Cheatography

gganimate Cheat Sheet

by Pranav V A via cheatography.com/126418/cs/24452/

Core Concepts

gganimate builds on ggplot2's grammar of graphics to provide functions for animation. You
add them to plots created with ggplot() the same way you add a geom
Main Function Groups

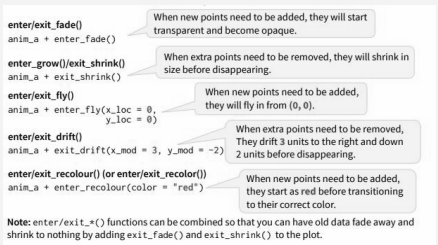
- **`transition_*()`**: What variable controls
change and how?
- **`view_*()`**: Should the axes change with the
data?
- **`enter/exit_*()`**: How does new data get
added the plot? How does old data leave?
- **`shadow_*()`**: Should previous data be "remembered" and shown with current data?
- **`ease_aes()`**: How do you want to handle the
pace of change between transition values?

Note: you only need a `transition_*()` or `view_*()` to make an animation. The other function groups enable you to add features or alter
gganimate's default settings .

Starting Plots

```
library(tidyverse)
library(gganimate)
a <- ggplot(diamonds, aes(carat, price)) + geom_point()
b <- ggplot(txhousing, aes(month, sales)) + geom_col()
c <- ggplot(economics, aes(date, psavert)) + geom_line()
```

enter/exit_*()



Note: `enter/exit_*()` functions can be combined so that you can have old data fade away and

transition_*()

Other transitions

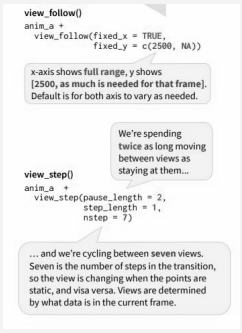
- **`transition_manual()`**: Similar to transition_states(), but without intermediate states.
- **`transition_layers()`**: Add layers (geoms) one at time.
- **`transition_components()`**: Transition elements independently from each other.
- **`transition_events()`**: Each element's duration can be controlled individually.

Baseline Animation

```
anim_a <- a +
transition_states(color,
transition_length = 3,
state_length = 1)
```

Saving animations

view_*()

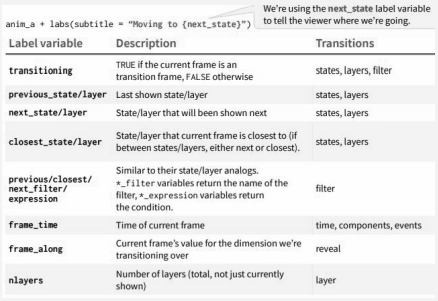


`view_zoom()`

`view_zoom()` works similarly to `view_step()`, except it changes the view by
zooming and panning.

Note: both `view_step()` and `view_zoom()` have `view_*_manual()` versions for setting views
directly instead of inferring it from frame data

Label variables

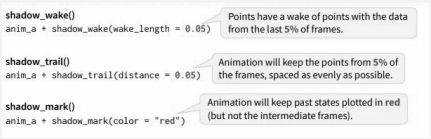


Label variable	Description	Transitions
`transitioning`	TRUE if the current frame is an transition frame, FALSE otherwise	states, layers, filter
`previous_state/layer`	Last shown state/layer	states, layers
`next_state/layer`	State/layer that will been shown next	states, layers
`closest_state/layer`	State/layer that current frame is closest to (if between states/layers, either next or closest).	states, layers
`previous/closest/ next_filter/ expression`	Similar to their state/layer analogs. `*_filter` variables return the name of the filter, `*_expression` variables return the condition.	filter
`frame_time`	Time of current frame	time, components, events
`frame_along`	Current frame's value for the dimension we're transitioning over	reveal
`nlayers`	Number of layers (total, not just currently shown)	layer

gganimate's transition_*() functions create label variables you can pass to (sub)titles and other labels
with the glue package. For example, transition_states() has next_state, which is the name of the
state the animation is transitioning towards. Label variables are different between transitions, and details
are included in the documentation of each.

shrink to nothing by adding `exit_fade()` and `exit_shrink()` to the plot

shadow_*()




```
animation_to_save <- anim_a +
exit_shrink()
anim_save("first_saved_animati-
on.gif", animation = animation-
_to_save)
Since the animation argument
uses your last rendered
animation by default, this also
works:
```

anim_a + exit_shrink()`

```
anim_save("second_saved_animat-
ion.gif")
```

`anim_save()` uses gifski to render the animation as a .gif file by default. You can use the renderer
argument for other output types including video files (av_renderer() or ffmeg_render-er()) or
spritesheets (sprite_renderer()):
requires you to have the av package installed

```
anim_save("third_saved_animati-
on.mp4",renderer = av_render-
er())
```

view_*()

`view_zoom()`

`view_zoom()` works similarly to `view_step()`, except it changes the view by
zooming and panning.

Note: both `view_step()` and `view_-zoom()` have
`view_*_manual()` versions for setting views
directly instead of inferring it from frame data