Getting started

This section provides guidance on how to create your first Kaa application that will work with the Kaa platform. In this guide we will show you how to create a simple desktop java application that will receive notifications from the Kaa server and display them on the console. We will define our own notification schema and use the generated java classes within our application.

Installation and configuration

Before you start using the Kaa framework, you need to install it. You can install Kaa in the single node mode or distributed mode. The installation procedure is described in Installation guide.

However, we recommend that you start exploring Kaa using Sandbox. Kaa Sandbox is an easy-to-use virtual environment that includes all the components that you need in order to learn Kaa, build a proof of concept and test your own applications locally. Sandbox also includes demo client applications.

The Sandbox setup procedure is straightforward and can be done instantly. See Sandbox guide for details.

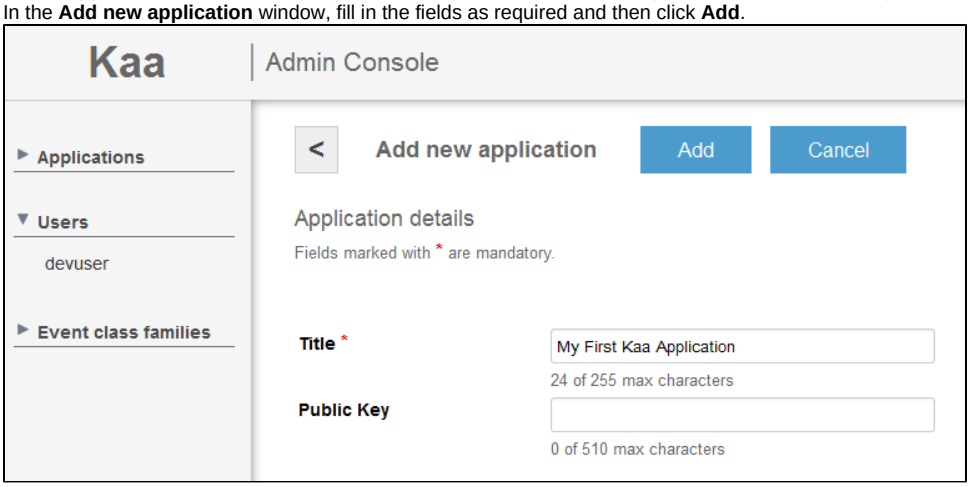
Your first Kaa application

To register a new application within a fresh Kaa server installation, you need to create users with the *tenant administrator* and *tenant developer* roles. T he tenant administrator is responsible for creating new applications in Kaa, and the tenant developer configures and generates SDKs for those applications. We suggest that you use Kaa Sandbox, which has a tenant administrator and tenant developer users already created.

Add application

To add an application, proceed as follows:

1. Open the Kaa admin UI in your browser (http://127.0.0.1:8080) and log in as a tenant administrator (user/password: admin/admin123).
2. Select **Applications** on the navigation panel on the left side and, in the **Applications** window that opens, click **Add new application.**
3. In the **Add new application** window, fill in the fields as required and then click **Add**.



After the application has been added, you may log out. We will not be using the tenant administrator role in this guide anymore.

