2.1 What web animation actually is

Before learning **GSAP** or **Glaze**, it is extremely important to understand what **web animation** actually *is*. If you understand this foundation well, then everything later—timelines, easing, scroll animations, stagger effects, interactive motion—will make much more sense. □□

Web animation is not just “making things move.” It is the controlled change of something on a webpage over time.

That sentence sounds simple, but it contains the whole idea:

1. **There is a thing**
   - for example:
     1. a heading
     2. a button
     3. an image
     4. a card
     5. a menu
     6. a section background
2. **There is a property that can change**
   - for example:
     1. position
     2. size
     3. opacity
     4. rotation
     5. color
     6. scale
3. **There is time**
   - the change does not happen instantly, but over a duration such as:
     1. `0.2s`
     2. `0.8s`
     3. `2s`
4. **There is a rule for how the change happens over that time**
   - for example:
     1. linear
     2. ease-in
     3. ease-out
     4. bounce
     5. elastic

So, if a card goes from invisible to visible in `0.6s` with a soft ease-out, that is web animation.

A simple definition

A practical definition is:

> **Web animation is the visual transition of one or more properties of an element over time, usually in response to page load, scrolling, hovering, clicking, or some other event.**

This means animation always involves these building blocks:

1. **Element**
2. **Property**
3. **Start state**
4. **End state**
5. **Duration**
6. **Timing/easing**
7. **Trigger**

Let’s look at each one carefully.

---

1. The element: what is being animated?

An animation always affects some part of the page.

That could be:

1. **A single HTML element**
   - for example:
     1. a button
     2. an image
     3. a heading
2. **A group of elements**
   - for example:

    1. several cards in a grid
    2. menu items
    3. testimonial slides
3. **A pseudo-element**
    - such as decorative parts created with CSS
4. **An SVG element**
    - for example:
        1. paths
        2. circles
        3. icons
        4. logos
5. **Even values not directly visible as “elements”**
    - like scroll position, numerical counters, or CSS custom properties

In your WordPress and BricksBuilder workflow, the “element” will often be something like:

1. a Bricks section
2. a container
3. a block
4. a heading
5. a button
6. an image
7. a class-selected group of elements

So when we animate, we are always asking:

> **“Which thing on the page do I want to change?**

---

2. The property: what exactly changes?

Animation is not movement only. That is a very common misunderstanding.

An animation can change many different kinds of properties.

Common visual properties

1. **Opacity**

    - makes something fade in or out
    - example:
        - `opacity: 0` to `opacity: 1`
2. **Position**
    - moves something horizontally or vertically
    - example:
        - from left to right
        - from lower position to upper position
3. **Scale**
    - makes something grow or shrink
    - example:
        - `scale: 0.8` to `scale: 1`
4. **Rotation**
    - turns an element
    - example:
        - `rotate: 0deg` to `rotate: 180deg`
5. **Size**
    - width, height, padding, etc.
6. **Color**
    - text color, background color, border color
7. **Blur or filter effects**
    - for modern visual transitions
8. **Clip or mask effects**
    - reveal animations, image wipes, text reveals
9. **SVG stroke values**
    - often used for “draw-on” effects
10. **Numerical values**
    - counters, progress bars, percentages

Important concept

Animation is simply:

$$
\text{property at time } t
$$

changing from one value to another.

For example:

$$
\text{opacity}: 0 \to 1
$$

or

$$
y: 50 \to 0
$$

GSAP makes this easy, but the underlying idea exists even without GSAP.

---

3. Start state and end state

Every animation needs at least two states:

1. **Start state**
2. **End state**

Example:

1. Start:
   - the element is invisible and slightly lower
2. End:
   - the element is visible and in its normal position

In CSS-like thinking, that might mean:

- start:
  - `opacity: 0`
  - `transform: translateY(30px)`
- end:
  - `opacity: 1`
  - `transform: translateY(0px)`

This is one of the most important mental models in animation:

> **Animation is the transition between states.**

Without states, there is no animation—only a static design.

Why this matters so much

When beginners say, “My animation doesn’t work,” the problem is often that one of these is unclear:

1. They do not know the start state.
2. They do not know the end state.
3. They did not set the start state properly.
4. The browser cannot interpolate the values the way they expect.

So before you animate anything, ask yourself:

1. Where does it begin?
2. Where does it end?
3. What values are changing?

That single habit will save you a lot of frustration later.

---

4. Time: animation is controlled change across duration

The next core ingredient is **time**.

If a property changes instantly, that is not really perceived as animation. It is just a jump.

