THE COMPLETE FRAMER GUIDE

From Zero to Dance Floor Ready

A 10-Chapter Study Guide to Master Framer

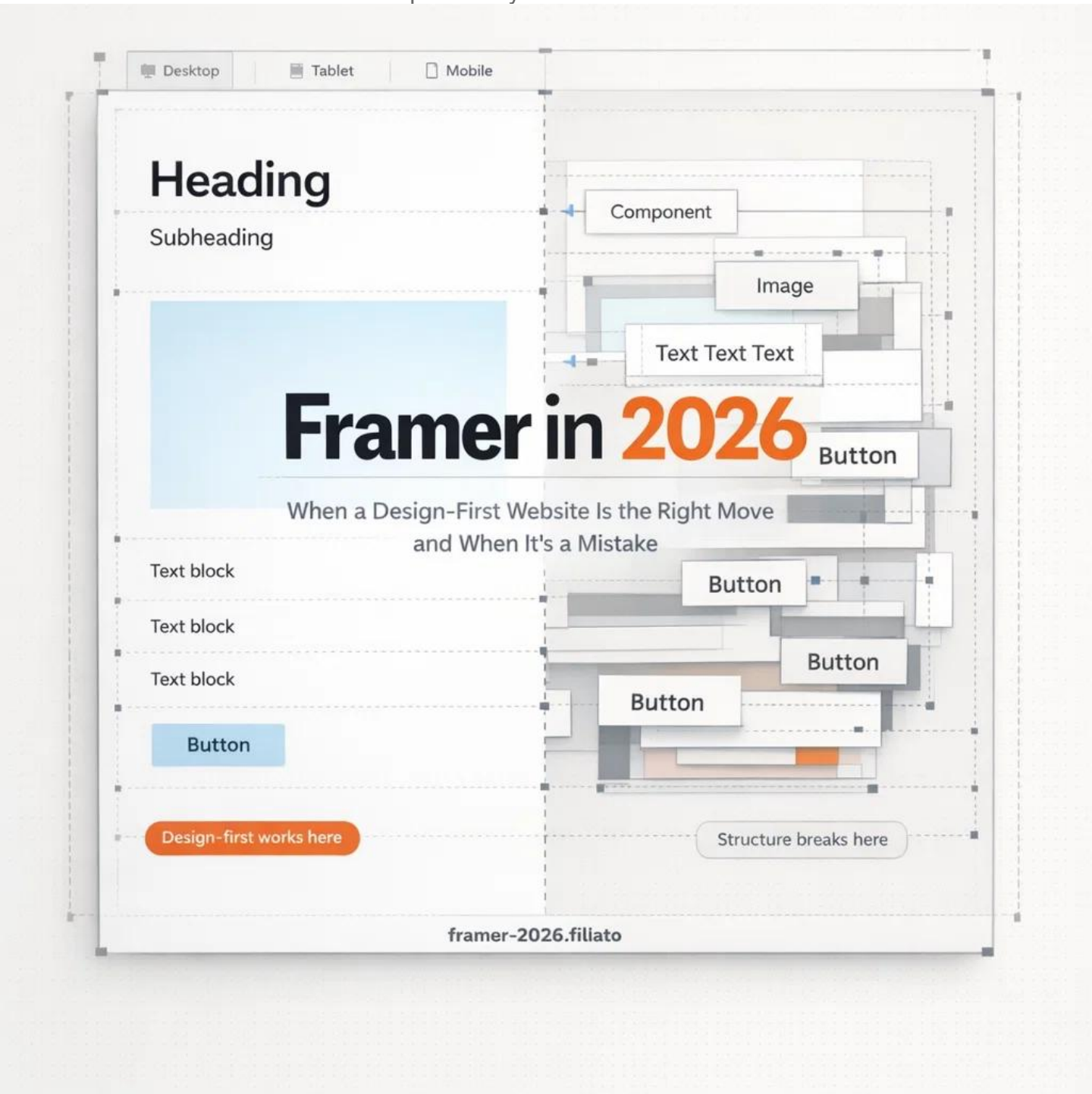

TABLE OF CONTENTS

Introduction: What is Framer?

Framer is a powerful web design and development platform that combines visual design with code, allowing designers and developers to create stunning, interactive websites without compromising on quality or performance. Unlike traditional website builders, Framer bridges the gap between design and development, offering professional-grade tools that produce production-ready code.

Think of Framer as your dance partner in web design—it moves with you, responds to your creativity, and helps you create beautiful, performant experiences. By the end of this guide, you'll be confidently creating websites that not only look amazing but also function flawlessly.

