Master Design Tokens

2025 • Workshop Workbook

DOOR3

Table of Contents:

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

2

Module 1: Workbook Overview

1.1 Introduction

Using design tokens in design systems streamlines the software development process by enabling teams to create and maintain a direct design-to-development pipeline. This facilitates team collaboration by ensuring consistency and efficiency, even across large organizations and platforms.

UX designers can now create, manage, and implement design decisions directly to the development workstream using tools like Figma and Tokens Studio. The design experts at DOOR3 have developed and successfully used this method in several projects, and we feel other designers would find it valuable in your own practice. We've worked hard to distill our method into this comprehensive guide.

1.2 Overview of Module Content

In this workshop, you can expect to cover a range of topics, organized into modules.

- **Design System Basics**
  - Outlines how to prepare a design system so you can add tokens easily.
- **Design Token Basics**
  - An introduction to tokens and their benefits that should familiarize you with the main concepts and techniques you'll need to build your design-to-dev pipeline.
- **Token Naming Conventions**
  - Establishing comprehensive methods to name and organize your tokens is essential to successful implementation. We will show you how to do it properly.
- **Building Your Token Structure**
  - Tokens work best within a set hierarchy that acts as a source of truth for teams. In this module, we discuss how to build a strong foundation.
- **Introduction To Tokens Studio**

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

3

- ○ Explore how to create and manage design tokens within Tokens Studio, including setting up tokens, organizing them into sets, and creating themes that can be easily switched based on different requirements.
- **Applying Tokens To Components**
  - ○ An overview of the Figma controls used to designate how your token sets should function when acting on a component.
- **Exporting Tokens Into Figma**
  - ○ Once your tokens are established in Tokens Studio, the next step is to export them into Figma variables. This integration ensures that your design tokens are not just theoretical concepts but are actively used in your design files, making updates and iterations a seamless process.
- **Syncing Designs to Development**
  - ○ Finally, we'll discuss how to effectively communicate your design tokens to developers, ensuring that your design system is faithfully translated into code. We'll cover best practices for collaboration, tools that facilitate this process, and how to maintain alignment between your design and development teams as your product evolves.
- **Design Token Governance**
  - ○ The effectiveness of design tokens depends on how well they are managed or governed. In this module, we provide clear guidelines, processes, and roles for how design tokens are created, maintained, and used.

By the end of this series, you'll have a comprehensive understanding of how to create a fully integrated design-to-development pipeline using Figma and Tokens Studio. This pipeline will not only enhance collaboration between designers and developers but also increase the efficiency and consistency of your entire software development process.

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

4

Module 2: Design System Overview

3.1 Introduction

Design systems are the backbone of modern product design, ensuring consistency by standardizing guidelines and reusable components. Figma, a powerful tool suite, has become a favorite among designers for creating and managing these systems. In this comprehensive guide, we'll delve into how to set up your design system in Figma, covering everything from the basics of setting up your component library to the latest advanced features available in Figma.

3.2 Understanding the Importance of a Design System

Before diving into the setup process, it's crucial to understand why a design system is essential for a successful UX practice. A well-constructed design system:

- **Ensures Consistency:** Across different platforms, products, and teams, a design system ensures that all UI components adhere to a unified design language.

- **Enhances Collaboration:** With a shared library of assets and components, teams can collaborate more effectively, reducing the risk of inconsistencies.

- **Increases Efficiency:** Designers and developers can work faster by reusing components and tokens, rather than creating new ones from scratch each time.

- **Facilitates Scalability:** As your product or brand grows, a design system allows you to scale your design language across new touchpoints without reinventing the wheel.

Figma provides a robust environment to build, manage, and scale your design system. Later, we'll go into detail about how to create tokens using the plugin Tokens Studio for Figma. For right now, let's explore how to set up a design system in Figma step by step.

3.3 Setting Up Typography and Color Systems

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

5

Typography and color are two of the most critical aspects of any design system. They define the visual identity and brand consistency across all digital products.

Typography in Figma Typography should be systematized in a way that covers all use cases, from large headings to small captions.

- **Define Text Styles:** Start by defining text styles for headings, subheadings, body text, captions, and any other text elements. Assign appropriate font sizes, weights, line heights, and letter spacing to each style.

- **Create a Hierarchy:** Establish a clear typographic hierarchy that guides users through your content. For example, `Heading-1` should be the largest and boldest text style, while `Body-Text` should be smaller and more readable for longer content blocks.

- **Use Consistent Naming Conventions:** Name your text styles clearly (e.g., `H1-Primary`, `H2-Secondary`, `Body-Text`) so that they can be easily identified and applied.

Color System in Figma A well-defined color system is essential for brand consistency and accessibility.

- **Create a Color Palette:** Start by defining your primary, secondary, and accent colors, as well as neutral shades like grays and whites. Ensure that these colors align with your brand's visual identity.

- **Set Up Color Styles:** In Figma, create color styles for each color in your palette. These can be named descriptively (e.g., `Primary-Blue`, `Secondary-Green`, `Neutral-Gray`) and applied across your components.

- **Consider Accessibility:** Ensure that your color palette meets accessibility standards, particularly in terms of contrast ratios. Figma provides plugins like Stark that can help you check color contrast and ensure your designs are accessible to all users.

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

6

3.4 Creating and Managing Icons

Icons are a crucial part of any design system, providing visual cues that enhance usability and communication.

Building an Icon Library Create a consistent set of icons that align with your brand's visual language. Each icon should be designed with similar stroke widths, corner radii, and proportions to maintain a cohesive look. You can also leverage ready-to-go icon libraries like Font Awesome to get started quickly.

- **Designing Icons:** Use Figma's vector tools to design icons directly within the platform. Maintain consistency by setting rules for grid sizes, padding, and stroke width.

- **Organizing Icons:** Group icons by categories (e.g., actions, navigation, social media) and name them clearly. This makes it easy for designers to find and use the right icon for their needs.

Converting Icons into Components Once your icons are designed, convert them into components so they can be reused across your designs.

- **Create Components:** Select the icon and use `Create Component` to make it reusable. Organize these components in your library for easy access.

- **Using Variants for Icons:** If your icons have different states or versions (e.g., filled vs. outlined), use Figma's Variants feature to group them under a single icon component. This keeps your library tidy and easy to navigate.

3.5 Leveraging Figma's Advanced Component and Token Features

Figma offers several [advanced features](#) that are essential for UX designers working with complex design systems. These advanced features work alongside tokens in order to promote consistency, maintain design standards, and streamline processes.

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

7

One of the key functionalities is component properties, which allow you to define and control the changeable aspects of a component. By tying specific design properties to components, you can dictate what other team members can modify, ensuring consistent usage and reducing errors. These properties are managed through a consolidated set of controls in the Design tab, streamlining the process of editing instances without diving into individual layers.

Figma provides various types of component properties:

- **Boolean Properties**: These control layer visibility, enabling true/false toggles for specific attributes. For instance, you can apply a boolean property to toggle an icon's visibility within a button component, reducing the need for multiple variants and simplifying the design system.
- **Instance Swap Properties**: These allow you to specify which instances within a component can be swapped, providing flexibility while keeping the design system manageable. Designers can select from predefined instances, streamlining the process of adjusting components like icon buttons without creating numerous variants.
- **Text Properties**: Text properties make it easy to edit text strings within a component while preserving the component's structure. This ensures that updates are consistent across the design, although rich text features must still be managed separately.

Most importantly, Figma also offers the ability to **expose nested instances** within components, making it easier to adjust deeply nested elements from the top level without extensive layer navigation. These salient features, combined with the component playground in Dev Mode, provide a robust framework for managing and scaling design systems effectively.

3.6 Scaling Your Design System

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

8

As your product grows, so should your design system. Scaling involves adding new components, adapting to different platforms, and ensuring consistency across all touchpoints. Think of it as a living document, which when set up correctly reduces maintenance costs and improves the sustainability to your design decisions. As we'll discuss later, tokens are particularly valuable in helping to scale efficiently and consistently.

Expanding Your Component Library As new features are added to your product, new components may be needed as well. Follow the same principles of consistency, reusability, and scalability when adding these components to your library.

- **Continuous Integration:** Regularly update your component library with new designs and improvements. Ensure that all team members are using the latest versions of components.

- **Cross-Platform Consistency:** If your product is available on multiple platforms (e.g., web, iOS, Android), ensure that your design system covers all necessary variations and adheres to platform-specific guidelines.

Maintaining and Evolving the System A design system is never truly "finished"; it evolves as your products and brand grow.

- **Governance:** Establish governance rules for how new components are added, how updates are made, and who is responsible for maintaining the system. This ensures consistency and quality as the system evolves.

- **Feedback and Iteration:** Encourage feedback from all users of the design system, from designers to developers. Use this feedback to make continuous improvements to the system.

3.7 Importance of a Strong Design System

Setting up a robust design system in Figma is the foundation for creating consistent, scalable, and efficient digital products. As we've discussed, components are the building

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

9

http://www.door3.com/

blocks that ensure your visual language is consistently applied across all products and platforms. However, establishing the initial design system is just the beginning.

The real value of a design system lies in its ability to evolve and grow alongside your product. As your brand expands and your digital presence becomes more complex, your design system must be able to adapt without compromising on consistency or quality. But what happens once your design system is set up?

How do you take this system and effectively bridge the gap between design and development? This is where the true potential of a design system comes to life — transforming static design files into dynamic, coded components that developers can use to build the final product.

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

10

http://www.door3.com/

Module 3: Design Token Overview

2.1 What Are Design Tokens?

Design tokens are text-based symbols that store visual design attributes such as colors, font sizes, spacing values, and more. These tokens act as the building blocks of your design system, ensuring consistency across different platforms and devices. By using design tokens, you can create a single source of truth for your design language, which can then be easily implemented by developers. For example:

- **Colors:** `Primary-Color`, `Secondary-Color`, `Background-Color`

- **Typography:** `Heading-Font`, `Body-Font`, `Button-Font`

- **Spacing:** `Padding-10px`, `Margin-5px`

Reusing these tokens across various components ensures consistency throughout the design. When set up correctly, they allow for easy updates—changing a token's value automatically updates all instances of its use across your designs.

## 2.2 Understanding Design Tokens

Design tokens are essential components of modern design systems, ensuring consistency and scalability across various digital products. They encode the core elements that define a design system's visual language, including colors, typography, spacing, and more. Creating and managing design tokens effectively can make a significant difference in the quality and efficiency of your design process.

Before diving into the tools and processes, it's important to understand what design tokens are and why they're so important.

Design tokens are platform-agnostic name:value pairs that store design information and can alias to other name:value pairs. These tokens act as the building blocks of your design

system, ensuring that design decisions are consistently applied across different platforms and devices. By using design tokens, you can create a single source of truth for your design language, which can then be easily implemented by developers.

Figma Variables are Figma's feature used to store and manage tokens for your design system. Variables can be applied to components directly, or be used as the underlying framework to build your styles. Once you create your tokens, you can export them to Figma and sync any changes as you continue developing them.
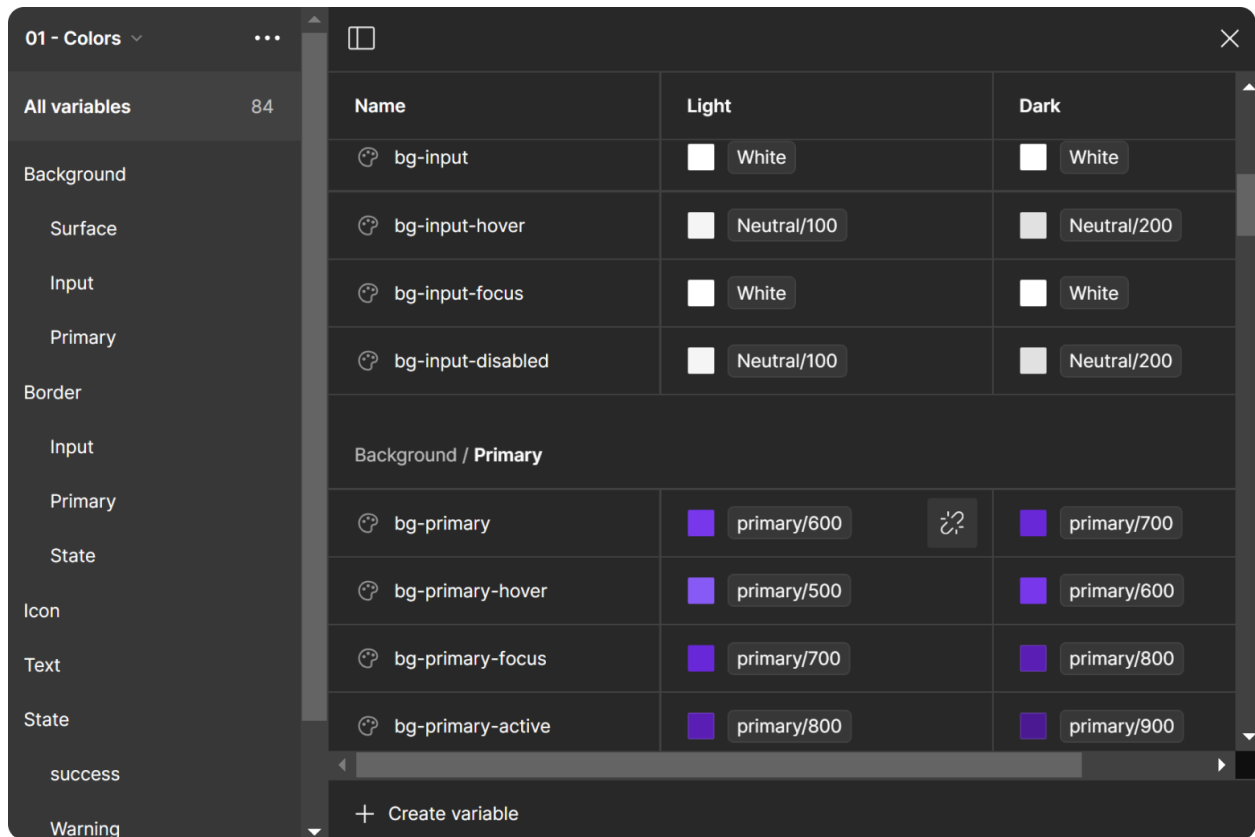
2.3 Why Design Tokens Matter

1. Consistency: Design tokens ensure that your brand's visual identity is applied consistently across all touchpoints.

2. Scalability: As your product or brand grows, design tokens allow you to scale your design language without starting from scratch.

3. Efficiency: By reusing tokens, designers and developers can work faster and more effectively, reducing the need for repetitive work.

4. Flexibility: Design tokens make it easy to update and maintain your design system. Change a token's value, and all instances where it's used will automatically update.
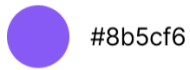
Hex Codes	Styles	Variables	Tokens
#8b5cf6	Primary Default	Violet 500	Violet 500
		↓	↓
		Primary Default	Primary Default
			↓
			--mode-content-primary-normal

Module 4: Design Token Naming Conventions

4.1 Introduction

Design tokens are an essential part of modern design systems, ensuring consistency and scalability across digital products by bridging the gap between design and development. However, many UX designers looking to apply design tokens in their own work find the process challenging to implement, especially across complex platforms. Establishing foundational characteristics like effective naming sets the stage for their successful application within your design-to-dev pipeline.

Establishing effective naming conventions might seem like a straightforward task, but it requires careful consideration and a strategic approach. The names you choose will impact how easily your team can understand, search, and apply these tokens across your design and development processes. In this module, we'll explore the principles and practices that can help you create a robust naming convention for your design tokens.

4.2 Why Naming Design Tokens Matters

Before diving into the specifics of naming conventions, it's important to understand why the naming of design tokens is so crucial. Design tokens are more than just styles; they are the language that designers and developers use to communicate visual decisions across platforms.

Clear and consistent naming of design tokens improves understanding by helping team members quickly grasp their purpose and usage, reducing cognitive load and minimizing errors. It also facilitates collaboration among interdisciplinary teams—such as designers, developers, and product managers—by establishing a shared language that reduces miscommunication. As the design system grows, a logical naming structure ensures scalability, preventing the system from becoming confusing or difficult to manage. Additionally, consistent naming conventions help maintain visual coherence across

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

14

http://www.door3.com/

components, platforms, and products, which is essential for large organizations with multiple teams working within the same design system.

4.3 Principles for Naming Design Tokens

When naming design tokens, there are several key principles to make your naming conventions practical and scalable.

1. Be Descriptive but Concise

Token names should clearly describe the attribute they represent, but also be concise, easily identified, and usable. Striking a balance between clarity and brevity is essential.

For example:

- `$color-primary-blue-dark`

If the context makes it clear that it's a shade of blue, you might use:

- `$color-primary-dark`

However, if the design system includes multiple primary colors, the color could be:

- `$color-primary-blue-dark`

2. Use a Consistent Structure

A consistent naming structure makes it easier to organize and locate tokens within your system. This structure often includes a combination of categories, properties, and modifiers.