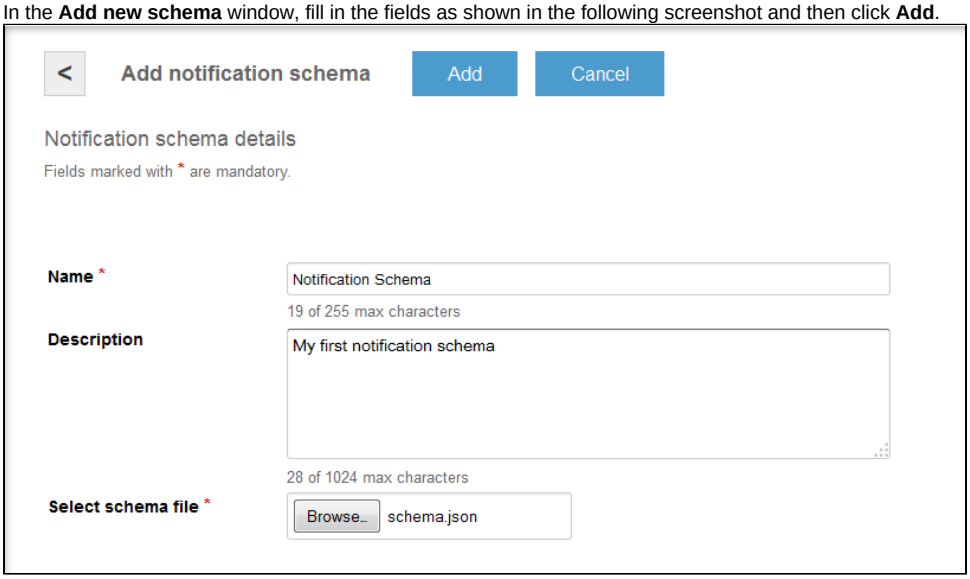
Create notification schema

The application that you have created in the previous step already includes the default versions of the profile, configuration, notification and log schemas ready for use. However, in this sample application, we will use a custom notification schema for demonstration purposes. To create and upload the schema, proceed as follows:

1. Create the *schema.json* file on your PC with the following schema definition:

```
{
    "type": "record",
    "name": "Notification",
    "namespace": "org.kaaproject.kaa.schema.example",
    "fields": [
        {
            "name": "message",
            "type": "string"
        }
    ]
}
```

2. Open the admin UI in your browser (http://127.0.0.1:8080) and log in as a tenant developer (user/password: devuser/devuser123).
3. Open the relevant **Notification schemas** window (**Applications -> My First Kaa Application -> Schemas -> Notification**) and click **Add new schema**.
4. In the **Add new schema** window, fill in the fields as shown in the following screenshot and then click **Add**.



As a result of this operation you will see two notification schemas in the list:

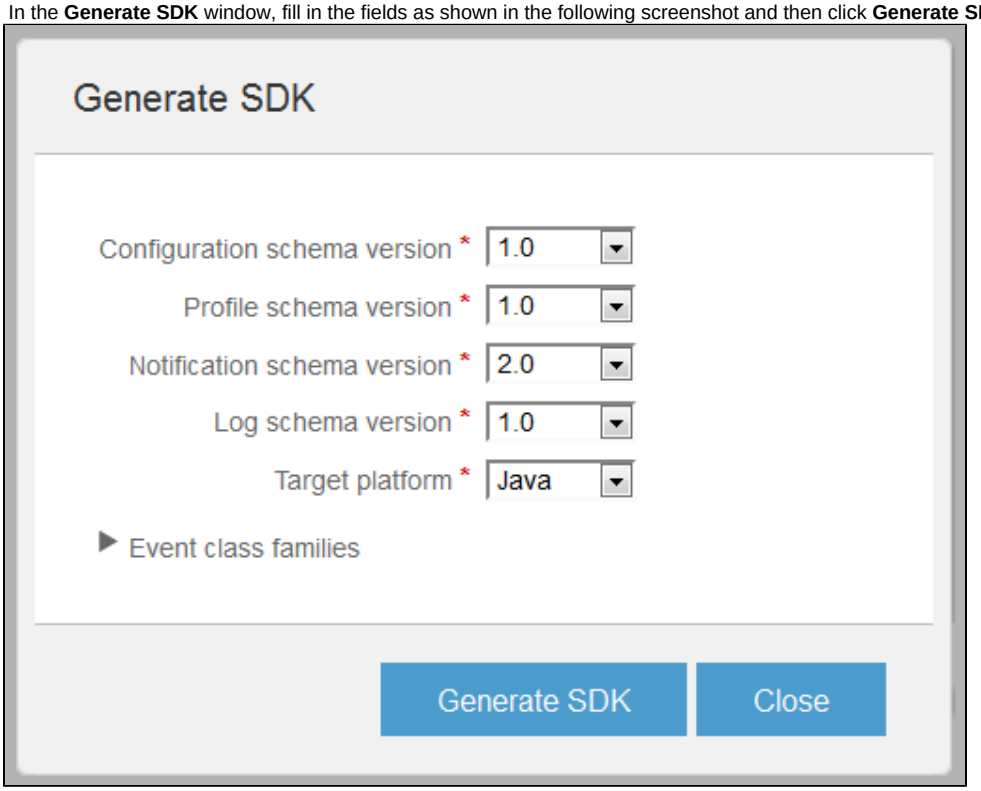
In this screenshot, version 2.0 is the notification schema that was just created. We will use this version for the SDK generation in the next step.