Chapter 1: Understanding Framer's Core Philosophy

1.1 Design Meets Code

Framer's fundamental philosophy is that designers shouldn't be limited by technical constraints, and developers shouldn't be constrained by inflexible tools. It achieves this through:

- Visual Design Tools: A canvas-based interface where you can drag, drop, and design intuitively
- Code Components: The ability to create custom React components when visual tools aren't enough
- Real-time Preview: See your changes instantly as you design
- Production-Ready Output: Websites that load fast and work perfectly across devices

1.2 The Component-Based Architecture

Everything in Framer is a component—from simple buttons to complex navigation systems. This approach provides:

- Reusability: Design once, use everywhere
- Consistency: Maintain design systems across your entire site
- Scalability: Update one component to change it everywhere
- Efficiency: Build faster by leveraging existing components

Example: Imagine designing a button. In traditional tools, you'd recreate it on every page. In Framer, you create it once as a component, then use instances across your site. Change the component's color, and every instance updates automatically.

1.3 Responsive by Default

Framer treats responsive design as a first-class citizen. Every element you create can adapt to different screen sizes through breakpoints, making it easy to ensure your designs look perfect on desktop, tablet, and mobile without starting from scratch.

Chapter 2: Getting Started with Framer

2.1 Setting Up Your Account

Step 1: Navigate to framer.com and click 'Start for Free'
Step 2: Sign up using your email, Google, or other authentication methods
Step 3: Choose your plan (Free plan is perfect for learning)

2.2 Understanding the Framer Interface

The Framer interface consists of four main areas:

The Toolbar (Top)

Located at the top of your screen, the toolbar contains:

- Tools: Frame, Text, Image, Shape tools
- View controls: Zoom, preview, device frames
- Publish button: Deploy your site live

The Canvas (Center)

Your main workspace where you design your pages. Think of it as an infinite canvas where your website comes to life.

The Layers Panel (Left)

Shows the hierarchical structure of your design. Every element you add appears here, organized by parent-child relationships.

The Properties Panel (Right)

Displays properties and controls for the selected element—size, position, styling, effects, and more.

2.3 Creating Your First Project

Exercise: Let's create a simple landing page

1. Click 'New Project' from your dashboard
2. Choose 'Blank' to start from scratch
3. Your canvas appears—this is your design playground
4. Click the Frame tool (or press F) and drag on the canvas to create a frame
5. In the properties panel, set the frame to 1440px wide (desktop size)

Congratulations! You've created your first frame. This frame will become your webpage.

Chapter 3: Mastering Frames and Layout

3.1 What Are Frames?

Frames are the building blocks of Framer. They're containers that hold other elements and define layout behavior. Think of frames as boxes that can contain text, images, other frames, and more.

3.2 Frame Properties

Size and Position

- Width (W) and Height (H): Set exact dimensions or use Auto, Fill, or Fit content
- X and Y: Position on the canvas (relative to parent)
- Rotation: Rotate elements at any angle

Fill and Stroke

- Fill: Background color, gradient, or image
- Stroke: Border color, width, and style
- Radius: Rounded corners (set all or individual corners)

3.3 Layout Systems: Stack, Grid, and Position

Stack Layout

Stack automatically arranges child elements vertically or horizontally with consistent spacing. Perfect for navigation bars, lists, and content sections.

Example: Create a navigation bar:

6. Create a frame (F)
7. Select it and change layout to Stack (horizontal)
8. Add text elements for 'Home', 'About', 'Contact'
9. Set gap to 32px for spacing between items
10. Set padding to 20px for internal spacing

Grid Layout

Grid creates rows and columns for complex layouts. Ideal for image galleries, product grids, and structured content.

Example: Create a 3-column feature section:

11. Create a frame, set layout to Grid
12. Set columns to 3, gap to 24px
13. Add frames inside—they'll automatically arrange in the grid

Position (Absolute)

Position layout gives you precise control, placing elements exactly where you want them. Use for overlapping elements, floating buttons, or custom compositions.

3.4 Constraints and Resizing

Constraints determine how elements behave when their parent resizes—crucial for responsive design.

- Left/Right constraints: Pin elements to edges
- Top/Bottom constraints: Control vertical positioning
- Center constraints: Keep elements centered
- Scale constraints: Elements grow/shrink proportionally

Pro Tip: Set a logo to 'Left + Top' constraints so it stays in the top-left corner when the window resizes.

Chapter 4: Typography and Text Styling

4.1 Adding and Styling Text

Text is the foundation of web communication. Framer gives you complete control over typography.