Animation becomes animation when the browser shows intermediate values between the start and end over a duration.

For example:

1. At `0s`
   - opacity is `0`
2. At `0.3s`
   - opacity may be around `0.5`
3. At `0.6s`
   - opacity reaches `1`

So the browser (or GSAP) calculates many little in-between states.

This is often called **interpolation**.

Simple idea of interpolation

If you move from:

$$
x = 0
$$

to

$$
x = 100
$$

over 1 second, the browser or animation engine computes the intermediate positions between 0 and 100 over that second.

That means the motion is not magic. It is calculated progression over time.

---

5. Easing: how the animation feels

This is where animation stops feeling mechanical and starts feeling natural.

Two animations can have:

1. the same start
2. the same end
3. the same duration

But still feel completely different because of **easing**.

What easing means

Easing describes **how speed changes during the animation**.

For example:

1. **Linear**
   - constant speed from start to finish
   - feels mechanical
2. **Ease-in**
   - starts slowly, then speeds up
3. **Ease-out**
   - starts quickly, then slows down near the end

4. **Ease-in-out**
   - starts slowly, speeds up, then slows down
5. **Bounce / elastic / back**
   - creates stylized motion

Why easing matters

Real-world objects rarely move at perfectly constant speed.

Think of:

1. a car starting to move
2. a drawer closing
3. a ball bouncing
4. a phone menu sliding in

They accelerate and decelerate.

So easing gives digital movement a more human, physical, or polished feeling.

A design truth

A large part of “professional-looking animation” is not just *what moves*, but *how it moves*.

Beginners often focus on dramatic motion:

- huge slides
- giant zooms
- wild rotations

Professionals often focus more on:

- subtle movement
- controlled timing
- good easing
- restraint

That is one of the biggest mindset shifts in motion design. □□

---

6. Trigger: when does the animation begin?

Animation usually does not happen randomly. It needs a trigger.

A trigger is the event or condition that starts the animation.

Common triggers on the web

1. **Page load**
   - when the page first appears
2. **Scroll**
   - when an element enters the viewport
   - when the page reaches a certain position
   - when animation is linked directly to scroll progress
3. **Hover**
   - when the mouse moves over an element
4. **Click / tap**
   - for menus, accordions, popups, tabs
5. **Focus**
   - important for forms and accessibility
6. **Time delay**
   - animation starts after a short wait
7. **Custom logic**
   - e.g. after AJAX load, after filtering, after a slider changes

In WordPress/BricksBuilder sites, many of your future animations will likely be triggered by:

1. section load
2. scroll into view
3. hover
4. click on a burger menu or button
5. scroll progress across a hero section

So another core question is:

> **What event causes this change to begin?**

---

7. Animation is communication, not decoration

This is one of the most important conceptual lessons.

Many people think animation is just visual decoration. Sometimes it is—but good animation usually has a purpose.

What animation can communicate

1. **Attention**
   - “Look here first.”
2. **Hierarchy**
   - “This item is more important than that one.”
3. **Relationship**
   - “These things belong together.”
4. **Cause and effect**
   - “You clicked this, so that happened.”
5. **Direction**
   - “This panel came from the side.”
   - “This dropdown belongs to this button.”
6. **State change**
   - “This item is now active/open/selected.”
7. **Feedback**
   - “The system received your action.”
8. **Continuity**
   - “The interface is changing, but in a way you can follow.”

Example

Imagine a menu opens instantly with no motion.

- It works.
- But it may feel abrupt.

Now imagine the menu fades and slides down slightly from the header.

- The user understands where it came from.
- The transition feels connected.
- The interaction feels more polished.

That is animation as communication.

---

8. Good animation reduces confusion

When interfaces change suddenly, users can lose track of what happened.

Animation helps bridge that gap.

For example:

1. A modal appears
   - if it simply pops in instantly, it can feel jarring
   - if it fades and scales in subtly, the brain follows the change more easily
2. A card expands
   - animation shows that the larger panel is connected to the smaller card
3. A mobile menu opens
   - motion helps users understand spatial relationship

In this way, animation is not just aesthetic—it improves **usability**.

---

9. But too much animation creates confusion

This is the other side of the coin.

Bad animation can make a website feel:

1. slow
2. chaotic
3. childish
4. distracting
5. hard to use

Common beginner mistakes

1. **Animating too many elements**
   - everything fades, slides, spins, bounces at once
2. **Using too much distance**
   - elements fly in from far away
3. **Using too much duration**
   - animations take too long and slow down the experience
4. **Using inappropriate easing**
   - bouncy effects on serious business websites
5. **Animating for no reason**
   - motion that adds no meaning or clarity
6. **Repeating reveal animations on every scroll**
   - can become annoying quickly

A practical principle

> **The best animation often feels inevitable, not flashy.**

The user should often feel:

- “That felt smooth.”
- not:
- “Wow, look at that crazy effect!”