Generate SDK

To generate the SDK for the new application, proceed as follows:

1. Select the **My First Kaa Application** application and click **Generate SDK.**

2. In the **Generate SDK** window, fill in the fields as shown in the following screenshot and then click **Generate SDK**.



After the SDK is generated, you will be presented with a window asking you to save a .jar file with the generated SDK. Specify the file name and location on your computer and then click **Save**. The SDK is now downloaded to your computer.

Please note that we are generating an SDK based on the default configuration, profile, and log schemas. These schemas are automatically populated during the application's creation. If necessary, you can overwrite them using the admin UI.

Sample client application

Once you have downloaded the SDK, you can use it in your sample project. The following code block illustrates a simple desktop java application that will receive notifications from the Kaa server and display them on the console.

```
package org.kaaproject.kaa.samples.nf;

import java.util.List;

import org.kaaproject.kaa.client.Kaa;
import org.kaaproject.kaa.client.KaaClient;
import org.kaaproject.kaa.client.KaaDesktop;
import org.kaaproject.kaa.client.notification.AbstractNotificationListener;
import org.kaaproject.kaa.client.notification.NotificationTopicListListener;
import org.kaaproject.kaa.client.profile.AbstractProfileContainer;
import org.kaaproject.kaa.common.endpoint.gen.Topic;
import org.kaaproject.kaa.schema.base.Profile;
import org.kaaproject.kaa.schema.example.Notification;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

public class NotificationSystemTestApp {

    private static final Logger LOG = LoggerFactory.getLogger(NotificationSystemTestApp.class);

    public static void main(String[] args) throws Exception {
        new NotificationSystemTestApp().launch();
        System.in.read();
    }

    private void launch() throws Exception {
        // Create instance of desktop Kaa application
        Kaa kaa = new KaaDesktop();
```

```
        // Create client for Kaa SDK
        KaaClient client = kaa.getClient();
        // Set simple profile container that reports current endpoint profile to
        // SDK.
        client.getProfileManager().setProfileContainer(new AbstractProfileContainer() {
            @Override
            public Profile getProfile() {
                return new Profile();
            }
        });
        // Starts Kaa SDK client
        kaa.start();
        LOG.info("Kaa SDK client started!");
        // Registering listener for topic updates
        client.getNotificationManager().addTopicListListener(new NotificationTopicListListener() {
            @Override
            public void onListUpdated(List<Topic> topicList) {
                LOG.info("Topic list updated!");
                for (Topic topic : topicList) {
                    LOG.info("Received topic with id {} and name {}", topic.getId(), topic.getName());
                }
            }
        });
        // Registering listener for notifications
        client.getNotificationManager().addNotificationListener(new AbstractNotificationListener() {
            @Override
            public void onNotification(String topicId, Notification notification) {
                LOG.info("Received notification {} for topic with id {}", notification, topicId);
            }
        });
    }
}
```

You can find the project source code in the attached archive. The project is built using Apache Maven. Please note that the downloaded SDK must be placed into the *lib* folder in order for the build to work.

Please import this project into your IDE as a Maven project and launch the *NotificationSystemTestApp* application. Once application is launched, you will see the following output in the application:

```
[...  INFO  NotificationSystemTestApp]  Kaa SDK client started!
```

