Feature Flags Overview

Feature flags allow your development team to turn various features on and off in your production environment for a percentage of your overall traffic without requiring a release cycle or, more importantly, a rollback. By using this tool, you can run experiments against a flagged feature and capture the results. Based on those results, your marketing team can then control naming, rolling out, implementing, and managing the feature.

The primary purpose of feature flags is to progressively roll out new functionality on your site and to allow others in your organization to control this rollout without having to depend on the developers who built the feature. Unlike normal test experiences that you build within Monetate, your team develops the feature and analyzes the data outside the platform.

Consider this example. Your site needs a new custom checkout flow, but you're unsure if what your development team has built is ready for widespread deployment. You want to test the new flow by first showing it to a small percentage of the site's user traffic so that you can identify any issues and fix them without inconveniencing your customers. By having a developer trigger the new flow with a feature flag, you can then achieve this real-world testing on a small scale.

Getting Started with Feature Flags

Before you can begin using feature flags, you must first contact your dedicated Customer Success Manager to have the feature flags component added to your commerce account. You should also ask your CSM any questions you may have about the JavaScript SDK your team of developers need for this component.

Once these steps are complete, your development team can create a feature flag in Monetate. (See Create a Feature Flag for instructions.) After you create the feature flag, you can then link it to the code developed for the new feature.

Setting Up Feature Flags on a Commerce Site

To set up feature flags on your site, you must add something similar to the code below to get the feature flag configuration data. You should only need to add this code once per page or site visit, depending on the setup of `featureFlags`.

```
var featureFlags = {};
function onPageLoad(...){
  // Set up variables needed to make the request
  const tuple = 'a-123456/p/commerce.com';
  const URL = 'https://sb.monetate.net/features/1/flags/' + tuple;
  const CONFIG = {
    method: 'GET'
  };
  // Make the request
  fetch(URL, CONFIG).then((response) => {
      return response.json();
    }).then((data) => {
      // Parse the flags and have them saved to a global variable
      featureFlags = data;
      // The mid can be any identifying value for the current user.
      // getCookie() is not provided, only here as an example
      const mid = getCookie('mt.v');
      window.monetate.featureFlag.parseTogglesConfig(featureFlags, mid);
  }
}
```

Customers and Feature Flag Tests

To illustrate how customers are deemed eligible for a flagged feature, imagine that your commerce site has feature 1 and feature 2, and two customers come to the site and are each assigned a Monetate ID (MID). Once they have their feature flag percentiles determined, the breakdown would look something like this:

- **Feature 1**
  - Customer A (MID 2.1111.11111) — 20%
  - Customer B (MID 2.2222.22222) — 80%
- **Feature 2**
  - Customer A (2.1111.11111) — 60%
  - Customer B (2.2222.22222) — 10%

In this example, notice that the percentile for each customer MID is calculated independently for each feature flag. As long as the customer's MID stays constant, those results are exactly the same for every visit.

Assuming that feature 1 and feature 2 each has its respective feature flag set to 60% and 10%, the eligibility looks something like this:

- **Feature 1**
  - Customer A (2.1111.11111) — True (20% ≤ 60%)
  - Customer B (2.2222.22222) — False (80% ≤ 60%)
- **Feature 2**
  - Customer A (2.1111.11111) — False (60% ≤ 10%)
  - Customer B (2.2222.22222) — True (10% ≤ 10%)

If you were to change the feature flag percentages, each customer's eligibility may also change because the percentiles stay the same and would compare differently to the new percentages. Therefore, if you were to change the percentages in the example used thus far to 10% and 70%, respectively, then the new eligibility calculations would be as follows:

- **Feature 1**
  - Customer A (2.1111.11111) — False (20% ≤ 10%)
  - Customer B (2.2222.22222) — False (80% ≤ 10%)
- **Feature 2**
  - Customer A (2.1111.11111) — True (60% ≤ 70%)
  - Customer B (2.2222.22222) — True (10% ≤ 70%)

Using Monetate ID to Identify Customers

In the previous example, the Monetate ID (MID) is used to determine if a customer should see the flagged feature. This determination is held constant for the MID on every visit. For example, customer A with the MID 2.1111.11111, is determined eligible to see feature 1 and will always see feature 1 until their Monetate ID changes or the traffic percentage for the feature flag changes, as shown previously. The change in MID is in line with the standard Monetate session constraints. This means that if the session for customer A ends for any reason, their Monetate ID changes, thus causing their eligibility determination to change as well.

Continuing with the previous example, the session for customer A ends and then they revisit the site. They would then be assigned a new MID: 2.3333.33333. Assuming that the percentages for the feature flags hold constant, the breakdown would look something like this:

- **Feature 1**
  - Customer A (MID 2.3333.33333) — 70%
  - Customer B (MID 2.2222.22222) — 80%
- **Feature 2**
  - Customer A (2.3333.33333) — 80%
  - Customer B (2.2222.22222) — 10%

As demonstrated here, the percentile for customer B stays constant since their MID has as well, whereas customer A received a new percentile to correspond to their new MID. This change would result in the following eligibility:

- **Feature 1**
  - Customer A (2.3333.33333) — False (70% ≤ 10%)
  - Customer B (2.2222.22222) — False (80% ≤ 10%)
- **Feature 2**
  - Customer A (2.3333.33333) — False (80% ≤ 70%)
  - Customer B (2.2222.22222) — True (10% ≤ 70%)

Using Other Means to Identify Customers

The ability for a customer to see a flagged feature is based on whatever ID is given to the feature flag SDK helper function. In the Monetate ID example, a customer's eligibility for viewing a feature is only held constant over their Monetate session. Although this approach is the most standard, the ID given to the feature flag SDK helper function can be anything. This means that if you use a different method to identify a customer by a more certain and constant metric, such as the Monetate customer ID or your own in-house account ID, their determined eligibility for a feature can be constant across all sessions and devices. The only caveat is that the added check for this consistent identifier means that the feature flag only applies to that subset of traffic rather than to all traffic.