For example, a common structure might be `[category]-[property]-[modifier]`. This would be represented as:

- `$color-background-hover`

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

15

or

- `$font-size-heading-large`.

3. Avoid Ambiguity

Ambiguous names can lead to confusion and misuse of tokens. Ensure that each name clearly distinguishes one token from another, particularly when they serve similar but distinct purposes.

One approach you could take is to avoid generic names without context, such as:

- `$color-secondary`

Instead, specify the use, such as:

- `$color-secondary-button`

4. Consider Future Scalability

As your design system evolves, you'll likely add new tokens. Choose names that can accommodate growth without requiring a complete overhaul of your naming convention.

For example, if you anticipate adding new shades of a color, start with a structure, such as:

- `$color-accent-1`, `$color-accent-2`

This will allow you to easily add `$color-accent-3` later.

5. Maintain a Shared Vocabulary

Ensure that your team has a shared understanding of the terms used in your token names. This might involve creating a glossary or style guide that defines key terms and their appropriate usage.

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

16

For example, define what "primary," "secondary," "tertiary," "hover," and other commonly used terms mean within the context of your design system.

4.4 Breaking Down the Components of Token Names

To create a robust naming convention, break down the components of a token name into specific levels. These levels can include categories, properties, concepts, and modifiers, each serving a distinct purpose.

1. Category

The category is the broadest level and typically represents a fundamental aspect of the design, such as color, typography, spacing, or elevation. For example:

- `$color`, `$font`, `$space`, `$shadow`.

2. Property

The property describes the specific attribute within the category. For color tokens, properties might include background, text, or border. For typography, properties might include size, weight, or line-height. For example:

- `$color-background`, `$font-size`, `$space-padding`.

3. Concept

Concepts group tokens within a category by their purpose or usage. For instance, color tokens might be grouped into concepts like feedback, action, or status. For example:

- `$color-feedback-error`, `$color-action-primary`, `$color-status-success`.

4. Modifier

Modifiers refine the token by specifying variants such as states (hover, active, disabled), scale (small, medium, large), or modes (light, dark). For example:

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

17

- `$color-background-hover`, `$font-size-heading-large`, `$color-action-primary-on-dark`.

4.5 Practical Examples of Token Naming

To illustrate how these principles and components come together, let's look at some practical examples of naming conventions for different types of tokens.

Color Tokens

Color tokens are one of the most common types in any design system. Here's how you might structure their names:

- **Primary Button Background Color:**
    - `$color-button-primary-background`
- **Error State Text Color:**
    - `$color-feedback-error-text`
- **Hover State for Link:**
    - `$color-link-hover`

Typography Tokens

Typography tokens manage text styles across your design system. Naming conventions here might include:

- **Heading 1 Font Size:**
    - `$font-size-heading-1`
- **Body Text Line Height:**
    - `$font-line-height-body`
- **Caption Font Weight:**
    - `$font-weight-caption`

Spacing Tokens

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

18

Spacing tokens ensure consistent margins, padding, and gaps across your designs. Examples might include:

- **Small Padding:**
  - `$space-padding-small`
- **Medium Margin:**
  - `$space-margin-medium`
- **Large Gap Between Components:**
  - `$space-gap-large`

4.6 Addressing Challenges in Token Naming

While the principles and structures outlined above provide a strong foundation, naming tokens isn't always straightforward. Here are some common challenges and how to address them.

Handling Multiple Themes

If your design system supports multiple themes (e.g., light and dark modes, brand variants), you'll need to account for these in your naming convention. One approach is to prepend or append theme identifiers to the token names. For example:

- `$color-primary-background-light`
- `$color-primary-background-dark`

Managing Complex Components

For complex components that involve multiple nested elements, naming can become unwieldy. In these cases, consider breaking down the component into its parts and naming tokens accordingly. For example:

- `$component-button-primary-background`
- `$component-button-primary-text`

Evolving Naming Conventions

As your design system matures, you may find that your initial naming convention no longer meets your needs. It's important to allow for flexibility and evolve your naming strategy as necessary, but do so with care to avoid disrupting your team's workflow. For example, when updating a naming convention, ensure that all instances of the old token names are updated consistently across your system.

Naming Hierarchies

As your product and brand evolve, your components will too, and extending your hierarchy to the level of semantic or component tokens gives your token system greater depth and specificity. Semantic tokens recodify core values like color with values based on form and usage. For example, while a core token could signify a certain shade of red, a semantic token would signify an error state or cancel action. Component tokens take semantic naming a step further to indicate characteristics of a particular component like a cancel button.

With more component level specificity, you have more granular control over the appearance of your design and a higher degree of integration between design and development, but at the cost of creating a potentially much larger token set. Our recommendation is to start simple and work at the semantic level, only adding component-level tokens when the need arises.

Semantic examples:

- `surface-primary`
- `border`
- `borderRadius-medium`

Component examples:

- `card-surface-primary`

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

20

- `inputField-border`
- `button-borderRadius`

4.7 Best Practices for Implementing Naming Conventions

To successfully implement your naming conventions, consider the following best practices:

- **Document Your Conventions:** Create a clear and accessible guide that outlines your naming conventions. Include examples, definitions, and any exceptions to the rules.
- **Use Automation Tools:** Tools like linters, token managers, and design system automation software can help enforce naming conventions and ensure consistency across your design and development environments.
- **Train Your Team:** Ensure that everyone involved in the design and development process understands the naming conventions and knows how to apply them. Regular training sessions or workshops can be beneficial.
- **Regularly Review and Update:** As your design system grows, regularly review your naming conventions to ensure they still meet the needs of your team. Be open to making adjustments as necessary.

4.8 The Impact of Thoughtful Naming on Your Design System

Naming your design tokens is a critical step in building a scalable, efficient, and user-friendly design system. By following the principles and best practices outlined in this post, you can create a naming convention that not only supports your current design needs but also allows your system to grow and evolve over time. Remember, the goal is to create a shared language that facilitates collaboration, ensures consistency, and ultimately enhances the quality of your digital products.

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

21

http://www.door3.com/

Module 5: Introduction to Tokens Studio

5.1 Creating Design Tokens with Tokens Studio

Tokens Studio is a dedicated tool for creating and managing design tokens, providing more advanced features for complex design systems. Here's how to create and manage design tokens using Tokens Studio.

Getting Started with Tokens Studio

Step 1: Install Tokens Studio

- Tokens Studio can be integrated with Figma as a plugin. Install the plugin from Figma's [community plugins section](#).

Step 2: Setting Up Your Project

- Create a new project in Tokens Studio. This project will serve as your central hub for managing all your design tokens.

Tokens Studio stores and exports your tokens as a single JSON file. While we'll explore using Tokens Studio with Figma, you can also utilize other third-party apps like Specify or Zeroheight to transform your tokens from your JSON file.

Creating and Managing Tokens in Tokens Studio and Figma

Step 1: Create and Define Token Sets in Tokens Studio for Figma

Tokens Studio allows you to categorize tokens in **sets** by type, such as colors, typography, spacing, etc. Start by defining categories that align with your design system.

We recommend utilizing a hierarchy to define different sets that serve the end goals of the design system. Foundational "reference" tokens can provide the basic guidelines, while

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

22

other "applied" tokens aggregate basic building blocks and are attributed to components directly. For example, in DOOR3's design systems we utilize several categories:

- **Core** tokens are the basic building blocks of your design system and are used as reference points for the tokens applied to your components.
  - Core tokens are foundational, but will not be surfaced in your final designs. This streamlines the design process and reduces the risk of misapplication.
- **Theme** tokens are subsets of core tokens that allow you to create different groupings for different uses.
  - Themes are especially useful because they facilitate modern UX design workflows. Global core tokens can be developed for brands or projects, organized into themes, and then are available for other designers to use without having to worry about the core tokens and their variables.
  - Themes' iterative nature makes them well-suited for developing white-label products that can cut down on extensive production processes.
- **Modes** are groupings or "semantic buckets" of color tokens that signify dynamic elements like light and dark, background and surface, but also things like content styles such as rest and active, borders, and states for text input fields, icons, and buttons.
- **Applied** tokens are the tokens that get applied directly to components. These can include things like sizing, color, topography, and effects. However, it is up to you how you ultimately structure your tokens.
  - We recommend this approach in order to maintain a separation between reference tokens (the underlying architecture) and the tokens which actually get applied to components. This helps simplify the exporting process and creates some guardrails for designers downstream.
  - Applied tokens can be defined at either the semantic or component level. Many DOOR3 designers set up mode sets as their applied sets because it simplifies the stack, but you might find a benefit in separating them out.

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

23

http://www.door3.com/

The benefit of this hierarchical approach is not only to keep your workspace uncluttered, but also to facilitate the multi-level workflows of complex team projects where multiple designers are responsible for different tasks. For example, a lead designer will often create the foundational core and alias tokens that comprise the themes used in a project. Then, downstream designers will be able to ideate within these themes without having to interact directly with Tokens Studio or the underlying token system generally.

Step 2: Create Your Design Tokens

The token creation process in Tokens Studio takes place in the context of the specific structure and hierarchy you will format through the use of **sets**. In order to get the most from your design tokens, it is important to consider how you will structure your sets into themes. This will ensure you will be able to successfully export them from Tokens Studio to Figma and apply them to your designs.

To start creating tokens, select the type you want by clicking the '+' icon to view the "New Token" modal. In that form, fill in a name for your token and a value (in this case a hex code color). You can then create your token!
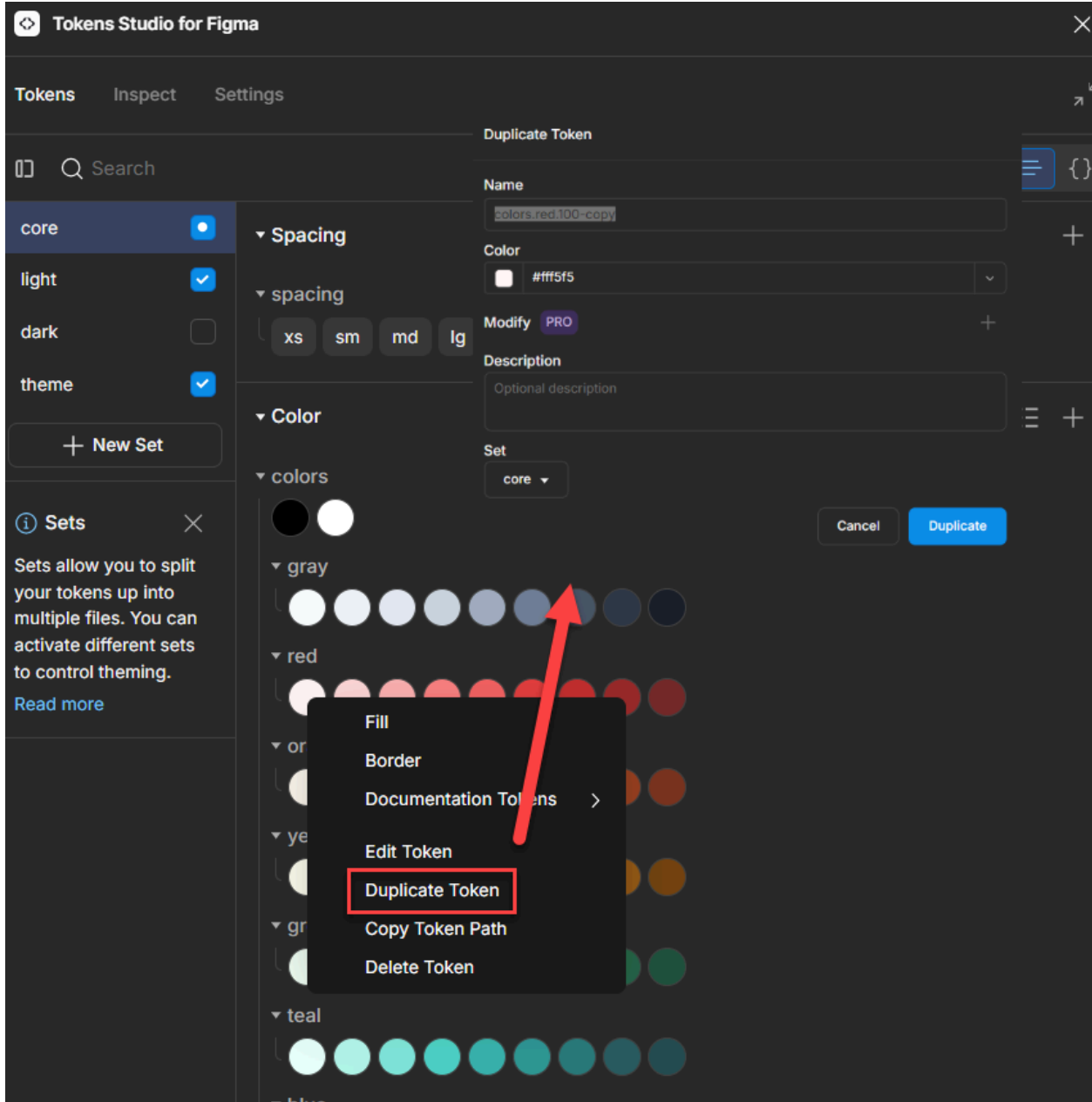
New Token

Name

Unique name

Color

\#

Modify PRO
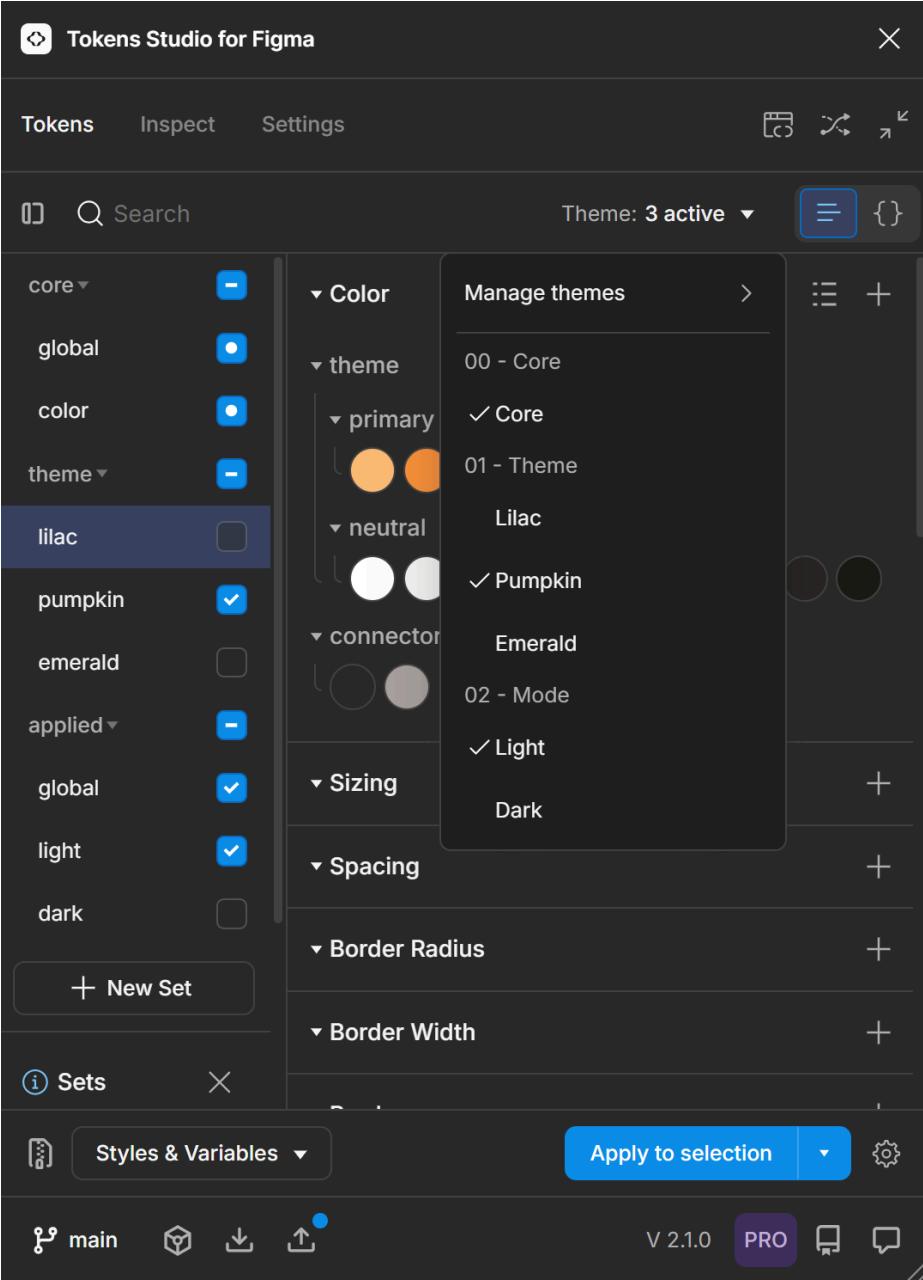
Description

Optional description

Cancel   Create

Once you familiarize yourself with creating different types of tokens, you can edit existing ones through the "Edit Token" option in the dropdown. Here, you can also duplicate the token or other options. As your token sets expand, you can add more buckets to organize them in groups.