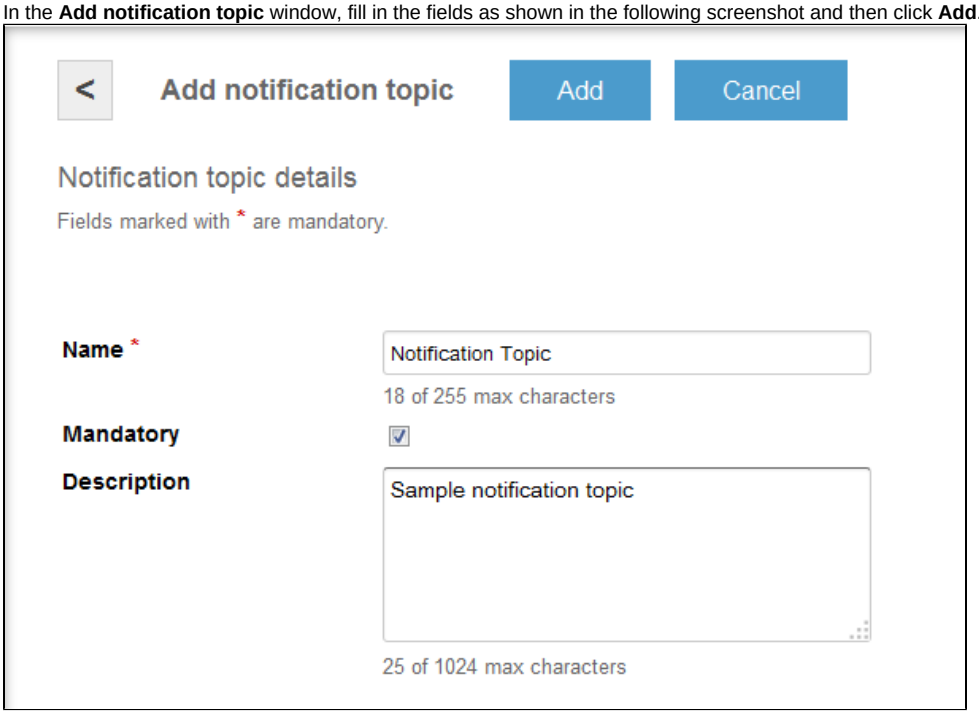
Create notification topic

To send your first Kaa notification, you need to create a notification topic and assign this topic to the default endpoint group.

To create a notification topic, proceed as follows:

1. Open the relevant **Notification topics** window (**Applications -> My First Kaa Application -> Notification topics**) and click **Add new notification topic**.

2. In the **Add notification topic** window, fill in the fields as shown in the following screenshot and then click **Add**.

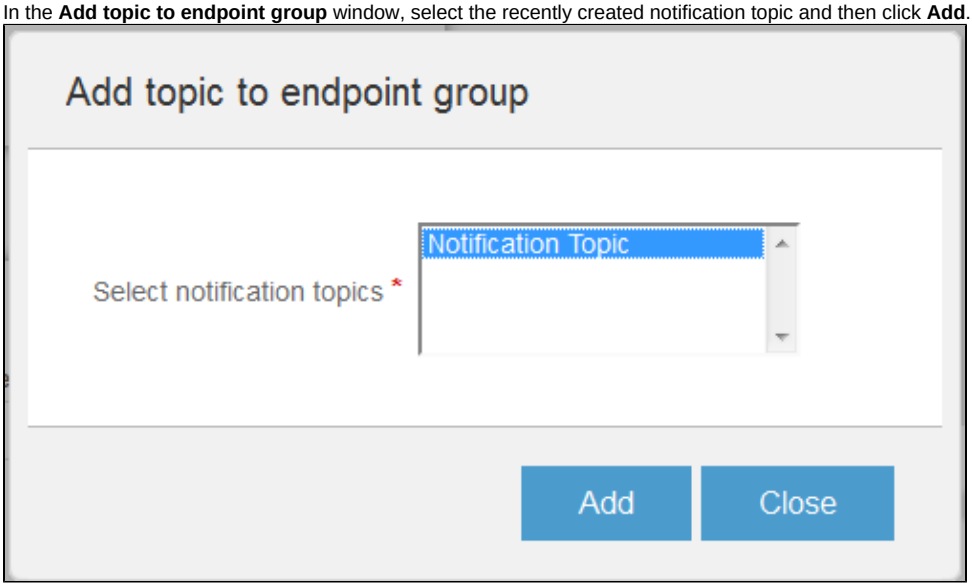


NOTE: We set the topic as mandatory in order to automatically subscribe the client application to notifications on this topic.

Once the topic is created, we will assign it to the default endpoint group, which contains all endpoints, including endpoints with our application.

To assign a notification topic to the default endpoint group, proceed as follows:

1. In the relevant **Endpoint groups** window (**Applications->My First Kaa Application->Endpoint groups**), select the *All* group.
2. In the **Endpoint group details** window, click **Add notification topic** at the bottom of the window.
3. In the **Add topic to endpoint group** window, select the recently created notification topic and then click **Add**.



After the topic is added to the endpoint group, you will see the following output in the application:

```
[...  INFO  NotificationSystemTestApp]  Topic list updated!
[...  INFO  NotificationSystemTestApp]  Received topic with id X and name Notification Topic
```

This is the first update from the Kaa server that provides the client with the information about the changes in the topic list. Now you can send notifications on this topic, and they will be delivered to your application.