To add text:

14. Press T or click the Text tool
15. Click on the canvas or drag to create a text box
16. Type your content
17. Use the properties panel to style it

4.2 Text Properties

- Font Family: Choose from Google Fonts or upload custom fonts
- Font Size: Set in pixels (16px is standard body text)
- Font Weight: Light (300), Regular (400), Medium (500), Bold (700)
- Line Height: Controls spacing between lines (1.5 is comfortable for body text)
- Letter Spacing: Adjust spacing between characters
- Color: Text color and opacity
- Alignment: Left, Center, Right, Justify

4.3 Creating Text Styles

Text Styles are reusable typography presets that ensure consistency across your site. Create them once, apply everywhere.

Creating a text style:

18. Style your text exactly how you want it
19. In the properties panel, click the style dropdown
20. Click 'Create Style'
21. Name it (e.g., 'Heading 1', 'Body Text', 'Caption')

Recommended text style hierarchy:

- Heading 1: 48-64px, Bold
- Heading 2: 36-48px, Bold
- Heading 3: 24-32px, Medium
- Body: 16-18px, Regular, line-height 1.5
- Caption: 12-14px, Regular

4.4 Advanced Typography Tips

- Limit fonts to 2-3 per project for cohesion
- Use font pairing: combine a serif heading with sans-serif body, or vice versa
- Create visual hierarchy through size, weight, and color contrasts
- Ensure sufficient contrast (WCAG recommends 4.5:1 for body text)
- Set max-width on body text (600-700px) for optimal readability

Chapter 5: Working with Components

5.1 Component Fundamentals

Components are reusable design elements. When you create a component, you can use it multiple times across your project. Update the main component, and all instances update automatically.

5.2 Creating Your First Component

Example: Create a button component

22. Create a frame (F) and size it 120×40px
23. Give it a blue fill (#0066FF), rounded corners (8px)
24. Add text 'Click Me' centered inside
25. Select the frame, right-click, choose 'Create Component'
26. Name it 'Primary Button'

Now you can drag this component from the Assets panel onto your canvas as many times as needed.

5.3 Component Properties and Variants

Variants allow a single component to have multiple states or styles. Think of a button with different colors or sizes.

Creating variants:

27. Select your component on the canvas
28. In the properties panel, click 'Add Variant'
29. Customize the new variant (e.g., change the fill to red for a 'danger' button)
30. Add a property to control which variant shows (e.g., 'Type' with options 'Primary' and 'Danger')

Now when you use the component, you can select which variant appears through the properties panel.

5.4 Overrides

Overrides let you customize individual instances of a component without breaking the connection to the main component.

Example: You have a card component. On one page, you want the card to have different text but maintain the same styling and structure.

- Place the component instance on your canvas
- Select it and change the text in the properties panel
- The text changes, but it remains connected to the main component's structure

5.5 Component Best Practices

- Create components for anything used more than once (buttons, cards, headers, footers)
- Name components clearly and consistently
- Organize components in the Assets panel with logical naming (e.g., 'Button/Primary', 'Button/Secondary')
- Use variants instead of creating separate components for similar elements

Chapter 6: Interactions and Animations

6.1 Understanding Interactions

Interactions make your website come alive. They respond to user actions—clicks, hovers, scrolls—and create engaging experiences.

6.2 Adding Basic Interactions

Example: Create a hover effect on a button

31. Select your button component
32. In the properties panel, find 'While Hovering' under Interactions
33. Click the '+' icon to add a hover state
34. Change the fill color to a lighter shade (e.g., #3385FF)
35. Adjust the transition settings (spring, duration)

Now when you preview, the button smoothly changes color when you hover over it!

6.3 Types of Interactions

- While Hovering: Triggers when the mouse is over an element
- While Pressing: Triggers when clicking/pressing an element
- On Tap: Triggers on click (can link to pages, URLs, or custom actions)
- While In View: Triggers when an element enters the viewport (scroll animations)
- While Scrolling: Creates scroll-based animations

6.4 Creating Page Transitions

Example: Navigate from one page to another

36. Select an element (like a button or link)
37. Add an 'On Tap' interaction
38. Choose 'Navigate to' and select your target page
39. Choose a transition (Instant, Push, Fade, Modal, etc.)

6.5 Scroll Animations

Scroll animations create dynamic experiences as users scroll down the page.