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

25

Step 3: Applying Tokens as Themes to Components in Figma

Token Studio in Figma also allows you to access **Themes** on the fly from the Manage Themes dropdown.

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

26

We recommend using Tokens Studio for Figma with the "Apply to selection" setting in the bottom right of the Tokens tab to keep processing times to a minimum as processing the whole file can take some time.
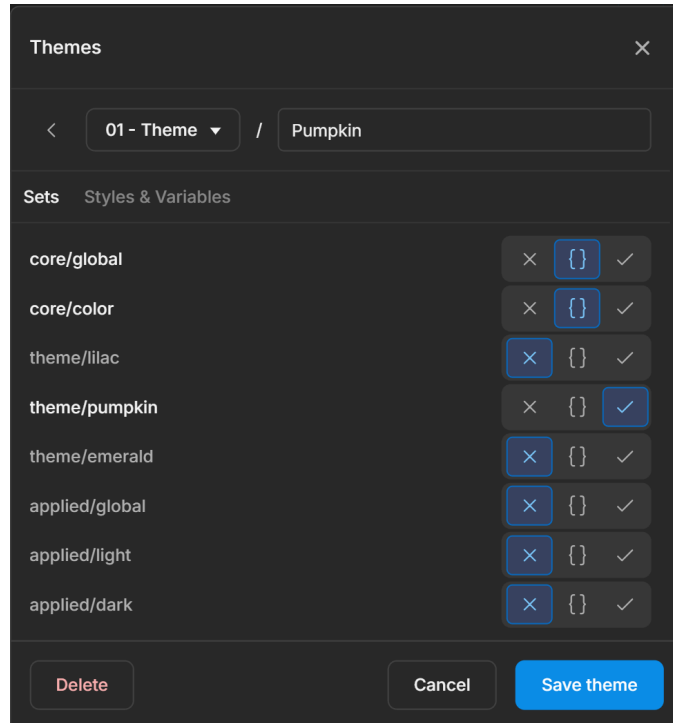
370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

27

To view components in either Light or Dark Theme, select the parent frame, then use the Figma Tokens plugin to switch between themes. The active theme lowest in the list will override the matching values in the themes listed above, but larger variant sets may take a moment to switch all elements to the chosen theme.

Note that this approach will not work if you have your tokens connected to Figma variables; if you've connected your tokens to variables, you'll be able to switch themes directly in the Figma design panel.

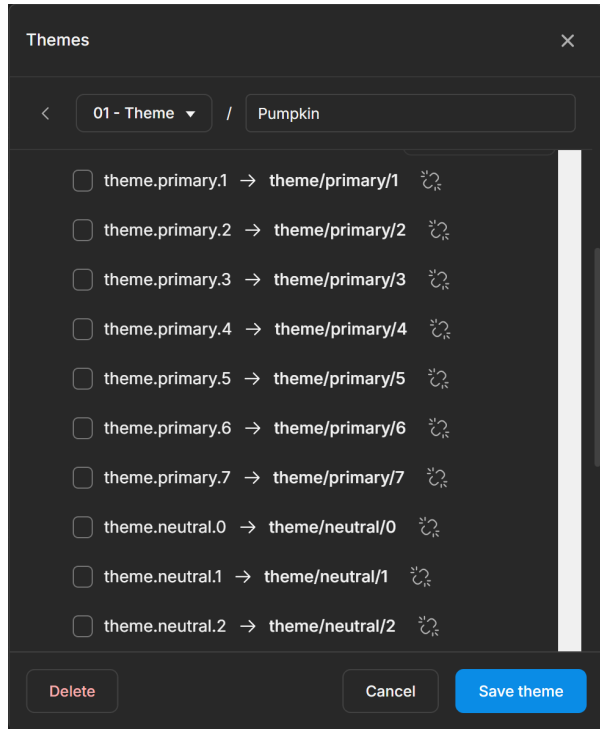You can manage and create themes in a dedicated menu:
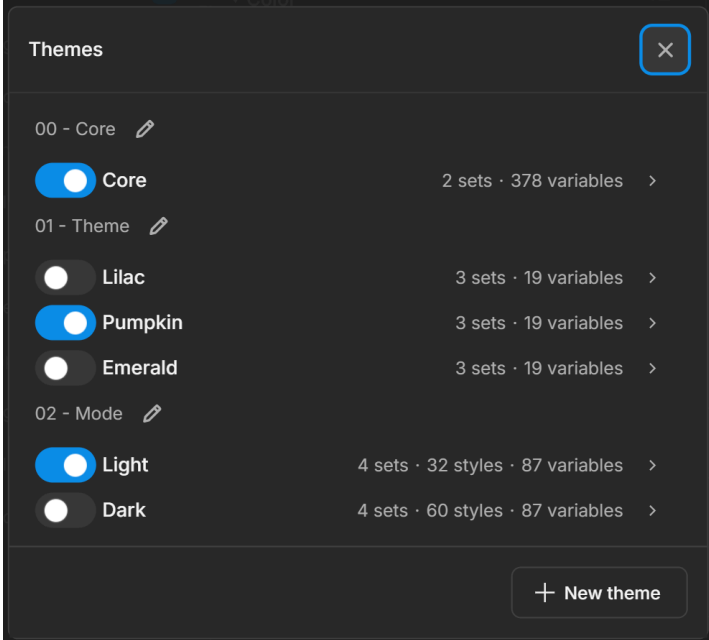
You can create your themes and customize them using the tokens you've created with a submenu's controls:

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

28

Understanding these controls is an **essential** part of exporting your tokens from Tokens Studio to Figma because you use these controls to designate how your sets should function when acting on a component. Remember, your core global tokens are used for reference only and should be marked as such. This allows them to function in the background while the applied tokens surface in the designs.
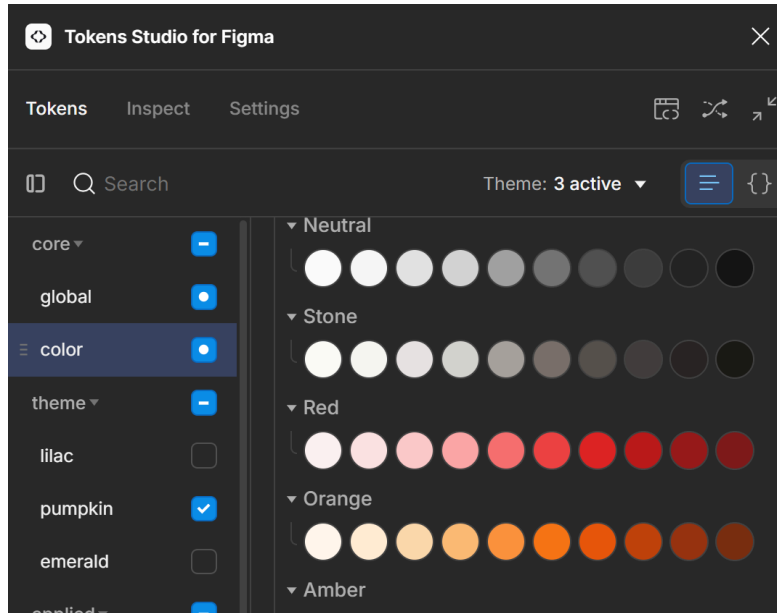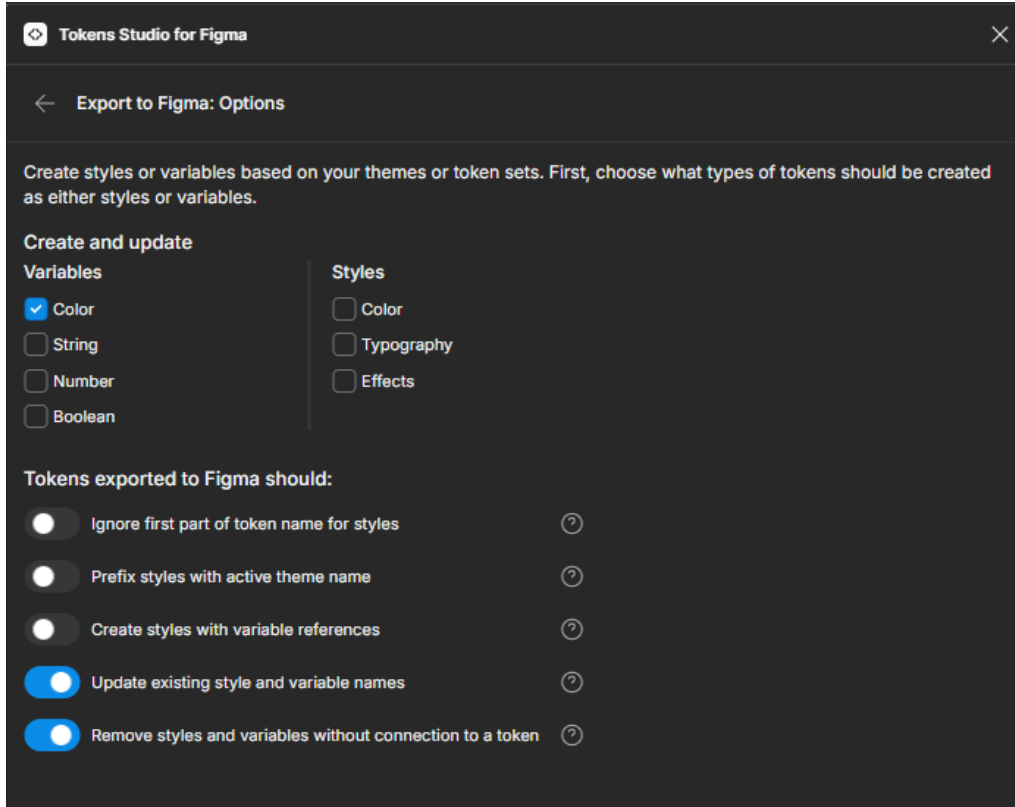
Defining your themes is key to successfully importing your tokens into Figma and applying them to your designs. This is especially true in practice, as themes invariably become complex. For example, if you switch to view the Figma variables in one of our example themes, you see it is composed of dozens of variables:

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

30

Step 4: Exporting tokens to Figma

Once you've created your tokens in Tokens Studio, you can export them into Figma as styles or variables in order to surface them for other designers in the Figma design panel.

In the "Styles & variables" menu, select "Export styles & variables." Check the variables and options you want to export. By selecting the last two options, "Update existing…" and "Remove styles…" you can ensure that any updates made to tokens in Tokens Studio are reflected in your Figma files.

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

*Mas de Roda 6-14. Sant Martí*
08005 Barcelona. Spain

31

After you confirm and select what sets to update, you can return to Figma to see your updated themes and variables. Note that there are some differences in terminology between Tokens Studio and Figma:

- In Tokens Studio, the individual groups of tokens are called "**sets**" which correspond to Figma "**groups**."
- However, what Tokens Studio calls "**themes**" are "**modes**" in Figma.
- Lastly, Token Studio "**theme groups**" map to "**collections**" in Figma.

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain
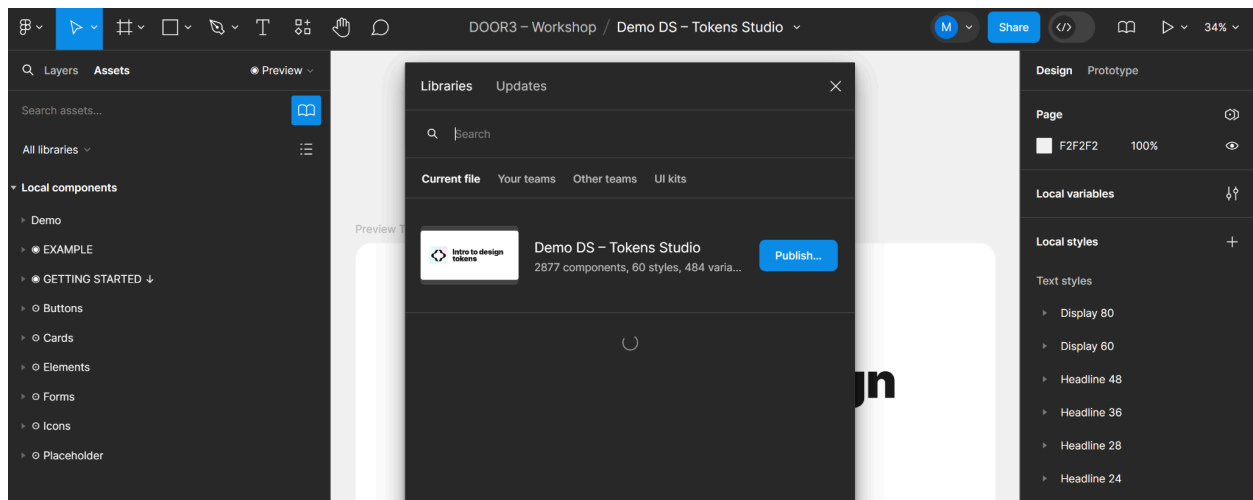
32

http://www.door3.com/

It seems confusing, we know. Once you familiarize yourself with this process, however, you'll find the connection fairly intuitive.


Applying tokens to components
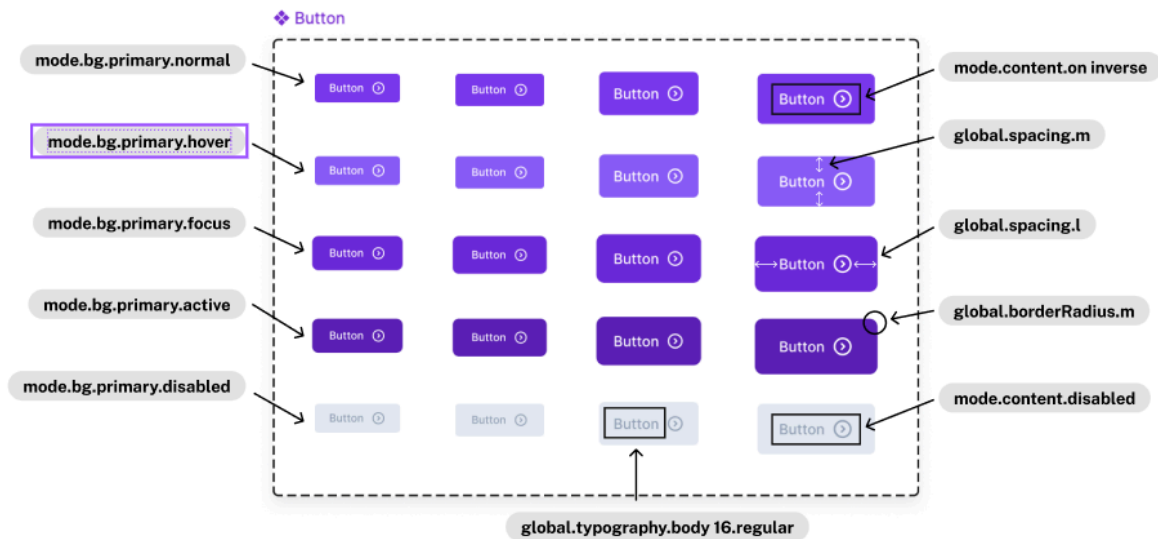
Step 5: Publish your changes in Figma

Once you have developed and integrated your design tokens into your design system, you will want to publish the changes across your designs. To do this, you will need to open the Library menu in Figma and publish your changes.

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

33

From this point on, the workflow should be familiar to any designer who's comfortable in Figma. The tokens that you've set up are connected to your styles, variables, and components, and your design system is poised for greatness.
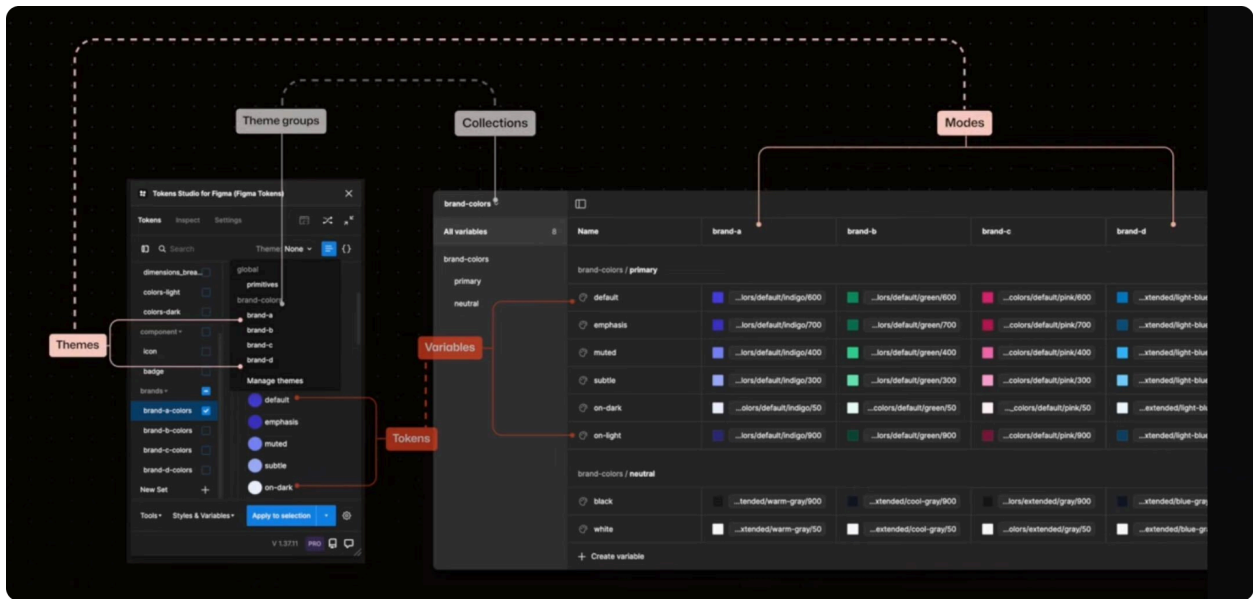
5.2 Advanced Token Management

Once you have mastered the process of creating, exporting, and publishing your tokens, there are still more skills to learn on your way to building your own design-to-dev pipeline.