Of course, for landing pages, portfolios, or creative sites, more expressive animation can be appropriate. But even then, it should be intentional.

---

10. Web animation is constrained by the browser

Unlike motion graphics software, web animation happens in a live environment.

That means it is affected by:

1. browser rendering
2. screen size
3. device performance
4. user input
5. scrolling

6. responsive layouts
7. accessibility preferences

So web animation is not the same as creating a video.

In a video

1. every frame is predetermined
2. playback is controlled
3. viewer interaction is limited

On the web

1. the user may scroll fast
2. the layout may change on mobile
3. images may load late
4. content may vary in height
5. the user may click during animation
6. the device may be slow

This is why web animation requires both **design thinking** and **technical thinking**.

You are not animating on a fixed canvas. You are animating inside a responsive, interactive system.

---

11. The browser “draws” animation frame by frame

At a conceptual level, animation is displayed as a sequence of visual updates.

The browser repeatedly redraws the page, often targeting around:

$$
60 \text{ frames per second}
$$

That means ideally about 60 visual updates each second.

Why this matters

If the browser can update smoothly, the animation feels fluid.

If it cannot, the animation may:

1. stutter
2. lag
3. skip frames
4. feel choppy

This is why performance matters in web animation.

Important beginner takeaway

Not all properties are equally efficient to animate.

Generally, the most animation-friendly properties are:

1. **transform**
2. **opacity**

These usually perform better than animating properties like:

1. width
2. height
3. top
4. left
5. heavy box-shadow changes
6. complex layout-triggering properties

You do not need to master performance yet, but you should already understand this core truth:

> **Animation is not only a visual matter; it is also a rendering and performance matter.**

This becomes especially relevant in WordPress, where pages often include:

1. many plugins
2. large images
3. dynamic content
4. page builder structures

So smooth animation requires some discipline.

---

12. Animation is often the illusion of movement

Here is another foundational insight:

Many times, the element is not “really moving” in the way beginners imagine. Instead, the browser is visually transforming it.

For example, when you animate with `transform: translateY(...)`, the browser is applying a transformation for display. This is often more efficient than changing layout position directly.

That means web animation is often about the **illusion** of movement rather than physically rearranging the document structure at every step.

This distinction becomes very important later when we talk about:

1. transforms
2. layout
3. reflow
4. repaint
5. performance

For now, the key idea is:

> **Good web animation often changes how something is rendered, not how the entire page layout is recalculated every moment.**

---

13. There are different categories of web animation

To understand web animation clearly, it helps to divide it into categories.

A. State-transition animation

This is when something changes from one interface state to another.

Examples:

1. button hover
2. accordion opening
3. modal appearing
4. tab switching
5. menu expanding

Purpose:

- communicate interface changes

B. Entrance/reveal animation

This is when an element enters view or becomes visible.

Examples:

1. hero text fading in on page load
2. cards revealing on scroll
3. image sliding up into view

Purpose:

- direct attention
- create rhythm
- make content feel staged

C. Continuous/decorative animation

This runs continuously or repeatedly.

Examples:

1. floating icons
2. pulsing indicators
3. looping background shapes

Purpose:

- add atmosphere
- bring a static layout to life

Danger:

- can become distracting if overused

D. Scroll-linked animation

Animation is tied to scroll progress.

Examples:

1. parallax
2. pinned sections
3. progress-based image scaling
4. storytelling sections

Purpose:

- create immersive narrative experiences

This is an area where GSAP is especially powerful.

E. Data/value animation

A number or progress indicator changes over time.

Examples:

1. count-up statistics
2. progress circles
3. percentage indicators

Purpose:

- make data feel dynamic and noticeable

F. Spatial/navigation animation

This helps users understand interface movement in space.

Examples:

1. off-canvas menu sliding in
2. page transition
3. card expanding into detail view

Purpose:

- maintain orientation
- preserve context

Understanding these categories helps you choose the *right kind* of motion for the job.

---

14. Animation has emotional tone

Motion changes how a brand feels.

The same content can feel:

1. elegant
2. playful
3. premium
4. technical
5. bold
6. calm
7. energetic
8. luxurious

depending on how it moves.

