MICROINTERACTIONS

Why Microinteractions Work

In a nutshell, microinteractions work because they appeal to the user's natural desire for acknowledgement. Microinteractions fine-tune human-centered design by:

- **More control through immediate feedback** — The user instantly knows their action was accepted, giving them more confidence in further usability.
- **Instructions** — Whether blatant or subtle, microinteractions can guide users in how to work the system.
- **Visual rewards** — Small but satisfying effects enhance the UX, and can facilitate a habit loop.
- **Meeting expectations** — In today's web design landscape, microinteractions are the norm — their absence makes a site seem bland.

How to design micro-interactions

Making micro-interactions is exciting for designers, because it is possible to experiment new design solutions and look for new ways to surprise the users. But for doing it you must keep in mind a few things :

- **Put yourself in the users'shoes** and use all that you have to figure out how they use your app.
- **Create functional animations**. Animations which have not only an aesthetic but which are able to enhance the user experience.
- **Have fun and entertain your users.** What the user feels when he uses the app is the reason behind the fact that he keeps using it. If the user enjoys the experience and finds it pleasant, he returns.
- **Do not be annoying.** Too many animations have the opposite effect on users. Annoying users make them stay away from your app.

Best Practices for Microinteractions

- Make it fast—usually within .4 seconds.
- Keep it visually unified throughout the site. (Be consistent with easing type, animation length, and style throughout.)
- Use what's available—Don't add more than necessary: use existing elements to deliver feedback if you can.
- Keep it simple. Remember that the microanimations are the backdrop, not the main show.

Auto-Animate Tutorial

Auto-Animate Requirements

Auto-animate looks at the differences in properties between the same object across artboards, and animates them with the chosen easing.

In order for objects to animate, they must be named the same across both artboards, and be the same *type* of layer. (Read on...)

Examples:

✓ Two rectangles both named “Button” will animate perfectly.

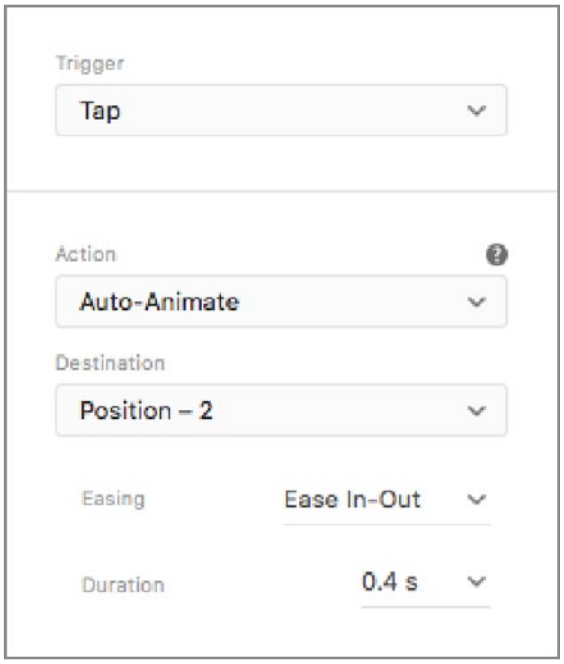
✗ Two rectangles, one named “Button” and the other “Button 2” will not.

✗ A rectangle and an ellipse both named “Morph” will not animate.

✓ A rectangle and an ellipse both named “Morph” and both converted into a path will animate beautifully.

Wiring it up

Once your artboards have been designed (again, making sure that the layer names/types are the same), jump into Prototype mode and drag a wire from the first artboard to the second. After choosing your desired trigger, make sure “Auto-Animate” is selected for the action.



Trigger: Tap

Action: Auto-Animate

Destination: Position – 2

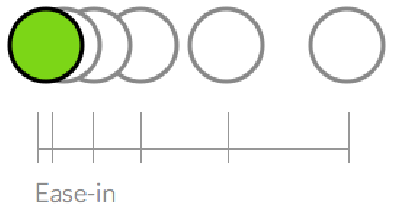
Easing: Ease In-Out

Duration: 0.4 s

Understanding Easing

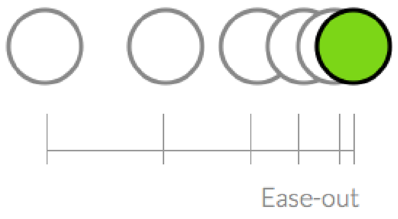
Easing is what keeps motion from feeling stiff and robotic.

Ease-in : when the ball starts out slow and builds up speed, it’s called ease-in

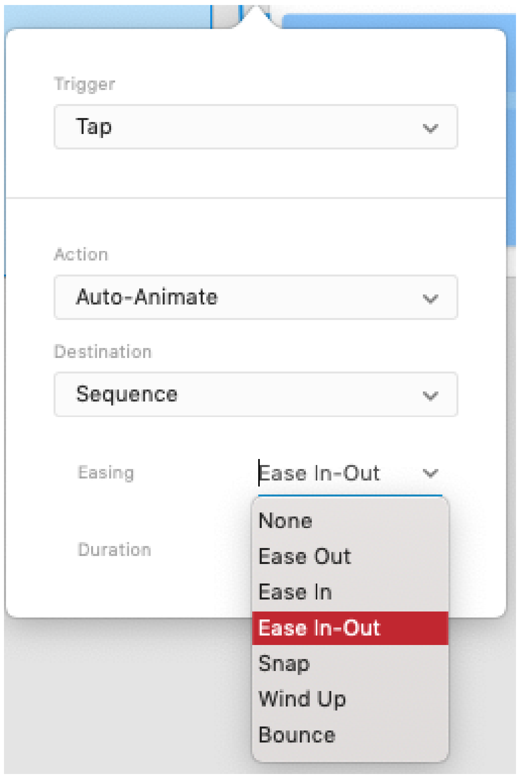


Ease-in: start slow and accelerate

Ease-out : when the ball starts out fast and gradually reduces speed, it’s called ease-out



Ease-out: start fast and decelerate



Trigger: Tap

Action: Auto-Animate

Destination: Sequence

Easing: Ease In-Out

Duration