Theming (“Brushes”) – Product Brief (Draft)

This document accompanies `docs/2026 – april release x/theming.requirements.md`.

It explains the theming (“brushes”) direction in non-technical terms: what problem it solves, how it fits with templates/blueprints, and how it can still satisfy the user request for “apply Set B’s look to Set A” without turning styling into another hard-coded per-card chore.

---

1. What We’re Solving

People build cards in sets. Sets evolve.

A very common workflow looks like this:

- Make **Set A** in one style (classic HQ look).
- Later make **Set B** with a different back, different fonts, or a cleaner look.
- Realize Set A would benefit from Set B’s look… and now you’re stuck doing repetitive edits card-by-card.
- Start **Set C** as a “total conversion” and want to restyle everything again.

This problem is not about card content (names, rules text, stats). It’s about presentation: fonts, colours, borders, backs, and other visual choices that should be consistent across many cards.

---

2. The Core Idea: A “Brush” Is a Saved Look

A **Brush** (theme) is simply a named “look” you can apply to cards.

Instead of editing 50 cards one-by-one, you pick a brush once:

- “OG HQ”
- “Set B – cleaner fonts”
- “Neon Total Conversion”

Then cards (or whole sets) can use that brush automatically.

---

3. Why “Brushes” Envelope the User Request

The user request can be read as:

> “Separate my content from my presentation so I can re-skin lots of cards quickly.”

Brushes are a direct solution because they support both of the workflows implied by that request:

3.1 Non-destructive re-skinning (the ideal)

Cards **reference** a brush. Changing the brush updates every card using it.

This is how you get:

- “Make Set A match Set B” by switching Set A’s brush.
- “Update the font everywhere” by editing the brush once.

3.2 Batch apply (the practical tool people expect)

Sometimes users do want a one-time “apply this look to these cards” operation.

Brushes can support that too, with guardrails:

- A preview of what will change.

- Options to preserve card-specific overrides.

This satisfies the “one click apply across many cards” expectation without forcing the entire system to be destructive.

---

4. Keep the Roadmap Clean: Templates vs Brushes

It’s easy to accidentally blend layout templating and theming into one huge system. We should avoid that.

The split is:

- **Layout Blueprint (Template)**: where things go (positions, bounds, stacks, groups).
- **Brush (Theme)**: how things look (fonts, colours, effects, frame/back assets).
- **Card Content**: the user’s actual information (title, rules text, stats, images).

This separation matters because it keeps each piece understandable and prevents “styling” features from reintroducing template-specific hacks.

---

5. What a Brush Covers (User-facing)

A brush should feel like it controls the “identity” of a set:

- Typography: title font, body font, sizes, emphasis styling
- Colour choices: border tint, header colours, text colours (where applicable)
- Frame/back assets: “this set uses these backs”, “this set uses this border style”
- Small visual effects: strokes/shadows where they’re part of the look

Importantly: a brush does **not** move things around. Layout remains the template's job.

---

6. "Border Colour" as the First Step

Border recolouring is a perfect example of a styling feature that belongs under brushes.

Even before a full brush system exists, we can implement border tinting as:

- A single "style setting" with a sensible default (today's brown)
- A clear upgrade path later where the brush supplies that value for a whole set

This lets us add meaningful customization without committing to the full theming UI immediately.

---

7. How We Make It Safe (Avoiding "Destructive Edits" Anxiety)

Users are right to worry about batch operations that overwrite work.

Brushes reduce that risk by defaulting to a reference-based model:

- "This card uses Brush X" is safer than "copy 20 style fields into this card forever."

When we do allow batch apply, we can make it safe with:

- Preview + confirmation
- Clear rules for what gets overwritten vs preserved
- A "reset to brush" concept for card-level exceptions

---

8. Fit With Export/Import and Collaboration

If someone shares a set (export/import), they want it to look the same when imported.

A brush model supports that cleanly:

- Export can include the brush definition (and any required assets) when needed.
- Imported cards can keep their style references instead of exploding into per-card duplicated style values.

This also helps with collaboration: multiple developers can work on the renderer and card parts while the brush model keeps styling consistent and centralized.

---

9. Delivery Approach (Phased)

This is intentionally staged so we deliver value without blowing up scope:

1. **Enable small style tokens** (e.g. border colour) with defaults and per-card override support.
2. **Introduce brushes as saved looks** and allow sets/cards to reference them.
3. **Add batch apply + safety UX** (preview, preserve overrides options).
4. (Optional) Add brush sharing/versioning tools as demand grows.

This approach supports the user's desired workflow while staying aligned with the broader architecture direction (blueprints first, then styling).

---

10. Summary

Brushes solve a real, repeated pain: “I wish I could restyle a whole set without re-editing every card.”

They also let us:

- Keep content and presentation cleanly separated
- Make global changes safely and consistently
- Support power-user workflows (batch apply) without making destructive editing the default

Most importantly, brushes are a theming system that sits *on top of* templates—so we can satisfy the user request without turning theming into a second, conflicting templating engine.