Manage Responsive Tokens Tokens Studio supports responsive design by allowing you to define a base font size so that your tokens will adapt based on screen size. This ensures that your design system is scalable and flexible.

Export Tokens for Development Once your tokens are set up, there are other tools available to transform your JSON into a format (e.g., XML, CSS) that can be easily integrated into your development workflow. This ensures that the design tokens are implemented consistently in code and will stay in sync with design assets in Figma.

5.3 Choosing the Right Approach for Your Team

What's the right approach for your organization?

Small design team

Large design team

Multiple design teams across the organization

Both Figma and Tokens Studio offer robust tools for creating and managing design tokens. Figma is excellent for designers who want to manage tokens directly within their design files, while Tokens Studio provides advanced features and greater control over token management, making it ideal for larger, more complex design systems.

When choosing the right tool, consider the size and complexity of your design system, your team's workflow, and how closely you need to collaborate with developers. Is your team small or large, or are you working with multiple design teams across an organization? As organizations scale up, the risks of design redundancies and inconsistencies grow larger, as does the misapplication of tokens.

A simple token schema that's well-defined and clearly named based on real-world use cases will provide the most value, while also encouraging widespread adoption. Whether you opt for Figma, Tokens Studio, or another tool, investing time in setting up your design tokens will help your designers and developers ship better products, faster.

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

35

Module 6: Applying Tokens To Components

6.1 Linking Tokens Across Components

As we've discussed in previous modules, design tokens are platform-agnostic name:value pairs that store design information, such as colors, typography, spacing, and shadows, and can alias to other name:value pairs. Linking tokens across your design system components frames them as a single source of truth, allowing designers to implement consistent styles across various components and platforms. However, in order to start, you should ensure your tokens and variables are compiled into a useful library.

6.2 Applying Tokens to Components Within a Design System

Once you've defined your tokens, the next step is integrating them into the relevant components of your design system. In Figma, this means selecting a component and assigning specific token values to its properties. For example, a button component might have its background color linked to a primary color token, the text linked to a typography token, and the padding connected to a spacing token.
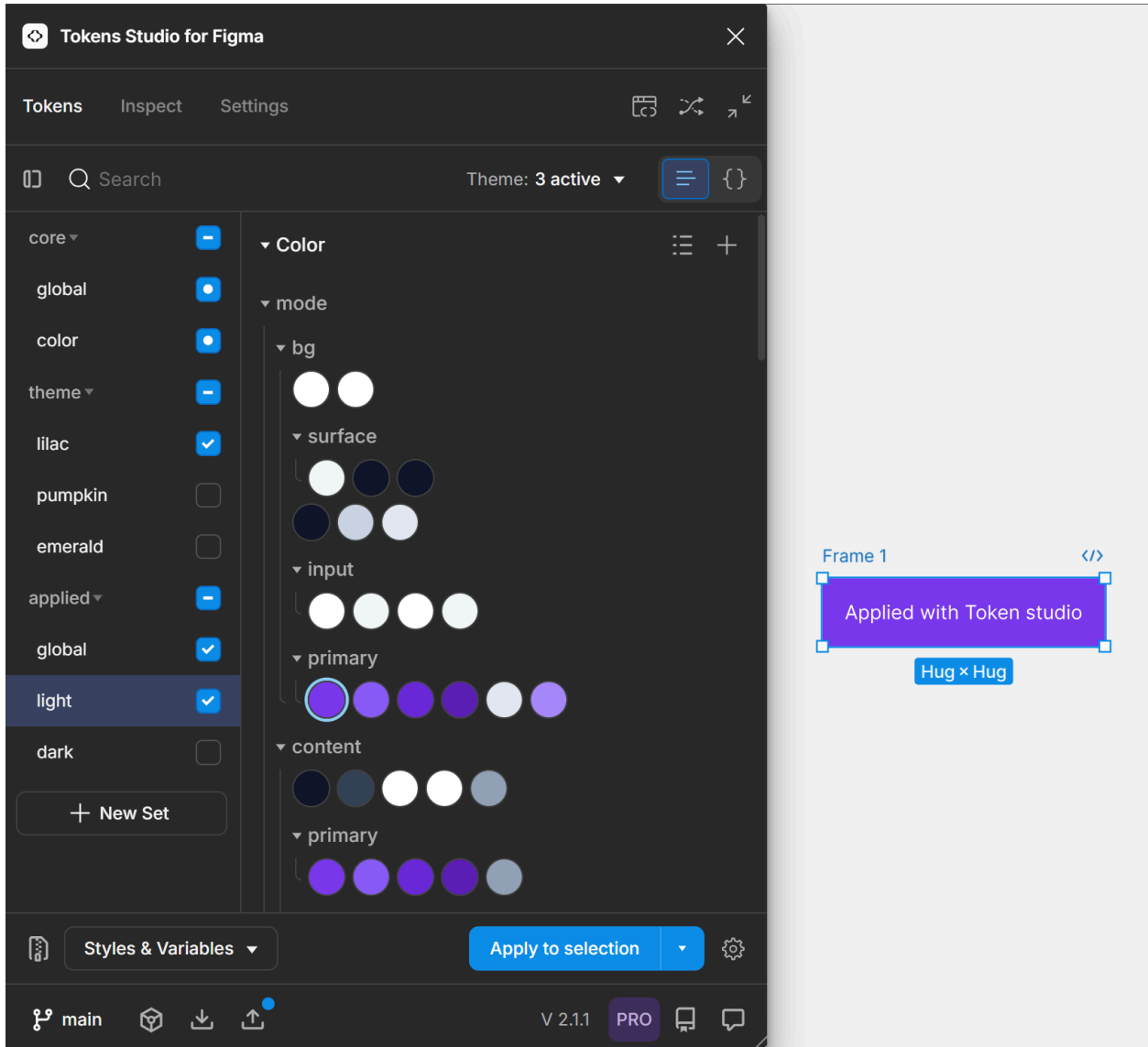
By linking tokens to components, any changes made to the tokens will automatically reflect across all instances of that component, ensuring consistency and reducing the need for manual updates. We'll use buttons as an example in this guide, but the same process can be applied to any component in your design system.

In this module, we'll explain four different methods for linking tokens to components. We'll focus on two recommended approaches, while also covering other alternatives for context.
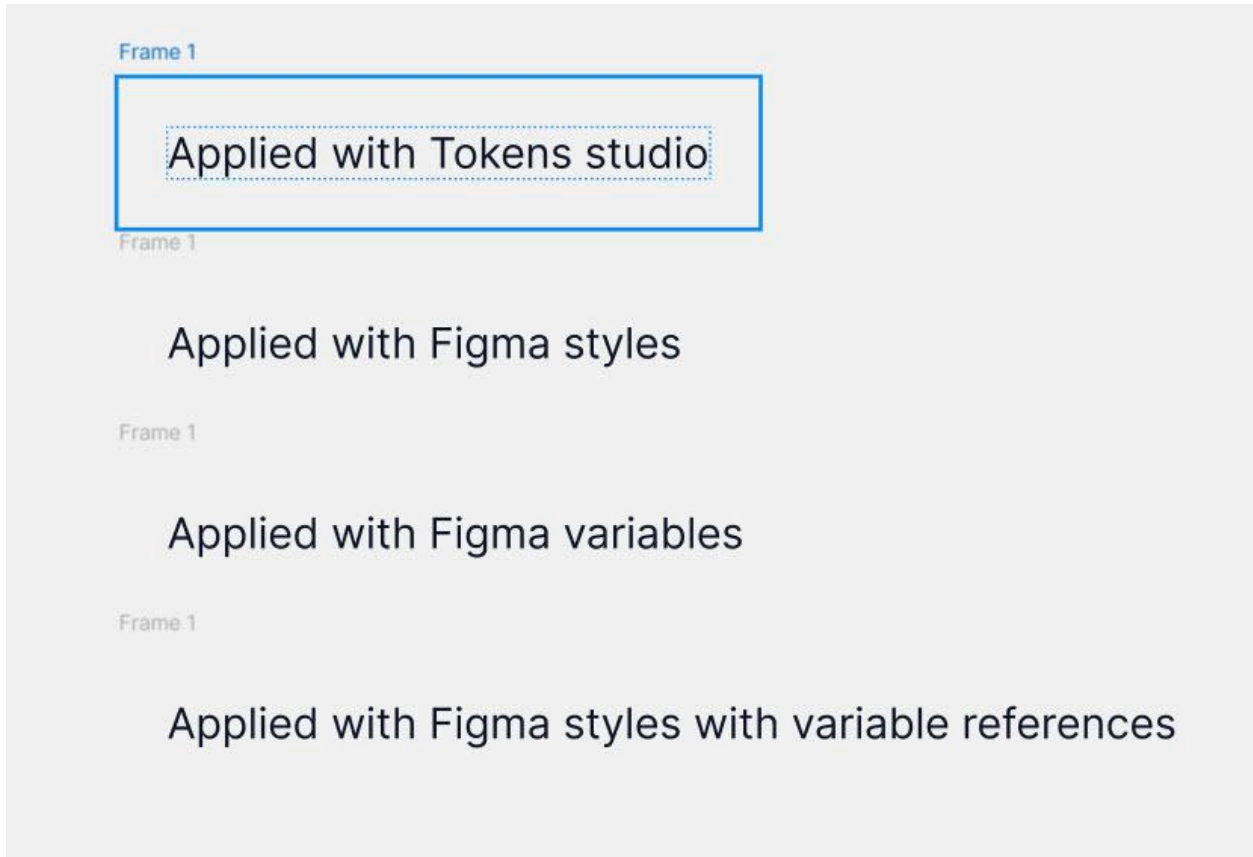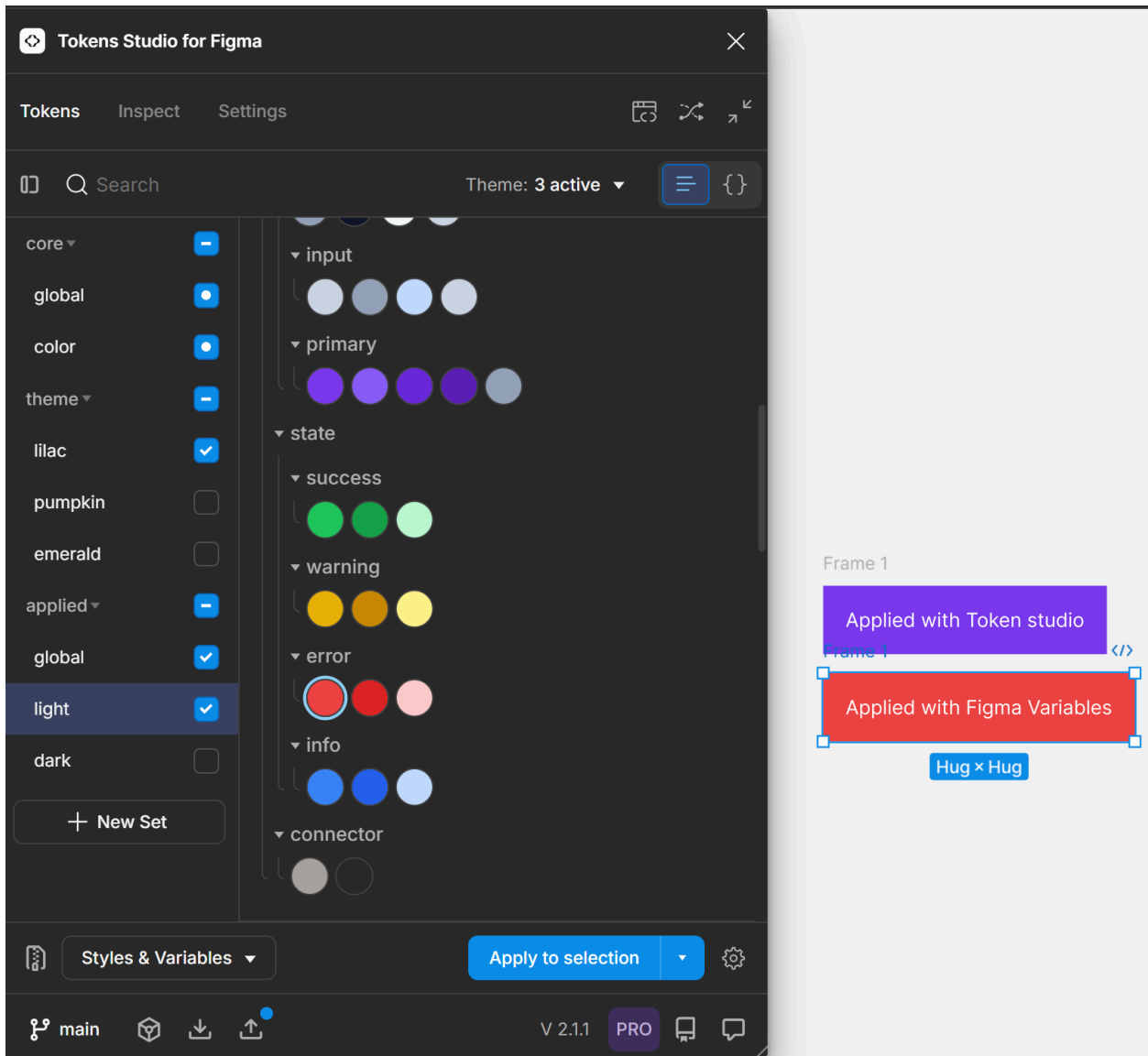
Method 1: Applying Tokens With Tokens Studio

- Select the button's frame in Figma.
- In Tokens Studio, choose an applied token theme (e.g., light mode, primary tokens) and assign the corresponding colors or other properties, such as spacing or typography.
- For the button's text, select the text element and choose the appropriate typography token.
- Once everything is assigned, click "Apply to Selection" in Tokens Studio.

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

37

http://www.door3.com/

Method 2. Applying Tokens with Figma Variables

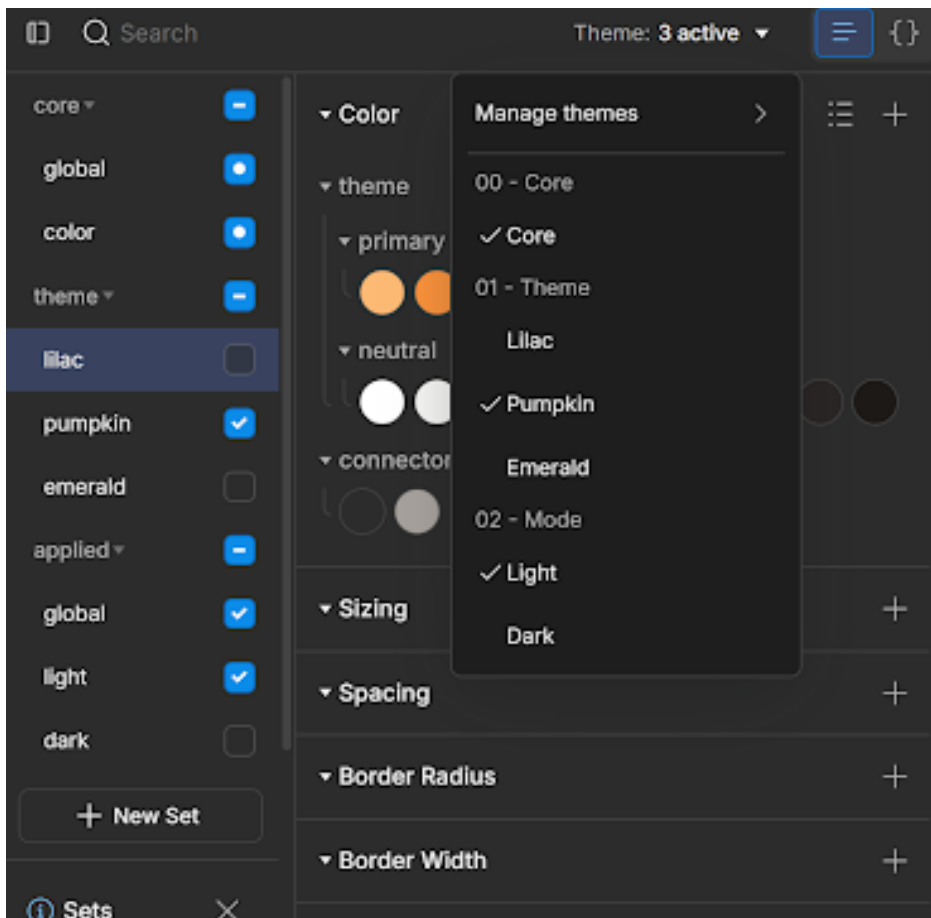- Export your variables from Tokens Studio.
- In Tokens Studio, select "color" variables, choose the last two options, and then click "Confirm."
- In the "Export to Figma" modal, select only the tokens you want to surface in Figma and click export. Be sure not to export your reference tokens to avoid misapplication.
- Select your component in Figma and open the "Fill" options in the side panel.