Example: Fade in elements as they appear

40. Select an element (like a card or image)
41. Set initial opacity to 0
42. Add 'While In View' interaction
43. Change opacity to 1 (fully visible)
44. Add a smooth spring transition

6.6 Animation Best Practices

- Keep animations subtle—they should enhance, not distract
- Use spring transitions (they feel natural and responsive)
- Typical durations: 0.3-0.5 seconds for micro-interactions, 0.6-1 second for larger animations
- Test on actual devices—animations can perform differently
- Respect user preferences (consider prefers-reduced-motion accessibility setting)

Chapter 7: Responsive Design with Breakpoints

7.1 Why Responsive Design Matters

Over 60% of web traffic comes from mobile devices. Responsive design ensures your website looks and functions perfectly on every screen size—from smartphones to ultrawide monitors.

7.2 Understanding Breakpoints

Breakpoints are specific screen widths where your design adapts. Framer provides three default breakpoints:

- Desktop: 1200px and above
- Tablet: 810px to 1199px
- Mobile: 0px to 809px

7.3 Designing for Different Screens

Example: Make a three-column grid responsive

Desktop (starting point):

- Create a grid frame with 3 columns
- Add cards to fill the grid

Tablet:

- Click the breakpoint selector (top toolbar)
- Select 'Tablet'
- Change grid to 2 columns
- Adjust spacing if needed

Mobile:

- Select 'Mobile' breakpoint
- Change grid to 1 column (stack)
- Increase card width to fill screen

Framer automatically handles the transitions between breakpoints!

7.4 Mobile-First vs Desktop-First

Mobile-First Approach (Recommended):

- Design for mobile screens first
- Then add complexity for larger screens
- Ensures core content works on small screens
- Faster load times on mobile devices

Desktop-First Approach:

- Design for desktop, then simplify for mobile
- Useful when desktop is the primary use case
- Can lead to bloated mobile experiences if not careful

7.5 Responsive Design Tips

- Use relative units (%, fr in grids) instead of fixed pixels where appropriate
- Hide/show elements at different breakpoints (use visibility settings)
- Test on real devices, not just the canvas preview
- Prioritize touch-friendly tap targets on mobile (44×44px minimum)
- Stack navigation horizontally on desktop, vertically (hamburger menu) on mobile

Chapter 8: Code Components and Advanced Features

8.1 When to Use Code Components

While Framer's visual tools handle most design needs, code components unlock unlimited possibilities. Use them when you need:

- Custom logic and interactivity
- API integrations (fetch data from external sources)
- Complex calculations or data manipulation
- Third-party library integrations
- Advanced animations beyond what interactions provide

8.2 Creating a Simple Code Component

Example: A counter button

45. Click the '+' icon in the toolbar
46. Select 'Code Component'
47. Name it 'Counter'
48. Write your React code

Sample code:

```
import { useState } from 'react'  export default function Counter() {   const [count, setCount] = useState(0)      return (     <div       onClick={() => setCount(count + 1)}      style={{         background: '#0066FF',         color: 'white',         padding: '16px 32px',         borderRadius: '8px',         cursor: 'pointer',         fontSize: '18px',         fontWeight: 'bold'       }}     >       Clicked {count} times     </div>   ) }
```

This creates a clickable button that counts how many times it's been clicked!

8.3 Adding Properties to Code Components

Make your code components flexible by adding customizable properties.

Example: Add a color property

```
import { addPropertyControls, ControlType } from 'framer'  export default function Counter({ color }) {   // component code...   return <div style={{ background: color }}>...</div> }  addPropertyControls(Counter, {   color: {     type: ControlType.Color,     defaultValue: '#0066FF'   } })
```

Now you can change the button color from the properties panel without editing code!

8.4 Using External Libraries

Import npm packages to extend functionality.

Example: Using date-fns for date formatting

```
import { format } from 'date-fns'  export default function CurrentDate() {   const today = format(new Date(), 'MMMM dd, yyyy')   return <div>{today}</div> }
```

8.5 CMS Integration

Framer integrates with headless CMS platforms, allowing you to manage content externally and pull it into your site dynamically. Popular integrations include:

- Framer CMS (built-in)
- Contentful
- Sanity
- Airtable
- Notion

Pro Tip: Use Framer CMS for simple blogs and content. For complex needs, integrate a dedicated headless CMS.

Chapter 9: Publishing and Performance Optimization

9.1 Publishing Your Site

Once your design is ready, publishing is simple:

49. Click the 'Publish' button in the top-right
50. Choose a site name (yoursite.framer.website)
51. Configure SEO settings (title, description, social image)
52. Click 'Publish'

Your site is live instantly! Updates publish in seconds with a single click.

