Chapter 6

Lab 6.2

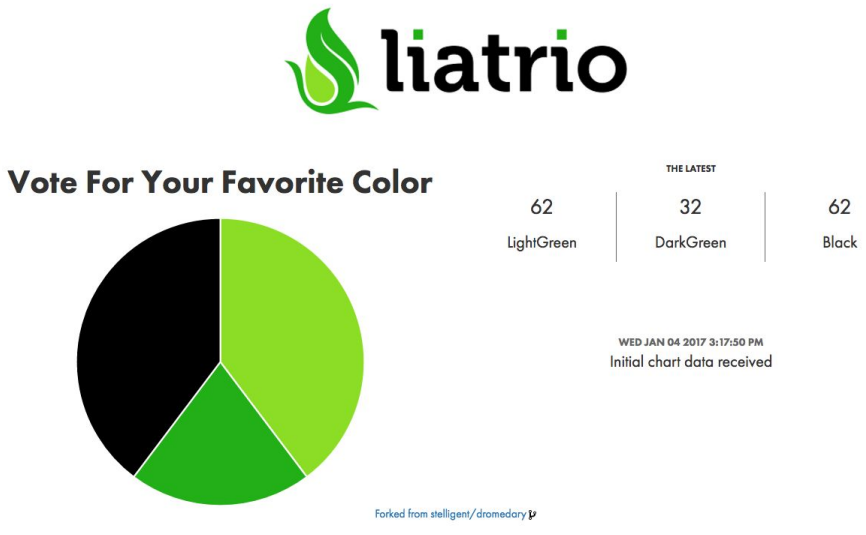
Feature Toggles

Feature toggles are a common software development pattern that allows for the activation/deactivation of various behaviors through some form of configuration. In this exercise, we will look at how Dromedary uses feature toggles to control the display of the build version number on its frontend.

Modifying the `feature-toggles.json` file

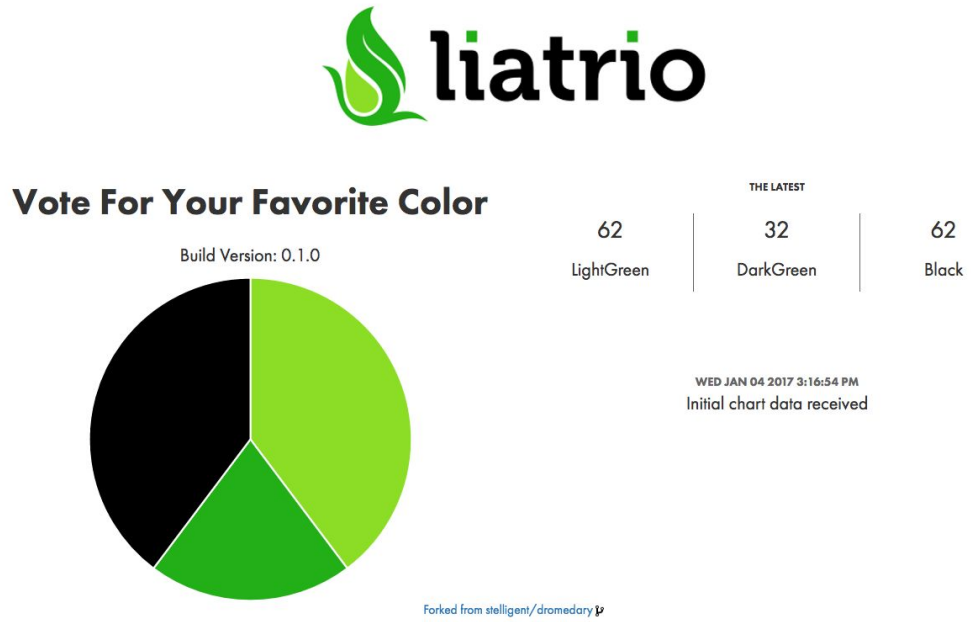
Dromedary uses a configuration file called **`feature-toggles.json`** to enable/disable features. For example, to disable the version-display feature, you can set **`version-display`** to **`false`** in the **`feature-toggles.json`** configuration file:

```
{
  'version-display': false
}
```



To re-enable it, set **version-display** back to **true** in the **feature-toggles.json** configuration file:

```
{
  'version-display': true
}
```



Implementation

Dromedary implements feature-toggling by serving the **feature** configuration file to the frontend, where features are enabled/disabled through basic control flow.

Feature toggle configuration is served on the **/feature-toggles** endpoint:

app.js

```
// GET requests to /feature-toggles return feature-toggles config object
app.get('/feature-toggles', function(req, res) {
  var featureFlags = require(__dirname + '/feature-toggles');
  sendJsonResponse(res, featureFlags);
});
```

Frontend requests the **feature-toggle** configuration to enable/disable features:

public/charthandler.js

```
// Load feature toggles config and enable/disable features accordingly
$.getJSON(apiBaseurl + '/feature-toggles', {}, function(featureFlags, status) {
  // version-display toggle
  if (featureFlags['version-display'] === true) {
    $('#build-version-message').show();
    loadBuildVersion();
  } else {
    $('#build-version-message').hide();
  }
});
```

A more sophisticated/secure way to implement feature toggles is to use a **Toggle Router**. For more information about this function, please visit http://martinfowler.com/articles/feature-toggles.html.

Use Cases

Dromedary's **`version-display`** feature is useful for debugging in non-production environments, but it can also be disabled as users might not want to see it. There are many other ways to use feature toggles to assist the development process in a CI/CD pipeline.

Release Toggles

Release toggles eliminate the need to create a new branch for every feature under development. Instead, features can be actively developed on the master branch, so long as they are kept disabled until they are production-ready.

Canary Releasing

Before fully releasing a new feature, it is often a good idea to perform a **canary release** by enabling the feature to only a small percentage of users. This allows developers to gauge the users' reception of a feature before publishing it to all users.

Permission Toggles

When a feature should be limited to a subset of users who have permission to access that feature (e.g. a feature that requires a subscription), permission toggles can be used to restrict access to that feature.

LaunchDarkly Exercise

Sign up for LaunchDarkly at https://launchdarkly.com/. After you register, you will get an email with a "Start Now" button. Click on that button and then insert the following link: https://app.launchdarkly.com/default/production/quickstart#/install-sdk/node. You should receive instructions on how to modify the Dromedary application and some new feature flags using the LaunchDarkly platform.

Note: These are some basic examples that show automated deployment techniques in a Continuous Delivery pipeline. For additional examples and real world complexity, please refer to Liatrio's Continuous Delivery page.