- To access the tokens, click on the "styles and variables" library (represented by the four-dot icon).
- Choose the relevant variable (the same one used in Tokens Studio), and you will see the selected token appear in the Figma sidebar and in Tokens Studio.
- Repeat the process for the text, assigning the appropriate typography or color token.
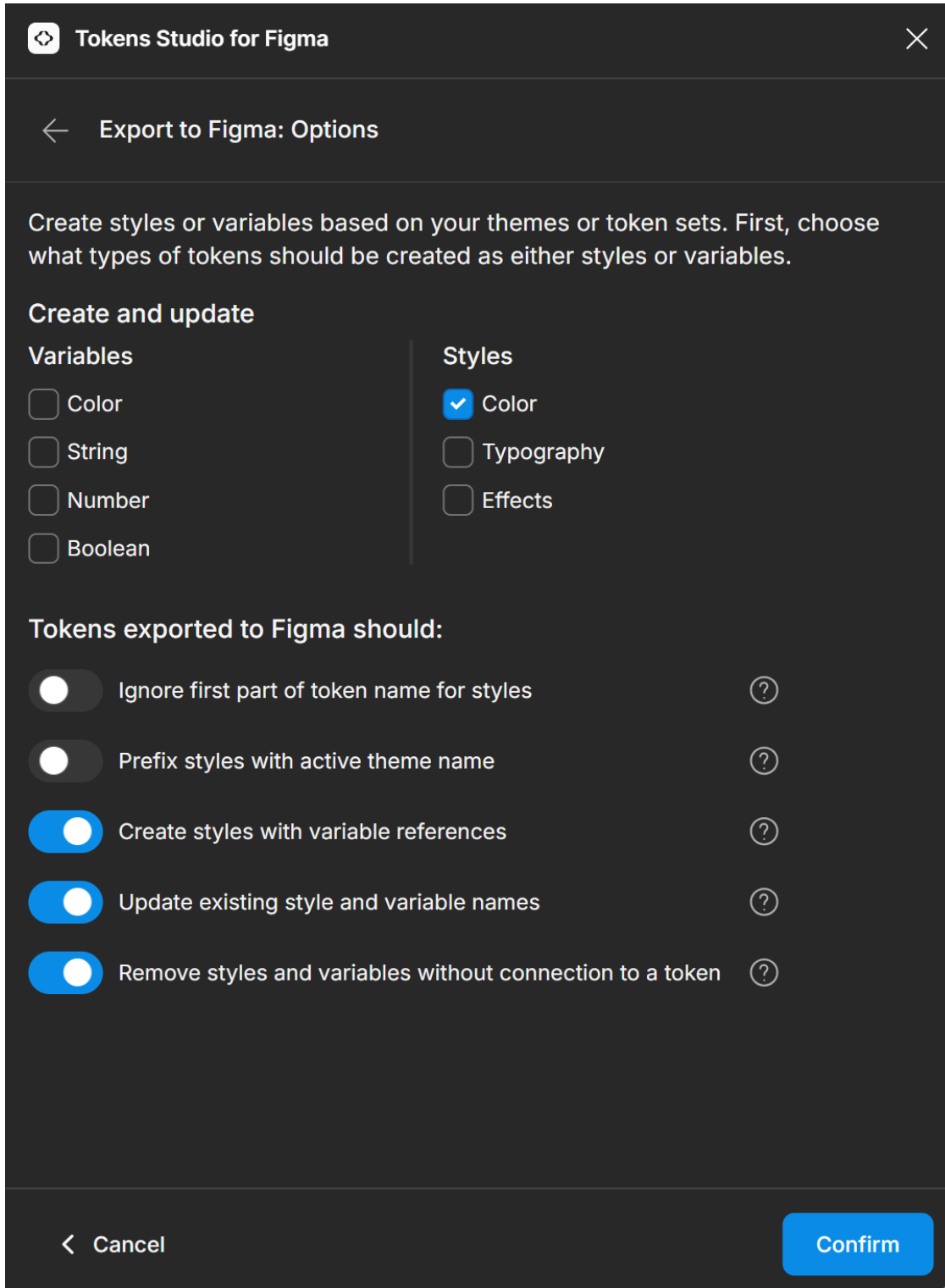

Method 3. Applying Tokens With Figma Styles

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

39

- Export your styles from Tokens Studio.
- In Tokens Studio, select "color" styles, choose the last two options, and then click "Confirm."
- In the "Export to Figma" modal, select only the tokens you want to surface in Figma and click export. Be sure not to export your reference tokens to avoid misapplication.
- When exporting styles in multiple modes, you'll need to export each mode set separately because styles will not switch modes automatically. In the Manage Themes dropdown, select the mode you want, then export your styles. Repeat this process for each mode you want to set up.

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
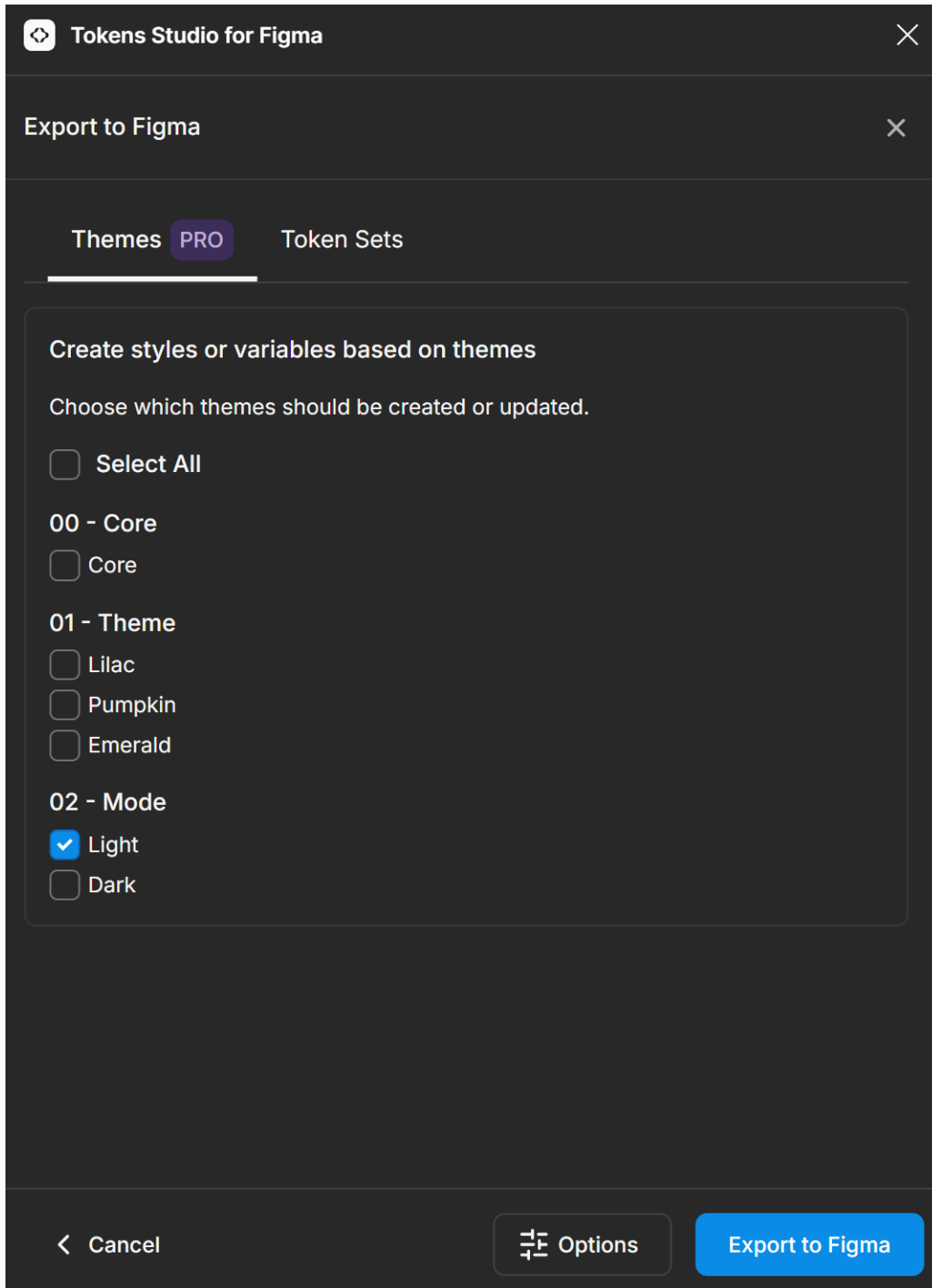08005 Barcelona. Spain

40

Back in Figma, navigate to the "Fill" menu and scroll past the square icons for variables until you find the circular icons representing styles. Select the appropriate style token for both the frame and text.

Method 4. Applying Tokens with Figma Styles With Variable Reference:

In the "Export to Figma: Options" modal in Tokens Studio, select the option to "Create styles with variable references" and click "Confirm." This will create a connection between your styles and variables, so that your styles alias to your variables and will resolve as variables if you detach them.

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

41

http://www.door3.com/

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

42

In the following modal, select a single mode token to apply.

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

43

In Figma, apply the tokens from the "Fill" menu to both the frame and text as you would in the previous examples.
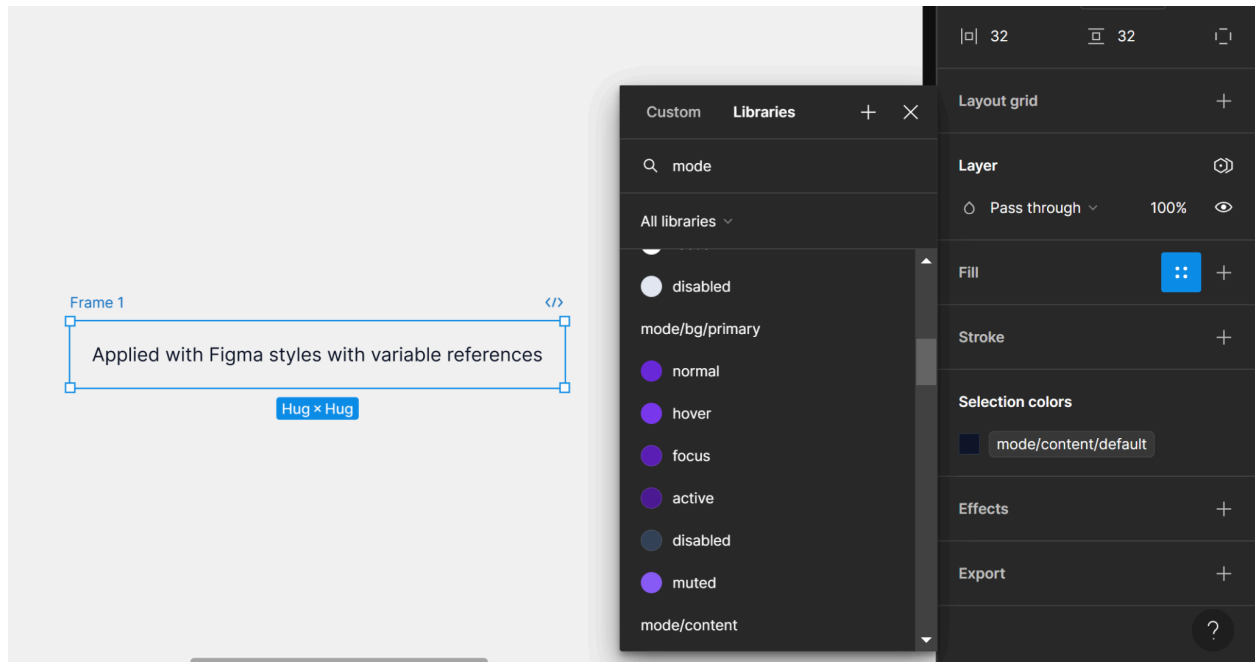
6.3 Which Approach is Right for Your Team?

Each method for linking tokens has its own merits, but due to current limitations in Figma Styles, we recommend the first two approaches: using Tokens Studio or Figma Variables. When using Figma Styles, we encountered issues where the connection between Figma and Tokens Studio wasn't appearing consistently and the tokens needed to be reapplied.

Let's now explore the key advantages of the recommended approaches.

Advantages of Applying Tokens with Tokens Studio

Tokens Studio allows for seamless reapplication of tokens, even after changes have been made in Figma. For instance, if you delete styles or variables from a component in Figma,

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
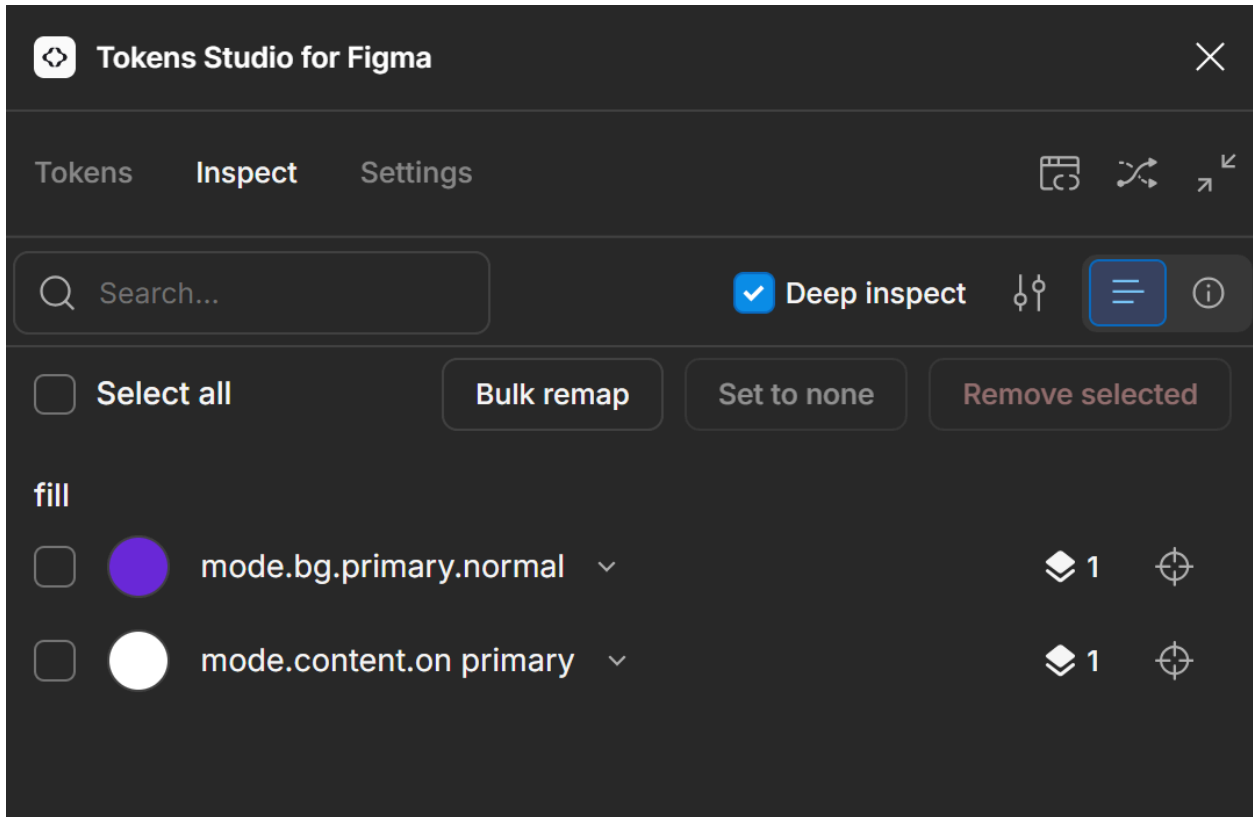*08005 Barcelona. Spain*

44

you can reload your tokens from Tokens Studio to revert back to the original design without losing consistency.