9.2 Custom Domains

Connect your own domain (e.g., www.yourbrand.com):

53. Go to Site Settings > Domain
54. Click 'Add Custom Domain'
55. Enter your domain and follow DNS configuration instructions
56. Wait for DNS propagation (usually 24-48 hours)

Framer automatically provisions SSL certificates for secure HTTPS connections.

9.3 Performance Optimization

Image Optimization

- Use WebP format when possible (smaller file sizes)
- Compress images before uploading (TinyPNG, ImageOptim)
- Set appropriate image sizes—don't upload 4K images for thumbnails
- Enable lazy loading for images below the fold

Code and Assets

- Minimize custom fonts (2-3 weights maximum)
- Remove unused components from your Assets panel
- Limit heavy animations on mobile devices
- Use system fonts as fallbacks

Testing Performance

Use these tools to audit your site:

- Google PageSpeed Insights
- Lighthouse (built into Chrome DevTools)
- GTmetrix

Aim for scores of 90+ on Performance, Accessibility, Best Practices, and SEO.

9.4 SEO Best Practices

- Set unique page titles and meta descriptions for each page
- Use semantic heading hierarchy (H1 > H2 > H3)
- Add alt text to all images
- Create a sitemap (Framer generates this automatically)
- Use descriptive URLs (/about-us not /page-2)
- Add Open Graph and Twitter Card metadata for social sharing

Chapter 10: Tips, Tricks, and Pro Workflows

10.1 Keyboard Shortcuts

Master these shortcuts to dramatically speed up your workflow:

Shortcut	Action
F	Frame tool
T	Text tool
R	Rectangle tool
O	Oval tool
Cmd/Ctrl + D	Duplicate
Cmd/Ctrl + G	Group
Cmd/Ctrl + K	Create component
Cmd/Ctrl + E	Export
Cmd/Ctrl + /	Search for anything
Z	Zoom tool
Space + Drag	Pan canvas
Cmd/Ctrl + 0	Zoom to fit
Cmd/Ctrl + 1	Zoom to 100%

10.2 Design System Workflow

Building a design system from the start saves time and ensures consistency:

57. Define your color palette (primary, secondary, neutrals, feedback colors)
58. Create text styles for all typography levels
59. Build core components (buttons, inputs, cards, modals)
60. Document spacing standards (8px grid system recommended)
61. Create reusable sections (headers, footers, CTAs)

10.3 Collaboration Features

- Invite team members (Site Settings > Team)
- Use comments to provide feedback directly on the canvas
- Version history lets you restore previous versions
- Real-time collaboration—see others' cursors as they design
- Share prototypes with clients via password-protected links

10.4 Framer Community and Resources

- Framer Marketplace: Download templates, components, and plugins
- Framer Community: Forum for questions and inspiration
- YouTube: Official tutorials and community creators
- Twitter/X: Follow @framer for updates and tips
- Documentation: https://www.framer.com/docs/

10.5 Common Mistakes to Avoid

- Not using components—leading to inconsistency and extra work
- Ignoring responsive design until the end
- Over-animating—keep it subtle and purposeful
- Not optimizing images—leading to slow load times
- Forgetting to test on actual devices
- Skipping SEO setup before publishing

10.6 Pro Tips for Efficiency

- Use Auto Layout (Stack) wherever possible—it makes responsive design effortless
- Name your layers descriptively—'Hero Section' not 'Frame 42'
- Keep your layers panel organized with groups
- Use the Search (Cmd/Ctrl + /) to quickly find anything
- Save commonly used color values in your color palette
- Work with a grid overlay for precise alignment
- Duplicate successful layouts and modify rather than starting from scratch

10.7 Your Next Steps

Congratulations on completing this comprehensive Framer guide! You now have the foundation to create stunning, professional websites. Here's how to continue your journey:

62. Build a personal portfolio to practice everything you've learned
63. Clone and study templates from the Framer Marketplace
64. Join the Framer Community and share your work
65. Take on a real client project to solidify your skills
66. Keep experimenting—the best way to learn is by doing

Conclusion: You're Ready for the Dance Floor

You've journeyed through the complete Framer ecosystem—from understanding its philosophy to mastering responsive design, creating components, adding animations, writing code, and publishing professional websites. You've learned not just what to do, but why and when to do it.

Framer is more than a tool; it's your creative partner in bringing web experiences to life. The skills you've gained here will serve you whether you're designing landing pages, complex web applications, or anything in between.

Remember: every great designer started exactly where you are now. What separates good from great is practice, experimentation, and a willingness to keep learning.

Now go forth and create something amazing. The dance floor is yours.