Examples

1. **Short, subtle fades and slides**
   - often feel clean and professional
2. **Elastic, bouncy movement**
   - often feels playful or youthful
3. **Large cinematic scroll transitions**
   - often feel premium or editorial
4. **Sharp, snappy micro-interactions**
   - often feel modern and tech-oriented

So animation is part of branding.

This matters for web designers and site builders because motion is not separate from design—it is part of the design language.

---

15. Animation should match context

A law firm website, a SaaS product website, a fashion brand site, and a creative portfolio should not all move the same way.

The right animation depends on:

1. brand personality
2. target audience
3. content density
4. user goals
5. performance requirements
6. device context

Example

A serious corporate site might use:

1. subtle fade-ups
2. restrained hover transitions
3. smooth menu movement

A creative agency site might use:

1. larger typography reveals
2. layered parallax
3. complex scroll storytelling
4. image masking effects

Neither is automatically better. The question is whether the motion supports the project.

---

16. Web animation is usually a combination of design and code

At first glance, animation seems purely visual. But on the web, it sits between design and programming.

Design side

You think about:

1. rhythm
2. emphasis
3. timing
4. hierarchy
5. brand feel
6. visual clarity

Code side

You think about:

1. selecting elements
2. setting values
3. defining duration
4. handling triggers
5. sequencing multiple steps
6. ensuring performance
7. making it responsive

This is exactly why tools like **GSAP** are so useful:

- they give you precise programmatic control

And why tools like **Glaze** are attractive:

- they reduce complexity for common animations

So in your learning journey:

1. **GSAP** helps you understand and control animation deeply
2. **Glaze** helps you apply animation quickly and simply in practical page-building workflows

But both still rely on the same underlying principles you are learning here.

---

17. Sequencing: one animation can lead into another

Animation is not always a single isolated effect.

Often, what makes a website feel polished is the sequencing of multiple animations.

Example:

1. section appears
2. heading fades in
3. text follows shortly after
4. buttons appear next
5. image scales in slightly
6. background shape moves subtly

This creates rhythm and structure.

Instead of everything happening at once, motion can guide the eye in a meaningful order.

This is one reason **timelines** are so important in GSAP later on: they let you orchestrate multiple changes as one coordinated sequence.

So web animation is not only:

- “move this thing”

It is also:

- “when should each thing happen relative to the others?”

That is a much more powerful way to think.

---

18. Animation can be user-controlled or system-controlled

Another useful distinction:

System-controlled animation

The website decides when and how the animation runs.

Examples:

1. page load reveal
2. auto-playing decorative loop
3. delayed entrance sequence

User-controlled animation

The user’s action directly controls the animation.

Examples:

1. hover effect
2. drag interaction
3. click-to-open menu
4. scroll-scrubbed animation

This distinction matters because user-controlled motion often needs to feel:

1. immediate
2. responsive
3. intuitive

Whereas system-controlled motion can be more choreographed.

---

19. Animation is not always visible motion

Sometimes the best animation is very subtle.

Examples:

1. a button background color transitioning smoothly

2. a form field border easing into an active state
3. a card shadow changing softly on hover
4. a tiny scale increase on interaction

These are still animations, even though they may not be dramatic.

This is important because beginners often think animation means:

- big motion
- obvious effects
- highly visible transitions

In reality, some of the most effective animation is almost invisible as a separate feature. It just makes the interface feel refined.

---

20. Accessibility matters in animation

Not all users experience motion the same way.

Some people are sensitive to movement, especially:

1. parallax
2. large motion effects
3. aggressive zooming
4. scroll-linked animations

This can cause discomfort or disorientation.

So good web animation should respect accessibility preferences, especially **reduced motion** settings.

Practical principle

If animation is essential, it should:

1. remain understandable
2. avoid causing discomfort
3. degrade gracefully

If animation is decorative, it should ideally be reducible or removable when necessary.

This is not just a technical detail—it is part of responsible design. ♿

---

21. Web animation is about change with intention

If you remember only one conceptual sentence from this whole lesson, let it be this:

> **Web animation is intentional visual change over time that helps communicate, guide, respond, or enhance experience.**

That definition is better than “making stuff move,” because it includes:

1. intention
2. time
3. visual change
4. user experience

That is the real foundation.

---

22. A mental model you can use immediately

Whenever you want to animate something, think through this checklist:

1. **What element am I animating?**
   - a heading, button, card, image, menu, section, SVG?
2. **What property is changing?**
   - opacity, position, scale, rotation, color, height?
3. **What is the start state?**
   - hidden, shifted down, smaller, rotated?
4. **What is the end state?**
   - visible, normal position, full size?
5. **How long should it take?**
   - fast, medium, slow?