To illustrate this, try the following:

- Copy your components and delete all styles and variables from them.
- For components linked with Tokens Studio, use the "Inspect" tab in the Tokens Studio modal to see that tokens remain applied, even after changes in Figma.
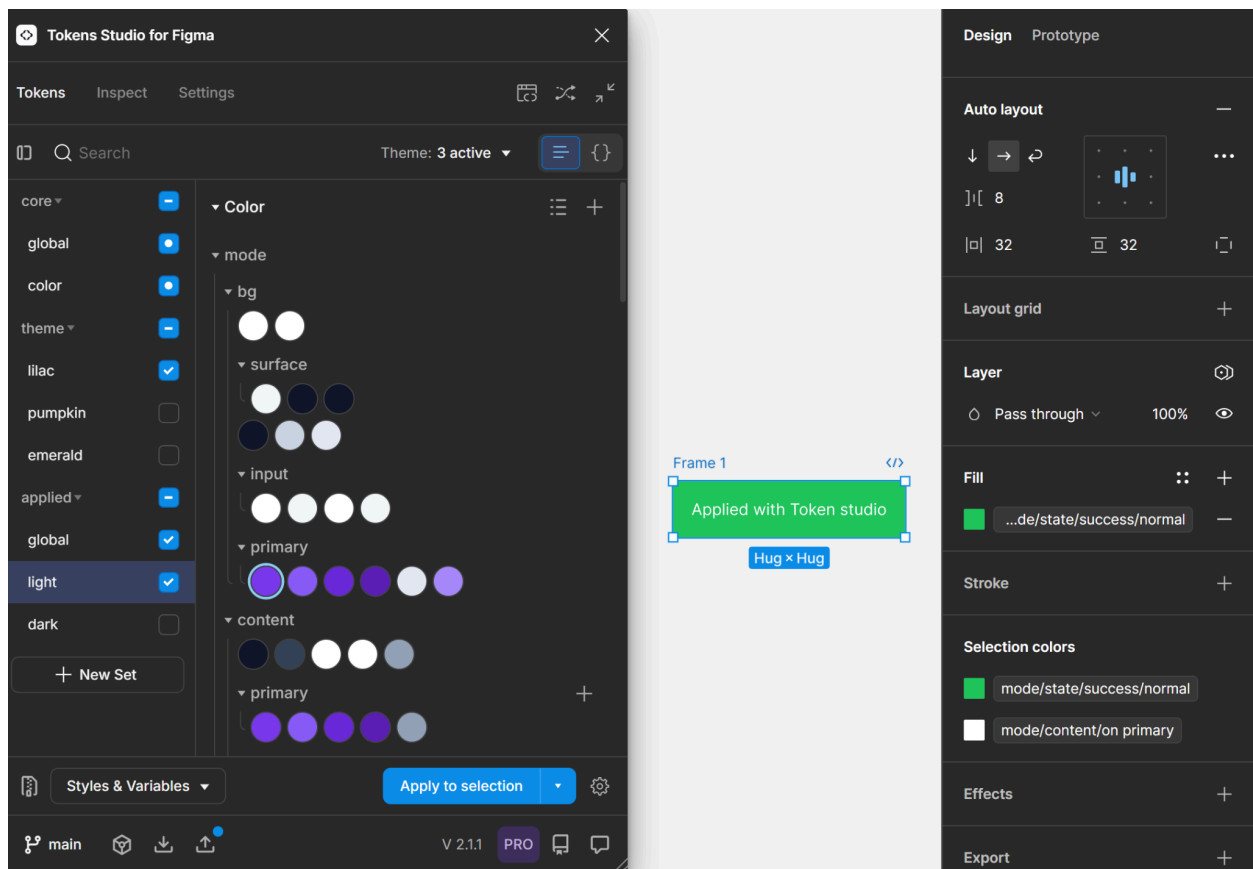
- For the final two buttons, remove the "Styles" selections in the "Fill" section.
- Now, if you click these and select the "Inspect" tab in the Tokens Studio modal, you can see the first component still has tokens applied.

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

45

However, the rest of the buttons show no tokens are applied. So, if you go ahead and select all four of the buttons and click "Apply to Selection," you'll see the tokens reapplied to the component (button) that was applied with Tokens Studio and nowhere else.

This is why we recommend using Tokens Studio to apply your tokens if you want a stable connection between the two. This strong connection between Tokens Studio and Figma provides a stable foundation, especially when managing large design systems or frequent updates. It's ideal for teams who want to centralize the control of component styling at the level of the design system.

Advantages of Applying Tokens With Figma Variables

Using Figma Variables to apply tokens offers a different approach that allows downstream designers more control and flexibility when styling components. Unlike Tokens Studio, where changes are driven from Tokens Studio itself, Figma Variables sync the token values

from Figma to Tokens Studio. This approach localizes the application of tokens within Figma and gives designers the freedom to make changes to component styling.

In our first example, for our component applied with Tokens Studio, if we change the token in the Figma "Fill" menu, it will change the color or other styling to match in Figma, but the original token is still applied through Tokens Studio, so that when tokens are updated, the fill change will be overwritten.

- In our example, if you select the component using variables, change the option in the "Fill" menu, and click "Apply to Selection" in Tokens Studio, you'll see that it syncs the two programs.
- By applying it as a variable, it changes the tokens in Tokens Studio. The changes are being driven by Figma, whereas in our previous example they emanate from Tokens Studio.
- If your team values direct control and collaboration through Figma's interface, applying tokens as variables may be a better fit.

Both methods offer valid and powerful solutions for managing design tokens. The key is to decide which method aligns best with your team's workflow and design system needs. Once you choose a method, commit to it for consistency across your organization. This will help avoid confusion, streamline updates, and ensure your design-to-dev pipeline remains efficient and effective.

6.4 Additional Recommendations

These approaches to applying tokens should cover your available options. However, there are some additional details that are worth considering if you want to master design tokens and all they offer.

Utilize Variants and States

Figma's support for component variants and states allows for more granular control over how tokens are applied. Variants represent different versions of a component, such as a button in its default, hover, and active states. By linking tokens to these variants, you can

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

47

ensure that changes to the design tokens are consistently reflected across all states of a component.

For instance, if a button's hover state uses a slightly lighter shade of the primary color, you can link this variant to a token representing that shade. Any updates to the primary color token will automatically adjust the hover state, ensuring consistency without additional manual effort.

Organize and Manage Tokens

As your design system expands, managing a large number of tokens becomes complex. It's crucial to organize your tokens in a way that facilitates ease of use and maintenance. In Figma, you can group tokens into categories such as "Colors," "Typography," "Spacing," and "Shadows." Within these categories, tokens can be further subdivided to reflect specific use-cases or design patterns.

Effective token organization not only simplifies the process of applying tokens to components but also makes it easier to update and scale your design system over time.

Test and Iterate

Once tokens are linked across components, it's essential to test the system to ensure that changes propagate as expected. This involves making changes to tokens and verifying that these changes are accurately reflected across all instances in your design.

Testing should be an iterative process, allowing you to refine your token system based on feedback and real-world use. As your design system evolves, you may need to adjust existing tokens or introduce new ones to accommodate emerging design requirements.

6.5 Challenges and Considerations for UX Designers

While the benefits of linking tokens across components are significant, there are also challenges and considerations that UX designers should be aware of.

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

48

Managing Complexity in Large Systems

As your design system grows, the number of tokens can quickly become overwhelming. It's important to establish clear guidelines for token creation, organization, and usage to prevent confusion and maintain the integrity of your design system. This includes defining naming conventions, categorizing tokens logically, and regularly auditing your token library to remove redundant or outdated tokens.

Ensuring Team-wide Adoption

For linked tokens to be effective, it's crucial that all team members adopt the practice consistently. This may require training and onboarding sessions to ensure that everyone understands how to use tokens correctly. Providing clear documentation and guidelines can also help facilitate team-wide adoption and reduce the risk of errors.

Tooling Limitations

While tools like Figma and Tokens Studio offer robust support for design tokens, there may be limitations depending on the complexity of your design system. It's important to evaluate your tools and workflows to ensure they can accommodate the specific needs of your project. In some cases, you may need to explore additional plugins or custom scripts to fully leverage the potential of linked tokens.

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

49

Module 7: Integrating Tokens Into Development

7.1 Introduction

Integrating design tokens from Figma with your code repository using Tokens Studio is a powerful way to maintain a consistent and efficient design-to-development workflow. This setup allows you to manage your design tokens with version control, ensuring that all team members are working with the latest updates, even when collaborating on different files.

In this module, we'll present a step-by-step guide to setting up this integration using the Tokens Studio for Figma plugin. We'll use GitHub as an example, but also provide an overview and resources for other repositories.

7.2 Syncing Tokens With a Repository

Since tokens are most useful in a team environment, it's also essential to have a version control in place to ensure that token values remain updated while the team members are working on different files.

To select one of the sync providers, in the **Settings** tab, you should find the **Add New** button. You will find the following options in the menu.

Sync options

Designers have several options to sync tokens depending on your team's needs. The following overview outlines the major options and some considerations for each. We'll be using GitHub as our example, but you can find more information here.
- Local Storage
    - By default, all tokens are stored on the document that you're working in. If you need to work in multiple documents, this option won't work as it means you would have to copy paste your tokens to the other document.
- URL

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

50

- Pulls tokens from a remote .json stored on your server. This is a read-only sync method.
- Generic Versioned Storage
    - An extension off of the URL storage. This storage provider supports read/only, read/write and read/write/create flows. Additionally it can be extended with arbitrary headers on the requests should you need more fine grained support for your end-point.
- JSONBin.io
    - You can use JSONBin.io by creating an account and syncing your tokens there. It's a free and easy way to sync your tokens, but they are not version controlled by default.
- GitHub
    - The recommended way as it hosts your code and design decisions in the same location. By connecting to a repository, you can pull and push tokens or even select a branch to push to. This allows you to explore design decisions before deploying them to a production environment. Additionally, we provide support for GitHub Enterprise for enhanced functionality and compatibility.
- Gitlab
    - Like GitHub, you can use GitLab to store your tokens in your repository.
- Azure DevOps
    - Allows you to store your tokens on ADO.
- Supernova
    - Allows you to sync your tokens between Tokens Studio and Supernova.

7.3 Syncing With GitHub

Storing tokens on GitHub is beneficial as it's version controlled; it allows you to work on different branches (think of a brand refresh that lives on a different branch than the current live version) and gives you varying access levels such as Read or Write access. You can also integrate GitHub Actions which will allow you to create a token pipeline with Style Dictionary or other tools.

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

51

To set up the Github sync for your file, follow these steps:

Step 1: Sign Up and Sign In to GitHub

- Sign up for an account at [GitHub](#) if you don't already have one.
- Log in to your account.
- [Create a new repository](#), name it and select your desired privacy settings
- Initialize it as **README** (a crucial step).

Step 2: Generate a Personal Access Token

In the **Settings** tab of your profile, scroll to the bottom to find **Developer Settings** where you should find [Personal Access Tokens](#).
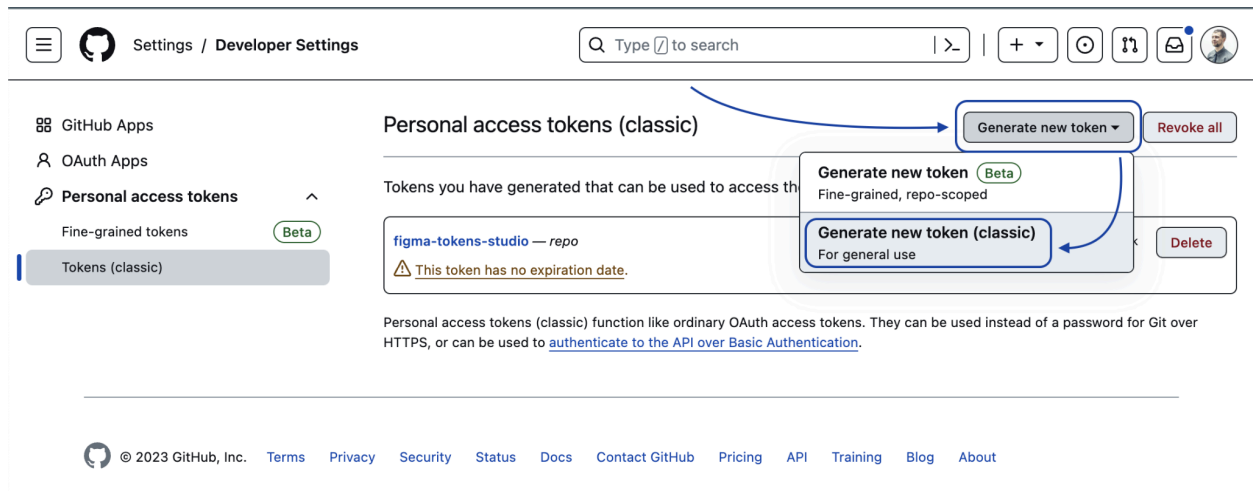
- Click on **Generate new token**.
- On the **New personal access token (classic)** page, fill in the **Note** input with a description for the token.
  - You can either create a personal access token with the older 'Tokens (Classic)', or with 'Fine-grained tokens':
- Set **Expiration** to **No Expiration**.
- Under the **Select scopes** section, check the **Repo** option.
- Click **Generate token** and copy the newly generated GitHub access token.
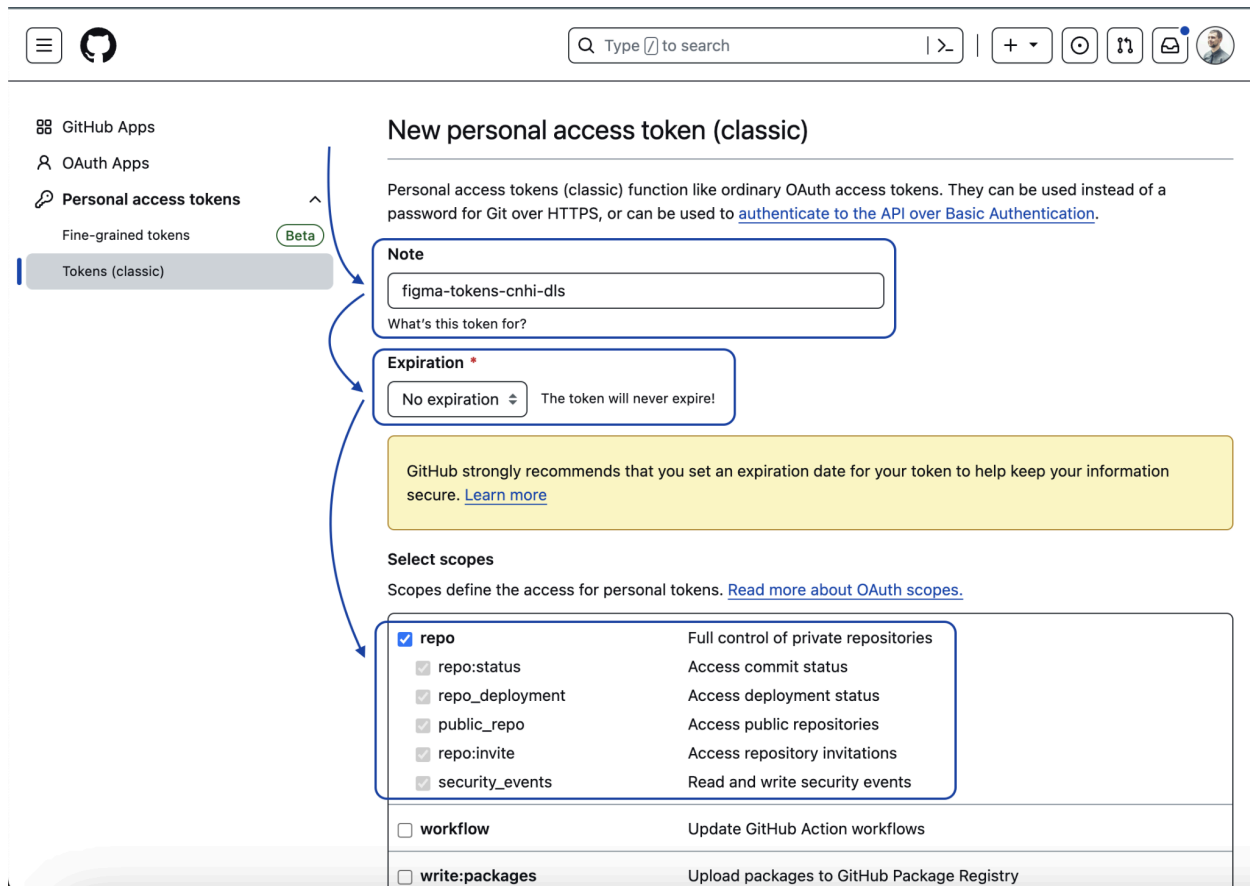
Once you've generated your token, make sure to copy and securely store it right away, as you won't be able to retrieve it later if you close the page. It's crucial to keep this token private and avoid sharing it with anyone who shouldn't have access to your repository. Think of the token as you would your personal password—its security is paramount. If the token is ever exposed or shared by mistake, you should immediately revoke and delete it to protect your repository's integrity.