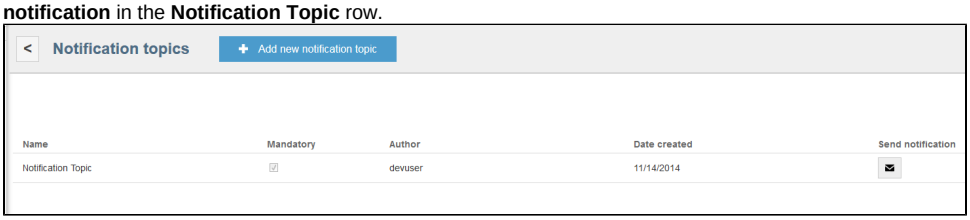
Create notification

To create a notification, proceed as follows:

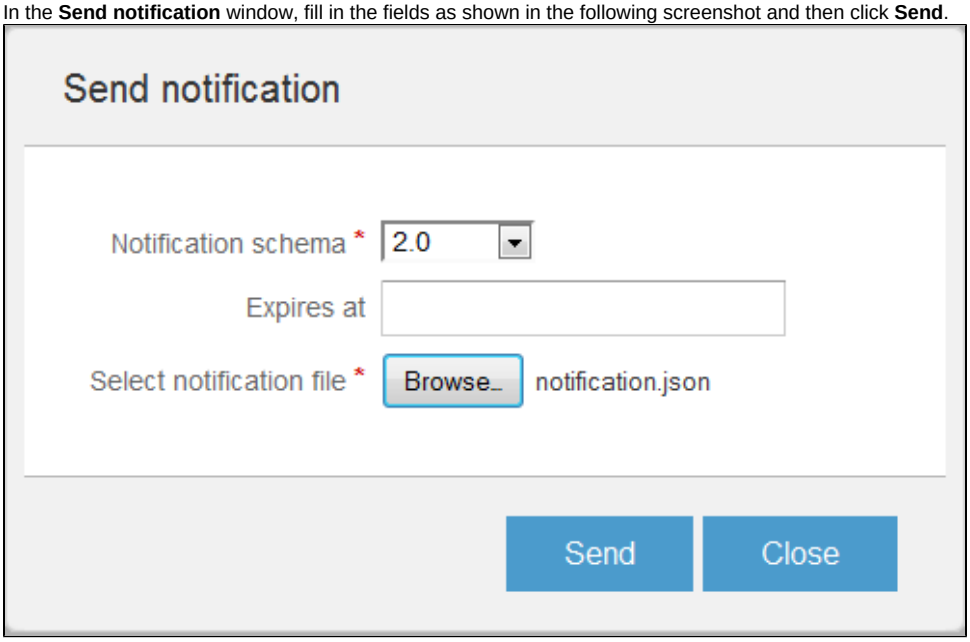
1. Create a *notification.json* file on your PC with the following contents:

```
{"message":"Hello from Kaa!"}
```

2. Open the relevant **Notification topics** window (**Applications -> My First Kaa Application -> Notification topics**) and click **Send notification** in the **Notification Topic** row.



3. In the **Send notification** window, fill in the fields as shown in the following screenshot and then click **Send**.



Once the notification is sent, you can see the following output in the application:

```
[...  INFO  NotificationSystemTestApp]  Received notification {"message": "Hello from Kaa!"} for topic with id X
```

Congratulations with your first Kaa application!

Further reading

Use the following guides and references to make the most of Kaa.

Guide	What it is for
Design reference	Use this reference to learn about features and capabilities of Kaa (Endpoint profiling, Events, Notifications, Logging, and other features).
Sandbox	Use this guide to try out Kaa in a private environment with demo applications.
Development environment setup	Use this guide to set up necessary environment for installing and programming Kaa.
Installation guide	Use this guide to install and configure Kaa either on a single Linux node or in a cluster environment.
Contribute to Kaa	Use this guide to learn how to contribute to Kaa project and which code/documentation style conventions we adhere to.

If you have any questions or need assistance, do not hesitate to use our forum.

Also, we will appreciate it if you help us improve the quality of this guide. Please let us know via email about any typos or factual mistakes that you find.

Thank you for using Kaa!