6. **What easing fits the feeling?**
   - smooth, snappy, playful, elegant?
7. **What triggers it?**
   - load, scroll, hover, click?
8. **Why does this animation exist?**
   - attention, clarity, feedback, branding, delight?
9. **Is it performant enough?**
   - especially on mobile?
10. **Is it accessible and not excessive?**
    - would it still feel good for real users?

If you begin every animation decision with these questions, you will already think more professionally than many people who just copy random effects from tutorials.

---

23. A few concrete real-world examples

Let’s make all of this more tangible.

Example 1: Fade-in text on scroll

What is happening?

1. Element:
   - heading
2. Property changes:
   - opacity
   - vertical position
3. Start state:
   - invisible
   - slightly lower
4. End state:
   - visible
   - normal position
5. Trigger:
   - heading enters viewport
6. Purpose:
   - draw attention gently as user scrolls

This is one of the most common animations on modern websites.

Example 2: Button hover effect

What is happening?

1. Element:
   - button
2. Property changes:
   - background color
   - scale
   - maybe shadow
3. Start state:
   - default style
4. End state:
   - highlighted style
5. Trigger:
   - hover
6. Purpose:
   - feedback and affordance

This tells the user:

- “This thing is interactive.”

Example 3: Mobile menu opening

What is happening?

1. Element:
   - menu panel
2. Property changes:
   - opacity
   - position
   - maybe clip-path or height
3. Start state:
   - hidden / off-canvas
4. End state:
   - visible / in place
5. Trigger:
   - click on menu icon
6. Purpose:
   - reveal navigation
   - preserve spatial logic

This is more than decoration; it improves orientation.

Example 4: Stats counting up

What is happening?

1. Element:
   - number text
2. Property changes:
   - numeric value
3. Start state:
   - `0`
4. End state:
   - e.g. `250`
5. Trigger:
   - scroll into view
6. Purpose:
   - make data more noticeable and engaging

This shows that animation is not only about visual transforms.

---

24. How this connects to GSAP later

Once you understand what animation actually is, GSAP becomes easier to understand.

Because GSAP is essentially a powerful way to define:

1. **what** changes
2. **from where**
3. **to where**
4. **over how long**
5. **with what easing**
6. **triggered by what**
7. **in what sequence**

For example, a GSAP animation often expresses exactly these ideas in code.

So the concepts come first; the syntax comes second.

This is why learning fundamentals matters so much.

---

25. How this connects to Glaze later

Glaze simplifies many common animation use cases by making it easier to apply motion without writing full custom GSAP logic every time.

But even when using a simplification layer, you still need to understand:

1. what your start state is
2. what your end state is
3. what trigger you want
4. whether the motion makes sense
5. whether it fits the design

So Glaze can reduce coding complexity, but it does not replace animation thinking.

That is why this topic is essential even for no-code or low-code workflows in WordPress and BricksBuilder.

---

26. The most important beginner mindset shift

Here is the mindset shift I would most like you to keep:

> **Do not think of animation as “adding effects.” Think of it as designing change.**

That small shift changes everything.

If you “add effects,” you often get:

1. random motion

2. overuse
3. inconsistent style
4. poor UX

If you “design change,” you start thinking about:

1. states
2. timing
3. meaning
4. interaction
5. clarity
6. consistency

That is the real beginning of mastering web animation.

Summary

Web animation is:

1. the change of an element’s properties over time
2. based on a start state and an end state
3. shaped by duration and easing
4. triggered by events like load, scroll, hover, or click
5. used to communicate, guide, clarify, and enhance experience
6. constrained by browser performance, responsiveness, and accessibility
7. most effective when intentional rather than decorative only

In short:

> **Web animation is the controlled visual behavior of a website over time.**

And that is the foundation for everything that comes next with **CSS transitions**, **GSAP**, **Glaze**, **ScrollTrigger**, timelines, and advanced interactions.

Mini self-check □□

If you want, test your understanding by answering these questions for yourself:

1. What are the 7 core building blocks of an animation?

2. Why is animation more than just movement?
3. What is the difference between start state and end state?
4. What does easing control?
5. Why can animation improve usability?
6. Why can too much animation harm usability?
7. Why are `transform` and `opacity` usually important for performance?
8. What is the difference between decorative animation and functional animation?

Revision #1
Created 2026-04-25 18:41:02 UTC by art10m
Updated 2026-04-25 18:44:25 UTC by art10m