Step 3: Configure Tokens Studio for Figma

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

53
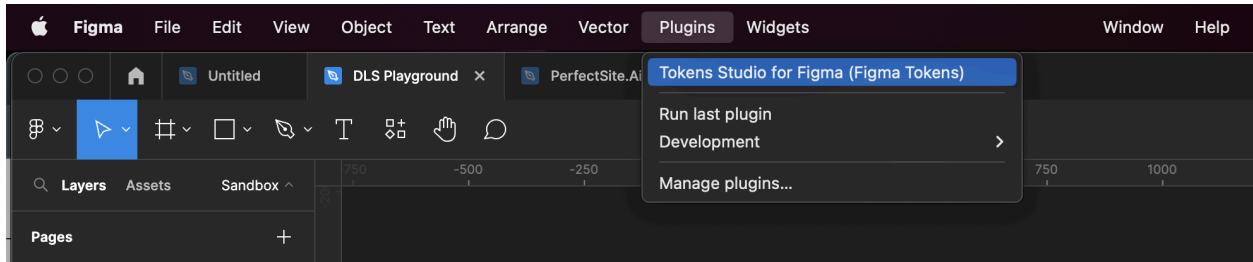
- Open your Figma project.

- Launch the Tokens Studio for Figma plugin.

- Switch to the **Settings** tab.

- Click on **Add new sync provider** and select the **GitHub** option.
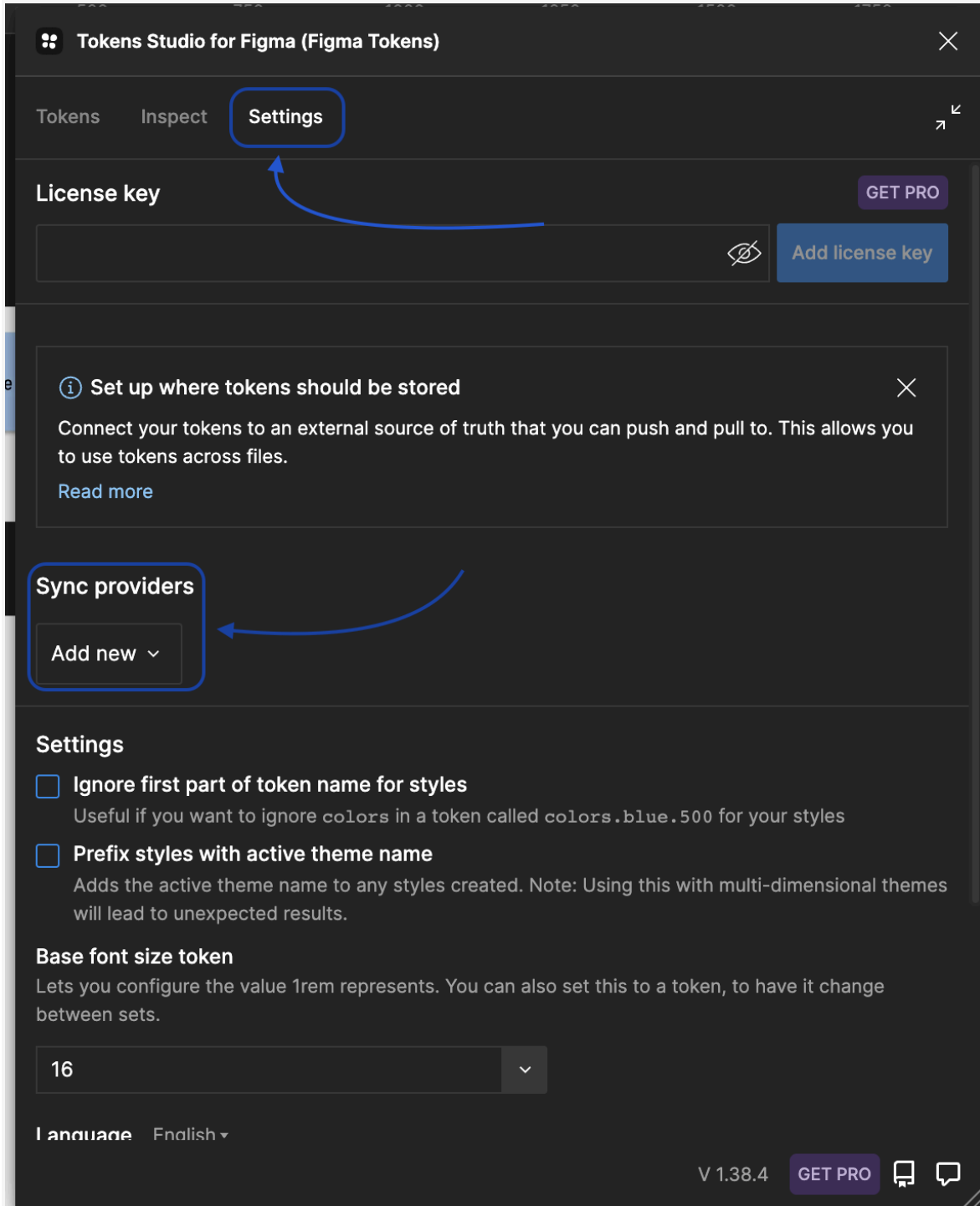
Step 4: Set Up GitHub as a Sync Provider

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

55

http://www.door3.com/

Add new credentials ✕

Add new GitHub credentials

Access tokens stored on your repository, push and pull tokens in a two-way sync.
Read more

Name

CNHI DLS Github repo connect

Personal Access Token

•••••••••••••••••••••••••••••••••

Repository (owner/repo)

Door3Dev/cnhi-dls

Branch

figma-design-tokens

File Path (e.g. tokens.json) or Folder Path (e.g. tokens)

/tokens/figma/tokens.json

Base URL (optional)

https://github.acme-inc.com/api/v3

Cancel   Save

- Enter the **Name** of the sync provider connection (only on the plugin UI).
- Paste the **Personal Access Token** you generated in GitHub.
- Enter the **Repository** value (example `tokens-studio/figma-plugin`).
- Select the Git repository **Branch** where all changes for the Figma tokens will be pushed and pulled from - `figma-design-tokens`.

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

56

- Specify the file path where the Figma tokens are located inside the Git repository - `/tokens/figma/tokens.json`.
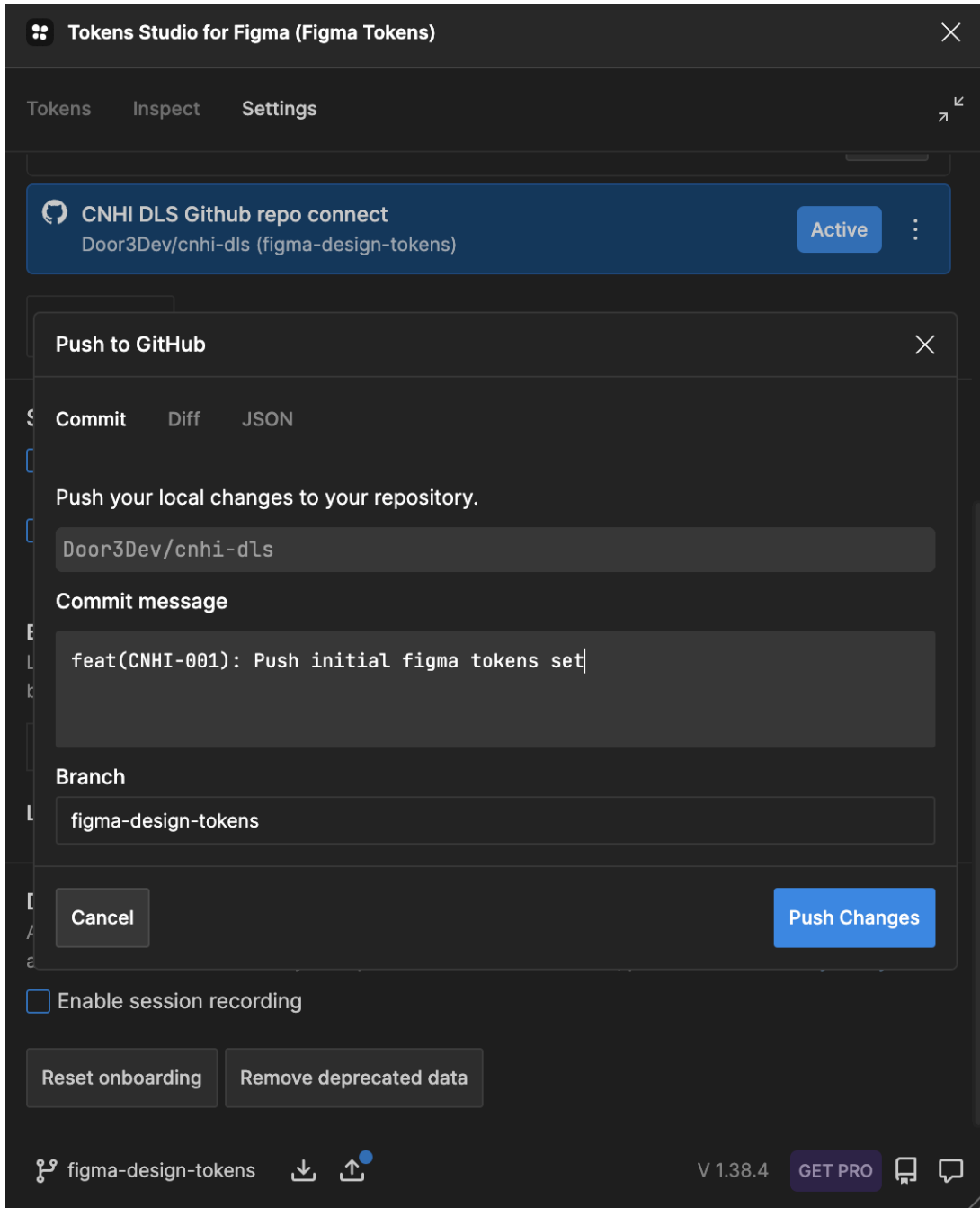- Click the **Save** button.

Tip: If your repository currently only contains a README file without any token files, you can initiate the setup by entering a forward slash '/' in the file path when you make your first commit through the plugin. This action will prepare your repository for use, allowing you to push your newly created tokens afterward.

Step 5: Sync Changes Between Figma and GitHub

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

57
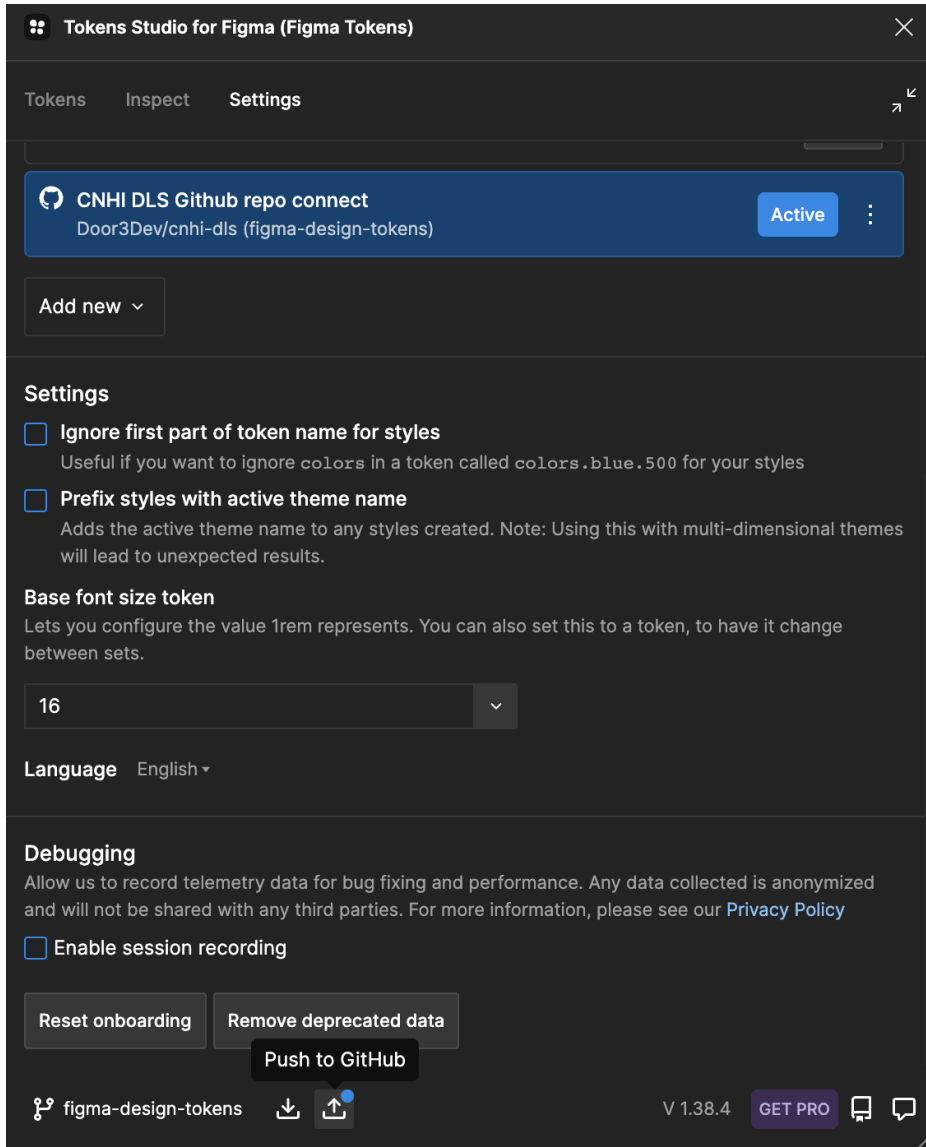
After configuring the sync provider, you'll notice a new set of items at the bottom of the Tokens Studio for Figma plugin window. This includes the GitHub branch name (`figma-design-tokens`) and two icons to pull changes from GitHub or push changes from Figma.

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

*Mas de Roda 6-14. Sant Martí*
08005 Barcelona. Spain

58

Once you've made local changes to the design tokens in Figma, click on the **Push** icon, which will be highlighted with a blue dot indicating changes have been made.

- A **Push to GitHub** modal will open, prompting you to enter a commit message, which should identify the details of the action for later reference.
- To finalize, click **Push changes**.

Note: You can also choose the branch to which you want to push, allowing you to experiment with new tokens on separate branches without affecting the main design system.

Step 6: Create a Pull Request

Once the setup is complete, you can start pushing and pulling tokens between Figma and GitHub.
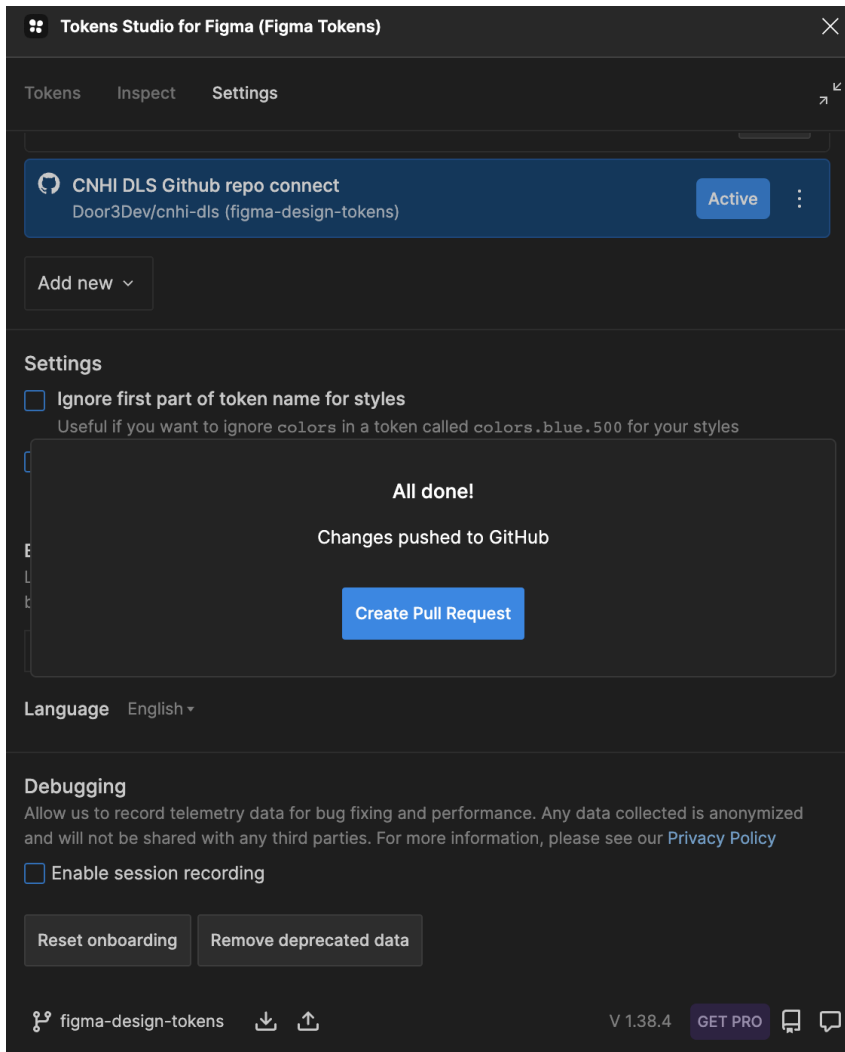
To notify developers of your changes:

- After pushing changes, you'll be prompted to **Create Pull Request**.
- Click the button, and you'll be redirected to GitHub.
- On the **Open Pull Request** page, enter a title, select reviewers, and click **Create pull request**.

Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also compare across forks. Learn more about diff comparisons here.

base: main ← compare: figma-design-tokens | ✓ **Able to merge.** These branches can be automatically merged.

Add a title

feat(CNHI-001): Push initial figma tokens set

Add a description

| Write | Preview | H B I ⚬ <> 🔗 | ⚬ ≡ ✓ | 📎 @ ↗ ↩ ☑ |

Add your description here...

Create pull request ▾

Reviewers ⚙

Request up to 15 reviewers

d3-

👥 **d3-konstantin-zaigraev** d3-konstantin-zaigraev

👥 **d3-laena-ilk**

👥 **d3-robert-miller**

👥 **d3-tom-mcclean** Tom Mcclean

👥 **d3-viacheslav-onyshchenko**

👥 **Door3Dev/d3-cnhi** d3-cnhi

Development

Use Closing keywords in the description to automatically close issues

Congratulations!

You've successfully integrated Figma with GitHub! Your design tokens are now synced, making collaboration between designers and developers more efficient than ever.

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

62

http://www.door3.com/

Module 8: Transforming Your Tokens For Development

7.1 Introduction

As design systems grow more complex and teams increasingly rely on design tokens to ensure consistency, the need to transform these tokens into various development languages becomes essential. This is where tools like Style Dictionary come into play. Style Dictionary is a robust framework that enables you to convert your design tokens into code tailored to different platforms, such as web, iOS, Android, and more. In this module, we'll walk you through the process of transforming your tokens into JSON and then into a specific development language using Style Dictionary.

7.2 Step-By-Step Guide

Step 1: Preparing Your Tokens as JSON

To begin the transformation process you must first ensure your design tokens are organized in a JSON format. This structure is crucial because it serves as the foundation for converting tokens into other formats. If you're using Figma or another design tool with Tokens Studio, you can easily export your tokens as a JSON file. Alternatively, if you're starting from scratch, you can manually define your tokens in a JSON file.

Here's a simple example of what a design token JSON might look like:

```
{

  "color": {

    "primary": {

      "value": "#FF5733",

      "type": "color"
```

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

63

```json
    },

    "secondary": {

      "value": "#33FF57",

      "type": "color"

    }

  },

  "font": {

    "base": {

      "value": "Arial, sans-serif",

      "type": "font"

    }

  }

}
```

This JSON structure categorizes tokens by their purpose, such as color or font, and assigns each a value and type.

Step 2: Installing Style Dictionary

With your tokens prepared in JSON, the next step is to install Style Dictionary. Style Dictionary is a tool that takes your JSON tokens and transforms them into platform-specific code. To get started, you'll need Node.js installed on your system. Then, you can install Style Dictionary using Node Package Manager (NPM).

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

64

Open your terminal and run the following command:

```
npm install style-dictionary
```

This command installs Style Dictionary globally, making it accessible for transforming your design tokens.

Step 3: Configuring Style Dictionary

Once Style Dictionary is installed, the next step is to configure it to transform your design tokens into the desired output formats. The configuration file, typically named `config.json` or `style-dictionary.config.js`, defines how the tokens should be transformed and what formats they should be output in.

Here's a basic example of a configuration file:

```
{

  "source": ["tokens/*.json"],

  "platforms": {

    "web": {

      "transformGroup": "web",

      "buildPath": "build/web/",

      "files": [{

        "destination": "variables.css",

        "format": "css/variables"

      }]
```

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

65

```json
    },

    "ios": {

      "transformGroup": "ios",

      "buildPath": "build/ios/",

      "files": [{

        "destination": "StyleDictionaryColor.h",

        "format": "ios/colors.h"

      }]

    },

    "android": {

      "transformGroup": "android",

      "buildPath": "build/android/",

      "files": [{

        "destination": "colors.xml",

        "format": "android/colors"

      }]

    }

  }
```

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

66

```
}
```

In this example, the configuration file specifies that the source tokens are located in the `tokens/` directory. The tokens are then transformed into CSS variables for web, Objective-C headers for iOS, and XML files for Android.

Step 4: Running the Transformation

With the configuration file in place, you can now run Style Dictionary to transform your tokens. From your terminal, navigate to the directory containing your configuration file and run:

```
npx style-dictionary build
```

This command processes your tokens according to the configuration file and outputs the transformed code into the specified directories.

Step 5: Customizing Your Transformations

While Style Dictionary provides a wide range of pre-built transforms, you may need to customize these to fit your project's specific needs. This can be done by creating custom transforms or modifying existing ones. Tokens Studio offers official transforms in a package called `@tokens-studio/sd-transforms`, which you can use as a starting point.

For example, if you need to output custom CSS properties instead of standard CSS variables, you can define a custom transform in your configuration file. Here's a snippet showing how you might configure a custom CSS transform:

```
{

  "transform": {
```

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

67

```
    "name": "custom/css",

    "type": "value",

    "matcher": function(prop) {

      return prop.attributes.category === 'color';

    },

    "transformer": function(prop) {

      return `--${prop.name}: ${prop.value};`;

    }

  }

}
```

This custom transform checks if the token's category is `color` and then outputs it as a custom CSS property.

Step 6: Using Style Dictionary Configurator

If you prefer a more visual approach, you can use the Style Dictionary Configurator — a web-based tool that allows you to transform your design tokens directly in your browser. This tool uses Style Dictionary under the hood and is perfect for experimenting with transformations without needing to install anything on your computer.

Simply upload your JSON file, configure the output formats, and let the tool handle the rest. This is especially useful for quick prototyping or testing different transformation setups.

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

68

Module 9: Design Token Governance

9.1 Introduction

Design tokens are essential to creating scalable, consistent, and efficient design systems. They also bridge the gap between designers and developers by ensuring that design decisions are carried out systematically, with uniformity across platforms and products. However, as powerful as design tokens are, their effectiveness depends on how well they are managed or governed. Without a robust governance framework, the benefits of design tokens can quickly unravel.

9.2 What Is Governance and Why It Matters

In this context, governance provides structure and consistency to design systems across teams and products. It ensures that design tokens are used appropriately, updated correctly, and applied consistently across all platforms. Governance helps align teams — designers, developers, and product managers — by providing clear guidelines, processes, and roles for how design tokens are created, maintained, and used.

Imagine a global retail brand operating across a mobile app, website, and in-store kiosks. If the company's web and mobile teams independently create a promotional banner that requires brand colors, padding, and fonts, they would rely on established governance to ensure consistency. They would first consult the design system or sync with the token guardian in order to make sure their choices adhere to established guidelines. After developing the new design, they could then work with the guardian to incorporate new patterns into the system, so that all teams have access to them.

9.3 Setting Up Your Design Token Governance Process

Design token governance isn't just about enforcing rules; it's about creating a process that encourages best practices, aligns teams, and ensures the long-term success of your design

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

69

system. Below is a step-by-step guide to establishing a governance framework for your design tokens.

Step 1: Define Roles and Responsibilities

One of the first steps in creating a governance process is to define the roles and responsibilities of everyone involved. This ensures that all stakeholders understand their part in managing design tokens, and it minimizes the risk of misunderstandings or bottlenecks.

Team Roles to Consider:

- **Design System Team**: Typically responsible for the creation, maintenance, and governance of the design tokens. This team ensures that all components adhere to established design principles.
- **Product Teams**: These teams use design tokens to build products and interfaces. They must follow the design system guidelines but may also request new tokens or updates.
- **Developers**: Developers play a critical role in implementing design tokens in code. They must ensure that tokens are applied correctly and work across different environments.
- **Governance Lead**: This role may fall to a design system manager or a UX lead who is responsible for overseeing the governance process and ensuring alignment across teams.

Individual Roles to Consider:

In practice, different individuals (or groups of individuals) take responsibility for various tasks in the workflow. The example below is from our practices here at DOOR3. Keep in mind, the above categories are general and can apply to one or more people depending on the size and complexity of your organization.

These individual roles are split up into two types:

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

70

Leadership:

- **Design Language (DL) Lead** owns, establishes, maintains, and distributes the design system guidelines. They make the ultimate decisions about the design systems.
- **Project Manager** coordinates resources, tasks and timelines, owns the component request backlog, and orchestrates communication.
- **UX Lead** owns and is responsible for the UX of one or more products. They work with the Design Language System (DLS) Guardian and UX Designers to define how design systems guidelines are applied.
- **Dev Lead** owns the shared component repository and oversees the development and maintenance of token transformation scripts.

Production:

- **DLS Guardian** designs and maintains one or more product-level design systems, ensuring consistency across product-level systems.
- **Token Guardian** owns and maintains the token "source of truth" and ensures all styles across the Design Language System (DLS) remain in sync.
- **UX Designer** identifies needs and gaps in the product level component library, providing functional requirements, and tests new and updated components.
- **Developer** owns and maintains platform-level dev libraries and infrastructure.

These roles work in tandem to preserve the continuity and stability of design systems by focusing on improving these key processes:

- **Documentation** through comprehensive records ensures clarity and accessibility for all product stakeholders.
- **Collaboration** fosters teamwork among designers, developers, and stakeholders for cohesive outcomes.
- **Governance** establishes guidelines and processes for sustainable maintenance and evolution.

Step 2: Establish a Clear Token Creation and Approval Process

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
*08005 Barcelona. Spain*

71

Design tokens should not be created haphazardly. A structured process for creating, reviewing, and approving new tokens ensures that they are thoughtfully designed and align with the broader design system.

Key Aspects of Token Creation:

1. **Token Definition**: When a new design token is needed, its purpose, usage, and relationship to other tokens should be clearly defined.
2. **Review and Approval**: Before a token is added to the design system, it should go through a formal review process. The design system team should ensure that it meets the organization's design standards and is necessary for scaling the system.
3. **Documentation**: Every token should be thoroughly documented to ensure proper usage and application. This includes its purpose, usage guidelines, and examples of where it should be applied.

A common approach is to use version control systems like GitHub or Jira to track changes and approvals, ensuring transparency and accountability across teams. Let's look at an example of a workflow approach that we've employed and found effective within a range of projects.

Design → dev workflow
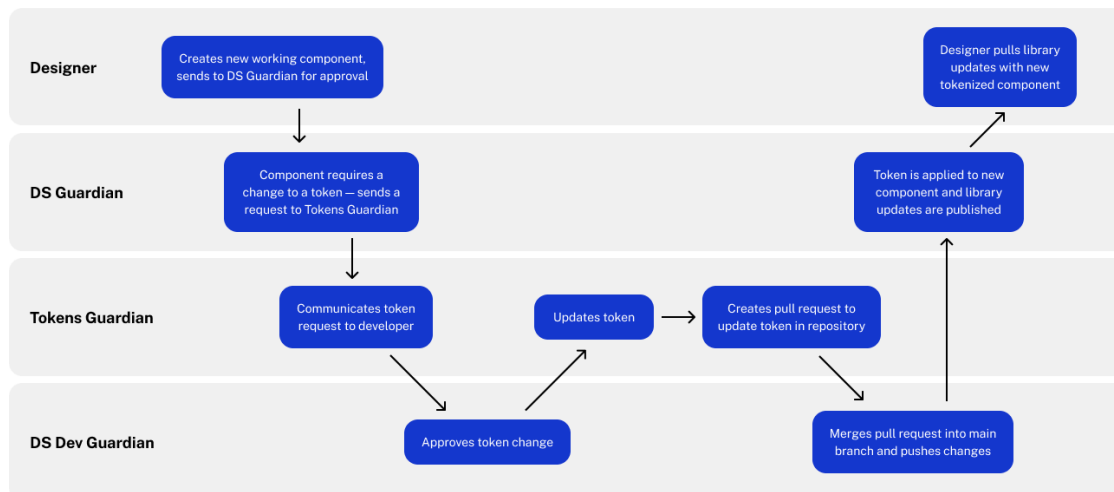
This workflow is designed so that each role serves to support the other while providing checks and balances in the process. This ensures consistency across complex teams and the entire organization. Each role not only serves to move the process forward, but also ensures visibility and collaboration.

Step 3: Implement Token Naming Conventions

Consistency in naming design tokens is critical for maintaining clarity and usability. Without a standard naming convention, it becomes difficult for designers and developers to find, use, and maintain tokens.

Best Practices for Naming Tokens:

- **Use a Hierarchical Structure**: Tokens should be named in a way that reflects their relationship to each other. For example, `color-primary`, `color-primary-dark`, and `color-primary-light` clearly indicate the relationship between different color tokens.
- **Be Descriptive**: Use names that accurately describe the token's function and purpose. Avoid vague or overly technical names.
- **Keep It Simple**: Avoid long, complex names. Tokens should be easy to read and understand at a glance.

Having a standardized naming convention allows teams to quickly identify the right tokens and apply them consistently.

Step 4: Establish Communication Channels for Feedback

Design tokens need to evolve over time to accommodate new features, products, or branding changes. A clear process for requesting changes or providing feedback is essential for maintaining the relevance and effectiveness of your token system.

Communication Channels to Consider:

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

73

http://www.door3.com/

- **Slack or Microsoft Teams**: Create dedicated channels for token-related discussions and questions. This ensures that all feedback and requests are centralized and easily accessible.
- **Issue Tracking Systems (Jira, GitHub, etc.)**: Use issue tracking tools to formally submit token requests or report problems. This ensures that requests are documented and can be prioritized effectively.
- **Regular Meetings**: Schedule regular check-ins between the design system team and product teams to discuss token-related issues, review requests, and provide updates.

By providing clear channels for feedback and communication, you can ensure that your design tokens remain aligned with the organization's evolving needs.

Step 5: Version Control and Change Management

As design tokens evolve, it's essential to manage changes in a way that minimizes disruption and ensures backward compatibility. Version control allows teams to track changes to design tokens and revert to previous versions if necessary.

Key Aspects of Version Control:

1. **Semantic Versioning**: Use semantic versioning to indicate the scope of changes to design tokens. For example, `v1.0.0` could represent the initial release, while `v1.1.0` indicates minor updates.
2. **Changelog**: Maintain a changelog that tracks all updates to design tokens, including why the change was made and its impact on the design system.
3. **Deprecation Strategy**: When tokens are updated or replaced, provide a clear deprecation strategy to ensure that old tokens are phased out gradually, without breaking existing components.

Change management is crucial for maintaining stability across the design system, especially as more teams and products rely on the tokens.

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

74

Step 6: Create a Token Review Process

To maintain the integrity of the design system, establish a regular review process for all tokens. This ensures that tokens remain relevant and are used correctly across the organization.

Steps for a Token Review Process:

1. **Periodic Audits**: Conduct regular audits of the design system to ensure that tokens are being used appropriately and are still necessary.
2. **Review Metrics**: Track the usage of tokens across products to identify under-used or redundant tokens.
3. **Cross-Team Collaboration**: Involve stakeholders from design, development, and product teams in the review process to get a holistic view of token usage and potential issues.

Step 7: Foster a Culture of Continuous Improvement

Governance is not a one-time task; it requires ongoing attention and refinement. Encourage teams to adopt a mindset of continuous improvement by regularly evaluating the effectiveness of the governance process and making adjustments as needed.

Tips for Continuous Improvement:

- **Gather Feedback**: Regularly solicit feedback from all teams involved in using the design tokens. This can provide valuable insights into areas for improvement.
- **Iterate on the Process**: Based on feedback and evolving needs, continuously iterate on the governance process to ensure that it remains effective and relevant.
- **Celebrate Successes**: Recognize and celebrate milestones, such as the successful rollout of new tokens or the resolution of governance issues. This reinforces the value of good governance and encourages teams to continue following best practices.

370 Lexington Ave. Suite 1806
New York, NY, 10003. United States
http://www.door3.com/

Mas de Roda 6-14. Sant Martí
08005 Barcelona. Spain

75

Final Thoughts

We hope you've enjoyed this deep dive into the world of design tokens and gained valuable insights along the way. We hope you found that this series has helped equip you with practical knowledge to streamline your workflow and elevate your design practice. Whether you're a newcomer or a seasoned designer, mastering these concepts will help you create more scalable, consistent, and impactful systems. We believe that what you've learned here will not only transform your work but also empower your team to work smarter, collaborate more effectively, and stay ahead in the ever-evolving field of UX design. Thank you for following along — stay tuned for more insights and best practices from DOOR3.