Email Templates

Namazu Elements Core Features · https://namazustudios.com/docs/namazu-elements-core/email-templates/

Elements version 3.9+

`EmailTemplateService` is a first-class platform service that stores the subject and HTML body used by transactional emails as persistent, editable records, addressed by a unique `key`. It replaces the older approach of hardcoding subject and body as element attribute defaults, so operators can change wording and styling without a redeploy, and custom Elements can register their own templates without any core platform changes.

Core templates

Two templates ship with the platform. Their keys use the reserved `dev.getelements.elements.` prefix and cannot be created or deleted through the API or CMS, only edited:

Key	Used by	Variables
`dev.getelements.elements.password_reset.email_template`	Password Reset (`PasswordResetService`)	`{link}` - full password-reset URL
`dev.getelements.elements.verification.email_template`	Email Verification	`{link}` - full verification URL

Both are seeded automatically with their built-in defaults the first time they are listed or requested, so they always appear in the CMS even on a brand-new deployment that has never sent either email.

Editing in the CMS

In the admin dashboard, go to **Other > Email Templates**. Core templates are marked with a **Core** badge; their key cannot be edited and they cannot be deleted. For each template you can edit the

name, subject, description, and HTML body, and use the preview action to render the body in an iframe with sample values substituted for any documented variables before saving. See CMS Feature Overview for a tour of the surrounding dashboard.

REST API

Method	Path	Description
`GET`	`/email_template`	Paginated list (`offset` , `count` query params).
`GET`	`/email_template/{idOrKey}`	Fetch by id or by key.
`POST`	`/email_template`	Create a new template. Rejected if `key` uses the reserved prefix.
`PUT`	`/email_template/{id}`	Update name, subject, description, or body. The key is immutable after creation.
`DELETE`	`/email_template/{id}`	Delete a template. Rejected for reserved (core) keys.

All endpoints require a Superuser session.

Using EmailTemplateService in custom Elements

`EmailTemplateService` is exported to element child injectors, so it can be injected directly, no additional ELM dependency is needed. Custom Elements register and fetch their own templates with `getOrCreateEmailTemplate` , an idempotent call that creates the row with the given defaults the first time it is seen and simply returns the existing row on every call after that (including after an operator has edited it in the CMS):

```
import dev.getelements.elements.sdk.service.schema.email.EmailTemplateService;
import dev.getelements.elements.sdk.service.email.EmailService;
import jakarta.inject.Inject;
import jakarta.inject.Named;

import static dev.getelements.elements.sdk.service.Constants.UNSCOPED;

public class WelcomeEmailService {
```

```
    private static final String WELCOME_EMAIL_KEY = "com.mystudio.mygame.welcome_email";

    private EmailTemplateService emailTemplateService;

    private EmailService emailService;

    public void sendWelcome(String toAddress, String displayName) {

        final var template = emailTemplateService.getOrCreateEmailTemplate(
            WELCOME_EMAIL_KEY,
            "Welcome Email",
            "Welcome to the game!",
            "<h2>Welcome, {name}!</h2><p>Thanks for joining. Good luck out there.</p>");

        final var body = template.getBody().replace("{name}", displayName);
        emailService.send(null, toAddress, template.getSubject(), body, true);
    }

    @Inject
    public void setEmailTemplateService(@Named(UNSCOPED) EmailTemplateService
emailTemplateService) {
        this.emailTemplateService = emailTemplateService;
    }

    @Inject
    public void setEmailService(EmailService emailService) {
        this.emailService = emailService;
    }
}
```

Because `getOrCreateEmailTemplate` only supplies its defaults the first time a key is seen, the template immediately becomes editable in the CMS under **Other > Email Templates**, right alongside the core templates, with no additional registration step. Use your own reverse-DNS key (e.g. `com.mystudio.mygame.welcome_email`), not the reserved `dev.getelements.elements.` prefix, which is rejected outside of this internal call.

Unlike the two core templates, there is no platform-wide registry of what variables a custom template supports. Document them for your own team elsewhere (e.g. in your Element's own README), since

the CMS only shows a variables reference for the core templates listed above.

See also

- Email Service - the underlying SMTP transport used to actually send the rendered subject and body.
- Email Verification - one of the two core templates, including token lifecycle and REST endpoints.
- CMS Feature Overview - a tour of the admin dashboard, including where Email Templates lives in the sidebar.