1 Driver API 3
   1.1 Driver ......................................................... 3
   1.2 Storage ......................................................... 5
Normandy is the server-side component of the Recipe Server. It serves bundles of JavaScript to various clients based on certain rules. The Normandy Actions repository contains the implementations of actions that are served by Normandy.
The Normandy driver is an object passed to all actions when they are created. It provides methods and attributes for performing operations that require more privileges than JavaScript is normally given. For example, the driver might provide a function for showing a notification bar within the Firefox UI, something normal JavaScript can’t trigger.

Environments in which actions are run (such as the Normandy system add-on) implement the driver and pass it to the actions before executing them. The driver is versioned, and the version of this documentation corresponds to the version of the driver that it documents.

1.1 Driver

The driver object contains the following attributes:

**testing**
Boolean describing whether the action is being executed in “testing” mode. Testing mode is mainly used when previewing recipes within the Normandy admin interface.

**locale**
String containing the locale code of the user’s preferred language.

**log**(message, level=’debug’)
Log a message to an appropriate location. It’s up to the driver where these messages are stored; they could go to the browser console, or to a remote logging service, or somewhere else. If level is ’debug’, then messages should only be logged if testing is true.

Arguments
- **message** – Message to log
- **level** – Level to log message at, such as debug, info, or error. Defaults to debug.

**showHeartbeat**(options)
Displays a Heartbeat survey to the user. Appears as a notification bar with a 5-star rating input for users to vote with. Also contains a “Learn More” button on the side. After voting, users are shown a thanks message and optionally a new tab is opened to a specific URL.

Arguments
- **message** – Primary message to display alongside rating stars.
- **thanksMessage** – Message to show after user submits a rating.
- **flowId** – A UUID that should be unique to each call to this function. Used to track metrics related to this user interaction.
• **postAnswerUrl** – URL to show users after they submit a rating. If empty, the user won’t be shown anything.

• **learnMoreMessage** – Text to show on the “Learn More” button.

• **learnMoreUrl** – URL to open when the “Learn More” button is clicked.

• **extraTelemetryArgs** – Object containing extra arguments to send to telemetry associated with this heartbeat call. Defaults to an empty object.

**Returns** A Promise that resolves with an event emitter.

The emitter returned by this function can be subscribed to using `on` method. For example:

```javascript
let heartbeat = await Normandy.showHeartbeat(options);
heartbeat.on('NotificationOffered', function(data) {
    // Do something!
});
```

All events are given a data object with the following attributes:

- **flowId** The `flowId` passed into `showHeartbeat`.
- **timestamp** Timestamp (number of milliseconds since Unix epoch) of when the event being emitted occurred.

The events emitted by the emitter include:

- **NotificationOffered** Emitted after the notification bar is shown to the user.
- **NotificationClosed** Emitted after the notification bar closes, either by being closed manually by the user, or automatically after voting.
- **LearnMore** Emitted when the user clicks the “Learn More” link.
- **Voted** Emitted when the user clicks the star rating bar and submits a rating. An extra `score` attribute is included on the data object for this event containing the rating the user submitted.
- **TelemetrySent** Emitted after Heartbeat has sent flow data to the Telemetry servers. Only available on Firefox 46 and higher.

**uuid()**
Generates a v4 UUID. The UUID is randomly generated.

**Returns** String containing the UUID.

**createStorage**(keyPrefix)
Creates a storage object that can be used to store data on the client.

**Arguments**

- **keyPrefix** – Prefix to append to keys before storing them, to avoid collision with other actions using the storage.

**Returns** Storage()

**location()**
Retrieves information about where the user is located.

**Returns** A Promise that resolves with a location object.

The location object has the following fields:

- **countryCode** ISO 3166-1 country code for the country the user has been geolocated to.

**saveHeartbeatFlow**(data)
Sends flow data from Heartbeat to the Input server. See the Input documentation for details about the data expected.
Arguments

- **data** – Object containing Heartbeat flow data.

**client()**
Retrieves information about the user’s browser.

**Returns** Promise that resolves with a client data object.

The client data object includes the following fields:

- **version** String containing the Firefox version.
- **channel** String containing the update channel. Valid values include, but are not limited to:
  - ‘release’
  - ‘aurora’
  - ‘beta’
  - ‘nightly’
  - ‘default’ (self-built or automated testing builds)
- **isDefaultBrowser** Boolean specifying whether Firefox is set as the user’s default browser.
- **searchEngine** String containing the user’s default search engine identifier.
- **syncSetup** Boolean containing whether the user has set up Firefox Sync.

### 1.2 Storage

**class Storage()**
Storage objects allow actions to store data locally on the client, using an API that is similar to localStorage, but is asynchronous.

**getItem(key)**
Retrieves a value from storage.

**Arguments**

- **key** – Key to look up in storage.

**Returns** A Promise that resolves with the value found in storage, or null if the key doesn’t exist.

**setItem(key, value)**
Inserts a value into storage under the given key.

**Arguments**

- **key** – Key to insert the value under.
- **value** – Value to store.

**Returns** A Promise that resolves when the value has been stored.

**removeItem(key)**
Removes a value from storage.

**Arguments**

- **key** – Key to remove.

**Returns** A Promise that resolves when the value has been removed.
C
client() (built-in function), 5
createStorage() (built-in function), 4

G
getItem() (built-in function), 5

L
locale (global variable or constant), 3
location() (built-in function), 4
log() (built-in function), 3

R
removeItem() (built-in function), 5

S
saveHeartbeatFlow() (built-in function), 4
setItem() (built-in function), 5
showHeartbeat() (built-in function), 3
Storage() (class), 5

T
testing (global variable or constant), 3

U
uuid() (built-in function), 4