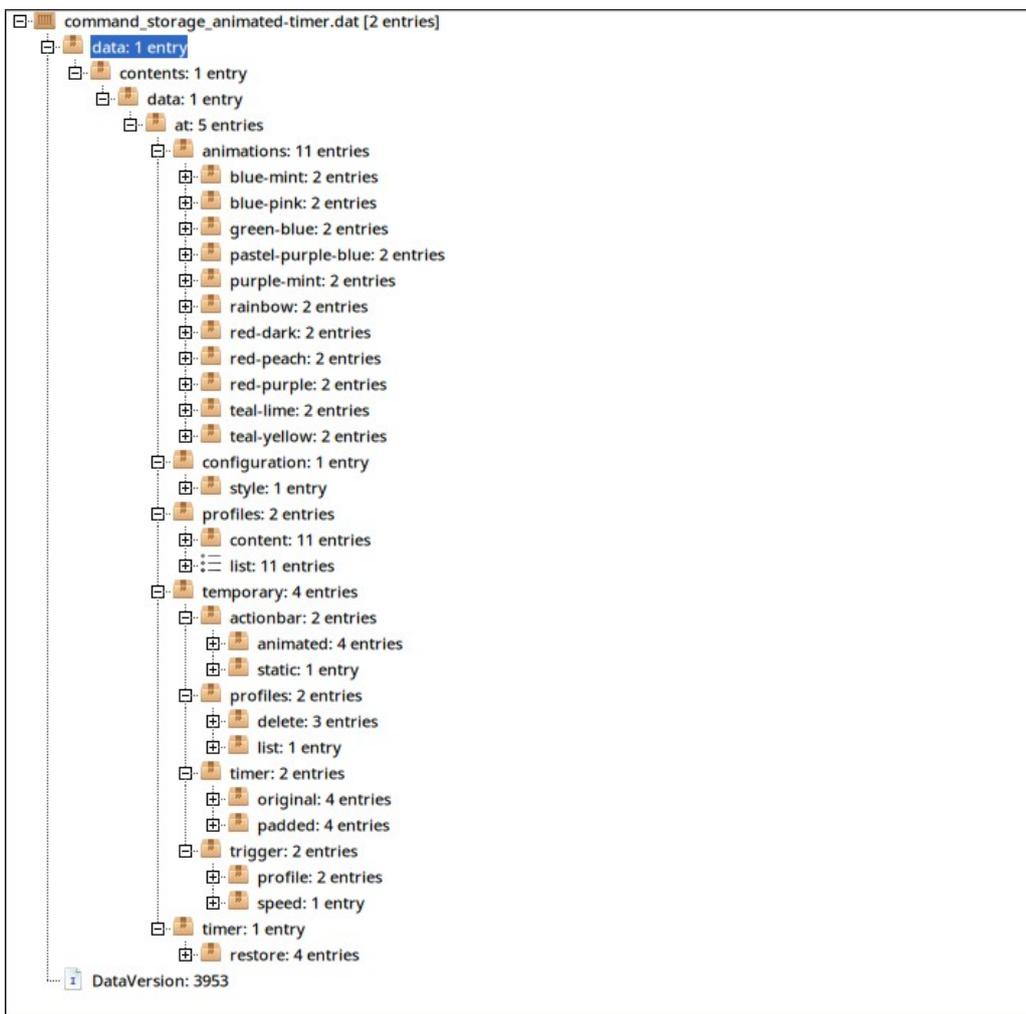
continue, pause, reboot, reset, animation, bold1, bold2, color1, color2, direction, italic1, italic2, mode, render, show, speed, style, visibility, delete, list, load, save, view, days, hours, minutes, seconds

Animated Timer Manual - Version: 240810 - AT Version: 1.0.1

Details : NBT storage

AT uses the storage namespace animated-timer:data.

Path	Description / purpose
at.animations	Holds animations that are used by style type animated
at.configuration	Part of current / running configuration
at.profiles	Stores pre-built and player-made profiles
at.temporary	Temporary storage for anything that is not permanent data
at.timer	Holds restore values (last manual time set)



Animated Timer Manual - Version: 240810 - AT Version: 1.0.1

Modification : Custom animations

May need a bit more technical knowledge but is possible nevertheless. Your first step should be to look at the code of function *at-misc:animation/load* - that is the place where pre-built animations are stored. Every animation follows this NBT storage pathing scheme:

at.animations.<animation-name>

An animation consists of two NBT tags: *count* & *colors*. The first one, *count*, is just an integer value containing the number of total colors (steps) per animation. The second, *colors*, is a list of strings containing the individual HEX codes that make up the animation.

The key is to find two colors that match and then generate a gradient between them. I've done this with an external tool: <https://colordesigner.io/gradient-generator>. If you have found a gradient that you are happy with, copy the colors and put them in the *colors* list.

AT does not loop around automatically. Going just from color A to B isn't sufficient - a full animation should look more like this: A > B > A. Best is to try around for yourself until you have your custom gradient.

If you are done, complete the *colors* list, write the final color count into *count*, come up with a great and unique animation name, save the file and reload using `/reload`.

Modification : Integration

Considerations to make when integrating AT in one of your projects:

- What kind of timer and style configuration do I need and how to do I implement it?
- Do I want something to happen if the timer pauses / continues / reaches zero?

Modification : Integration : Configuration

Part of the configuration work for AT can be done via functions in *at-conf*. Although this just works, it can be tedious or inefficient to run the same command twice or in a specific order to get the style you prefer.

Depending on what you are trying to do, it may make sense to look at functions in *at-dcfg* namespace. Any command there just does what it's named like and none of them give any output in form of tellraws.

Animated Timer Manual - Version: 240810 - AT Version: 1.0.1

Modification : Integration : Extraction

The namespace *at-dext* contains four functions that are automatically invoked when a specific action happens within the timer:

Function	Invoked at
at-dext:continue	Timer start or continue via /function at:continue or trigger
at-dext:pause	Timer pause via /function at:pause or trigger
at-dext:reset	Timer reset via /function at:reset or trigger Timer reboot via /function at:reboot or trigger
at-dext:zero	Timer in mode DOWN reaches zero